Netkiller 手札系列电子书 http://www.netkiller.cn

Netkiller Blockchain 手札

陈景峰 著

HYPERLEDGER

BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

Netkiller Blockchain 手札

目录

5. 连接控制台
    5.1. 指定 geth.ipc 文件位置
    5.2. IPC 方式连接
    5.3. TCP 连接控制台
    5.4. WebSocket 方式
6. 账号管理
    6.1. 新建账号
    6.2. 查看账号
    6.3. 从私钥导入以太坊地址
7. 配置自动解锁账号
8. 运行JS
9. 节点管理
10. 启动挖矿
    10.1. 挖矿线程数
    10.2. 指定旷工账号
11. 运行智能合约
12. Ropsten测试网络
13. 静态节点
14. JavaScript Console
    14.1. personal 管理
        14.1.1. 创建账号
        14.1.2. 列出账号
        14.1.3. 解锁账号
    14.2. eth 管理
        14.2.1. 矿工账号
        14.2.2. 余额
            14.2.2.1. 单位转换
            14.2.2.2. 一次检查所有账号余额
        14.2.3. 解锁账号
        14.2.4. 转账
        14.2.5. 查看挂起的交易

    4.2. Truffle console 控制台
    4.3. create
        4.3.1. contract 创建合约
        4.3.2. test 创建单元测试
    4.4. migrate
    4.5. compile
    4.6. test
    4.7. watch
5. 合约开发
    5.1. 构造方法
6. truffle console
    6.1. 获取账号列表
    6.2. 余额
    6.3. 实例化合约
    6.4. 访问 public 变量
    6.5. 调用 public 函数
7. 测试
    7.1. balanceOf
    7.2. transfer
8. TRUFFLE BOXES
9. Zeppelin Solidity - OpenZeppelin is a library for writing secure Smart Contracts on Ethereum.
    9.1. ERC20
    9.2. ERC872

16. web3.js - 1.0.0
1. 开发环境
    1.1. Ropsten 测试网
2. truffle-contract
3. 连接到以太坊客户端
    3.1. http 方式
    3.2. WebSocket 方式
    3.3. IPC 方式
4. web3

        21.3. TestRestController
        21.4. 测试
18. web3.py - A python interface for interacting with the Ethereum blockchain and ecosystem.
    1. 安装 web3.py 开发环境
        1.1. CentOS
        1.2. MAC OS
    2. 连接到以太坊节点
        2.1. HTTP
        2.2. IPC
        2.3. Websocket
    3. 交易
        3.1. 发送 ETH
        3.2. 签名发送 ETH
    4. ERC20 代币合约
        4.1. 签名发送ERC20代币
19. Ethereum iOS
20. Ethereum Developer APIs
    1. API Keys
    2. 账号
        2.1. 余额
        2.2. 查询区块
        2.3. 查询区块
    3. 查询交易
        3.1. 检查合约执行状态
    4. Geth/Parity Proxy APIs
        4.1.
    5. JSON RPC 原生交口调用
21. infura
    1. Infura 3.0
    2. websocket
        2.1. 订阅 newBlockHeaders
    3. 配置 Truffle

5. Unhandled rejection Error: Returned error: The method personal_unlockAccount does not exist/is not available
6. Error: exceeds block gas limit
7. Migrations.sol:11:3: Warning: Defining constructors as functions with the same name as the contract is deprecated. Use "constructor(...) { ... }" instead.
8. Exception in thread "main" rx.exceptions.OnErrorNotImplementedException: Invalid response received: okhttp3.internal.http.RealResponseBody@6c25e6c4

III. Hyperledger

24. Hyperledger Fabric v2.0.0

1. 安装 Hyperledger Fabric v1.1.x

1.1. 依赖工具

1.2. 安装Docker

1.3. 安装 Node.js 环境

1.4. 安装 hyperledger 1.1.0

1.5. 手工安装 hyperledger v 1.1.0 开发环境

1.5.1. 登录 docker

1.5.2. Docker 安装

1.5.3. 编译安装

1.6. 启动 docker 虚拟机

1.7. 管理 hyperledger

1.7.1. CouchDB 管理界面

1.8. 部署 chaincode

1.8.1. channel 管理

1.8.1.1. 列出 channel

1.8.1.2. 创建 Channel

1.8.1.3. 加入 Channel

1.8.2. 部署连

1.8.3. 查询合约

1.8.4. 调用合约

25. Hyperledger Fabric 运维
    1. 背景
    2. 部署拓扑
        2.1. 依赖关系
        2.2. 准备物理机
    3. cli 管理节点安装
        3.1. 安装 Docker 镜像
        3.2. docker-compose-cli.yaml
        3.3. 启动 cli 节点
        3.4. 生成证书和创世区块
            3.4.1. 创建配置文件
                3.4.1.1. crypto-config.yaml
                3.4.1.2. configtx.yaml
            3.4.2. 生成证书
            3.4.3. 生成创世区块
            3.4.4. 生成通道配置文件
            3.4.5. generate anchor peer transaction
        3.5. 清理 Docker 容器
    4. CA 节点安装
        4.1. 安装 Docker 镜像
        4.2. docker-compose-ca.yml
        4.3. 启动 CA 节点
    5. CouchDB 节点
        5.1. 安装 Docker 镜像
        5.2. 安装 CouchDB
        5.3. 启动 CouchDB
        5.4. 备份与恢复 CouchDB
    6. Orderer 节点安装
        6.1. 安装 Docker 镜像
        6.2. docker-compose-orderer.yml
        6.3. 启动 Orderer 节点

35. IPFS WebUI
    1. 配置 CORS
    2. 访问 Web UI
    3. HTTP 网关
        3.1. 查看网关地址
        3.2. 添加测试文件
        3.3. 配置代理服务器
        3.4.
        3.5. 监听地址
36. IPFS 集群配置
    1. 手工添加节点
        1.1.
        1.2.
37. IPFS API
    1. 启动 IPFS API
    2. 原始 HTTP API
        2.1. 查看节点
        2.2. 上传文件
    3. Infura IPFS API
        3.1. 查看文件
        3.2. 下载文件
    4. java-ipfs-api
        4.1. Maven 配置
        4.2. 查看版本号
        4.3. 添加文件到 IPFS
        4.4. 添加文本到 IPFS
        4.5.
    5. js-ipfs-api
        5.1. 开发环境
        5.2. 链接到 IPFS
38. IPFS Faq
    1. 一个大文件将会被分块存储

2. Ethereum
    2.1. web3.js Document
    2.2. Standardizing of HD wallet derivation paths
3. NXP（恩智浦）相关产品
    3.1. MifareUltralight
    3.2. MifareClassic
    3.3. Mifare
4. NFC 数据格式
    4.1. NFC 标准
    4.2. NDEF (NFC Data Exchange Format)

表格清单

Netkiller Blockchain 手札

本文作者最近在找工作，有意向致电 13113668890

ISBN#
ISBN#

Mr. Neo Chan, 陈景峯(BG7NYT)

中国广东省深圳市望海路半岛城邦三期
518067
+86 13113668890

<netkiller@msn.com>

电子书最近一次更新于 2020-11-15 03:25:12 .

http://www.netkiller.cn
http://netkiller.github.io
http://netkiller.sourceforge.net

微信订阅号 netkiller-ebook
(微信扫描二维码)
QQ：13721218 请注明“读者”
QQ群：128659835 请注明“读者”

知乎专栏 | 多维度架构

2020-11-15 03:25:12

内容摘要

这一部关于区块链开发及运维的电子书。

为什么会写区块链电子书？因为2018年是区块链年，区块链是一个风口，前几个风口我都错过了。例如web2.0, 云， 大数据等等，都从身旁擦肩而过。所以我要抓住这次。

这本电子书是否会出版（纸质图书）？ 不会，因为互联网技术更迭太快，纸质书籍的内容无法实时更新，一本书动辄百元，很快就成为垃圾，你会发现目前市面的上区块链书籍至少是一年前写的，内容已经过时，很多例子无法正确运行。所以我不会出版，电子书的内容会追逐技术发展，及时跟进软件版本的升级，做到内容最新，至少是主流。

这本电子书与其他区块链书籍有什么不同？市面上大部分区块链书籍都是用2/3去讲区块链原理，只要不到 1/3 的干货，干货不够理论来凑，通篇将理论或是大谈特谈区块链行业，这些内容更多是头脑风暴，展望区块链，均无法落地实施。本书与那些书籍完全不同，不讲理论和原理，面向应用落地，注重例子，均是干货。

写作原则，无法落地的项目作者绝对不会写。凡是写入电子的内容均具备可操作，可落地。

电子书更新频率？每天都会有新内容加入，更新频率最迟不会超过一周，更新内容请关注 https://github.com/netkiller/netkiller.github.io/commits/master

本文采用碎片化写作，原文会不定期更新，请尽量阅读原文。
http://www.netkiller.cn/blockchain/index.html

您的打赏是我的写作动力： http://www.netkiller.cn/blockchain/donations.html

接受 ETH 捐赠： 0x3e827461Cc53ed7c75A29187CfF39629FCAE3661

喜大普奔！读者币（Netkiller eBook Reader Coin - NBRC）正式开始空投，请在钱包中添加NBRC代币即可看到1000枚读者币。建议使用 Ethereum Wallet (Mist) 操作 NBRC，在 CONTRACTS菜单点击WATCH TOKEN按钮，输入合约地址 0x4488ed050cd13ccfe0b0fcf3d168216830142775 。注意imtoken 计算 Gas 有问题，转账会失败。MetaMask 建议 Gas Price 4GWei, Gas Limit 200000 费用 0.0004 左右的ETH

表 1. 企业招聘信息广告位，区块链工作机会

LOGO 广告位招租 LOGO 广告位招租 LOGO 广告位招租

广告发布，请联系 13113668890

致读者

Netkiller 系列电子书始于 2000 年，风风雨雨走过20年，将在 2020 年终结，之后不在更新。作出这种决定原因很多，例如现在的阅读习惯已经转向短视频，我个人的时间，身体健康情况等等.....

感谢读者粉丝这20年的支持

虽然电子书不再更新，后面我还会活跃在知乎社区和微信公众号

自述

《Netkiller 系列 手札》是一套免费系列电子书，netkiller 是nickname 从1999 开使用至今，“手札” 是札记，手册的含义。

2003年之前我还是以文章形式在BBS上发表各类技术文章，后来发现文章不够系统，便尝试写长篇技术文章加上章节目录等等。随着内容增加，不断修订，开始发布第一版，第二版.....

IT知识变化非常快，而且具有时效性，这样发布非常混乱，经常有读者发现第一版例子已经过时，但他不知道我已经发布第二版。

我便有一种想法，始终维护一个文档，不断更新，使他保持较新的版本不过时。

第一部电子书是《PostgreSQL 实用实例参考》开始我使用 Microsoft Office Word 慢慢随着文档尺寸增加 Word 开始表现出力不从心。

我看到PostgreSQL 中文手册使用SGML编写文档，便开始学习Docbook SGML。使用Docbook写的第一部电子书是《Netkiller Postfix Integrated Solution》这是Netkiller 系列手札的原型。

至于“手札”一词的来历，是因为我爱好摄影，经常去一个台湾摄影网站，名字就叫“摄影家手札”。

由于硬盘损坏数据丢失 《Netkiller Postfix Integrated Solution》的 SGML文件已经不存在；Docbook SGML存在很多缺陷UTF-8支持不好，转而使用Docbook XML.

目前技术书籍的价格一路飙升，动则￥80，￥100，少则￥50，￥60. 技术书籍有时效性，随着技术的革新或淘汰，大批书记成为废纸垃圾。并且这些书技术内容雷同，相互抄袭，质量越来越差，甚至里面给出的例子错误百出，只能购买影印版，或者翻译的版本。

在这种背景下我便萌生了自己写书的想法，资料主要来源是我的笔记与例子。我并不想出版，只为分享，所有我制作了基于CC License 发行的系列电子书。

本书注重例子，少理论（捞干货），只要你对着例子一步一步操作，就会成功，会让你有成就感并能坚持学下去，因为很多人遇到障碍就会放弃，其实我就是这种人，只要让他看到希望，就能坚持下去。

1. 写给读者

为什么写这篇文章

有很多想法,工作中也用不到所以未能实现，所以想写出来,和大家分享.有一点写一点,写得也不好,只要能看懂就行,就当学习笔记了.

开始零零碎碎写过一些文档，也向维基百科供过稿，但维基经常被ZF封锁，后来发现sf.net可以提供主机存放文档，便做了迁移。并开始了我的写作生涯。

这篇文档是作者20年来对工作的总结,是作者一点一滴的积累起来的，有些笔记已经丢失，所以并不完整。

因为工作太忙整理比较缓慢。目前的工作涉及面比较窄所以新文档比较少。

我现在花在技术上的时间越来越少，兴趣转向摄影，无线电。也想写写摄影方面的心得体会。

写作动力:

曾经在网上看到外国开源界对中国的评价，中国人对开源索取无度，但贡献却微乎其微.这句话一直记在我心中，发誓要为中国开源事业做我仅有的一点微薄贡献

另外写文档也是知识积累，还可以增加在圈内的影响力.

人跟动物的不同,就是人类可以把自己学习的经验教给下一代人.下一代在上一代的基础上再创新,不断积累才有今天.

所以我把自己的经验写出来,可以让经验传承

没有内容的章节:

目前我自己一人维护所有文档，写作时间有限，当我发现一个好主题就会加入到文档中，待我有时间再完善章节，所以你会发现很多章节是空无内容的.

文档目前几乎是流水帐试的写作，维护量很大，先将就着看吧.

我想到哪写到哪,你会发现文章没一个中心,今天这里写点,明天跳过本章写其它的.

文中例子绝对多,对喜欢复制然后粘贴朋友很有用,不用动手写,也省时间.

理论的东西,网上大把,我这里就不写了,需要可以去网上查.

我爱写错别字,还有一些是打错的,如果发现请指正.

文中大部分试验是在Debian/Ubuntu/Redhat AS上完成.

写给读者

至读者:

我不知道什么时候，我不再更新文档或者退出IT行业去从事其他工作，我必须给这些文档找一个归宿，让他能持续更新下去。

我想捐赠给某些基金会继续运转，或者建立一个团队维护它。

我用了20年时间坚持不停地写作，持续更新，才有今天你看到的《Netkiller 手扎》系列文档，在中国能坚持20年，同时没有任何收益的技术类文档，是非常不容易的。

有很多时候想放弃，看到外国读者的支持与国内社区的影响，我坚持了下来。

中国开源事业需要各位参与，不要成为局外人，不要让外国人说:中国对开源索取无度，贡献却微乎其微。

我们参与内核的开发还比较遥远，但是进个人能力，写一些文档还是可能的。

系列文档

下面是我多年积累下来的经验总结，整理成文档供大家参考:

Netkiller Architect 手札
Netkiller Developer 手札
Netkiller PHP 手札
Netkiller Python 手札
Netkiller Testing 手札
Netkiller Cryptography 手札
Netkiller Linux 手札
Netkiller FreeBSD 手札
Netkiller Shell 手札
Netkiller Security 手札
Netkiller Web 手札
Netkiller Monitoring 手札
Netkiller Storage 手札
Netkiller Mail 手札
Netkiller Docbook 手札
Netkiller Version 手札
Netkiller Database 手札
Netkiller PostgreSQL 手札
Netkiller MySQL 手札
Netkiller NoSQL 手札
Netkiller LDAP 手札
Netkiller Network 手札
Netkiller Cisco IOS 手札
Netkiller H3C 手札
Netkiller Multimedia 手札
Netkiller Management 手札
Netkiller Spring 手札

Netkiller Perl 手札

Netkiller Amateur Radio 手札

2. 作者简介

陈景峯 (ㄔㄣˊ ㄐㄧㄥˇ ㄈㄥ)

Nickname：netkiller | English name: Neo chen | Nippon name: ちんけいほう (音訳) | Korean name: 천징봉 | Thailand name: ภูมิภาพภูเขา | Vietnam: Trần Cảnh Phong

Callsign: BG7NYT | QTH: ZONE CQ24 ITU44 ShenZhen, China

程序猿，攻城狮，挨踢民工, Full Stack Developer, UNIX like Evangelist, 业余无线电爱好者（呼号：BG7NYT）,户外运动，山地骑行以及摄影爱好者。

《Netkiller 系列 手札》的作者

成长阶段

1981年1月19日(庚申年腊月十四)出生于黑龙江省青冈县建设乡双富大队第一小队

1989年9岁随父母迁居至黑龙江省伊春市，悲剧的天朝教育，不知道那门子归定，转学必须降一级，我本应该上一年级，但体制让我上学前班，那年多都10岁了

1995年小学毕业，体制规定借读要交3000两银子(我曾想过不升初中)，亲戚单位分楼告别平房，楼里没有地方放东西，把2麻袋书送给我，无意中发现一本电脑书BASIC语言，我竟然看懂了，对于电脑知识追求一发而不可收，后面顶零花钱，压岁钱主要用来买电脑书《MSDOS 6.22》《新编Unix实用大全》《跟我学Foxbase》。。。。。。

1996年第一次接触UNIX操作系统，BSD UNIX, Microsoft Xinux(盖茨亲自写的微软Unix，知道的人不多)

1997年自学Turbo C语言，苦于没有电脑，后来学校建了微机室才第一次使用QBASIC(DOS 6.22 自带命令)，那个年代只能通过软盘拷贝转播，Trubo C编译器始终没有搞到，

1997年第一次上Internet网速只有9600Bps, 当时全国兴起各种信息港域名格式是www.xxxx.info.net, 访问的第一个网站是NASA下载了很多火星探路者拍回的照片，还有“淞沪”sohu的前身

1998~2000年在哈尔滨学习计算机，充足的上机时间，但老师让我们练打字（明伦五笔/WT）打字不超过80个/每分钟还要强化训练，不过这个给我的键盘功夫打了好底。

1999年学校的电脑终于安装了光驱，在一张工具盘上终于找到了Turbo C, Borland C++与Quick Basic编译器，当时对VGA图形编程非常感兴趣，通过INT33中断控制鼠标，使用绘图函数模仿windows界面。还有操作 UCDOS 中文字库，绘制矢量与点阵字体。

2000年沉迷于Windows NT与Back Office各种技术，神马主域控制器，DHCP，WINS，IIS，域名服务器，Exchange邮件服务器，MS Proxy, NetMeeting...以及ASP+MS SQL开发；用56K猫下载了一张LINUX。ISO镜像，安装后我兴奋的24小时没有睡觉。

职业生涯

2001 年来深圳进城打工,成为一名外来务工者. 在一个4人公司做PHP开发，当时PHP的版本是2.0, 开始使用Linux Redhat 6.2.当时很多门户网站都是用FreeBSD,但很难搞到安装盘，在网易社区认识了一个网友,从广州给我寄了一张光盘，FreeBSD 3.2

2002 年我发现不能埋头苦干,还要学会"做人".后辗转广州工作了半年，考了一个Cisco CCNA认证。回到深圳重新开始，在车公庙找到一家工作做Java开发

2003 年这年最惨,公司拖欠工资16000元,打过两次官司2005才付清.

2004 年开始加入分布式计算团队,目前成绩，工作仍然是Java开发并且开始使用PostgreSQL数据库。

2004-10月开始玩户外和摄影

2005-6月成为中国无线电运动协会会员,呼号BG7NYT,进了一部Yaesu FT-60R手台。公司的需要转回PHP与MySQL，相隔几年发现PHP进步很大。在前台展现方面无人能敌，于是便前台使用PHP，后台采用Java开发。

2006 年单身生活了这么多年,终于找到归宿. 工作更多是研究PHP各种框架原理

2007 物价上涨,金融危机，休息了4个月（其实是找不到工作）,关外很难上439.460中继，搞了一台Yaesu FT-7800.

2008 终于找到英文学习方法， 《Netkiller Developer 手札》，《Netkiller Document 手札》

2008-8-8 08:08:08 结婚,后全家迁居湖南省常德市

2009《Netkiller Database 手札》,2009-6-13学车，年底拿到C1驾照

2010 对电子打击乐产生兴趣，计划学习爵士鼓。由于我对Linux热爱，我轻松的接管了公司的运维部，然后开发运维两把抓。我印象最深刻的是公司一次上架10个机柜，我们用买服务器纸箱的钱改善

伙食。我将40多台服务器安装BOINC做压力测试，获得了中国第二的名次。

2011 平凡的一年，户外运动停止，电台很少开，中继很少上，摄影主要是拍女儿与家人，年末买了一辆山地车

2012 对油笔画产生了兴趣，活动基本是骑行银湖山绿道，

2013 开始学习民谣吉他，同时对电吉他也极有兴趣；最终都放弃了。这一年深圳开始推数字中继2013-7-6日入手Motorola MOTOTRBO XIR P8668，Netkiller 系列手札从Sourceforge向Github迁移；年底对MYSQL UDF，Engine与PHP扩展开发产生很浓的兴趣，拾起遗忘10+年的C，写了几个mysql扩展（图片处理，fifo管道与ZeroMQ），10月份入Toyota Rezi 2.5V并写了一篇《攻城狮的苦逼选车经历》

2014-9-8 在淘宝上买了一架电钢琴 Casio Privia PX-5S pro 开始陪女儿学习钢琴，由于这家钢琴是合成器电钢，里面有打击乐，我有对键盘鼓产生了兴趣。

2014-10-2号罗浮山两日游，对中国道教文化与音乐产生了兴趣，10月5号用了半天时间学会了简谱。10月8号入Canon 5D Mark III + Canon Speedlite 600EX-RT香港过关被查。

2014-12-20号对乐谱制作产生兴趣（https://github.com/SheetMusic/Piano），给女儿做了几首钢琴伴奏曲，MuseScore制谱然后生成MIDI与WAV文件。

2015-09-01 晚饭后拿起爵士鼓基础教程尝试在Casio Privia PX-5S pro演练，经过反复琢磨加上之前学钢琴的乐理知识，终于在02号晚上，打出了简单的基本节奏，迈出了第一步。

2016 对弓箭（复合弓）产生兴趣，无奈天朝法律法规不让玩。每周游泳轻松1500米无压力，年底入 xbox one s 和 Yaesu FT-2DR, 同时开始关注功放音响这块

2017 7月9号入 Yamaha RX-V581 功放一台，连接Xbox打游戏爽翻了，入Kindle电子书，计划学习蝶泳，果断放弃运维和开发知识体系转攻区块链。

2018 从溪山美地搬到半岛城邦，丢弃了多年攒下的家底。11 月开始玩 MMDVM，使用 Yaesu FT-7800 发射，连接MMDVM中继板，树莓派，覆盖深圳湾，散步骑车通联两不误。

2019 卖了常德的房子，住了5次院，哮喘反复发作，决定停止电子书更新，兴趣转到知乎，B站

2020 准备找工作

职业生涯路上继续打怪升级

3. 如何获得文档

下载 **Netkiller** 手札 **(epub,kindle,chm,pdf)**

Netkiller 手札 2020版

Netkiller 手札 2019版

Netkiller 手札 2018版

Netkiller 手札 2017版

Netkiller 手札 2017版

通过 **GIT** 镜像整个网站

https://github.com/netkiller/netkiller.github.com.git

$ git clone https://github.com/netkiller/netkiller.github.com.git

镜像下载

整站下载

```
wget -m http://www.netkiller.cn/index.html
```

指定下载

```
wget -m wget -m http://www.netkiller.cn/linux/index.html
```

Yum 下载文档

获得光盘介质，RPM包，DEB包，如有特别需要，请联系我

YUM 在线安装电子书

http://netkiller.sourceforge.net/pub/repo/

```
# cat >> /etc/yum.repos.d/netkiller.repo <<EOF
[netkiller]
name=Netkiller Free Books
baseurl=http://netkiller.sourceforge.net/pub/repo/
enabled=1
gpgcheck=0
gpgkey=
EOF
```

查找包

```
# yum search netkiller

netkiller-centos.x86_64 : Netkiller centos Cookbook
netkiller-cryptography.x86_64 : Netkiller cryptography Cookbook
netkiller-docbook.x86_64 : Netkiller docbook Cookbook
netkiller-linux.x86_64 : Netkiller linux Cookbook
netkiller-mysql.x86_64 : Netkiller mysql Cookbook
netkiller-php.x86_64 : Netkiller php Cookbook
netkiller-postgresql.x86_64 : Netkiller postgresql Cookbook
netkiller-python.x86_64 : Netkiller python Cookbook
netkiller-version.x86_64 : Netkiller version Cookbook
```

安装包

```
yum install netkiller-docbook
```

4. 打赏（Donations）

If you like this documents, please make a donation to support the authors' efforts. Thank you!

您可以通过微信，支付宝，贝宝给作者打赏。

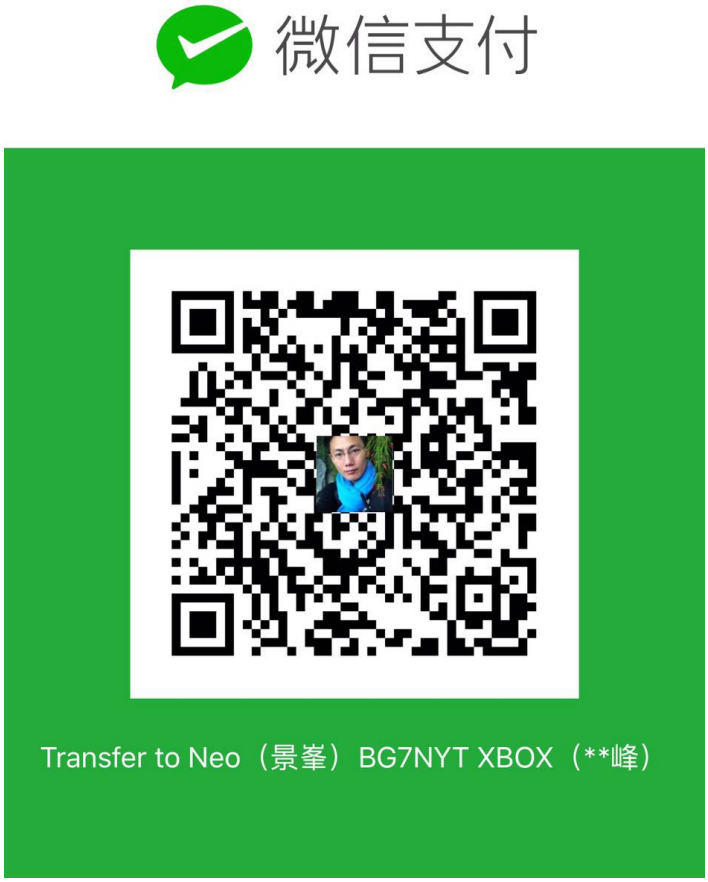
银行(Bank)

招商银行(China Merchants Bank)

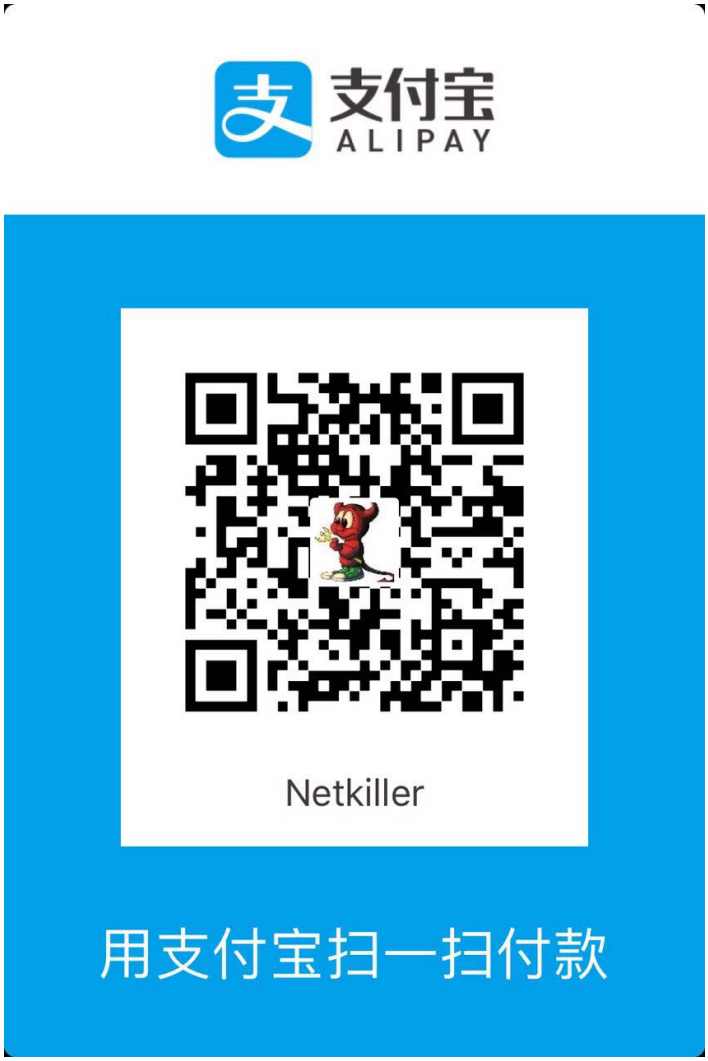
开户名：陈景峰

账号：9555500000007459

微信（Wechat）



支付宝（Alipay）



PayPal Donations

https://www.paypal.me/netkiller

5. 联系方式

主站 http://www.netkiller.cn/

备用 http://netkiller.github.io/

繁体网站 http://netkiller.sourceforge.net/

联系作者

Mobile: +86 13113668890

Email: netkiller@msn.com

QQ群: 128659835 请注明“读者”

QQ: 13721218

ICQ: 101888222

注：请不要问我安装问题！

博客 Blogger

知乎专栏 https://zhuanlan.zhihu.com/netkiller

LinkedIn: http://cn.linkedin.com/in/netkiller

OSChina: http://my.oschina.net/neochen/

Facebook: https://www.facebook.com/bg7nyt

Flickr: http://www.flickr.com/photos/bg7nyt/

Disqus: http://disqus.com/netkiller/

solidot: http://solidot.org/~netkiller/

SegmentFault: https://segmentfault.com/u/netkiller

Reddit: https://www.reddit.com/user/netkiller/

Digg: http://www.digg.com/netkiller

Twitter: http://twitter.com/bg7nyt

weibo: http://weibo.com/bg7nyt

Xbox club

我的 xbox 上的ID是 netkiller xbox，我创建了一个俱乐部 netkiller 欢迎加入。

Radio

CQ CQ CQ DE BG7NYT:

如果这篇文章对你有所帮助,请寄给我一张QSL卡片, qrz.cn or qrz.com or hamcall.net

Personal Amateur Radiostations of P.R.China

ZONE CQ24 ITU44 ShenZhen, China

Best Regards, VY 73! OP. BG7NYT

守听频率 DMR 438.460 -8 Color 12 Slot 2 Group 46001

守听频率 C4FM 439.360 -5 DN/VW

MMDVM Hotspot:

Callsign: BG7NYT QTH: Shenzhen, China

YSF: YSF80337 - CN China 1 - W24166/TG46001

DMR: BM_China_46001 - DMR Radio ID: 4600441

第 1 章 区块链

区块链可以说是2018年最火的技术，相信很多开发者已经跃跃欲试投入到区块链开发队伍当中来，可是又感觉无从下手，你会发现世面上的书籍大多是将理论纸上谈兵，都是一些无法落地的异想天开的想法，本书将用大量实例讲解如何让技术落地。

目前区块链技术无论是 Ethereum 还 Hyperledger 都处在高速发展阶段，每次版本迭代更新变化巨大，至少还需要一到三年才能变成成熟的技术。

我个人认为区块链的出现不是仅仅是简单意义上技术的革新，若干年后回过头来再看区块链，很可能是一个人类社会体系的变革伊始，是一个里程碑。

1. 什么是区块链?

很多书籍谈到区块链都从比特币开始，媒体也经常把比特币拉出来说事，首先要高清一个问题：比特币是区块链，但区块链并不是比特币。

区块链是什么？一句话，它是一种特殊的（非关系型）分布式数据库，这种数据库只能做插入和查找操作，并且没有管理员。

首先，区块链的主要作用是储存信息。任何需要保存的信息，都可以写入区块链，也可以从里面读取，所以它是数据库。

其次，任何人都可以架设服务器，加入区块链网络，成为一个节点。区块链的世界里面，没有中心节点，每个节点都是平等的，都保存着

整个数据库。你可以向任何一个节点，写入/读取数据，因为所有节点最后都会同步，保证区块链一致。

2. 什么是智能合约?

智能合约是区块数据业务逻辑的封装.

你可以理解为存储过程+数据库结构，这样应该很好理解了把?

访问只能合约就如同访问存储过程。在合约中定义的变量是不能直接访问的，只能通过函数操作他。

所以非常类似数据库定义了表结构，但是不能直接 select, insert, delete, update 数据，只能通过存储过程操作数据库一样。

3. 我们应该怎么做?

很简单，通过IP地址与端口号连接到区块链系统，通过API（通常是json-rpc）调用合约方法完成一笔交易，产生一笔区块记录。

理论上区块链比数据库简单。

4. 如何学习区块链

我学习区块链技术是没有看过任何书籍的，我采用的是碎片化学习方法，主要是通过搜索引擎和官方文档。我比较擅长自学，也很少和人交流。

也曾试图购买书籍，但是我发现这些书籍没有多大价值，几乎三分之二的内容在谈原理，理论的东西，剩下三分之一的内容，无非就是安装、配置、Helloword 实例。另外书籍的出版周期通常是半年至一年，等书籍出版出来，内容早已经过时，软件版本的差异导致书中的例子运行不了，所以我放弃了购买书籍的想法，同时萌生了自己要写一本以干货内容为主的电子书，尽量在书中回避理论的东西，软件版本我选择当前的主流版本，直接上例子，只要你对照步骤 Step by Step 实验就能成功，这种成就感会驱使你继续学习下去。

学习中遇到碰壁无法解决的问题可以借助搜索引擎解决，这是最好的学习工具。

我主张学习区块链不要看太多的原理，快速过一遍即可，很多书中从比特币开始讲起，我觉的是没有必要的。

学习区块链有两个方向，一个是代币开发，另一个才是区块链开发，现在媒体将两个方向混为一谈，这是两个独立的方向。研究前者的人称为币圈，研究后者的人称为链圈。

如果你想从事代币开发那么目标很明确以太坊是最佳选择，你的学习内容是代币合约开发，合约部署，web3合约操作，代币上交易所等等.....

业界所指区块链并非代币开发，而是解决去中心化，期望区块链技术帮助企业解决实际问题。通常Ethereum和Hyperledger两种方案都能满

足企业需求，你需要自己判断选择哪个方案。

区块链是一种工具，就如同手机是通讯工具，你不需要学习通信原理和计算机原理，一样可以使用手机。区块链的发展一定是趋向傻瓜化，越来越容易使用和开发。

5. 币圈与链圈

币圈是一帮专注炒币，甚至自己发币的人。他们专注于市场行情，追涨杀跌；而链圈多为技术人员，认为单凭自己的技术，便可对行业做出巨大颠覆。

6. 区块链能做什么

区块链具有去中心化安全性、可追溯、不可篡改等特性。

区块链目前的底层只适合做，低频高价值的业务。例如区块链+征信，区块链+资产，区块链+支付，区块链+供应链，房地产+区块链（登记，转账）

7. 区块链不能解决的问题

你能保证上链的数据绝对不会被篡改；但你不能保证，上传的数据是真的。

区块链不能解决的问题:

- 用户上传假数据
- 物品被调包
- 高频交易

我们举一个现实中的例子“身份证”，例如身份证是可能证明你是你，但是别人可以拿着你的身份证冒充你。另外你不能保证户籍人员在录入身份信息的时候不出错。现实中我们常有身份证重号或信息有误的情况。

并不是实施了区块链技术就安全无忧了，安全分为很多层，区块链只能做到存储层的安全。例如安全分为用户层，应用层，逻辑层，存储层等等。区块链无法解决用户层，应用层，逻辑层等安全问题，他只能保证存储在硬盘上的区块不被修改。

8. 理解去中心化

传统数据库是中心化的，它通过一个IP地址和一个端口号为应用程序提供服务，后来出现了“主从”和“主主”结构，去中心化就是一种“多主”结构。

与数据库相比区块链的去中心化更为复杂，他们的数据同步不是简单的二进制日志同步，而是通过加密传输，节点共识后才做数据存储。

9. 理解不可撰改

很多人被这句话误导，认为区块链的数据一旦创建是永久不能修改的，所以它安全。其实不然，区块链的数据可以修改，但不能撰改。

首先你要搞明白什么是撰改和修改，撰改是指非法修改区块链数据，而修改则是合法变更数据。

区块链上的数据是可以修改的，无论存储多久的数据，随时可以修改里面内容。

通常撰改区块链数据多指数据存储层面的修改。而修改则是通过 chaincode 提供的修改函数变更区块链里面的数据。

举例一个场景例子，在征信系统中，用户有时被拉入黑名单，但用户缴纳欠费后应该立即将其移到白名单中，这个过程就需要修改区块链上的数据。

另外我还告诉你，多数区块链平台没有用户认证权限管理模块。所以无法控制区块中的那些数据可能修改，那些不能修改，那些数据XXX用户可以修改等等。即使有些区块链平台具备权限控制，颗粒度也无法想目前的数据库那些细。

10. 理解分布式记账

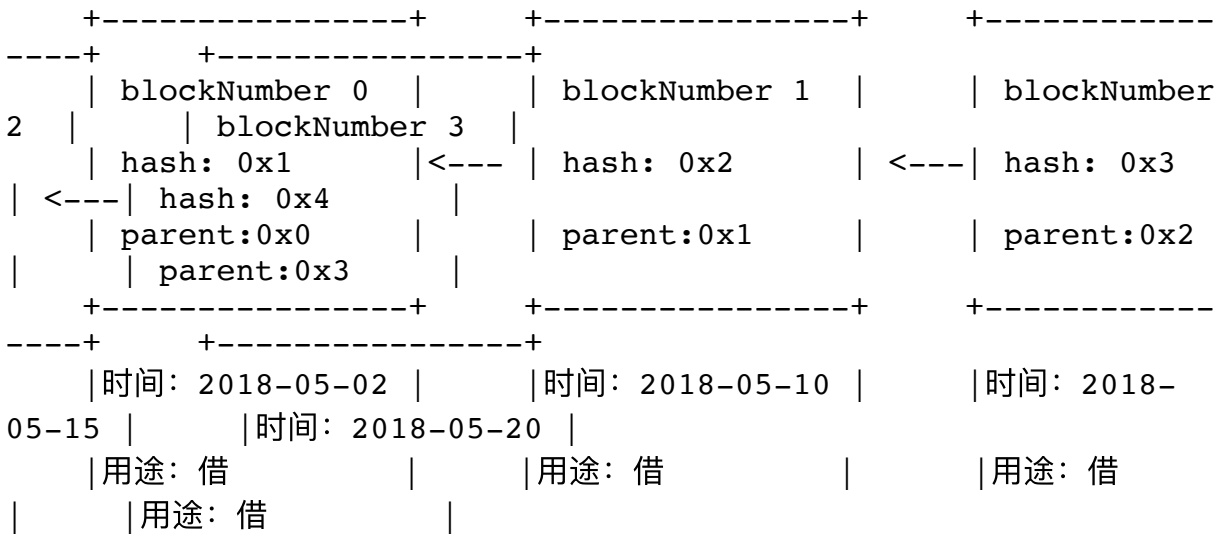
首先说明区块链中提到的账本与记账等等词汇是与会计无关的词汇。

我们传统理解的账本是指二位表格，记录某年某月产生的费用。

时间	用途	金额
2018-05-02	借	500
2018-05-10	还	500
2018-05-15	借	500
2018-05-20	借	500

如果账目比较多，可以拆账，将不同分类的账目，放到特定账本中。另外二位表格可以通过时间索引或者分类索引等等，快速找到一笔账目。

区块链是怎么记账的?



```
    |金额：500          |      |金额：500          |      |金额：500
|      |金额：500          |
    +-----------------+     +-----------------+     +------------
----+     +-----------------+
```

区块链可以理解为是传统账本的行列矩阵做这转换，每个事件收尾相连指向上一个区块地址形成链状，区块链不能通过分类拆分账本，所有账目全部在一个链条上。

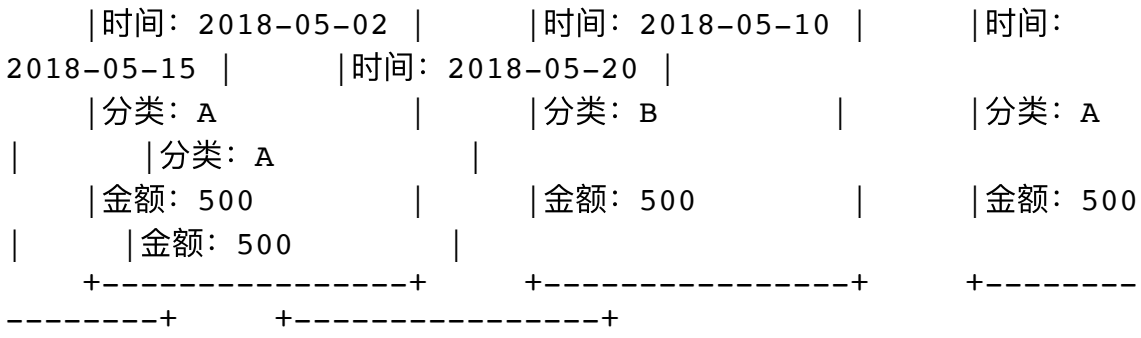
什么是分布式记账？上面链状的数据结构将保存在所有的区块链节点上，形成分布式集群，这就是分布式记账。

虽然区块链解决了分布式记账，但是也有很多弊端。我说过互联网上很多关于区块的文章都是臆想，纸上谈兵，他们根本没有实操经验。

下面我们讲讲区块链账本存在的问题

- 区块链不能键索引，无法快速搜索区块中的数据，必须依赖区块链以外的中心化技术，例如搜索引擎，数据库。例如 etherscan.io 就是将以太坊上的区块重新入库，借助数据库实现数据检索。
- 区块链只能顺序检索，中心化账本我们汇总账目只需做 sum 求和操作，而区块链必须从 blockNumber 0 开始一次向后读取，运算成本极高。
- 所有账目均在一个链上，不同分类混在一起，彼此相连。

```
    +-----------------+     +-----------------+     +--------
--------+     +-----------------+
    | blockNumber 0   | --> | blockNumber 1   | --> |
blockNumber 2   | --> | blockNumber 3   |  --> 时间轴
    +-----------------+     +-----------------+     +--------
--------+     +-----------------+
```



- 无法归档

  中心化数据库，可以归档一段时间内的数据，归档数据是冷数据，几乎不会再查询，这样一来中心数据库中的数据量减少，剩下的热数据处理起来非常快。我们有很多技术处理归档数据，将归档数据备份到存储介质上的解决方便有很多，也非常成熟。例如压缩，去重复等等，以减少存储成本开销。

  而区块链从诞生之日起到今日所有数据必须放在热数据区。任何新增节点都必须从区块0开始同步，并且保持每日同步到最新区块，否则将无法交易。区块一直在膨胀，随时区块链的普及，交易量猛增，总有一天将不堪重负。

  例如BTC(比特币) 安装钱包后，需要从92年的0区块开始同步，至少需要一周的时间，并且占用你硬盘203G的空间。

  ETH（以太坊）采用 fast 模式也需要200G的磁盘空间同步一周左右。就算采用最新的 light 模式，同步过程中经常出现中断，没有peers节点，断断续续，体验极度不好。

- 区块链没有事务处理

  因为区块链是首尾相连的，只能在尾部添加新区块，区块无法修改，所以区块链无法做事务处理。

```
Block 0 -> Block 1 -> Block 2 -> Block 3 -> Block 4
```

试想一下上面的 Block 2 回滚会怎样？如果 Block 2 回滚，Hash 值产生变化，后面所有区块都作废。所以区块链无法实现事务处理。

超级账本(Hyperledger Fabric)记不了帐

Hyperledger Fabric 中文名称叫超级账本，这个翻译坑害了无数人。 Hyperledger Fabric 跟账本没有任何关系。

实际工作中我使用 Hyperledger Fabric 实现了类似以太坊ERC20代币的功能，发行一个代币后将发行金额写入一个总账，然后从总账中项其他账号转账，用户消费后将金额从用户转会总账。

问题来了，因为超级账本没有事务处理，也无法串行执行每一笔操作，当并发执行的时候，账目出现混乱。

区块链无法将一组业务逻辑放到事物中执行。这样在实际的开发中我们只能依赖应用层，只能在应用层上实现事物锁的功能。由于区块存储在多个节点上，共识时间无法预计，不知道 stub.PutState（异步写入）执行完成的具体时间，无法达到100%无误。对于财务数据来不得半点马虎，我还是决定放弃这个功能，专为传统数据库。

所以超级账本记不了

- TPS:Transactions Per Second(每秒传输的事物处理个数)

1. 区块链是异步执行，你无法知道什么时候才能完成这笔交易，无法实现瞬间到账。

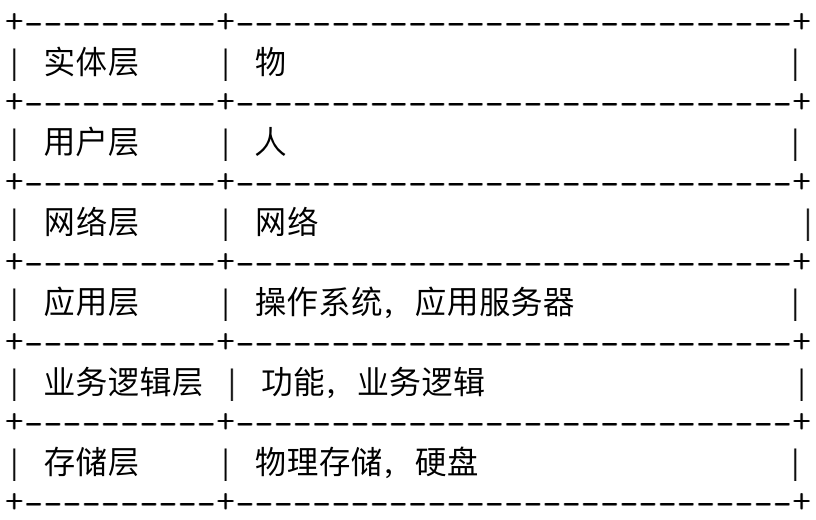
2. 交易阻塞

- 蛋疼的 gas 费用

总结：用区块链记账很蛋疼。

11. 安全问题

我将安全划分为六层，分别是：



并不是实施了区块链技术就安全无忧了，安全分为很多层，区块链只能做到网络层和存储层的安全。区块链无法解决用户层，应用层，逻辑层等安全问题。

网络层，因为区块链是通过公私钥体系加密数据库传输与存储，所以在网络上传输区块链数据是安全的。

存储层，区块链存储是加密的，链与链之间通过通过Hash算法连接，保证数据不被修改，修改会有什么后果呢？首先链就会断裂。修改一处数据，当前位置后面的数据全部需要更新，就算你更新成功，其他共识节点，也会视为你的更新是非法，所以数据撰改不可能。

12. 区块链落地面临的问题

12.1. 性能问题

区块链目前的底层只适合做，低频高价值的业务。

区块链的读取性能通常是没有问题的，但是区块链的写入实际上无论你用多少个服务器节点都不能提升，因为写入区块需要做共识算法，这步操作，会在所有节点上进行，同时还需要加密运算，这些操作都是 CPU 密集型操作。所以写入操作是存在瓶颈的。

解决这个问题，我想出了几种方案:

性能解决方案

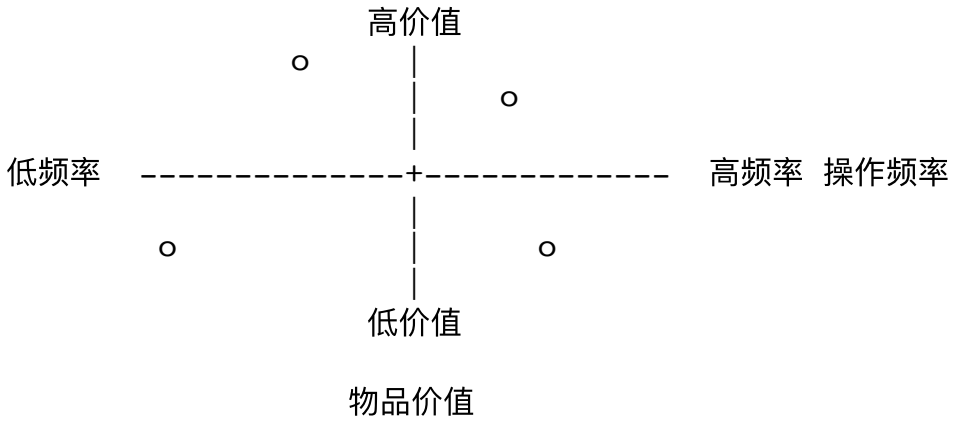
- 通过消息队列技术异步写入，将需要写入的区块放入队列，异步完成上链操作。
- 并行写入，我们可以建设多个区块链平台。多个平台同时服务于业务。

为了达到去中心化并行写入，我们将在客户端通过算法，匹配服务器。而不是在两个平台前面增加负载均衡。因为这样又回到了中心化系统。

12.2. 颗粒度问题

朔源的颗粒度问题，例如“红酒”的溯源，我们是将单位溯源做到箱呢？还是打，或是瓶呢？

我们用“四象限法则”分析



通过观察上面图，我们可以看到可以有四种情况，低频低价值，低频高价值，高频高价值，高频低价值

我认为对于低频高价值和高频高价值的业务，尽量做到最小颗粒度。

而对于低频低价值和高频低价值的业务，可以颗粒度更粗。

12.3. 区块链不能替代传统数据

回归技术本质，我认为区块链技术本身是一种追求分布一致性的数据库。

我们学过数据库的，都知道CAP理论。CAP理论是指的是在一个分布式系统中， Consistency（一致性）、 Availability（可用性）、Partition tolerance（分区容错性），三者不可得兼。大多数区块链，放弃了一些可用性，偏向了一致性和分区容错。

区块链并非能解决所有问题，虽然他也算是一种数据库，它能解决问题十分有限，它的数据管理和查询能力还打不到 NoSQL 的水平，更别提 SQL 的复杂应用。所以在实际的应用中，区块链不能替代数据，只能互补。

所以在项目实施前，仔细想想自己需求，真的需要区块链吗？还是需要区块链上的一些特性？例如数据不可撰改。如果仅仅是需要区块链的某一个特性。我们可以针对这个需求，思考一下能否使用传统数据库解决。

12.4. 链上，链下数据一致性问题

既然区块链替代不了传统数据库，那么必然要在项目中同时使用两种技术。这样问题来了，会有两份数据，一份存储在链下，即传统数据库，另外一部分数据上链，这样就有两份重复的数据，那么怎样保证他们的一致性呢？

非链上的原生资产在上链过程中，有一个重要的问题：原生资产的真实性问题，即链上资产、链下资产如何保持一致性问题。

区块链和比特币网络不同，比特币是在链上产生的，它与区块链密布可分，是一体的，所以它的数据安全性是自闭环的。而我们的链下数据并不是在区块链中产生的，将链下数据与区块链对接上链

我们需要考虑几个问题

1. 怎样能保证数据的真实和一致性呢？
2. 当出现不一致的时候以哪个为准呢？
3. 当需要读取数据时，是走连上，还是链下呢？
4. 什么数据上链，什么数据不上链？

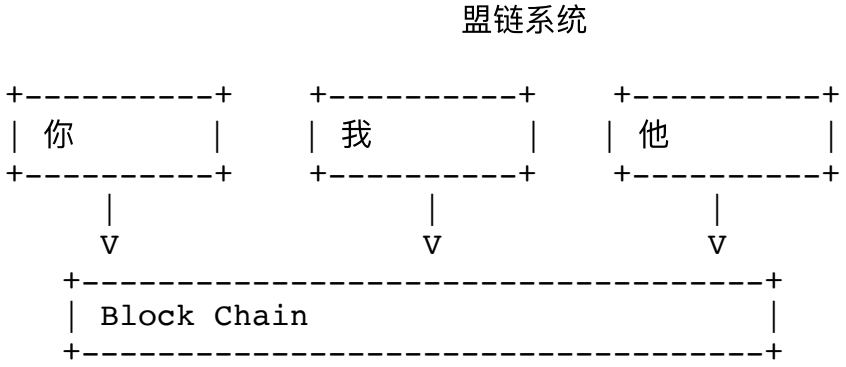
下面回答上面提出的问题

- 两端都将数据做一次hash，可以快速对比是否数据一致
- 我认为以连上数据为准较好，因为数据库数据更容易被撰改。

- 前台走上链，后台走数据库，前台是为用户提供服务的，所以要走连上数据，后台是管理的，可以直接走数据库，然后保证数据库与区块链数据一致。

- 共享数据上链，私有数据不上链，想象一下在盟链系统中，需要共享给其他成员数据，之前采用 Api接口，有了区块链更好的解决了资源共享问题。

例如下图我们将共享数据上链，在联盟中共享。私有数据（不能公开的数据）放到自己本地数据库，或者私链中。



13. 区块链未来

我认为区块链未来发展一定是易用，傻瓜化。

例如区块链的SQL网关，通过SQL访问区块链。

14. 区块链的六层模型

区块链技术的模型是由自下而上的数据层、网络层、共识层、激励层、合约层和应用层组成。

第一层“数据层”，封装了底层数据区块的链式结构，以及相关的非对称公私钥数据加密技术和时间戳等技术，这是整个区块链技术中最底层的数据结构。这些技术是构建全球金融系统的基础，数十年的使用证明了它非常安全的可靠性。而区块链，正式巧妙地把这些技术结合在了一起。

第二层“网络层”，包括P2P组网机制、数据传播机制和数据验证机制等。P2P组网技术早期应用在BT这类P2P下载软件中，这就意味着区块链具有自动组网功能。

第三层“共识层”，封装了网络节点的各类共识机制算法。共识机制算法是区块链的核心技术，因为这决定了到底是谁来进行记账，而记账决定方式将会影响整个系统的安全性和可靠性。目前已经出现了十余种共识机制算法，其中比较最为知名的有工作量证明机制（PoW，Proof of Work）、权益证明机制（PoS，Proof ofStake）、股份授权证明机制（DPoS，Delegated ProofofStake）等。数据层、网络层、共识层是构建区块链技术的必要元素，缺少任何一层都将不能称之为真正意义上的区块链技术。

第四层“激励层”，将经济因素集成到区块链技术体系中来，包括经济激励的发行机制和分配机制等，主要出现在公有链当中。在公有链中必须激励遵守规则参与记账的节点，并且惩罚不遵守规则的节点，才能让整个系统朝着良性循环的方向发展。而在私有链当中，则不一定需要进行激励，因为参与记账的节点往往是在链外完成了博弈，通过强制力或自愿来要求参与记账。

第五层“合约层”，封装各类脚本、算法和智能合约，是区块链可编程特性的基础。比特币本身就具有简单脚本的编写功能，而以太坊极大的强化了编程语言协议，理论上可以编写实现任何功能的应用。如果把比特币看成是全球账本的话，以太坊可以看作是一台“全球计算机”，任何人都可以上传和执行任意的应用程序，并且程序的有效执行能得到保证。

第六层“应用层”，封装了区块链的各种应用场景和案例，比如搭建在以太坊上的各类区块链应用即部署在应用层，而未来的可编程金融和可编程社会也将会是搭建在应用层。

15. 共识机制

共识机制，就是所有记账节点之间如何达成共识，去认定一个记录的有效性，这既是认定的手段，也是防止篡改的手段。目前主要有四大类共识机制：PoW、PoS、DPoS和分布式一致性算法。

15.1. PoW（Proofof Work，工作量证明）

PoW机制，也就是像比特币的挖矿机制，矿工通过把网络尚未记录的现有交易打包到一个区块，然后不断遍历尝试来寻找一个随机数，使得新区块加上随机数的哈希值满足一定的难度条件。找到满足条件的随机数，就相当于确定了区块链最新的一个区块，也相当于获得了区块链的本轮记账权。矿工把满足挖矿难度条件的区块在网络中广播出去，全网其他节点在验证该区块满足挖矿难度条件，同时区块里的交易数据符合协议规范后，将各自把该区块链接到自己版本的区块链上，从而在全网形成对当前网络状态的共识。

优点：完全去中心化，节点自由进出，避免了建立和维护中心化信用机构的成本。只要网络破坏者的算力不超过网络总算力的50%，网络的交易状态就能达成一致。

缺点：目前比特币挖矿造成大量的资源浪费；另外挖矿的激励机制也造成矿池算力的高度集中，背离了当初去中心化设计的初衷。更大的问题是PoW机制的共识达成的周期较长，每秒只能最多做7笔交易，不适合商业应用。

15.2. PoS（Proofof Stake，权益证明）

PoS机制，要求节点提供拥有一定数量的代币证明来获取竞争区块链记账权的一种分布式共识机制。如果单纯依靠代币余额来决定记账者必然使得富有者胜出，导致记账权的中心化，降低共识的公正性，因

此不同的PoS机制在权益证明的基础上，采用不同方式来增加记账权的随机性来避免中心化。例如点点币（Peer Coin）PoS机制中，拥有最多链龄长的比特币获得记账权的几率就越大。NXT和Blackcoin则采用一个公式来预测下一记账的节点。拥有多的代币被选为记账节点的概率就会大。未来以太坊也会从目前的PoW机制转换到PoS机制，从目前看到的资料看，以太坊的PoS机制将采用节点下赌注来赌下一个区块，赌中者有额外以太币奖，赌不中者会被扣以太币的方式来达成下一区块的共识。

优点：在一定程度上缩短了共识达成的时间，降低了PoW机制的资源浪费。

缺点：破坏者对网络攻击的成本低，网络的安全性有待验证。另外拥有代币数量大的节点获得记账权的几率更大，会使得网络的共识受少数富裕账户支配，从而失去公正性。

15.3. DPoS（DelegatedProof-Of-Stake，股份授权证明）

DPoS很容易理解，类似于现代企业董事会制度。比特股采用的DPoS机制是由持股者投票选出一定数量的见证人，每个见证人按序有两秒的权限时间生成区块，若见证人在给定的时间片不能生成区块，区块生成权限交给下一个时间片对应的见证人。持股人可以随时通过投票更换这些见证人。DPoS的这种设计使得区块的生成更为快速，也更加节能。从某种角度来说，DPOS可以理解为多中心系统，兼具去中心化和中心化优势。

优点：大幅缩小参与验证和记账节点的数量，可以达到秒级的共识验证。

缺点：选举固定数量的见证人作记账候选人有可能不适合于完全去中心化的场景。另外在网络节点数少的场景，选举的见证人的代表性也不强。

16. SHA-256

sha256 是一个摘要算法与常见的md5作用相同，因为区块链的数据量安全性md5,sha1都已经无法满足。实际上 sha512标准也已经出来，目前区块链主流采用 sha256，主要是考虑性能和存储空间。

目前区块链数据存储hash值占了 1/3 ~ 2/3

```
neo@netkiller ~ % sha256sum /etc/hosts
be0b4786b1169533329b2ab5292d8d1c16bbea5bd24c882a983ab4b754a398c
8  /etc/hosts

neo@netkiller ~ % sha1sum /etc/hosts
48bea81a95c75efa52608a9d384126be3131e40f  /etc/hosts

neo@netkiller ~ % sha224sum /etc/hosts
65bf519aa94d15c3e62406b8e1f8dd8b7fb513a5dac439606f954822
/etc/hosts

neo@netkiller ~ % sha384sum /etc/hosts
65478594d45d1f1054bea0f19b320607682a2c96c6efe39de9ab0a176a2623b
94728937346b1ede6b92cb36c013b0d93  /etc/hosts

neo@netkiller ~ % sha512sum /etc/hosts
275a4258fb570d897c67a65a65e73ac1e38cdb4166b1fdf290d7f6361892f81
d663cbab51a32cf827a4d4bc26edcd0a34bed36c170dc373058f3606c02c1c5
ae  /etc/hosts
```

17. Base58编码

Base64是常见的可读性编码算法，所谓Base64，即是说在编码过程中使用了64种字符：大写A到Z、小写a到z、数字0到9、“+”和“/”。

Base58是Bitcoin中使用的一种编码方式，主要用于产生Bitcoin的钱包地址。相比Base64，Base58不使用数字"0"，字母大写"O"，字母大写"I"，和字母小写"i"，以及"+"和"/"符号。

IPFS 也使用了 Base58

18. Merkle

而Merkle名字来源于其发明者Ralph Merkle，他在1979年获得了哈系树的专利。

19. BIP39协议:使用助记词生成确定性钱包

BIP:39
层：应用层
标题：使用助记词生成确定性钱包秘钥
作者：Marek Palatinus <slush@satoshilabs.com>
Pavol Rusnak <stick@satoshilabs.com>
Aaron Voisine <voisine@gmail.com>
Sean Bowe <ewillbefull@gmail.com>
状态：已经被提议
类型：标准化跟踪
创建日期：2013-09-10
译者：kimziv

19.1. 摘要

这个BIP描述了使用助记码或者助记句子（简称助记词）--一组便于记忆的单词来生成确定性钱包。

这个BIP由两部分构成：生成助记词和把生成的助记词转化成一个二进制种子。这个种子后面会更急类似于BIP32的方法生成确定性钱包。

19.2. 动机

与处理原始的二进制或者十六进制的钱包种子相比，在人机交互过程中助记词是更胜一筹的。这些助记单词可以被写在纸上或者通过电话说出来。

本指南旨在通过人类可读的转录来传输计算机生成的随机性。并不是将用户创建的句子（也称为脑钱包）处理到钱包种子中的方法。

19.3. 生成助记词

助记符必须以32位的倍数编码熵。随着熵的安全性提高，同时句子的长度也在增加。我们将初始熵长度称为ENT。ENT允许的大小为128-256位。

首先，生成ENT位的初始熵。通过取第一个生成的校验和

ENT/32

它的SHA256哈希的位。该校验和附加到初始熵的末尾。接下来，这些连接的比特位被分成多个11位的组，每个组用从0-2047的数字编码，用作单词表的索引。最后，我们将这些数字转换为单词，并将加入的所有单词组成助记句。

下表描述了初始熵长度（ENT），校验和长度（CS）和生成助记词（MS）的长度之间的关系。

CS = ENT / 32
MS = (ENT + CS) / 11

ENT	CS	ENT + CS	MS
128	4	132	12
160	5	165	15
192	6	198	18
224	7	231	21
256	8	264	24

19.4. 单词表

理想的单词列表具有以下特点:

- 智能选词

  单词列表以这种方式创建：输入前四个字母来就足以明确地标识这个单词;

- 避免相似的单词

  "build" and "built", "woman" and "women", or "quick" and "quickly" 这样的词对，不仅使记忆困难，而且更容易出错，更难猜到;

- 排序的单词列表

  排序的单词列表允许更有效地查找代码字（即，实现可以使用二分搜索而不是线性搜索）

这也允许使用字典树（前缀树），例如用于更好的压缩

单词表可以包含本土字符，但必须使用规范化形式兼容性分解（NFKD）以UTF-8编码。

19.5. 从助记词到种子

用户可以决定用密码保护他们的助词。如果密码不存在，则使用空字符串“”代替。

要通过助记词创建一个二进制种子，我们使用助记符作为密码（UTF-8 NFKD）和字符串“mnemonic”+ passphrase 作为盐（再次以UTF-8 NFKD）来调用PBKDF2函数。迭代计数设置为2048，HMAC-SHA512用作伪随机函数。派生密钥的长度为512位（= 64字节）。

该种子可以随后用于使用BIP-0032或类似方法产生确定性钱包。

助记词转换为二进制种子完全独立于生成这个助记词。这导致相当简单的代码; 助记词结构没有约束，客户可以自由地实现自己的单词列表，甚至是整个助记词的生成器，允许字典列表中的输入错误检测或其他用途的灵活性。

虽然使用的助记词可能不是通过“生成助记词”部分中描述的算法生成的，但这是不建议的，软件必须使用单词表计算助记词的校验和，如果无效则发出警告。

所描述的方法还提供似乎可信的可否认性，因为每个密码短语产生一个有效的种子（因此产生确定性钱包），但是只有正确的那一个才能使所需的钱包可用。

19.6. 单词列表

https://github.com/bitcoin/bips/blob/master/bip-0039/bip-0039-wordlists.md

如果一个 HD 钱包助记词是 12 个单词，一共有 2048 个单词可能性，如何算出随机的生成的助记词可能性是一个排列问题，根据公式： n!/(n - r)! ，既 2048!/(2048-12)! = 5.271537971301488476000309317528 2e+39。

Wordlists

1. English https://github.com/bitcoin/bips/blob/master/bip-0039/english.txt
2. Japanese https://github.com/bitcoin/bips/blob/master/bip-0039/japanese.txt
3. Korean https://github.com/bitcoin/bips/blob/master/bip-0039/korean.txt
4. Spanish https://github.com/bitcoin/bips/blob/master/bip-0039/spanish.txt

5. Chinese (Simplified) https://github.com/bitcoin/bips/blob/master/bip-0039/chinese_simplified.txt
6. Chinese (Traditional) https://github.com/bitcoin/bips/blob/master/bip-0039/chinese_traditional.txt
7. French https://github.com/bitcoin/bips/blob/master/bip-0039/french.txt
8. Italian https://github.com/bitcoin/bips/blob/master/bip-0039/italian.txt

19.7. 开发库

19.7.1. Node.js

https://www.npmjs.com/package/bip39

19.7.2. Python

https://github.com/trezor/python-mnemonic

19.7.3. 其他实现

```
Elixir: https://github.com/izelnakri/mnemonic

Objective-C: https://github.com/nybex/NYMnemonic

Haskell: https://github.com/haskoin/haskoin

.NET C# (PCL): https://github.com/Thashiznets/BIP39.NET

.NET C# (PCL): https://github.com/NicolasDorier/NBitcoin

JavaScript: https://github.com/bitpay/bitcore-mnemonic,
https://github.com/bitcoinjs/bip39 (used by blockchain.info)

Ruby: https://github.com/sreekanthgs/bip_mnemonic

Rust: https://github.com/infincia/bip39-rs

Swift: https://github.com/CikeQiu/CKMnemonic

C++:
https://github.com/libbitcoin/libbitcoin/blob/master/include/bitcoin/bitcoin/wal
let/mnemonic.hpp

C (with Python/Java/Javascript bindings):
https://github.com/ElementsProject/libwally-core
```

19.8. Netkiller 助记词词库

HD Wallet 采用 2048 个单词，或者汉字作为助记词，这些词库对外公开，很多钱包仅仅使用path第一个地址并且没有加密。如果你知道某个用户的助记词中的11各词的排列顺序，那么我们就可以通过穷举方法，算出所有地址的私钥，如果碰巧找到了已经在使用的地址。就可以将里面的ETH全部转走。

为了增加 HD Wallet 的安全，我做了一个词库，这个词库不对外公开，并且使用的汉字均是不常用汉字。只能复制粘贴，几乎很难使用输入法输入该汉字。

同时path 还做了分层，和索引地址。分层采用时间维度，索引采用随机数， Seed 做加密处理。助记词共 15 个汉字。

效果如下

```
汉字助记词: 欈 戀 髂 響 麠 戳 邂 蠢 鶖 躒 蠈 鲮 警 趱 巆
地址: 0x4949225eab0121d1e0b0eeb286a12b04ff596471
私钥: b5ce4ac958fbdcd385d6ae850c1870c6da7b990981363c25036a31ba06be6636

汉字助记词: 瓢 彎 曤 纀 鸇 鸄 鶹 纚 鱱 謫 艣 鼴 黭 纆 贛
地址: 0x430097d16819108068a7af22a116285e54bc3e6b
私钥: 3b78431a43a2c69e861870f0eff1d54d3965247ca5e588a9f907904f9ea5b822

汉字助记词: 聾 鱦 驥 鬘 鰕 蘲 韃 鏻 礭 鑀 瓤 蕮 壢 躠 罋
地址: 0x641fd58728cf08bc8795d41cfd3885a4f1c8dced
私钥: b7c2ff2a39e3a534e6e89288b05b4a283b10b34b2dfca2b336676729c7a68ad1

汉字助记词: 轕 鶼 瀝 钁 麝 鸑 瀧 纞 躐 嘯 樧 鱁 贛 癖 躤
地址: 0xae6ad7cf3e31556bc7262e25fba2ebad9954d08b
私钥: c989fe5c108b0bd33e5731919e09c30c639a4ff29fb4e66fe3052975855181f6

汉字助记词: 鱉 鋼 鰊 彏 鱞 鷮 鼯 瞪 晨 鬙 衢 斀 櫜 鸗 襫
地址: 0x7fc6bca55c51ab3b4266d8f67d63c196ac874d93
私钥: 53e755075653a64867f885e702ca0a2612bdd13ec2bed0df647bf568a639bc46
```

20. Ethereum vs Hypterledger Fabic vs EOS 对比

表 1.1. 智能合约对比

	Ethereum	Hypterledger Fabic	EOS
合约语言	Solidity	Go/Node.js/Java	C/C++
难易程度	容易	较容易	难
成熟度	非常成熟	较成熟	不成熟
调试环境	有Remix,Truffle，钱包合约工具等等	没有较好的调试环境	只能手工排查
扩展性	Solidity 无法扩展，不能引用外部开发库	可以扩展，导入第三方库	可以扩展，并且支持STL开发库

第 2 章 区块链探索

1. 以太坊物流场景解决方案

网上谈关于物流行业区块链的文章很多，但是你会发现找遍互联网也找不到具体怎样将物流落地到区块链的文章，于是我只能自己捣鼓。

背景，使用区块链记录物流信息，实现信息朔源。

我想法是，将物流信息放到区块链中，实现物流中转信息的添加，当用户签收后合约关闭，不再允许增加新信息。

首先，每个物流单一张合约

其次，以太坊账号代表转运站，或者用户，这里我们使用5个账号分别代表不同的角色。

```
pragma solidity ^0.4.20;

contract Logistics {

    enum State { New, Reviewed, Pending, Shipping, Received }

    struct Node {
        address owner;  // 中转站
        string date;    // 转运日期
        State status;   // 状态
        string message; // 留言信息
    }

    mapping (uint  => Node) stations;

    uint number = 1;
    string name;                          //商品名称
    bool close = false; //合约状态

        function Logistics(string _name) public {
        name = _name;
        }
    function getName() public view returns(string){
        return name;
    }

    // 增加物流中转信息
    function put(address _owner,string _date, State _status, string _message ) public{
        if(close == false){
            Node memory node = Node(_owner,_date,_status,_message);
            stations[number] = node;
            number = number + 1;
        }
        if (_status == State.Received) {
            close = true;
        }
    }

        // 获得中转信息
    function get(uint _number) public view returns(address, string, State, string) {
        require(_number < number);
```

```
        Node memory node = stations[_number];

                return (node.owner, node.date, node.status, node.message);
        }

        // 或者转中站数量
        function getNode() public view returns(uint){
            return number;
        }
}
```

保存合约到 Truffle 的 contracts/Logistics.sol

部署代码

```
neo@MacBook-Pro ~/ethereum/truffle % cat migrations/1_initial_migration.js

var Logistics = artifacts.require("./Logistics.sol");

module.exports = function(deployer) {
  deployer.deploy(Logistics,"Mackbook");
};
```

Mackbook 就是商品名称。

编译部署合约

```
neo@MacBook-Pro ~/ethereum/truffle % truffle compile --all
Compiling ./contracts/Logistics.sol...
Writing artifacts to ./build/contracts

neo@MacBook-Pro ~/ethereum/truffle % truffle migrate --reset
Using network 'development'.

Running migration: 1_initial_migration.js
  Replacing Logistics...
  ... 0x14b6b6bfb84383b8325f5e97a6b7a5c1d1f5c2e162a4bd201b93a9d30cd75d8e
  Logistics: 0x1cff61b8259f05f4bbf7aa4f769321e5fa70b22d
Saving successful migration to network...
  ... 0x26d544c8db7b1cf06034963e5f5bea7b28d11e7295a018f1b80a7555c38f26e7
Saving artifacts...
```

启动开发环境

```
neo@MacBook-Pro ~/ethereum/truffle % truffle develop
Truffle Develop started at http://localhost:9545/

Accounts:
(0) 0x627306090abab3a6e1400e9345bc60c78a8bef57
(1) 0xf17f52151ebef6c7334fad080c5704d77216b732
(2) 0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef
(3) 0x821aea9a577a9b44299b9c15c88cf3087f3b5544
(4) 0x0d1d4e623d10f9fba5db95830f7d3839406c6af2
(5) 0x2932b7a2355d6fecc4b5c0b6bd44cc31df247a2e
(6) 0x2191ef87e392377ec08e7c08eb105ef5448eced5
```

```
(7) 0x0f4f2ac550a1b4e2280d04c21cea7ebd822934b5
(8) 0x6330a553fc93768f612722bb8c2ec78ac90b3bbc
(9) 0x5aeda56215b167893e80b4fe645ba6d5bab767de

Private Keys:
(0) c87509a1c067bbde78beb793e6fa76530b6382a4c0241e5e4a9ec0a0f44dc0d3
(1) ae6ae8e5ccbfb04590405997ee2d52d2b330726137b875053c36d94e974d162f
(2) 0dbbe8e4ae425a6d2687f1a7e3ba17bc98c673636790f1b8ad91193c05875ef1
(3) c88b703fb08cbea894b6aeff5a544fb92e78a18e19814cd85da83b71f772aa6c
(4) 388c684f0ba1ef5017716adb5d21a053ea8e90277d0868337519f97bede61418
(5) 659cbb0e2411a44db63778987b1e22153c086a95eb6b18bdf89de078917abc63
(6) 82d052c865f5763aad42add438569276c00d3d88a2d062d36b2bae914d58b8c8
(7) aa3680d5d48a8283413f7a108367c7299ca73f553735860a87b08f39395618b7
(8) 0f62d96d6675f32685bbdb8ac13cda7c23436f63efbb9d07700d8669ff12b7c4
(9) 8d5366123cb560bb606379f90a0bfd4769eecc0557f1b362dcae9012b548b1e5

Mnemonic: candy maple cake sugar pudding cream honey rich smooth crumble sweet treat

truffle(develop)>
```

开发环境会创建10个账号用户测试。我们需要使用前5个账号，每个账号代表一个转运站，或者用户

进入控制台验证合约

```
var contract;
Logistics.deployed().then(function(instance){contract=instance;});
contract.getName();

contract.put("0x627306090abab3a6e1400e9345bc60c78a8bef57","2018-02-20",0,"寄包裹");
contract.get(1);
contract.put("0xf17f52151ebef6c7334fad080c5704d77216b732","2018-02-21",1,"包裹揽件");
contract.get(2);
contract.put("0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef","2018-02-22",2,"运输处理中");
contract.get(3);
contract.put("0x821aea9a577a9b44299b9c15c88cf3087f3b5544","2018-02-23",3,"运输处理中");
contract.get(4);
contract.put("0x0d1d4e623d10f9fba5db95830f7d3839406c6af2","2018-02-24",4,"包裹收到");
contract.get(5);
contract.getNode();


contract.put("0x0d1d4e623d10f9fba5db95830f7d3839406c6af2","2018-02-22",5,"已经收到包裹，合约关闭，不
允许在修改");
contract.get(6);
```

操作演示如下

```
truffle(development)> var contract;
undefined
truffle(development)> Logistics.deployed().then(function(instance){contract=instance;});
undefined
truffle(development)> contract.getName();
'Mackbook'
truffle(development)> contract.put("0x627306090abab3a6e1400e9345bc60c78a8bef57","2018-02-
20",0,"寄包裹");
{ tx: '0x74992b7cccb214600ac2f1257486053202736714cf7e9e69fb62cba692bc6592',
  receipt:
   { transactionHash: '0x74992b7cccb214600ac2f1257486053202736714cf7e9e69fb62cba692bc6592',
     transactionIndex: 0,
     blockHash: '0xc838fb9c5352544f4d743b170d146a9ef1b1ef6a30019c33e2a77df24e808964',
```

```
     blockNumber: 86,
     gasUsed: 98633,
     cumulativeGasUsed: 98633,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.get(1);
[ '0x627306090abab3a6e1400e9345bc60c78a8bef57',
  '2018-02-20',
  BigNumber { s: 1, e: 0, c: [ 0 ] },
  '寄包裹' ]
truffle(development)> contract.put("0xf17f52151ebef6c7334fad080c5704d77216b732","2018-02-
21",1,"包裹揽件");
{ tx: '0x3f8dcd5f0d9a9ec60942e6a1c73556dfcfde59354fc24474ffc8e32b9b00ac61',
  receipt:
   { transactionHash: '0x3f8dcd5f0d9a9ec60942e6a1c73556dfcfde59354fc24474ffc8e32b9b00ac61',
     transactionIndex: 0,
     blockHash: '0x96c889cae1001265bcdf32c808770a7f9f0c325467912524c10100bc04cf8271',
     blockNumber: 87,
     gasUsed: 113889,
     cumulativeGasUsed: 113889,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.get(2);
[ '0xf17f52151ebef6c7334fad080c5704d77216b732',
  '2018-02-21',
  BigNumber { s: 1, e: 0, c: [ 1 ] },
  '包裹揽件' ]
truffle(development)> contract.put("0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef","2018-02-
22",2,"运输处理中");
{ tx: '0x1ebe589e6b63479f9542ba67650d63757ca45ac38cb43d395b5bc2a573d0363b',
  receipt:
   { transactionHash: '0x1ebe589e6b63479f9542ba67650d63757ca45ac38cb43d395b5bc2a573d0363b',
     transactionIndex: 0,
     blockHash: '0x83edf5fc1e38062dafc49a21b3d9a1fa0f9ddfb0f2e749b2b1945d03360a5209',
     blockNumber: 88,
     gasUsed: 114081,
     cumulativeGasUsed: 114081,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.get(3);
[ '0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef',
  '2018-02-22',
  BigNumber { s: 1, e: 0, c: [ 2 ] },
  '运输处理中' ]
truffle(development)> contract.put("0x821aea9a577a9b44299b9c15c88cf3087f3b5544","2018-02-
22",3,"运输处理中");
{ tx: '0x44b2bf7853e6b4c86f732bb8f1bcee17f00e0f850530e359753b4d7c55c35b4d',
  receipt:
   { transactionHash: '0x44b2bf7853e6b4c86f732bb8f1bcee17f00e0f850530e359753b4d7c55c35b4d',
     transactionIndex: 0,
     blockHash: '0x7e79ca2570f5045f4c226805866803f898109d238518fa1e5abe6b4ee4c1c552',
     blockNumber: 89,
     gasUsed: 114081,
     cumulativeGasUsed: 114081,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.get(4);
[ '0x821aea9a577a9b44299b9c15c88cf3087f3b5544',
  '2018-02-22',
  BigNumber { s: 1, e: 0, c: [ 3 ] },
  '运输处理中' ]
truffle(development)> contract.put("0x0d1d4e623d10f9fba5db95830f7d3839406c6af2","2018-02-
```

```
22",4,"包裹收到");
{ tx: '0xb2b0223dc7cc90744a97ea002ecd468796d7596e38f8bb105c9f2103da6dfa19',
  receipt:
   { transactionHash: '0xb2b0223dc7cc90744a97ea002ecd468796d7596e38f8bb105c9f2103da6dfa19',
     transactionIndex: 0,
     blockHash: '0xeb1051e80fe920fc166288036e6d27b38aca27144d2b636decade338f787371b',
     blockNumber: 90,
     gasUsed: 134156,
     cumulativeGasUsed: 134156,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.get(5);
[ '0x0d1d4e623d10f9fba5db95830f7d3839406c6af2',
  '2018-02-22',
  BigNumber { s: 1, e: 0, c: [ 4 ] },
  '包裹收到' ]
truffle(development)> contract.getNode();
BigNumber { s: 1, e: 0, c: [ 6 ] }
truffle(development)>
```

合一已经关闭，添加不会出错，但是没有数据进入区块中，使用 contract.get(6); 获取数据会抛出异常。

```
truffle(development)> contract.put("0x0d1d4e623d10f9fba5db95830f7d3839406c6af2","2018-02-
22",3,"已经收到包裹，合约关闭，不允许在修改");
{ tx: '0x72999fc308f2f3bc1f70fbc919c8b08594f177318dc4e57dd5ea590248e9a6cc',
  receipt:
   { transactionHash: '0x72999fc308f2f3bc1f70fbc919c8b08594f177318dc4e57dd5ea590248e9a6cc',
     transactionIndex: 0,
     blockHash: '0xa3d9bc835bd5de6067271baa7899c3aaada6088362371b5139f4fa7cbd9f4050',
     blockNumber: 91,
     gasUsed: 29360,
     cumulativeGasUsed: 29360,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.get(6);
Error: VM Exception while processing transaction: revert
    at XMLHttpRequest._onHttpResponseEnd
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js:509:1)
    at XMLHttpRequest._setReadyState
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js:354:1)
    at XMLHttpRequestEventTarget.dispatchEvent
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js:64:1)
    at XMLHttpRequest.request.onreadystatechange
(/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/httpprovider.js:128:1)
    at /usr/local/lib/node_modules/truffle/build/webpack:/~/truffle-provider/wrapper.js:134:1
    at /usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/requestmanager.js:86:1
    at Object.InvalidResponse
(/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/errors.js:38:1)
truffle(development)>
```

这个合约还不是很完善，仅仅是作者的想法，是否在实际项目中可行，尚未知，区块链应用场景实例的文章还比较少，只能摸索前进。

2. 区块链防伪溯源应用场景

食品安全溯源

下面的方案，同样适合药品安全溯源

2.1. 背景

需求是通过区块链跟踪产品，实现产品产地，生产，流通等环节溯源。

需求归纳，需要实现下面几点：

产品具备通用的属性，例如名称，价格，重量，颜色，体积等等

生产销售链条跟踪

涉及环节，农产品的供应链是一个非常复杂的过程，涉及多方，农业局、卫生局、药监局、工商局、环保局等多个部门交织其中。

参与者角色，我们为每个环节的参与者分配一个以太坊账号，例如每个供应商一个账号，每个代理商一个账号。这样任何一方经手后都会使用自己的账号想合约中添加数据。

2.2. 如何实现

首先，要理解什么是物品，什么是数据。

其次，要明白物品跟数据的关系，物品怎么跟数据建立关系。

食品是物品，物品与数据并无关联，所谓溯源是指一连串的数据。数据是可以伪造的，我们可以伪造一连串溯源数据，然后上链。这些数

据都是安全正常供应链数据伪造到。所以数据只是数据，与物品没有任何关系。

怎样把物品和数据建立关系呢？我们通常会借助中间载体。例如

1. 贴二维码，RFID标签，NFC标签 。或者通过激光打码到物品上
2. 装箱，装瓶，装盒子
3. 提取生物特征信息

贴二维码，RFID和NFC标签是常规做法，将物品上贴上标签，标签带有唯一标示，标签与区块链上的数据建立关联。这样当读取标签的时候，就能调出区块链上的数据。

物品装进盒子，盒子上有唯一标示，盒子与区块链数据建立关联。

生物特征是指，例如人与数据建立连接，可以采集面部数据，眼部虹膜数据，指纹，牙齿，甚至DNA..... 然后用这些特征与数据建立关系。食品溯源通常使用贴标签和装盒子的方法，不会使用生物识别技术。

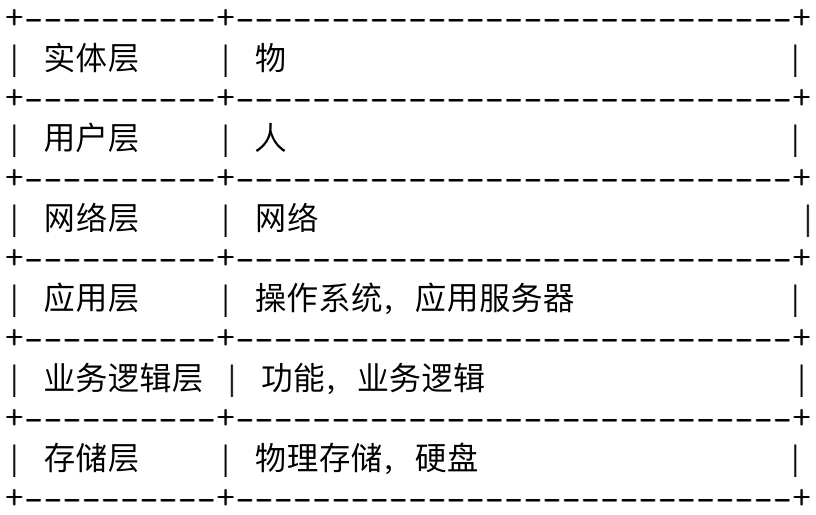
仍然存在的问题:

首先说说贴标签，这种方式并不能保证100%真实，揭开标签，转帖到其他商品上，理论上那个产品就是真实的。虽然有易碎纸技术，道高一尺魔高一丈，作假者有各种手段揭开标签。RFID/NFC理论上不可造假，实际上是可以了，只是作假成本的问题，是否值得。

在说说装箱，装瓶防伪以茅台为代表，虽然厂家可以使用一次性瓶子，盒子，箱子，制假者还是前面说的那句道高一尺魔高一丈。它们在茅台瓶子上打孔，采用真瓶装假酒的手法作假。

2.3. 安全问题

我将安全划分为六层，分别是：



并不是实施了区块链技术就安全无忧了，安全分为很多层，区块链只能做到网络层和存储层的安全。区块链无法解决用户层，应用层，逻辑层等安全问题，他只能保证存储在硬盘上的区块不被修改。

因为区块链仅仅能解决数据存储层的安全问题，不能保证上链的数据是真实的，上链前绝对不会被篡改；所以仅仅朔源，不考虑防伪是没有意义的，防伪仍然是重中之重。

2.4. 防伪问题

如何做防伪呢，这个领域很多公司已经探索多年，各种高科技应用，武装到牙齿，但仍没有解决假货问题。

区块链的出现很可能是一个突破，我们只需将现有成熟的防伪技术与区块链结合即可。

现在流行的访问技术太多了，我倾向于采用二维码，RFID，NFC技术，二维码与互联网紧密相连。

2.5. 性能问题

区块链目前的底层只适合做，低频高价值的业务。

区块链的读取性能通常是没有问题的，但是区块链的写入实际上无论你用多少个服务器节点都不能提升，因为写入区块需要做共识算法，这步操作，会在所有节点上进行，同时还需要加密运算，这些操作都是 CPU 密集型操作。所以写入操作是存在瓶颈的。

解决这个问题，我想出了几种方案:

性能解决方案

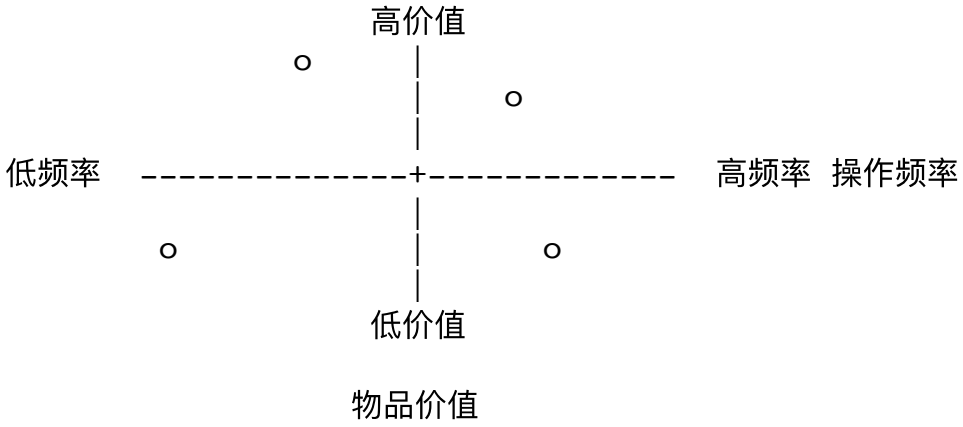
- 通过消息队列技术异步写入，将需要写入的区块放入队列，异步完成上链操作。
- 并行写入，我们可以建设多个区块链平台。多个平台同时服务于业务。

为了达到去中心化并行写入，我们将在客户端通过算法，匹配服务器。而不是在两个平台前面增加负载均衡。因为这样又回到了中心化系统。

2.6. 颗粒度问题

朔源的颗粒度问题，例如“红酒”的溯源，我们是将单位溯源做到箱呢？还是打，或是瓶呢？

我们用“四象限法则”分析



通过观察上面图，我们可以看到可以有四种情况，低频低价值，低频高价值，高频高价值，高频低价值

我认为对于低频高价值和高频高价值的业务，尽量做到最小颗粒度。

而对于低频低价值和高频低价值的业务，可以颗粒度更粗。

2.7. 存储规划

如果是高频低价值的业务，那么溯源数据源源将会不断的被添加到区块，以此同时区块的访问率极低。迟早会达到一个临界值。

所以你要规划存储，例如溯源数据的过期时间，对于 hyperledger 可以使用 DelState(key) 删除历史数据。

如果是高频高价值的业务是否要考虑永久保留数据呢?

这些问题都是需要考虑的。因为目前我们还不知道区块链的存储临界值。

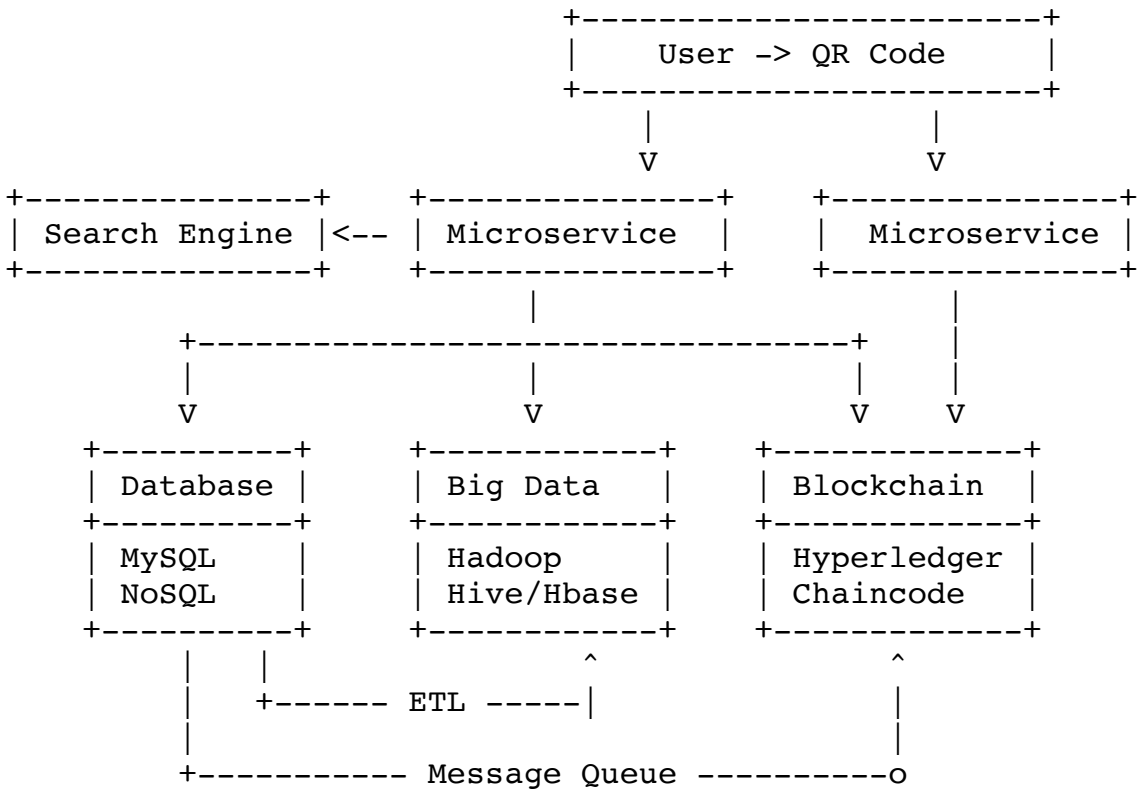
2.8. 大数据问题

区块链替代不了数据库，它与数据库是互补关系。

对于低频的业务，通常传统数据库足以应付。那么对于高频操作的业务呢？暂时可能没有问题，但总有一天会遇到瓶颈。

综上所述，溯源项目数据库规划决不能少。同时还要考虑数据仓库和后期数据挖掘。因为用户使用微信或者我们的APP扫描二维码，我们可以获得很多有价值的数据。

手上没有 Vision 使用文本简单的绘制了一幅图



区块链之外的很多复杂的需求我们需要借助大数据系统和搜索技术。

区块链的弱点是无法做复杂的查询，这里我们会用到搜索引擎技术解决，实际上搜索引擎角色是给区块链做索引。

上图数据写入时，保存了四份，分别在搜索引擎，关系型数据库，数据仓库和区块的

具体怎么实现，有很多方式，这里就不讨论了，否则就跑题了。

2.9. BI商业智能

数据采集，大数据分析

溯源信息的查询是通过用户手机终端实现，有几种途径，微信扫二维码，APP扫二维码，微信小程序等等。

扫码的同时还可以收集用户数据，我们可以收集到很多有价值的数据，例如地理位置，手机号码，性别，年龄等等.....

有了这些数据便可以挖掘出有价值的数据，甚至可以将数据提供给生产企业作参考。

传统销售数据只能跟踪到地域，也就是统计出地域销量，没法监控到最后一公里的数据，而我们主要是采集商品最后一公里的数据。

我们能做到用户消费后，呼叫中心立即跟进回访，还能在用户快用完商品是向用户推送促销信息，以及客服二次跟进。

大数据能做什么?

1. 用户行为分析，用户的喜好，这些数据能为后面精准推送提供支持。

2. 消费与地理分析的关系

3. 年龄段与购买力的关系

4. 区域产品的存量，例如：用户扫描了一次二维码，可能用户就已经使用了改产品。我们就知道该地区投放的1000件商品被消耗了意见。

5. 性别与消费习惯

6. 两次间隔消费时间

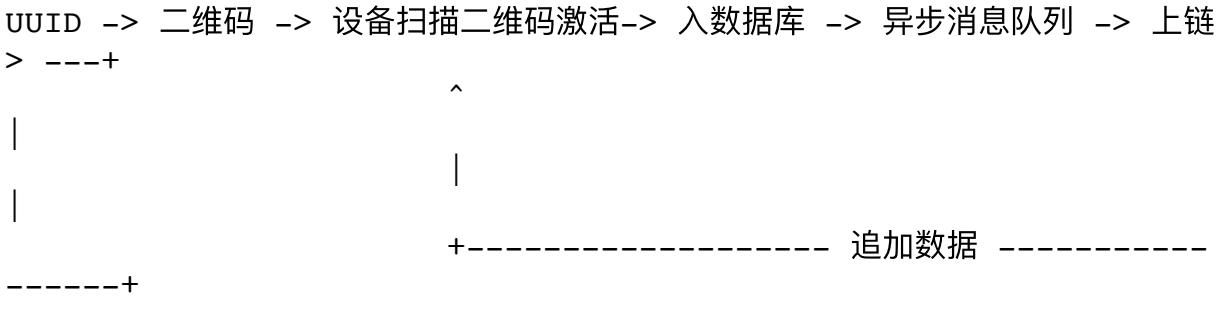
7. 活跃用户和沉睡用户

2.10. 采集终端

溯源数据怎么录入呢？例如我们开发一个设备，二维码扫描枪，内置安卓系统。

我们不清楚他们的教育背景以及学习能力，所以设计原则是尽量傻瓜化，降低数据录入难度和学习难度，终端开机后互动教学，走一遍流程即可上手。

首先将溯源环节的每个节点通过后台事先写入数据库，接下来通过GIS地理信息系统匹配。



终端会帮助用户欲录入信息，用户可以在信息基础上修改或者重写。同时终端支持图片，图像记录上传。

对于图片还能实现 EXIF 数据保存，包括图片描述信息，地理信息等等.....

2.11. 多媒体数据

这里我们需要考虑是否需要记录多媒体数据，这里的多媒体指图像，声音，甚至3D扫描数据等等.....

对于图片、音频与视频，我们可以将它集成到采集终端上，然后异步上传到去中心化的分布式文件系统中。

去中心化的分布式文件系统能实现，一张图片一个hash值，通过hash值访问图片，图片被同步到相邻节点实现去中心化，图片被修改hash值随之变化数据便无效。

2.12. 物流接口

使用物流单好通过物流公司提供的借口获得物流数据，然后写入到区块。

2.13. 如何激励用户

防伪技术做了，区块链溯源也做了，那么对于用户来说，他可能懒得去扫你的二维码，怎么办呢?

这里需要激励用户，怎样激励用户，我的方案是送代币。

首先代币不仅能够购买物品，还能交易，流通，形成一个小的商业闭环。其次目前代币已经泛滥 99% 可能是空气币，这里我们需要将代币的价值与物品对价，类似金本位/银本位。

怎样操作呢？例如一个代币等于一斤水果，无论代币怎样炒作，最终用户不想玩下去了，就来换水果，也可以是大米，食用油等等...

关于怎样使用代币来做积分系统请参考我的另一篇文章 《使用代币替代传统积分系统》，你可以在搜索引擎中找到

根据业务需要，可以发行布置一套币，例如水果币，流量币，话费币，每种币的功能不同，这些币可以在交易所中撮合交易，例如卖出水果币，换成流量币等等。

由于国家的法规问题，代币系统设计原则一定是代币只能用来购买商城中的物品，不能直接兑换成RMB，否则会触碰到国家的红线。但是通过交易所，币币之间兑换我们就控制不了了。

另外扫描二维码显示溯源防伪信息的同时我们有很多可以操作空间，可以获取用户地理位置，手机号码等等信息，为后面大数据分析埋点。

用户激励手段

1. 分享激励
2. 好评激励
3. 用户等级激励
4. 代币激励
5. 用户排名，PK排行榜
6. 成就勋章
7. 身份标签，黄马甲：）

等等，手段众多，目的是让用户查询溯源信息，手机用户数据，鼓励代币消费等等.....

2.14. 上链

并不是所有数据都上链，哪些数据上链呢?

产地(出生、生长)、采购、加工(检疫、屠宰)、库存、运输、销售、配送等等.....

2.15. 以太坊解决方案

我们设计一个简单的合约，模拟上面提到的解决方案

```
pragma solidity ^0.4.20;

contract Trace {

    enum State { Origin, Factory, QA, Shipping, Received,
Pending }

    string name;
    uint price;
    uint weight;
    bool lock = false;  //合约锁
    bool close = false; //合约状态
    uint number = 1;
    uint attr_number = 1;

    mapping (address  => string) guestbook; //客户留言本

        struct Attribute {
            address owner;       // 供应商
        string name;             // 属性的名字
        string date;             // 生产日期
        string desc;             // 描述信息

    }
```

```
    mapping (uint  => Attribute) attribute;

    struct Logistics {
        address owner;  // 中转站
        string date;    // 转运日期
        State status;   // 状态
        string message; // 留言信息
    }
    mapping (uint  => Logistics) stations;

    function Trace(string _name, uint _price, uint _weight)
public {
        name = _name;
        price = _price;
        weight = _weight;
        }
    // 名称
    function getName() public view returns(string){
        return name;
    }
    // 价格
    function getPrice() public view returns(uint){
        return price;
    }
    // 重量
    function getWeight() public view returns(uint){
        return weight;
    }

     // 增加商品属性
    function putAttribute(address _owner,string _name, string
_date, string _desc ) public{
        if(lock == false){
                        Attribute memory item =
Attribute(_owner, _name,_date,_desc);
                        attribute[attr_number] = item;
                        attr_number = attr_number + 1;
        }
    }

        // 获得属性
    function getAttribute(uint _attr_number) public view
returns(address, string, string, string) {
        require(_attr_number < attr_number);
        Attribute memory item = attribute[_attr_number];
```

```
                return (item.owner, item.name, item.date,
item.desc);
        }

    // 增加物流中转信息
    function putLogistics(address _owner,string _date, State
_status, string _message ) public{
        if(close == false){
            Logistics memory node =
Logistics(_owner,_date,_status,_message);
            stations[number] = node;
            number = number + 1;
            lock = true;
        }
        if (_status == State.Received) {
            close = true;
        }
    }

        // 获得中转信息
    function getLogistics(uint _number) public view
returns(address, string, State, string) {
        require(_number < number);

        Logistics memory node = stations[_number];

                return (node.owner, node.date, node.status,
node.message);
        }

        // 或者转中站数量
        function getLogisticsCount() public view returns(uint){
            return number;
        }

    // 客户留言
        function addGuestbook(address _owner, string message)
public{
            guestbook[_owner] = message;
        }
}
```

怎样使用这个合约呢？合约部署，需要输入三个参数，分别是名称，价格和装量

```
Trace(string _name, uint _price, uint _weight)
```

产品属性可以在出厂前设置，一旦出厂进入物流阶段就不允许在更改了。

2.15.1. 应用场景一

调用合约案例一，这是没有经过深加工的原产品案例。例如 Trace("山羊肉", 25, 50)

```
var contract;
Trace.deployed().then(function(instance){contract=instance;});
contract.getName();
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef57","颜色", "", "黑色")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef57","产地", "", "内蒙古")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef57","出生", "2017-01-12", "XXX牧场")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef57","宰杀", "2018-02-12", "XXX宰杀")

contract.putLogistics("0x627306090abab3a6e1400e9345bc60c78a8bef57","2018-02-20",0,"XXX牧场");
contract.putLogistics("0x627306090abab3a6e1400e9345bc60c78a8bef57","2018-02-20",1,"XXX屠宰公司");
contract.putLogistics("0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef","2018-02-22",2,"XXX检验检疫");
contract.putLogistics("0xf17f52151ebef6c7334fad080c5704d77216b732","2018-02-21",3,"XXX一级经销商");
contract.putLogistics("0x821aea9a577a9b44299b9c15c88cf3087f3b55
```

```
44","2018-02-23",3,"XXX二级经销商");
contract.putLogistics("0x821aea9a577a9b44299b9c15c88cf3087f3b55
44","2018-02-24",3,"XXX批发中心");
contract.putLogistics("0x821aea9a577a9b44299b9c15c88cf3087f3b55
44","2018-02-25",3,"XXX超市");
contract.putLogistics("0x0d1d4e623d10f9fba5db95830f7d3839406c6a
f2","2018-02-26",4,"用户包裹收到");

contract.getNode(); // 获得物流经过的转运站数量
```

2.15.2. 应用场景二

调用合约案例二，这是深加工的产品案例。例如 Trace("牦牛肉干", 80, 500)

```
var contract;
Trace.deployed().then(function(instance){contract=instance;});
contract.getName();
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef
57","调和油", "2016-10-10", "银龙鱼牌")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef
57","辣椒粉", "2016-10-30", "西藏XXX公司生产")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef
57","生抽", "2016-01-12", "XXX生抽, XXX生产")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef
57","山梨酸钾", "2017-02-12", "XXX生产")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef
57","防腐剂", "2017-02-12", "XXX生产")
contract.putAttribute("0x627306090abab3a6e1400e9345bc60c78a8bef
57","牦牛肉", "2017-02-12", "XXX牧场")

contract.putLogistics("0x627306090abab3a6e1400e9345bc60c78a8bef
57","2018-02-20",0,"XXX牧场");
contract.putLogistics("0x627306090abab3a6e1400e9345bc60c78a8bef
57","2018-02-20",1,"XXX公司生产");
contract.putLogistics("0xc5fdf4076b8f3a5357c5e395ab970b5b54098f
ef","2018-02-22",2,"XXX通过QA、QC");
contract.putLogistics("0xf17f52151ebef6c7334fad080c5704d77216b7
```

```
32","2018-02-21",3,"XXX一级经销商");
contract.putLogistics("0x821aea9a577a9b44299b9c15c88cf3087f3b55
44","2018-02-23",3,"XXX二级经销商");
contract.putLogistics("0x821aea9a577a9b44299b9c15c88cf3087f3b55
44","2018-02-24",3,"XXX批发中心");
contract.putLogistics("0x821aea9a577a9b44299b9c15c88cf3087f3b55
44","2018-02-25",3,"XXX超市");
contract.putLogistics("0x0d1d4e623d10f9fba5db95830f7d3839406c6a
f2","2018-02-26",4,"用户包裹收到");

contract.getNode(); // 获得物流经过的转运站数量
```

2.15.3. 用户留言

```
contract.addGuestbook("0x0d1d423e623d10f9d10f9d10f9d10f9d10f9fb
a5","东西好吃，下次还买，给好评");
```

2.16. Hyperledger 解决方案

由于家里在刷墙，服务器收起来了，没有开发环境，只能提供部分参考代码，无法提供合约完整代码，只是给大家一个思路，原理很上面以太坊的合约类似。

2.16.1. 溯源合约涉及

```
package main

import "fmt"
import "encoding/json"

const (
        Origin = iota   // 0
```

```
        Factory         // 1
        QA                      // 2
        Shipping                // 3
        Received                // 4
        Pending                 // 5
        Supermarket             // 6
)

type structElement struct {
        Name string `json:"name"`
        Company string `json:"company"`
        Description string `json:"description"`
}

type structLogistics struct {
    Stations string `json:"stations"`   // 中转站
        Date    string  `json:"date"`  // 转运日期
        Status  uint8   `json:"status"`  // 状态
    Message    string `json:"message"` // 留言信息
}

type Trace struct {
        Name string     `json:"name"`
        Address string  `json:"address"`
        Attribute       map[string]string
`json:"attribute"`
        Element         []structElement
`json:"element"`
        Logistics       map[string]structLogistics
`json:"logistics"`
}

func (trace *Trace) setName(_name string) {
    trace.Name = _name
}

func (trace *Trace) getName() string {
    return trace.Name
}

func (trace *Trace) putAttribute(_key string, _value string) {
    trace.Attribute[_key] = _value
}

func (trace *Trace) putLogistics(_key string, _value
structLogistics) {
    trace.Logistics[_key] = _value
```

```
}

func main(){

        trace := &Trace{
                Name: "牦牛肉干",
                Address: "内蒙古呼和浩特",
                Attribute: map[string]string{},
                Element: []structElement{structElement{Name:"塑
料袋",Company: "XXX塑料制品有限公司", Description: "外包
装"},structElement{Name:"辣椒粉",Company: "XXX调味品有限公司",
Description: "采摘年份2016-10-10"},structElement{Name:"调和
油",Company: "XXX调味品有限公司", Description: "生产日期2016-10-
10"}},
                Logistics: map[string]structLogistics{}}

        trace.putAttribute("Color","Red")
        trace.putAttribute("Size","10")
        trace.putAttribute("Weight","100kg")

        trace.putLogistics("1", structLogistics{"呼和浩
特","2016-10-15", Origin, "牦牛收购"})
        trace.putLogistics("2", structLogistics{"呼和浩
特","2016-10-18", Factory, "牦牛宰杀"})
        trace.putLogistics("3", structLogistics{"呼和浩
特","2016-10-15", QA, "经过质检"})
        trace.putLogistics("4", structLogistics{"北京市","2016-
10-15", Shipping, "运输中"})
        trace.putLogistics("5", structLogistics{"杭州市","2016-
10-15", Shipping, "XXX冷库"})
        trace.putLogistics("5", structLogistics{"深圳市","2016-
10-15", Supermarket, "XXX超市"})
        trace.putLogistics("5", structLogistics{"龙华区","2016-
10-15", Received, "用户签收"})


        traceJson, _ := json.Marshal(trace)
        fmt.Println(string(traceJson))

}
```

2.16.1.1. 食品安全朔源

```
        trace := &Trace{
                Name: "牦牛肉干",
                Address: "内蒙古呼和浩特",
                Attribute: map[string]string{},
                Element: []structElement{structElement{Name:"塑
料袋",Company: "XXX塑料制品有限公司", Description: "外包
装"},structElement{Name:"辣椒粉",Company: "XXX调味品有限公司",
Description: "采摘年份2016-10-10"},structElement{Name:"调和
油",Company: "XXX调味品有限公司", Description: "生产日期2016-10-
10"}},
                Logistics: map[string]structLogistics{}}

        trace.putAttribute("Color","Red")
        trace.putAttribute("Size","10")
        trace.putAttribute("Weight","100kg")

        trace.putLogistics("1", structLogistics{"呼和浩
特","2016-10-15", Origin, "牦牛收购"})
        trace.putLogistics("2", structLogistics{"呼和浩
特","2016-10-18", Factory, "牦牛宰杀"})
        trace.putLogistics("3", structLogistics{"呼和浩
特","2016-10-15", QA, "经过质检"})
        trace.putLogistics("4", structLogistics{"北京市","2016-
10-15", Shipping, "运输中"})
        trace.putLogistics("5", structLogistics{"杭州市","2016-
10-15", Shipping, "XXX冷库"})
        trace.putLogistics("5", structLogistics{"深圳市","2016-
10-15", Supermarket, "XXX超市"})
        trace.putLogistics("5", structLogistics{"龙华区","2016-
10-15", Received, "用户签收"})
```

2.16.1.2. 水平移植

这个方案可以水平移植到其他领域，例如 药品安全溯源

```
        trace := &Trace{
                Name: "强身大力丸",
                Address: "深圳是XXX制药有限公司",
                Attribute: map[string]string{},
                Element: []structElement{
                        structElement{Name:"枸杞",Company: "宁夏
XXX农业有限公司", Description: "采摘年份2016-10-10，10g"},
                        structElement{Name:"茯苓",Company: "河南
XXX农业有限公司", Description: "采摘年份2016-10-10，20kg"},
                        structElement{Name:"XXX",Company: "XXX
有限公司", Description: "生产日期2016-10-10"},
                        structElement{Name:"XXX",Company: "XXX
有限公司", Description: "生产日期2016-10-10"},
                        ...
                        ...
                        structElement{Name:"塑料包装",Company:
"XXX有限公司", Description: "生产日期2016-10-10"},
                        structElement{Name:"包装盒",Company:
"XXX有限公司", Description: "生产日期2016-10-10"}
                        },
                Logistics: map[string]structLogistics{}}

        trace.putAttribute("Color","Red")
        trace.putAttribute("Size","10")
        ...
        ...
        trace.putAttribute("Weight","100kg")

        trace.putLogistics("1", structLogistics{"呼和浩
特","2016-10-15", Origin, "原材料...."})
        trace.putLogistics("2", structLogistics{"呼和浩
特","2016-10-18", Factory, "生产...."})
        trace.putLogistics("3", structLogistics{"呼和浩
特","2016-10-15", QA, "经过质检"})
        trace.putLogistics("3", structLogistics{"XXX市药品监督
局","2016-10-15", QA, "经过质检"})
        trace.putLogistics("4", structLogistics{"北京市","2016-
10-15", Shipping, "运输中"})
        trace.putLogistics("5", structLogistics{"杭州市","2016-
10-15", Shipping, "XXX冷库"})
```

```
        trace.putLogistics("5", structLogistics{"深圳市","2016-
10-15", Supermarket, "XXX超市"})
        trace.putLogistics("5", structLogistics{"龙华区","2016-
10-15", Received, "用户签收"})
```

合约落地，还需要做一些调整已适应实际场景。但基本思路是通的。

2.16.2. 积分通正（代币）

我发现用以太坊思维，将以太坊代币合约搬到 hyperledger 上，一样可以实现代币的功能，这个代币除了不能上交易所，基本满足我们替代积分系统的需求，下面是我写了这样一个合约，在超级账本上实现类似以太坊的代币转账功能。

```
package main

import (
        "bytes"
        "encoding/json"
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        sc "github.com/hyperledger/fabric/protos/peer"
)

// Define the Smart Contract structure
type SmartContract struct {
}

type Token struct {
        Owner                   string  `json:"Owner"`
        TotalSupply     uint     `json:"TotalSupply"`
        TokenName               string  `json:"TokenName"`
        TokenSymbol     string  `json:"TokenSymbol"`
        BalanceOf               map[string]uint
`json:"BalanceOf"`
}
```

```
func (token *Token) initialSupply(){
        token.BalanceOf[token.Owner] = token.TotalSupply;
}

func (token *Token) transfer (_from string, _to string, _value
uint){
        if(token.BalanceOf[_from] >= _value){
                token.BalanceOf[_from] -= _value;
                token.BalanceOf[_to] += _value;
        }
}

func (token *Token) balance (_from string) uint{
        return token.BalanceOf[_from]
}

func (token *Token) burn(_value uint) {
        if(token.BalanceOf[token.Owner] >= _value){
                token.BalanceOf[token.Owner] -= _value;
                token.TotalSupply -= _value;
        }
}

func (token *Token) burnFrom(_from string, _value uint) {
        if(token.BalanceOf[_from] >= _value){
                token.BalanceOf[_from] -= _value;
                token.TotalSupply -= _value;
        }
}

func (token *Token) mint(_value uint) {

        token.BalanceOf[token.Owner] += _value;
        token.TotalSupply += _value;

}

func (s *SmartContract) Init(stub shim.ChaincodeStubInterface)
sc.Response {
        return shim.Success(nil)
}

func (s *SmartContract) initLedger(stub
shim.ChaincodeStubInterface) sc.Response {

        token := &Token{
                Owner: "netkiller",
```

```
                TotalSupply: 10000,
                TokenName: "代币通正",
                TokenSymbol: "COIN",
                BalanceOf: map[string]uint{}}

        token.initialSupply()

        tokenAsBytes, _ := json.Marshal(token)
        stub.PutState("Token", tokenAsBytes)
        fmt.Println("Added", tokenAsBytes)

        return shim.Success(nil)
}

func (s *SmartContract) transferToken(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 3 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        tokenAsBytes, _ := stub.GetState(args[0])
        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)
        token.transfer(args[1],args[2],args[3])

        tokenAsBytes, _ = json.Marshal(token)
        stub.PutState(args[0], tokenAsBytes)

        return shim.Success(nil)
}

func (s *SmartContract) balanceToken(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        tokenAsBytes, _ := stub.GetState(args[0])
        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)
        amount := token.balance(args[1])
```

```
        return shim.Success(amount)
}

func (s *SmartContract) Invoke(stub
shim.ChaincodeStubInterface) sc.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to
interact with the ledger appropriately
        if function == "balanceToken" {
                return s.balanceToken(stub, args)
        } else if function == "initLedger" {
                return s.initLedger(stub)
        } else if function == "transferToken" {
                return s.transferToken(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}

// The main function is only relevant in unit test mode. Only
included here for completeness.
func main() {

        // Create a new Smart Contract
        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
}
```

合约代码的测试

```
func main(){

        token := &Token{
                Owner: "netkiller",             // 代币管理者
```

```
                TotalSupply: 10000,                 // 代币发行总量
                TokenName: "积分连",                 // 代币名称
                TokenSymbol: "NEO",                 // 代币符号 NEO
                BalanceOf: map[string]uint{}}

        token.initialSupply()            // 初始化代币

        fmt.Println(token.balance("netkiller")) // 查询余额

        token.transfer("netkiller","neo", 100)  // 转账，这里账号使用用户ID，没有使用以太坊钱包那样的哈希值，因为哈希值不便于记忆。

        fmt.Println(token.balance("netkiller"))
        fmt.Println(token.balance("neo"))

}
```

我们可以建立很多套这样的比，例如水果币，蔬菜币，流量币...

开发一个小型交易所难度也不大，让用户在交易所中交易这些币。

2.17. 总结

区块链技术不能彻底解决食品的防伪和溯源问题，但是他能增加造假的难度。目前还不能解决源头造假的问题。

举例阳澄湖大闸蟹怎么用区块链造假，首先将外地蟹运往阳澄湖洗澡。将螃蟹打捞上来，期间记录过程，拍摄视频，然后贴上二维码，RFID，NFC标签，在装上有防伪的箱子（带有IoT技术，开箱需要扫码，中间运输过程中一旦开启时将上报公司，且箱子是一次性的，开启后无法在还原，只能报废），同时也给物流公司的集装箱上了电子锁，电子锁有记录GPS轨迹的功能，还有加速度传感器，被开启或中途长时间停车都会通过 IoT 技术通知公司。武装到牙齿了吧！！！

一切就绪后，准备防伪证书，视频，产品信息，将这些数据上链。发货，记录供应链数据，物流数据，GPS行车轨迹，中转站。最后到大商超，上架。所有数据收集完毕，更新区块链，追加供应链数据。

现在用户下载溯源APP或小程序，扫码或NFC可以看到溯源数据和整个流程的视频记录。哇真实，可信。同时扫码还有区块链积分的激励。

整个过程，除了大闸蟹是假的，其他环节全是真实的。最终所谓的区块链溯源仅仅成为了一种营销工具和手段。它并不能保证物品是真实性。

同理，黑龙江五常大米，福建武夷山茶叶也可以用类似方法造假。防伪技术解决了中间环节的安全问题。例如中间调包，但是源头控制不了。

真正解决食品安全问题需要靠法制和公民的道德。这是一个礼崩乐坏，底层互害的社会，日本不需要任何区块链溯源，即使是落后的泰国也不需要什么区块链的加持来保证食品安全。

3. 以太坊·电影院场景区块链应用探索

最近一直在思考区块链在各种场景下的落地问题。

下面是电影院场景区块链应用探索，这是我的一个设想，区块链如何在院线场景落地的一些思路。

为此我写一个这样的智能合约，实现了构造方法描述一部电影票价，坐位数量，可以实现订票，扣款，退票，还款等功能。

3.1. 合约文件

```
pragma solidity ^0.4.21;

// author: netkiller
// home: http://www.netkiller.cn
// QQ:13721218

contract Movie {
  address public publisher;     //电影院
  string name;                              //影片名称
  uint price;                               //票价
  uint public seat;                         //坐位数量
  mapping (address => uint) public audience;

  bool play = false;                        //电影是否已经开播，开播后不允许买票和退票。

  //合约构造方法
  function Movie(string _name, uint _price, uint _seat) public{
    publisher = msg.sender;
    name = _name;
    price = _price;
    seat = _seat;
  }

  //获取剩余坐位数量
```

```
  function getName() public view returns (string){
      return name;
  }

  //有时需要开放预留坐位，调整坐位数量
  function changeSeat(uint _seat) public {
    if (msg.sender != publisher) { return; }
    if (play == true){ return; }
    seat = _seat;
  }

  //获取剩余坐位数量
  function getSeat() public view returns (uint){
      return seat;
  }

  //买票方法，参数买票者，票数，买票后扣除用户以太币。
  function buyTicket(address _audience, uint _ticket) public
payable returns (bool success) {
    if (_ticket >= seat) { return false; }
    if (play == true){ return false; }
    uint amount = price * _ticket;      //计算票价

    if (this.balance >= amount) {
        _audience.transfer(_audience.balance - amount);
        publisher.transfer(publisher.balance + amount);
        audience[_audience] = _ticket;
        seat -= _ticket;
    }

    return true;
  }

  //退票
  function refundTicket(address _audience, uint _ticket) public
{
    if (msg.sender != publisher) { return; }
    if (play == true){ return; }
    uint amount = price * _ticket;

    if (audience[_audience] <= _ticket) {

      if (publisher.balance >= amount) {
        _audience.transfer(_audience.balance + amount);
        publisher.transfer(publisher.balance - amount);
        audience[_audience] -= _ticket;
```

```
        seat += _ticket;

      }
    }
  }

  //播放电影，锁定
  function playMovie() public {
      play = true;
  }

  //销毁合约
  function destroy() public{
    if (msg.sender == publisher) {
      selfdestruct(publisher);
    }
  }
}
```

3.2. 合约用法

例如现在要上映一部影片步骤是,首先实例化合约，然后部署合约

```
Movie("黑客帝国", "25", 80)
```

```
var contract;
Movie.deployed().then(function(instance){contract=instance;});
contract.getName(); //获得影片名字

contract.buyTicket("0x627306090abab3a6e1400e9345bc60c78a8bef57"
,1) // 购买 1 张票
contract.buyTicket("0x627306090abab3a6ebc60c78a8bef571400e9345"
,5) // 购买 5 张票
contract.buyTicket("0xf17f52151EbEF6C7334FAD080c5704D77216b732"
```

```
,2) // 购买 2 张票

...
...
contract.refundTicket("0x627306090abab3a6ebc60c78a8bef571400e93
45", 2 ) // 退 2 张票

contract.audience.call().then(console.log);

contract.playMovie() // 电影开播，锁定这个合约
```

4. 游戏领域区块链探索

如何将区块链嫁接到游戏领域，我做了很多思考，经过分析总结，发现下面几项内容非常适合上链。

上链内容

- 积分代币

  如果说区块链应用于游戏领域，可能99%的人首先会想到是代币，的确游戏领域实施区块链连，代币必不可少。但是区块链不等于代币。

- 游戏装备
- 人物属性
- 关卡任务

下面我们要思考为什么需要将游戏数据放到区块链上，玩游戏的人都知道私服，私人架设游戏服务器，私服上玩游戏遇到最大的问题就是公平性。管理员可以随意调整服务器参数。

私服存在哪些问题呢?

- 修改游戏装备属性
- 修改生命与魔法值
- 关卡参数
- 人物属性
- 随意封账号

这是我们在私服上遇到的最大问题，那么官方服务器就公平吗？不一定，对于弱势的玩家只能相信游戏公司的承诺。

有了区块链技术，我们能做什么呢？例如我们将用户装备数据等数据上链，这样保证了装备永远属于玩家

区块链能做什么？

- “点” 奖励采用代币实现，可以实现流通，兑换，消费等等.....
- 爆出装备立即上链
- 用户等级属性上链
- 用户状态上链
- 关卡数据上链

了凸显公平性，我们采用公链，查询用户数据可以使用接口，也可以直接到公链上查询。

下面详细讲解具体怎么实现。

4.1. 游戏代币

传统币 Point (点) 仅仅是一个数字，数字存在数据库中，例如

```
Username         | Point (Integer)
-----------------------
Neo              | 1000
Jam              | 500
```

因为仅仅是一个数字，管理员可以随意修改，黑客也可随意修改，例如

```
update member set point = 1000000000000 where username = 'Neo'
```

瞬间就有 1000000000000 点。由于是在数据库中修改，没有日志，不知道谁操作的，可能是开发人员，可以是管理员，也可能是黑客。

如何消费“点呢”，例如消费 100 个点：

```
update member set point = point - 100 where username = 'Neo'
```

传统币“点”，只是一个数字做加法和减法运算，安全性主要依赖于开发团队的能（期望别出BUG），运维团队的能力（被别黑客攻击），以及DBA（数据库管理员）的节操。

审计全靠开发人员打印出的日志。

现在我们再看看数字货币，跟很多朋友聊天中发现，他们还没有理解什么是币，他们认为数字代币花掉就没了（消失了），然后通过挖矿不停的产生新币，这种理解完全错误。

数字币是这样运作的，首先发行时设置了币的总量例如 1000000，然后将所有代币存入发行者账号，例如 account 1

```
account                  | coin
---------------------------------
account1         | 1000000
account2         | 0
```

```
account3         | 0
account4         | 0
account5         | 0
```

现在 account2 游戏在线1小时奖励 10 个币，系统从账号account1转账5个币给 account2

```
account                  | coin
---------------------------------
account1         | 999990
account2         | 10
account3         | 0
account4         | 0
account5         | 0
```

以此类推，从 account1 转账给其他账号。

```
account                  | coin
---------------------------------
account1         | 999960
account2         | 10
account3         | 10
account4         | 10
account5         | 10
```

现在 account3 消费 5个币买了装备。从 account3 转 5 个币到 account1

```
account                  | coin
---------------------------------
account1         | 999965
account2         | 10
account3         | 5
account4         | 10
```

```
account5        | 10
```

现在你应该看懂了把，代币是流通的，总量是不变的。account1 账号负责币的发行，回收等等工作。

同时任何转账将产生区块，历史数据永久记录。

下面是一个高级代币合约，地址
https://github.com/ibook/NetkillerAdvancedToken

```
pragma solidity ^0.4.20;

/******************************************/
/*       Netkiller ADVANCED TOKEN         */
/******************************************/
/* Author netkiller <netkiller@msn.com>   */
/* Home http://www.netkiller.cn           */
/* Version 2018-03-05                     */
/* Version 2018-03-06 - Add Global lock   */
/******************************************/

interface tokenRecipient { function receiveApproval(address
_from, uint256 _value, address _token, bytes _extraData) public;
}

contract NetkillerAdvancedToken {
    address public owner;
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 2;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    uint256 public sellPrice;
    uint256 public buyPrice;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
```

```
allowance;

    // This generates a public event on the blockchain that will
notify clients
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);
    event Approval(address indexed owner, address indexed
spender, uint256 value);

    mapping (address => bool) public frozenAccount;

    /* This generates a public event on the blockchain that will
notify clients */
    event FrozenFunds(address target, bool frozen);

    bool lock = true;

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function NetkillerAdvancedToken(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        owner = msg.sender;
        totalSupply = initialSupply * 10 ** uint256(decimals);
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
        symbol = tokenSymbol;                               //
Set the symbol for display purposes
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    modifier isLock {
```

```
        require(!lock);
        _;
    }

    function setLock(bool _lock) onlyOwner {
        lock = _lock;
    }

    function transferOwnership(address newOwner) onlyOwner
public {
        owner = newOwner;
    }

    /* Internal transfer, only can be called by this contract */
    function _transfer(address _from, address _to, uint _value)
isLock internal {
        require (_to != 0x0);                                  //
Prevent transfer to 0x0 address. Use burn() instead
        require (balanceOf[_from] >= _value);                  //
Check if the sender has enough
        require (balanceOf[_to] + _value > balanceOf[_to]); //
Check for overflows
        require(!frozenAccount[_from]);                        //
Check if sender is frozen
        require(!frozenAccount[_to]);                          //
Check if recipient is frozen
        balanceOf[_from] -= _value;                            //
Subtract from the sender
        balanceOf[_to] += _value;                              //
Add the same to the recipient
        Transfer(_from, _to, _value);
    }

    /**
     * Transfer tokens
     *
     * Send `_value` tokens to `_to` from your account
     *
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transfer(address _to, uint256 _value) public {
        _transfer(msg.sender, _to, _value);
    }

    /**
     * Transfer tokens from other address
     *
```

```
     * Send `_value` tokens to `_to` in behalf of `_from`
     *
     * @param _from The address of the sender
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(_value <= allowance[_from][msg.sender]);     //
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    /**
     * Set allowance for other address
     *
     * Allows `_spender` to spend no more than `_value` tokens
in your behalf
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     */
    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        Approval(msg.sender, _spender, _value);
        return true;
    }

    /**
     * Set allowance for other address and notify
     *
     * Allows `_spender` to spend no more than `_value` tokens
in your behalf, and then ping the contract about it
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     * @param _extraData some extra information to send to the
approved contract
     */
    function approveAndCall(address _spender, uint256 _value,
bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
        if (approve(_spender, _value)) {
```

```
            spender.receiveApproval(msg.sender, _value, this,
_extraData);
            return true;
        }
    }

    /**
     * Destroy tokens
     *
     * Remove `_value` tokens from the system irreversibly
     *
     * @param _value the amount of money to burn
     */
    function burn(uint256 _value) onlyOwner public returns (bool
success) {
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        Burn(msg.sender, _value);
        return true;
    }

    /**
     * Destroy tokens from other account
     *
     * Remove `_value` tokens from the system irreversibly on
behalf of `_from`.
     *
     * @param _from the address of the sender
     * @param _value the amount of money to burn
     */
    function burnFrom(address _from, uint256 _value) onlyOwner
public returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
Update totalSupply
        Burn(_from, _value);
        return true;
```

```
    }

    /// @notice Create `mintedAmount` tokens and send it to
`target`
    /// @param target Address to receive the tokens
    /// @param mintedAmount the amount of tokens it will receive
    function mintToken(address target, uint256 mintedAmount)
onlyOwner public {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
        Transfer(0, this, mintedAmount);
        Transfer(this, target, mintedAmount);
    }

    /// @notice `freeze? Prevent | Allow` `target` from sending
& receiving tokens
    /// @param target Address to be frozen
    /// @param freeze either to freeze it or not
    function freezeAccount(address target, bool freeze)
onlyOwner public {
        frozenAccount[target] = freeze;
        FrozenFunds(target, freeze);
    }

    /// @notice Allow users to buy tokens for `newBuyPrice` eth
and sell tokens for `newSellPrice` eth
    /// @param newSellPrice Price the users can sell to the
contract
    /// @param newBuyPrice Price users can buy from the contract
    function setPrices(uint256 newSellPrice, uint256
newBuyPrice) onlyOwner public {
        sellPrice = newSellPrice;
        buyPrice = newBuyPrice;
    }

    /// @notice Buy tokens from contract by sending ether
    function buy() payable public {
        uint amount = msg.value / buyPrice;               //
calculates the amount
        _transfer(this, msg.sender, amount);              //
makes the transfers
    }

    /// @notice Sell `amount` tokens to contract
    /// @param amount amount of tokens to be sold
    function sell(uint256 amount) public {
        require(this.balance >= amount * sellPrice);      //
checks if the contract has enough ether to buy
```

```
        _transfer(msg.sender, this, amount);              //
makes the transfers
        msg.sender.transfer(amount * sellPrice);          //
sends ether to the seller. It's important to do this last to
avoid recursion attacks
    }

  function transfer(address _to, uint256 _value, bytes _data)
public returns (bool) {
    require(_to != address(this));
    transfer(_to, _value);
    require(_to.call(_data));
    return true;
  }

  function transferFrom(address _from, address _to, uint256
_value, bytes _data) public returns (bool) {
    require(_to != address(this));

    transferFrom(_from, _to, _value);

    require(_to.call(_data));
    return true;
  }

  function approve(address _spender, uint256 _value, bytes
_data) public returns (bool) {
    require(_spender != address(this));

    approve(_spender, _value);

    require(_spender.call(_data));

    return true;
  }

}
```

这个代币合约实现了，增发，减持，全局锁，账号冻结/解冻 等等功能。

4.2. 玩家属性与游戏装备

下面的合约实现了游戏玩家上链，上链信息有玩家属性，例如肤色，眼睛，头发，血统等等。身上的穿戴物品包括武器等等。

```
pragma solidity ^0.4.20;

contract Player {

    address public owner;

    string name;
    bool lock = false;  //合约锁
    uint number = 1;
    uint attr_number = 1;

    mapping (address  => string) guestbook; //客户留言本

        struct Attribute {
        string key;              // 属性的名字
        string value;   // 属性值

    }
    mapping (uint  => Attribute) attribute;

    struct Wear {
        string name;         // 装备名
        string desciption;  // 信息
        string attribute;   // 属性，存储json 数据。例如生命+10，魔法
+5，冰冻系...

    }
    mapping (uint  => Wear) wear;

    function Player(string _name) public {
        name = _name;
        }

        modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    // 名称
    function getName() public view returns(string){
```

```
        return name;
    }
    function setLock(bool _lock) onlyOwner public {
        lock = _lock;
    }
     // 增加人物属性，例如肤色，眼睛，头发等等
    function putAttribute(string _key, string _value) onlyOwner
public{
        if(lock == false){
                        Attribute memory item = Attribute(_key,
_value);
                        attribute[attr_number] = item;
                        attr_number = attr_number + 1;
        }
    }

        // 获得属性
    function getAttribute(uint _attr_number) public view
returns(string, string) {
        require(_attr_number < attr_number);
        Attribute memory item = attribute[_attr_number];

                return (item.key, item.value);
        }

    // 增加装备信息，穿戴物品，武器，
    function putWear(string _name, string _description, string
_attribute ) onlyOwner public{
        if(lock == false){
            Wear memory node =
Wear(_name,_description,_attribute);
            wear[number] = node;
            number = number + 1;
            lock = true;
        }
    }

        // 获得信息
    function getWear(uint _number) public view returns(string,
string, string) {
        require(_number < number);

        Wear memory item = wear[_number];

                return (item.name, item.desciption,
item.attribute);
        }
```

```
        // 数量
        function getWearCount() public view returns(uint){
            return number;
        }

    // 客户留言
    function addGuestbook(address _owner, string message)
onlyOwner public{
            guestbook[_owner] = message;
        }
}
```

4.3. 装备属性与规范

假设我们开发一个游戏平台，很多厂商可以在这个平台上出售游戏。

例如屠龙宝刀只有一把，但是实际情况只要能赚钱，游戏厂商可以卖给了10个玩家，甚至更多。

为了公平起见，对于稀有的装备管理，我们要求游戏厂商在平台上备案。包括装备属性，应该归谁所有等等

4.4. 物品合成计算

区块链还可用于物品合成计算或者叫炼金术等等

很早的时候玩《暗黑破坏神III》 里面已一个盒子，放入符文，可以根据公式合成其他属性的符文，我任务这个需求可以使用区块链来完成。

另外在我玩XBOX游戏《巫师3》 中的炼金术，铸造，药水合成等等，我逗人都可以使用区块链完成。

4.5. 实施步骤

如果着手一个游戏项目上链，我们需要怎么做呢?

上链步骤

- 收集需求，收集公司的内部上链需求，听取所有部门的建议和诉求。

  收集内容例如，代币发行量多少？代币小数位数，代币名称，是否会增发，是否需要冻结，代币怎样流通，怎样回收

  Dapp 的 UI 设计，各种功能流程

- 分析需求，因为需求来自各种部门，各种岗位等等，他们不一定从事需求分析工作，所以我们需求对他们的需求过滤，分析，然后给出初步的PRD文档（产品需求文档）

- 根据收集的需求设计合约和Dapp

  根据需求设计Dapp

  系统架构设计，软件架构设计，技术选型；需要考虑扩展性，灵活性，并发设计，数据安全，部署，后期监控，各种埋点（统计、监控）

- 准备环境，我们需要三个环境，开发，测试，生产（运营）。

- 项目启动

  运维部门准备环境，开始建设监控系统

  开发部门开发合约和Dapp

  测试部门准备测试用例，测试环境

- 测试

Alpha 阶段，将合约部署到测试环境，测试合约的每个函数的工作逻辑，确保无误。因为合约一旦部署将不能更改，只能废弃使用新的合约，所以这个测试步骤必须重视。

Beta 阶段，将测试合约部署到测试网，例如 Ropsten ，可以邀请公司内部测试

- 部署生产环境

  部署合约，将合约部署到主网

  Dapp 部署到生产环境。

- 验收测试，在生产环境做最后验收测试
- 代币上交易所

5. 以太坊竞猜活动区块链探索

合约实现了报名，退出，参加人数控制，竞猜次数控制，公布答案，获奖名单等等功能

```
pragma solidity ^0.4.20;
// Author netkiller<netkiller@msn.com>
// Home http://www.netkiller.cn
contract Guess {

    address public owner;

    string name;         //活动名称
    bool start = false; //合约锁
    uint number;                              //参赛人数统计
    uint public quota;  //名额限定
    mapping (address => string) public registrantsPaid; //参加活
动
    uint maxCounter = 3;         //最大竞猜次数
    mapping (address  => uint) counter; //竞猜次数统计

    string public question; //竞猜问题
    mapping (uint  => string) public options; //选项

    // 答案结构
    struct Answer {
        address player;
        uint answer;
    }

    mapping (uint  => Answer) public answer; //答案
    uint answerIndex = 0;

    //公布最终答案
    uint public expose;

    //获奖名单
    mapping (address  => uint) winner;
```

```
    function Guess(string _name, uint _quota) public {
        name = _name;
        quota = _quota;
        number = 0;
        }

        modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    // 获取活动名称
    function getName() public view returns(string){
        return name;
    }
    function setStatus(bool _start) onlyOwner public {
        start = _start;
    }
    function setQuestion(string _question) public {
        question = _question;
    }
    // 增加人物属性，例如肤色，眼睛，头发等等
    function putOptions(uint _key, string _value) onlyOwner
public{
        if(start == false){
                options[_key] = _value;
        }
    }
    function join(string _password) onlyOwner public returns
(bool success) {
                require(start == ture);
        if (number >= quota) { return false; }
        registrantsPaid[msg.sender] = _password;
        number++;

        return true;
      }
      function changeQuota(uint _quota) onlyOwner public {
        quota = _quota;
      }
    function quit() onlyOwner public {
                require(start == false);
        //require (registrantsPaid[msg.sender] == _password);
        delete registrantsPaid[msg.sender];
        number--;
    }
    function setGuess(uint _answer) public{
```

```
        require(start == ture);
        if(maxCounter > counter[msg.sender]){
            counter[msg.sender]++;
            answer[answerIndex] = Answer(msg.sender, _answer);
            answerIndex++;
        }

    }
    //揭秘答案
    function setExpose(uint _expose) onlyOwner public {
            require(start == ture);
        expose = _expose;

        for(uint i=0;i<answerIndex;i++)
        {
            Answer memory ans = answer[i];
            if(ans.answer == expose){
                winner[ans.player] = ans.answer;
            }
        }
    }

        // 数量
        function getCount() public view returns(uint){
            return number;
        }

}
```

6. 使用代币替代传统积分系统

首先我们使用代币是为了取代传统的积分机制。因为代币的“币”特性能够实现流通，交易等等，实现一个闭环生态系统。而传统的积分只能内部使用，无法流通，外接不承认加分的价值，积分无法交易，流通。

其次我们并非是为了ICO炒作，然后割韭菜，代币取代积分是刚性需求。

这里假设您已经看我我之前写的文章，知道什么是代币，并且能用钱包部署代币合约。例如现在合约已经部署到主网，已经可以使用钱包转账代币，甚至代币已经上了交易所，接下来我们要做什么呢?

接下来的工作是将代币和网站或者手机App打通，只有将代币整合到现有业务场景中代币才有意义。

6.1. 规划

6.1.1. 账号规划

我认为我们需要下面几种角色的账号

- 代币发行账号，负责发行代币，管理代币
- 收款账号，用户代币流通中的收款，这个账号应该由财务人员负责，相当于企业对公账号。
- 交易所账号，用来对接交易所
- 用户账号，普用户的账号，依赖接受代币，消费代币，交易代币。

6.1.2. 日志规划

日志

1. 开户日志
2. 绑定日志
3. 送币日志
4. 商城日志，代币转账日志

6.1.3. 监控规划

监控

监控内容

1. 额度监控，监控账号额度变化
2. 交易监控，监控任何一笔交易
3. 异常监控

6.1.4. 代币构成规划

例如总量发行1亿，1亿代币怎样构成的?

代币构成

1. 1000万是空投赠送给 XXX 个用户
2. 3000万作为福利发给公司员工

3. 3000万放在交易

4. 3000万放在网站流通

6.2. 实施步骤

如果着手一个游戏项目上链，我们需要怎么做呢?

上链步骤

- 收集需求，收集公司的内部上链需求，听取所有部门的建议和诉求。

  收集内容例如，代币发行量多少？代币小数位数，代币名称，是否会增发，是否需要冻结，代币怎样流通，怎样回收

  Dapp 的 UI 设计，各种功能流程

- 分析需求，因为需求来自各种部门，各种岗位等等，他们不一定从事需求分析工作，所以我们需求对他们的需求过滤，分析，然后给出初步的PRD文档（产品需求文档）

- 根据收集的需求设计合约和Dapp

  根据需求设计Dapp

  系统架构设计，软件架构设计，技术选型；需要考虑扩展性，灵活性，并发设计，数据安全，部署，后期监控，各种埋点（统计、监控）

- 准备环境，我们需要三个环境，开发，测试，生产（运营）。

- 项目启动

运维部门准备环境，开始建设监控系统

开发部门开发合约和Dapp

测试部门准备测试用例，测试环境

- 测试

  Alpha 阶段，将合约部署到测试环境，测试合约的每个函数的工作逻辑，确保无误。因为合约一旦部署将不能更改，只能废弃使用新的合约，所以这个测试步骤必须重视。

  Beta 阶段，将测试合约部署到测试网，例如 Ropsten ，可以邀请公司内部测试

- 部署生产环境

  部署合约，将合约部署到主网

  Dapp 部署到生产环境。

- 验收测试，在生产环境做最后验收测试

- 代币上交易所

6.3. ERC20 代币合约

合约部署这里加就不介绍了，可以参考《Netkiller Blockchain 手札》

合约地址：
https://raw.githubusercontent.com/ibook/TokenERC20/master/contracts/TokenERC20.sol

这个合约提供增发，冻结，所有权转移等等功能。

```
pragma solidity ^0.4.21;

interface tokenRecipient { function receiveApproval(address
_from, uint256 _value, address _token, bytes _extraData)
public; }

contract TokenERC20 {
    address public owner;
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
allowance;

    // This generates a public event on the blockchain that
will notify clients
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);

    mapping (address => bool) public frozenAccount;
    event FrozenFunds(address target, bool frozen);

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function TokenERC20(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        owner = msg.sender;

        totalSupply = initialSupply * 10 ** uint256(decimals);
```

```
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
        symbol = tokenSymbol;                               //
Set the symbol for display purposes
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    /**
     * Internal transfer, only can be called by this contract
     */
    function _transfer(address _from, address _to, uint _value)
internal {
        // Prevent transfer to 0x0 address. Use burn() instead
        require(_to != 0x0);
        // Check if the sender has enough
        require(balanceOf[_from] >= _value);
        // Check for overflows
        require(balanceOf[_to] + _value > balanceOf[_to]);
        // Save this for an assertion in the future
        uint previousBalances = balanceOf[_from] +
balanceOf[_to];
        // Subtract from the sender
        balanceOf[_from] -= _value;
        // Add the same to the recipient
        balanceOf[_to] += _value;
        Transfer(_from, _to, _value);
        // Asserts are used to use static analysis to find bugs
in your code. They should never fail
        assert(balanceOf[_from] + balanceOf[_to] ==
previousBalances);
    }

    function transfer(address _to, uint256 _value) public {
        require(!frozenAccount[msg.sender]);
        _transfer(msg.sender, _to, _value);
    }

    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(!frozenAccount[msg.sender]);
        require(_value <= allowance[_from][msg.sender]);     //
```

```
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        return true;
    }

    function approveAndCall(address _spender, uint256 _value,
bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
        if (approve(_spender, _value)) {
            spender.receiveApproval(msg.sender, _value, this,
_extraData);
            return true;
        }
    }

    function burn(uint256 _value) onlyOwner public returns
(bool success) {
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        Burn(msg.sender, _value);
        return true;
    }


    function burnFrom(address _from, uint256 _value) onlyOwner
public returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
```

```
Update totalSupply
        Burn(_from, _value);
        return true;
    }


  function transfer(address _to, uint256 _value, bytes _data)
public returns (bool) {
    require(_to != address(this));
    transfer(_to, _value);
    require(_to.call(_data));
    return true;
  }

  function transferFrom(address _from, address _to, uint256
_value, bytes _data) public returns (bool) {
    require(_to != address(this));

    transferFrom(_from, _to, _value);

    require(_to.call(_data));
    return true;
  }

  function approve(address _spender, uint256 _value, bytes
_data) public returns (bool) {
    require(_spender != address(this));

    approve(_spender, _value);

    require(_spender.call(_data));

    return true;
  }


    function transferOwnership(address _owner) onlyOwner public
{
        owner = _owner;
    }
    function mintToken(address target, uint256 mintedAmount)
public onlyOwner {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
        Transfer(0, owner, mintedAmount);
        Transfer(owner, target, mintedAmount);
    }
```

```
    function freezeAccount(address target, bool freeze) public
onlyOwner {
        frozenAccount[target] = freeze;
        FrozenFunds(target, freeze);
    }
}
```

6.4. 打通用户注册

这里我们要实现新用户在网站上开户为其创建一个钱包账号。

对于现有的注册流程界面部分无需修改，只需在创建账号逻辑的环节增加一段代码，去以太坊创建账号。

```
web3.eth.personal.newAccount('!@superpassword').then(console.lo
g);
```

创建账号后

```
                    注册成功
-------------------------
用户名：netkiller
性别：男
...
...
...
钱包账号：0x627306090abab3a6e1400e9345bc60c78a8bef57      [ 下载备
份 ]
```

* （提醒）请下载备份您的钱包文件，并请牢记你的密码，一旦丢失无法找回。

点击下载按钮后将对应账号文件提供给用户，文章通常在 keystore 类似这种格式 UTC--2018-02-10T09-37-49.558088000Z--5c18a33df2cc41a1beddc91133b8422e89f041b7

有了这个账号文件，用户可以导入到Ethereum Wallet 或者 MetaMask 中。

6.5. 现有用户怎么处理

新用户我们可以在用户注册的时候为其创建一个钱包账号。那么老用户呢?

老用户我们提供“绑定”功能，如果用户已有以太坊账号，就可以将以太坊账号与网站账号做绑定。

通常我们需要一个页面

当前用户名：netkiller
以太坊钱包：_____
手机验证码：_____ [获取验证码]

[绑定] [取消]

填写钱包账号，然后点击获取验证码，然后输入验证码，点击 “提交” 完成账号的绑定。

如果你想在平台上提供转账等高级操作，你还需要让用户上传 UTC--2018-02-10T09-37-49.558088000Z--5c18a33df2cc41a1beddc91133b8422e89f041b7 文件，如果采用上传方案，就不需要绑定了，因为在文件中（json格式）address 就是账号。

当前用户名：netkiller
以太坊钱包：_____ [浏览]
手机验证码：_____ [获取验证码]

[上传] [取消]

6.6. 赠送代币

对于新开户，或者老用户绑定了钱包。我们通常要意思一下，就是送点币。

送币有两种方式，第一种是转账给用户，缺点是需要花店gas(气)，第二种是采用空投方式，就是在用户查询余额的时候送币。

先来说说第一种，转账方式。

```
fs = require('fs');
const Web3 = require('web3');
const web3 = new Web3('http://localhost:8545');
web3.version
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x05A97632C197a0496bc939C4e666c2E03Cb95DD4";
const toAddress = "0x2C687bfF93677D69bd20808a36E4BC2999B4767C";

var coinbase;

web3.eth.getCoinbase().then(function (address){
  coinbase = address;
  console.log(address);
});

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});

contract.methods.balanceOf('0x5c18a33DF2cc41a1bedDC91133b8422e8
```

```
9f041B7').call().then(console.log).catch(console.error);
contract.methods.balanceOf('0x2C687bfF93677D69bd20808a36E4BC299
9B4767C').call().then(console.log).catch(console.error);

web3.eth.personal.unlockAccount(coinbase,
"netkiller").then(console.log);
contract.methods.transfer('0x2C687bfF93677D69bd20808a36E4BC2999
B4767C', 100).send().then(console.log).catch(console.error);

contract.methods.balanceOf('0x2C687bfF93677D69bd20808a36E4BC299
9B4767C').call().then(console.log).catch(console.error);
```

第二种是空投币

```
uint totalSupply = 100000000 ether;     // 总发行量
uint currentTotalAirdrop = 0;                    // 已经空投数量
uint airdrop = 1 ether;                 // 单个账户空投数量

// 存储是否空投过
mapping(address => bool) touched;

// 修改后的balanceOf方法
function balanceOf(address _owner) public view returns (uint256
balance) {
    if (!touched[_owner] && currentTotalAirdrop < totalSupply)
{
        touched[_owner] = true;
        currentTotalAirdrop += airdrop;
        balances[_owner] += airdrop;
    }
    return balances[_owner];
}
```

空投代币省了 Gas，但是带来一个问题，就是实际代币发行量成了 totalSupply * 2 ，因为创建合约的时候代币全部发给了 msg.sender ，空投只能使用增发币，无法去 msg.sender 扣除的空投数量。

空投币不好管理发行量。有一种做法，就是发行的时候分为2分，一份是 coinbase(msg.sender) 的 另一份是空投的。

6.7. 赚取代币

这里我们举例几个场景

发放代币的方法

1. 电商平台可以通过活动，订单量等等条件将代币发放给用户
2. 智能穿戴，例如鞋子，手环，可以根据用户的运动值这算成代币，发放给用户
3. 游戏平台，用户在线时间，电子竞技赢得分数都可以这算成代币，发放给用户

6.8. 用户登录

第一个界面一定是，请输入用户名和密码，然后提交登录。

登录后进入用户信息页面

用户登录成功

```
------------------------------
当前用户名：netkiller
...
...

钱包账号：0x627306090abab3a6e1400e9345bc60c78a8bef57

------------------------------
当前余额：1000 NEO
```

NEO是代币符号，假设叫NEO，这是我的英文名。实际上NEO已经被其他代币使用：（

获取账号余额代码

```
fs = require('fs');
const Web3 = require('web3');
const web3 = new Web3('http://localhost:8545');
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x05A97632C197a0496bc939C4e666c2E03Cb95DD4";
const fromAddress =
"0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7";   //用户账号
const toAddress = "0x2C687bfF93677D69bd20808a36E4BC2999B4767C";
//收款账号

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: fromAddress , gas: 100000});
contract.methods.balanceOf(toAddress).call().then(console.log).
catch(console.error);
```

6.9. 积分商城

这里是消费代币的地方，可以使用代币对兑换礼品，购买物品等等。

用户花出去代币去向应该是，用户收款的财务账号。

```
fs = require('fs');
const Web3 = require('web3');
const web3 = new Web3('http://localhost:8545');
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x05A97632C197a0496bc939C4e666c2E03Cb95DD4";
const fromAddress =
"0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7";   //用户账号
```

```
const toAddress = "0x2C687bfF93677D69bd20808a36E4BC2999B4767C";
//收款账号

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: fromAddress , gas: 100000});

web3.eth.personal.unlockAccount(fromAddress,
"netkiller").then(console.log);
contract.methods.transfer(toAddress,
10).send().then(console.log).catch(console.error); //花费代币 10
contract.methods.balanceOf(toAddress).call().then(console.log).
catch(console.error);
```

6.10. 代币报表

报表是用来展示网站数据变化的图标，这里只列出与代币有关的报表。

6.10.1. 曾币报表

用来展示每日，每周，每月.... 赠送，空投的代币量

6.10.2. 积分商城报表

进账财务数据，每日，每周，每月....

6.11. 代币交易

代币上交易所后，用户间就可以了。

我们使用另外一个交易所账号，参与代币交易，可以卖币（回收），买币（发行）等等操作，实现代币的闭环流通。

7. 区块链征信解决方案探索

翻看了无数的文章没有找到一篇关于谈征信在区块链上怎么落地的文章。也在各种区块链微信群和QQ群中问了一圈，也没有人知道怎么落地。

现在的情况是大家都知道区块链做征信没问题，区中心化，不可撰改，简直是征信系统而设计的。那么怎么使项目落地呢？没有一篇文章谈到这个问题。可能有些大公司已经实现了，处于技术保密，没有分享。

看来只能靠自己了，以太坊和超级账本一直在研究，最近一段时间研究以太坊比较多，但是发现征信这个系统是在不适合在以太坊上实现，于是有回到超级账本上。

回到超级账本上感觉有点不适应，超级账本没有 Token ， 超级账本的合约实现与以太坊完全不同。两个系统是两种思维解决同一个区块链需求。

一看 hyperledger 就是当前IBM风格，hyperledger 特点，体系庞大，结构复杂，难以理解，运维复杂。简单的问题用复杂的方式思考，做出一个复杂的系统，可用性极差。IBM的产品特点是，你只能他们合作，一旦合作（上了船）就摔不掉，从他们小型机，到中间件产品，以及各种行业解决方案。IBM的系统出了问题，只有IBM的人才能解决。

随者 hyperledger 开源，我希望 hyperledger 的风格能脱离IBM的影子。

回到正题，研究了几个 hyperledger 提供的 Example 后，对怎么实现征信需求，有了一点思路。

首先 chaincode 合约并不复杂，由两个核心方法组织，分别是Init和Invoke。其次数据操作类似 map 数据结构, shim.ChaincodeStubInterface 接口提供了 get, put, del 等操作。

7.1. 需求分析与概要设计

证信系统信息查询问题，一怎样查询？二查询哪些信息？

区块不是关系型数据库，无法实现SQL那样的发杂查询，所以设计接口要尽量迎合区块链的，有些情况需要妥协，适应区块链的弱点和不足。

但是我们可以让数据库和区块链同时存在，相互弥补不足。

数据库部分我这就就不讲了，区块链的实现方式是，使用身份证码号查询，返回 json 数据。

7.2. 数据结构

首先我们定义一个结构体用来存储身份信息，征信信息远不止这几项，请根据你的实际情况定义即可

```
package main

import "fmt"
import "encoding/json"

type Person struct {
        No string       `json:"no"`
        Name string     `json:"name"`
        Sex     bool    `json:"sex"`
        Age int         `json:"age"`
    Address string      `json:"address"`
}

func main(){
```

```
        person := &Person{"430725198001190911","景
峯",true,30,"Shenzhen,China"}

        personJson, _ := json.Marshal(person)

        fmt.Println(string(personJson));

        person1 := &Person{
        No: "430725198001190911",
                Name: "Neo Chen",
                Sex: true,
                Age: 35,
                Address: "Shenzhen, China"}

    json2, _ := json.Marshal(person1)
    fmt.Println(string(json2))
}
```

编译，运行，测试定义json是否正确。

```
neo@MacBook-Pro ~/golang/contract % rm -rf person &&  go build
person.go  && ./person
{"no":"430725198001190911","name":"景
峯","sex":true,"age":30,"address":"Shenzhen,China"}
{"no":"430725198001190911","name":"Neo
Chen","sex":true,"age":35,"address":"Shenzhen, China"}
```

最终我们只需要结构体复制到合约代码中。

```
type Person struct {
        No string       `json:"no"`
        Name string     `json:"name"`
        Sex     bool    `json:"sex"`
        Age int         `json:"age"`
    Address string      `json:"address"`
}
```

7.3. 将征信资料写入区块链

通过下面的函数，将征信数据写入到区块链上。

```
func (s *SmartContract) createPerson(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

    if len(args) != 6 {
        return shim.Error("Incorrect number of arguments.
Expecting 6")
    }

    var person = Person{No: args[1], Name: args[2], Sex:
args[3], Age: args[4], Address: args[5]}

    personAsBytes, _ := json.Marshal(person)
    stub.PutState(args[0], personAsBytes)

    return shim.Success(nil)
}
```

7.4. 查询区块数据

通过下面方法查询链上的征信资料。

```
func (s *SmartContract) queryPerson(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

    if len(args) != 1 {
        return shim.Error("Incorrect number of arguments.
Expecting 1")
    }
    personAsBytes, _ := stub.GetState(args[0])
    return shim.Success(personAsBytes)
}
```

7.5. 删除区块

通过下面方法删除征信数据。

```
func (s *SmartContract) deletePerson(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

    if len(args) != 1 {
        return shim.Error("Incorrect number of arguments.
Expecting 1")
    }
    personAsBytes, _ := stub.GetState(args[0])

    err= stub.DelState(args[0])
        if err != nil {
                return shim.Error("Failed to delete Student
from DB, key is: "+args[0])
        }

    return shim.Success(personAsBytes)
}
```

8. Hyperledger fabric 银行应用探索

一直想写这篇文章，可是我个人对银行系统了解甚少，网上很多文章有多拿银行来举例，铺天盖地的文章，却没有一篇告诉你究竟如何落地。

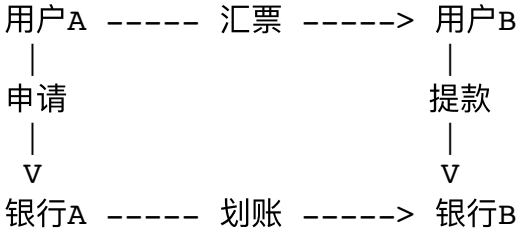
其中不少文章中提到银行SWIFT系统，什么事 SWIFT 呢?

8.1. 电汇年代

这要从电汇说起，年轻时候上学，每个学期都有一笔学费，那时主要交通是铁路，携带现金非常不便，母亲将5000元人民币缝在我的贴身内裤边上。到了学校拆开线取出，味道好极了，呵呵。

后来从同学那里得知，可以使用邮局汇款，首先去邮局，拿一个特殊信封填好地址，然后将钱交给工作人员。一周以后信封皮会记挂号信到收款人手里，那个这个信封去指定邮局取款。

记得第二学期就出现电汇，银行提供的电汇服务比邮政的速度快，也比邮局更方便。



电汇是用户A银行提出申请，将钱交给A银行。银行马上通过网络通知B银行，用户B就可以提款。没有多久邮政的系统也换成这套系统了。

那个年代只有拨号网络，帧中继，ATM(是一种网络，不是ATM取款机) 等窄带网络，现在用宽带上网的90后无法想法那个网速。

8.2. 通存通取年代

ISDN，DDN 专线的出现，才有了稳定窄带通信，银行网点互联成为可能。

MasterCard万事达，Visa维萨卡率先进入中国。很快银行内部就能实现转账了，ATM机互联成为可能，这时结算全部使用 MasterCard万事达，Visa维萨卡。银行需向MasterCard万事达，Visa维萨机构支付费用。促使一些有实力银行研发自己的内部系统，但是这些系统仅限内部使用，如果银行卡上没有MasterCard/Visa 的只能本银行使用，无法跨行。没有实力的银行则会选择门槛稍低的 INTERLINK、PLUS 等机构合作。

98年在学校寝室，一位同学拿了一张牡丹万事达卡，全寝室传看，那个年代在哈尔滨提款机都不普及，只有较大的银行网点才有。

2000来深圳，那时有个深银联，深圳首先实现了跨行业务，2年后（2002年）银联出现，国家强推，所有银行开始支持银联。当时银行卡上同时有两个标示，MasterCard/Visa 和 银联。

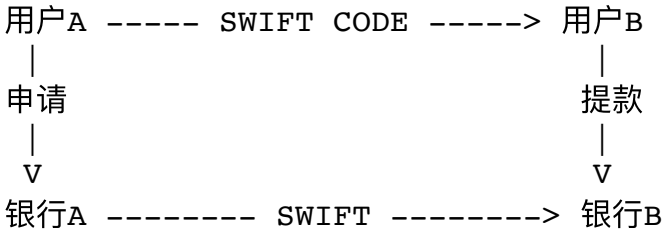
同年招商银行率先推出网上银行。招商银行电脑部一度成为IT红黑榜上评价最高的公司，可惜没有跟上互联网的步伐.....

目前新开的银行卡已经看不到 MasterCard/Visa 标志了，甚至香港的银行卡也默认使用银联，出国旅游要刷 UnionPay 跨境支付非常方便。

8.3. 跨境汇款

跨境汇款你别想像银联一样，填写一个姓名一个账号，点击一下转账按钮这样的操作。每个国家的政策也不同，政策不允许这样操作。

跨境只能汇款，无法转账，又回到了电汇时代，只是不再主要纸质的汇票了



跨境汇款必须依赖 SWIFT 系统，由于我国的政策问题，个人很少涉及跨境业务，所以多数人对 SWIFT 不是很了解。 如果你在香港开一张银行卡例如汇丰银行，拿到银行给的信封里，就会有一个 SWIFT 码。

你会遇到这个问题，无法输入对方的名字，例如：

```
Nickname：netkiller
English name：Neo chen
Nippon name：ちんけいほう (音訳)
Korean name：천징봉
Thailand name：ภูมิภาพภูเขา
Vietnam：Trần Cảnh Phong
```

所以就需要每个银行一个代码，每个账号一个代码，由于全世界银行太多，银行系统各种各样，每个国家的语言也不同，难以达成一致，联合国也没有能力统一标准。新建一套系统不太可能，所以80年代的标准仍然沿用至今。

使用 SWIFT 面临的问题

- 网络速度慢

- 手续费高
- 技术落后
- 不能实时到账
- 脆弱容易被攻击

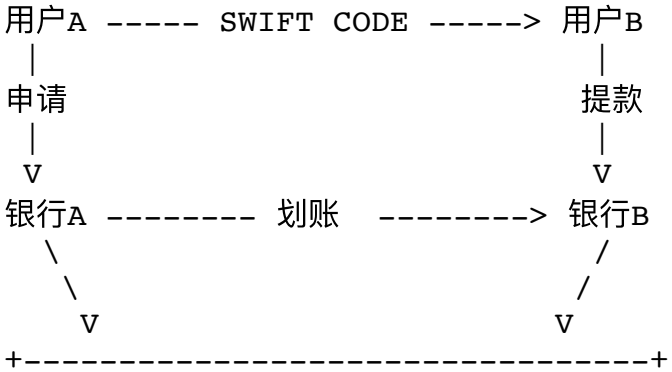
SWIFT的诞生甚至早于数字时代，可以说是目前最好的跨境通讯和交易系统，但它的确需要与时俱进。

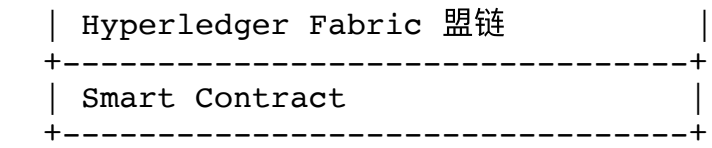
牵头做一个世界银联不太可能，世界各国银行无法想政府一样，一个红头文件，下面招办，行政手段推动。且业务差异大，系统复杂超乎想象，这个中心数据库谁来管理呢?

SWIFT早就意识到了这些问题，并宣布进军区块链，同时加入超级账本项目（Hyperledger Project）成为会员。可以肯定下一个版本的SWIFT灰使用区块链技术，一步一步逐渐取代就系统。

8.4. 区块链能做什么

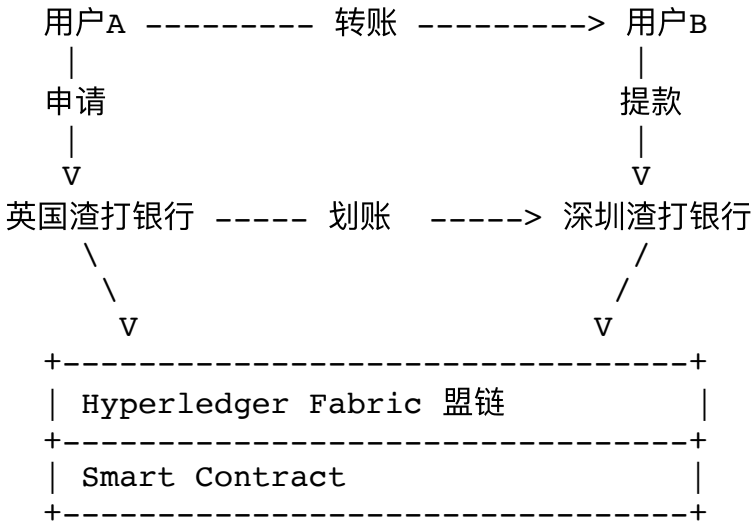
区块链可以解决银行哪些痛点，先说说 SWIFT 2.0 （区块链SWIFT）我想SWIFT仍然会兼容现有的协议。SWIFT CODE协议仍然会保留。短时间不可能被取代SWIFT CODE因为这个体系已经使用很多年。





后端将会被区块链取代

另外银行的跨国业务将会走自己的区块链，不再依赖 SWIFT， 因为费用等问题。



8.5. 智能合约怎么落地

我对银行业务实在不了解，这里只能设想一下场景。下面是我之前写的一个Token合约，我将它应用到这个场景中

```
package main

/*
------------------------------------------------
Author: netkiller <netkiller@msn.com>
```

```
Home: http://www.netkiller.cn
Data: 2018-03-20 11:00 PM
-------------------------------------------------
CORE_PEER_ADDRESS=peer:7051 CORE_CHAINCODE_ID_NAME=token3:1.0
chaincode/token/token3
peer chaincode install -n token3 -v 1.0 -p
chaincodedev/chaincode/token
peer chaincode instantiate -C myc -n token3 -v 1.0 -c '{"Args":
[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
peer chaincode invoke -C myc -n token3 -c
'{"function":"createAccount","Args":["coinbase"]}'
peer chaincode invoke -C myc -n token3 -v 1.0 -c
'{"function":"showAccount","Args":["coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"balanceAll","Args":["coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":["Netkiller
Token","NKC","1000000","coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":["NEO
Token","NEC","1000000","coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"setLock","Args":["true"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"setLock","Args":["false"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"mintToken","Args":["NKC","5000","coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"createAccount","Args":["netkiller"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"transferToken","Args":
["coinbase","netkiller","NKC","100"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"balance","Args":["netkiller","NKC"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"frozenAccount","Args":["netkiller","true"]}'
-------------------------------------------------
*/

import (
        "encoding/json"
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)
```

```
type Msg struct{
        Status  bool    `json:"Status"`
        Code    int             `json:"Code"`
        Message string  `json:"Message"`
}

type Currency struct{
        TokenName               string  `json:"TokenName"`
        TokenSymbol     string  `json:"TokenSymbol"`
        TotalSupply     float64 `json:"TotalSupply"`
}

type Token struct {
        Lock            bool    `json:"Lock"`
        Currency        map[string]Currency
`json:"Currency"`
}

func (token *Token) transfer (_from *Account, _to *Account,
_currency string, _value float64) []byte{

        var rev []byte
        if (token.Lock){
                msg := &Msg{Status: false, Code: 0, Message: "锁
仓状态，停止一切转账活动"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if(_from.Frozen ) {
                msg := &Msg{Status: false, Code: 0, Message:
"From 账号冻结"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if( _to.Frozen) {
                msg := &Msg{Status: false, Code: 0, Message: "To
账号冻结"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if(!token.isCurrency(_currency)){
                msg := &Msg{Status: false, Code: 0, Message: "货
币符号不存在"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if(_from.BalanceOf[_currency] >= _value){
```

```
                _from.BalanceOf[_currency] -= _value;
                _to.BalanceOf[_currency] += _value;

                msg := &Msg{Status: true, Code: 0, Message: "转账
成功"}
                rev, _ = json.Marshal(msg)
                return rev
        }else{
                msg := &Msg{Status: false, Code: 0, Message: "余
额不足"}
                rev, _ = json.Marshal(msg)
                return rev
        }

}
func (token *Token) initialSupply(_name string, _symbol string,
_supply float64, _account *Account) []byte{
        if _,ok := token.Currency[_symbol]; ok {
                msg := &Msg{Status: false, Code: 0, Message: "代
币已经存在"}
                rev, _ := json.Marshal(msg)
                return rev
        }

        if _account.BalanceOf[_symbol] > 0 {
                msg := &Msg{Status: false, Code: 0, Message: "账
号中存在代币"}
                rev, _ := json.Marshal(msg)
                return rev
        }else{
                token.Currency[_symbol] = Currency{TokenName:
_name, TokenSymbol: _symbol, TotalSupply: _supply}
                _account.BalanceOf[_symbol] = _supply

                msg := &Msg{Status: true, Code: 0, Message: "代币
初始化成功"}
                rev, _ := json.Marshal(msg)
                return rev
        }

}

func (token *Token) mint(_currency string, _amount float64,
_account *Account) []byte{
        if(!token.isCurrency(_currency)){
                msg := &Msg{Status: false, Code: 0, Message: "货
```

```
币符号不存在"}
                rev, _ := json.Marshal(msg)
                return rev
        }
        cur := token.Currency[_currency]
        cur.TotalSupply += _amount;
        token.Currency[_currency] = cur
        _account.BalanceOf[_currency] += _amount;

        msg := &Msg{Status: true, Code: 0, Message: "代币增发成
功"}
        rev, _ := json.Marshal(msg)
        return rev

}
func (token *Token) burn(_currency string, _amount float64,
_account *Account) []byte{
        if(!token.isCurrency(_currency)){
                msg := &Msg{Status: false, Code: 0, Message: "货
币符号不存在"}
                rev, _ := json.Marshal(msg)
                return rev
        }
        if(token.Currency[_currency].TotalSupply >= _amount){
                cur := token.Currency[_currency]
                cur.TotalSupply -= _amount;
                token.Currency[_currency] = cur
                _account.BalanceOf[_currency] -= _amount;

                msg := &Msg{Status: false, Code: 0, Message: "代
币回收成功"}
                rev, _ := json.Marshal(msg)
                return rev
        }else{
                msg := &Msg{Status: false, Code: 0, Message: "代
币回收失败，回收额度不足"}
                rev, _ := json.Marshal(msg)
                return rev
        }

}
func (token *Token) isCurrency(_currency string) bool {
        if _, ok := token.Currency[_currency]; ok {
                return true
        }else{
                return false
```

```
        }
}
func (token *Token) setLock(_look bool) bool {
        token.Lock = _look
        return token.Lock
}
type Account struct {
        Name                    string  `json:"Name"`
        Frozen                  bool    `json:"Frozen"`
        BalanceOf               map[string]float64
`json:"BalanceOf"`
}
func (account *Account) balance (_currency string)
map[string]float64{
        bal     :=
map[string]float64{_currency:account.BalanceOf[_currency]}
        return bal
}

func (account *Account) balanceAll() map[string]float64{
        return account.BalanceOf
}

// -----------
const TokenKey = "Token"

// Define the Smart Contract structure
type SmartContract struct {

}

func (s *SmartContract) Init(stub shim.ChaincodeStubInterface)
pb.Response {

        token := &Token{Currency: map[string]Currency{}}

        tokenAsBytes, err := json.Marshal(token)
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Init Token %s \n",
string(tokenAsBytes))
        }
        return shim.Success(nil)
}

func (s *SmartContract) Query(stub shim.ChaincodeStubInterface)
```

```
pb.Response {
        function, args := stub.GetFunctionAndParameters()
        if function == "balance" {
                return s.balance(stub, args)
        } else if function == "balanceAll" {
                return s.balanceAll(stub, args)
        } else if function == "showAccount" {
                return s.showAccount(stub, args)
        }
        return shim.Error("Invalid Smart Contract function
name.")
}

func (s *SmartContract) Invoke(stub shim.ChaincodeStubInterface)
pb.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to interact
with the ledger appropriately
        if function == "initLedger" {
                return s.initLedger(stub, args)
        } else if function == "createAccount" {
                return s.createAccount(stub, args)
        } else if function == "initCurrency" {
                return s.initCurrency(stub, args)
        } else if function == "setLock" {
                return s.setLock(stub, args)
        } else if function == "transferToken" {
                return s.transferToken(stub, args)
        } else if function == "frozenAccount" {
                return s.frozenAccount(stub, args)
        } else if function == "mintToken" {
                return s.mintToken(stub, args)
        } else if function == "balance" {
                return s.balance(stub, args)
        } else if function == "balanceAll" {
                return s.balanceAll(stub, args)
        } else if function == "showAccount" {
                return s.showAccount(stub, args)
        } else if function == "showToken" {
                return s.showToken(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}
```

```
func (s *SmartContract) createAccount(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        key  := args[0]
        name  := args[0]
        existAsBytes,err := stub.GetState(key)
        fmt.Printf("GetState(%s) %s \n", key,
string(existAsBytes))
        if string(existAsBytes) != "" {
                fmt.Println("Failed to create account, Duplicate
key.")
                return shim.Error("Failed to create account,
Duplicate key.")
        }

        account := Account{
                Name: name,
                Frozen: false,
                BalanceOf: map[string]float64{}}

        accountAsBytes, _ := json.Marshal(account)
        err = stub.PutState(key, accountAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("createAccount %s \n",
string(accountAsBytes))

        return shim.Success(accountAsBytes)
}
func (s *SmartContract) initLedger(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        return shim.Success(nil)
}
func (s *SmartContract) showToken(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("GetState(%s)) %s \n", TokenKey,
string(tokenAsBytes))
```

```
        }
        return shim.Success(tokenAsBytes)
}

func (s *SmartContract) initCurrency(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        if len(args) != 4 {
                return shim.Error("Incorrect number of
arguments. Expecting 4")
        }

        _name  := args[0]
        _symbol:= args[1]
        _supply,_:= strconv.ParseFloat(args[2], 64)
        _account := args[3]

        coinbaseAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Coinbase before %s \n",
string(coinbaseAsBytes))

        coinbase := &Account{}

        json.Unmarshal(coinbaseAsBytes, &coinbase)

        token := Token{}
        existAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("GetState(%s)) %s \n", TokenKey,
string(existAsBytes))
        }
        json.Unmarshal(existAsBytes, &token)

        result := token.initialSupply(_name,_symbol,_supply,
coinbase)

        tokenAsBytes, _ := json.Marshal(token)
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Init Token %s \n",
string(tokenAsBytes))
        }
```

```
        coinbaseAsBytes, _ = json.Marshal(coinbase)
        err = stub.PutState(_account, coinbaseAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Coinbase after %s \n",
string(coinbaseAsBytes))



        return shim.Success(result)
}

func (s *SmartContract) transferToken(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 4 {
                return shim.Error("Incorrect number of
arguments. Expecting 4")
        }
        _from           := args[0]
        _to                     := args[1]
        _currency       := args[2]
        _amount,_       := strconv.ParseFloat(args[3], 32)

        if(_amount <= 0){
                return shim.Error("Incorrect number of amount")
        }

        fromAsBytes,err := stub.GetState(_from)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("fromAccount %s \n", string(fromAsBytes))
        fromAccount := &Account{}
        json.Unmarshal(fromAsBytes, &fromAccount)

        toAsBytes,err := stub.GetState(_to)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("toAccount %s \n", string(toAsBytes))
        toAccount := &Account{}
        json.Unmarshal(toAsBytes, &toAccount)

        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
```

```
                return shim.Error(err.Error())
        }
        fmt.Printf("Token %s \n", string(toAsBytes))
        token := Token{Currency: map[string]Currency{}}
        json.Unmarshal(tokenAsBytes, &token)

        result := token.transfer(fromAccount, toAccount,
_currency, _amount)
        fmt.Printf("Result %s \n", string(result))

        fromAsBytes, err = json.Marshal(fromAccount)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(_from, fromAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("fromAccount %s \n",
string(fromAsBytes))
        }

        toAsBytes, err = json.Marshal(toAccount)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(_to, toAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("toAccount %s \n", string(toAsBytes))
        }

        return shim.Success(result)
}
func (s *SmartContract) mintToken(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 3 {
                return shim.Error("Incorrect number of
arguments. Expecting 3")
        }
        _currency       := args[0]
        _amount,_       := strconv.ParseFloat(args[1], 32)
        _account        := args[2]

        coinbaseAsBytes,err := stub.GetState(_account)
        if err != nil {
```

```
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Coinbase before %s \n",
string(coinbaseAsBytes))
        }

        coinbase := &Account{}
        json.Unmarshal(coinbaseAsBytes, &coinbase)

        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Token before %s \n", string(tokenAsBytes))

        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)

        result := token.mint(_currency, _amount, coinbase)

        tokenAsBytes, err = json.Marshal(token)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Token after %s \n", string(tokenAsBytes))

        coinbaseAsBytes, _ = json.Marshal(coinbase)
        err = stub.PutState(_account, coinbaseAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Coinbase after %s \n",
string(coinbaseAsBytes))
        }

        fmt.Printf("mintToken %s \n", string(tokenAsBytes))

        return shim.Success(result)
}

func (s *SmartContract) setLock(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
```

```
        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _look := args[0]

        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        // fmt.Printf("setLock - begin %s \n",
string(tokenAsBytes))

        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)

        if(_look == "true"){
                token.setLock(true)
        }else{
                token.setLock(false)
        }

        tokenAsBytes, err = json.Marshal(token)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("setLock - end %s \n", string(tokenAsBytes))

        return shim.Success(nil)
}
func (s *SmartContract) frozenAccount(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _account        := args[0]
        _status         := args[1]

        accountAsBytes,err := stub.GetState(_account)
```

```
        if err != nil {
                return shim.Error(err.Error())
        }
        // fmt.Printf("setLock - begin %s \n",
string(tokenAsBytes))

        account := Account{}

        json.Unmarshal(accountAsBytes, &account)

        var status bool
        if(_status == "true"){
                status = true;
        }else{
                status = false
        }

        account.Frozen = status

        accountAsBytes, err = json.Marshal(account)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(_account, accountAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("frozenAccount - end %s \n",
string(accountAsBytes))
        }

        return shim.Success(nil)
}

func (s *SmartContract) showAccount(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }
        _account        := args[0]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Account balance %s \n",
```

```
string(accountAsBytes))
        }
        return shim.Success(accountAsBytes)
}

func (s *SmartContract) balance(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }
        _account        := args[0]
        _currency       := args[1]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Account balance %s \n",
string(accountAsBytes))
        }

        account := Account{}
        json.Unmarshal(accountAsBytes, &account)
        result := account.balance(_currency)

        resultAsBytes, _ := json.Marshal(result)
        fmt.Printf("%s balance is %s \n", _account,
string(resultAsBytes))

        return shim.Success(resultAsBytes)
}

func (s *SmartContract) balanceAll(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }
        _account        := args[0]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Account balance %s \n",
```

```
string(accountAsBytes))
        }

        account := Account{}
        json.Unmarshal(accountAsBytes, &account)
        result := account.balanceAll()
        resultAsBytes, _ := json.Marshal(result)
        fmt.Printf("%s balance is %s \n", _account,
string(resultAsBytes))

        return shim.Success(resultAsBytes)
}

// The main function is only relevant in unit test mode. Only
included here for completeness.
func main() {

        // Create a new Smart Contract
        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
}
```

部署链码，然后实例化链码

```
peer chaincode install -n token3 -v 1.0 -p
chaincodedev/chaincode/token
peer chaincode instantiate -C myc -n token3 -v 1.0 -c '{"Args":
[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
```

首先初始化账号，需要将现有账号同步到区块链上，这是上链操作。账号分为两种，一个是 coinbase 银行的总账号，另外是用户账号

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"createAccount","Args":["coinbase"]}'
```

初始化外币，银行有多少外币存量，初始化到 coinbase 账号

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":
["Chain","RMB","1000000000","coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":
["Japan","JPY","1000000000","coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":
["USA","USD","1000000000","coinbase"]}'
```

为用户创建账号

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"createAccount","Args":["netkiller"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"createAccount","Args":["neo"]}'
```

同步用户账号中的外币

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"transferToken","Args":
["coinbase","netkiller","RMB","10000"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"transferToken","Args":
["coinbase","netkiller","USD","100"]}'

peer chaincode invoke -C myc -n token3 -c
'{"function":"transferToken","Args":
["coinbase","neo","RMB","10000"]}'
peer chaincode invoke -C myc -n token3 -c
```

```
'{"function":"transferToken","Args":
["coinbase","neo","USD","100"]}'
```

现在区块链上的用户资金已经跟数据库中的资金同步。

现在netkiller这个用户需要给neo转账 50 美金

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"transferToken","Args":
["netkiller","neo","USD","50"]}'
```

现在 netkiller 账号中又 50美金，neo 账号中有 150美金。

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"balanceAll","Args":["netkiller"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"balanceAll","Args":["neo"]}'
```

如果 neo 账号被法院申请冻结了，怎么办？可以使用下面方法设置冻结账号

```
peer chaincode invoke -C myc -n token3 -c
'{"function":"frozenAccount","Args":["neo","true"]}'
```

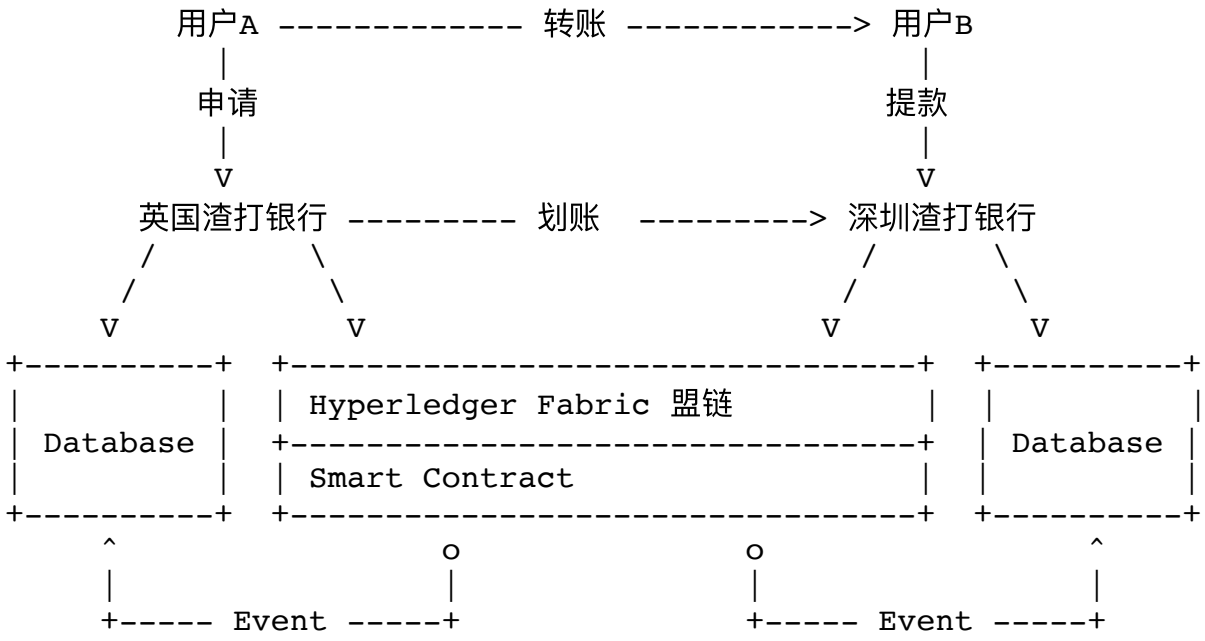
此时 neo 账号被冻结，资金无法转入，转出。

8.6. 总结

以上仅仅是提供一个区块链学历案例，作者也设想了一个场景。但是真是情况怎样作者也不清楚。

上面通正合约没有考虑到汇率和中间价问题，需要进一步完善。

另外还需要一个守护进程订阅 Event 状态，每做一笔转账交易，event 会受到然后将这个操作同步到中心化数据库。



9. 区块链医院应用探索

9.1. 背景

当今医院面临的问题很多，产生很多医疗事故和医疗纠纷。问题主要体现在，医疗信息不透明，药品监管不力，电子病历不能共享，医疗影像资料一次性消费。

各医院的医院信息系统 HIS(Hospital Information System) 在一个局域网中运行，无法信息共享

还有药品和器械定价，采购，流通，溯源都需要监管。

在区块链没有出现之前，中心化的数据库无法解决这些问题，为什么呢？，谁来建设这个中心库，谁来管理，怎样使用，产生的费用谁来承担，等等无法达成一致。

区块链的出现迎刃而解，区块链是分布式共识数据库，加入医院盟链的医院都有各自的的节点，首先是公平性，都能达成一致。其次是安全性所有医院的节点达成共识后，才能存储数据。

最终实现医疗和健康信息共享，医疗流程透明化，医疗事故责任可追溯.....

9.2. 药品和器械上链

9.2.1. 药品上链

使用区块链技术建立一个药品数据库。

将世面上所有销售的药品上链，上链内容包含，药品名称，说明书，价格，图片.....

药品的生产，质检，物流等等溯源信息上链

药品数据库，价格透明

9.2.2. 器械上链

重复使用的器械，我们需要将使用过程上链，例如什么时间，什么地点，谁使用过，事后消毒过程等等

一次性使用的器械，一品一码对一次性使用的器械信息上链，使用状态，是否销毁等等。

9.3. 电子病历上链

目前的病历掌握在各个医院手上的，患者自己并不掌握，所以病人就没有办法获得自己的医疗记录和历史情况，这对患者就医会造成很大的困扰，因为医生无法详尽了解到你的病史记录。

虽然各医院的病例已经电子化，但是医院与医院之间的是不共享的。如果病人需要转院，因为医院因为隐私原因是不能直接把病历给转院的那家的，那么需要在当前医院把病历取出来，如果还需要保险报销的话，那么这个会更为麻烦。

建立医院盟链，病历上链一定是未来的趋势，通过区块链共享病例，能为患者和医生带来便利。在我们就医的过程中，如果医生能够获得更多的病例信息，有利于病情诊断。

病例一旦上链，电子病例不可篡改

9.3.1. 医学影像上链

例如 X 光片，CT，超声波等等医学影像资料，目前的现状是一次性使用，看一次病拍一次，看完丢掉，下次再拍。除非患者有意保存，否则医生很难看到一个时间段内的医学影像资料。

所以医学影像资料上链也很重要

9.4. 健康信息

另外现在又很多穿戴设备可以记录脉搏、运动数据、例如跑步，健身等等数据。

9.5. 出生证明

新生儿出生证明

9.6. 保险

9.6.1. 保险信息上链

就诊过程中医生经常询问患者，是否购买了商业保险以及社会保险，这样医生可以根据患者的承受能力为患者提供可承受的治疗方案。

9.6.2. 区块链解决出险理赔过程

传统出险，需要患者保存好单据，填写好出险单，提交病历，收据等资料交给保险公司。保险公司收到单据后审核，如果没有问题转账到患者银行卡上。

保险公司加入到盟链中，可以直接访问患者病历，无需患者再提交纸质单据。通过智能合约完成整个理赔过程，也降低了患者理赔操作难度。

9.7. 智能合约

暂无时间写

10. 艺术品区块链溯源防伪平台

10.1. 都有哪些角色参与其中

参与艺术品上链，鉴定，交易包含了下面几种角色。

平台至少有三种角色会

- 用户
- 机构
- 鉴定师
- 艺术家

所以我们需要为不同的角色提供不同的App应用。

用户端：功能包括防伪查询，链上资产的浏览，权益转让，资产拍卖，资产抵押，社区互动，分享，数字资产行情，钱包等等

机构端：负责信息收集，信息整理，数据提交，数据审查，资产上链，资产划拨等等

鉴定师：负责数字资产的鉴定，需要有相关资质。

10.2. 需要运用的防伪技术

防伪溯源涉及的技术栈

- 纸纹防伪（PaperPrint），纸纹防伪即纸纹防伪技术。它是一种基于提取和识别每张纸与生俱来的、独一无二且无法仿造、克隆的自然纤维纹理作为防伪特征（即纸纹）来实现防伪的新型防伪技术。

- 荧光防伪油墨，使用荧光油墨印刷技术，在特定波长的紫外线或者红外线下才能看到。荧光二维码，荧光印章，荧光指纹，荧光暗记，布满整个宣纸的荧光图案。我们可以为每个艺术家定制带有荧光图案的专属纸张。
- DNA防伪,将艺术家的血液滴在书画上或者头发夹在宣纸中间。据说某中世纪著名画家将自己的精液和油画颜料混合：）
- 特征识别防伪，类似我们手机的面部识别，记录物品的特征，例如使用电子显微镜平射纸文理，画面局部等等。油画还可以拍摄X光片。
- 3D 建模扫描，例如3D扫描仪，将物品的3D数据记录下来。
- 激光内雕，例如施华洛世奇将 logo 内雕在他的水晶制品中。
- QRcode 二维码，用于链上数据查询
- NFC(Near Field Communication) 有两个作用，一可以存储数据，二用来防伪，因为生产相同UID的NFC芯片难度极大，门槛很高。
- RFID(Radio Frequency Identification) 是 NFC 的一种，区别是不能存储数据，NFC 不能替代 RFID，RFID 可以实现资产盘点，以及安防。
- GPS 定位与地图，记录用户位置，资产位置，机构位置，鉴定师位置，还能实现次产跟踪，例如两次查询资产的GPS坐标，不在安全范围，将视为被盗，系统将通知机构或用户。
- 高清相机、高清视频设备
- 安防设备，CCTV监控，门禁等等

举例，书画怎样做防伪

以国画为例，我们首先找一个宣纸企业合作，在宣纸生产过程中，将NFC芯片夹在宣纸中间，然后我们使用荧光油墨在宣纸上印刷防伪图案。也可以采用人民币上的金属丝方案，最后拍摄纸纹。视频记录下艺术家的整个作画过程，然后再拍摄特征数据。最后使用激光打孔技

术，在宣纸上打出防伪图案数据，针孔极小肉眼难以分辨，每个防伪数据都是唯一的。

NFC 标签可以使用易碎纸粘贴在艺术品上，缺点是寿命较短。使用PVC材料又容易撕下。

油画可以讲NFC芯片放在纺织布中，使用堆彩技术的画家可以讲NFC芯片覆盖（埋植）在颜料下面

10.3. 技术架构

10.3.1. 前端技术

由于 H5 技术无法满足我们的需求，例如相机，麦克风，NFC，定位...等等。我们重点放在 App 开发，H5 紧紧用于官网，区块链浏览器，资讯，等等。

由于使用了很多手机上技术，原生App更适合，而混合开发 React Native，Vue.js, Flutter 不在我们选择之列。

微信小程序可以考虑，但是如果涉及 Token 可能随时会被下架。

10.3.2. 微服务端

服务端设计为可以水平扩展，可以随时根据用户量，扩展服务器规模。

Nginx 负载均衡，HTTP2 （安卓 Okhttp 已经很好的支持 http2）

框架采用 Spring cloud

数据库开发使用 JPA

接口认证 Oauth2 + Jwt

10.3.3. 存储层

MongoDB

Redis

10.3.4. 消息队列层

Kafka

10.3.5. 搜索层

ELK(ElasticSearch, Logstash, Kibana)

搜索是非常重要的功能，因为区块链只能通过 hash 值取出链上的数据，虽然 Hyperledger Fabrc 在数据使用 CouchDB 时提供了 World State 的一些高级搜索功能，但是仍然不能满足我们的需求。

所以链上数据需要存储一份在搜索引擎中，搜索引擎的分词功能，可以提供快速精准的搜索服务。

搜索引擎的工作流程是:

```
User --> Phone App --> Nginx --> Spring cloud --> Elastsearch --> Hyperledger Fabric
```

10.3.6. 区块链

我们不做山寨链，我发现很多国内企业热衷于做山寨链，什么事山寨链呢，就是在现有的区块链(Ethereum, Hyperledger Fabric 或 EOS) 的基础上二次开发，首先开发山寨链需要大量的资金人力，私链是没有任何意义的，没有公信力。即使目前的现有区块链无法满足我们的需求，可以通过架构调节去适应他。

所以我们只用最成熟的产品：

Hyperledger Fabric 盟链：主要用于资产上链，链上资产查询

Ethereum 公链：用于 Token ，由于 Hyperledger Fabric 无法实现 Token，所以我们仍然需要以太坊。（作者写过一篇文章关于为什么 Hyperledger Fabric 不能实现 Token，请兴趣自己在网上搜索）

EOS Token/资产上链：由于在我设计这个系统之时 EOS 还没有 Release 所以当时没有考虑 EOS。现在我们可以使用 EOS，甚至替换掉 Hyperledger Fabrc + Ethereum 方案。因为 EOS 即能实现资产上链，也能实现 Token。 这里我们将资产也在 EOS 上链一份，同时也支持 EOS 发的 Token。

IPFS 星际文件系统：用于存储多媒体数据，例如图片，视频。（注意：IPFS 暂时不支持流媒体，我的解决方案是上链同事复制一份到 nginx 中，并开启 mp4 流媒体功能）

注：虽然以太坊目前尝尝拥堵，但是很多应用场景仍是不可替代的。

10.3.7. 支持层

监控 Zabbix

10.4. RFID/NFC

10.4.1. RFID

RFID基本概念:

RFID(Radio Frequency Identification)的缩写，即射频识别，俗称电子标签。 RFID射频识别是一种非接触式的自动识别技术，它通过射频信号自动识别目标对象并获取相关数据，识别工作无须人工干预，可工作于各种恶劣环境。 RFID是一种简单的无线系统，只有两个基本器件，该系统用于控制、检测和跟踪物体。系统由一个询问器(或阅读器)和很多应答器(或标签)组成。

RFID包括： 低频125KHz 主要是动物管理 中频 一般指433MHz（这个频段一般也是有源的 也有做高速收费） 高频13.56MHz 公交卡 身份证都是这个频段。 超高频860-960MHz 主要用在物流和停车场管理。 微波2.45GHz ETC用这个频段的多

10.4.2. NFC

NFC基本概念:

NFC(Near Field Communication)缩写，即近距离无线通讯技术。由飞利浦公司和索尼公司共同开发的一项无线技术。NFC由非接触式射频识别及互联互通技术整合演变而来，可以在移动设备、消费类电子产品、PC和智能控件工具间进行近距离无线通信。NFC提供了一种简单、触控式的解决方案，可以让消费者简单直观地交换信息、访问内容与服务。 NFC技术特点： 1、 在13.56MHz频率运行距离在20公分内； 2、 传输速度可分106Kbits/sec，212 Kbits/sec，424 Kbits/sec； 3、 运作可分主动与被动模式。主动模式需使用电池，也需要独立发射模组；被动模式不需使用电池，但无法独立发射讯号； 4、 已成为ISO/IEC IS 18092国家标准、ETSI TS 102 190标准、EMCA-340标准。

目前主流手机都带有 NFC 是近场通信功能，安卓手机对 NFC 方案全开放，苹果手机暂时开放部分功能.

10.4.3. RFID/NFC 两种技术的差异

RFID/NFC 比较

- 距离，RFID远，NFC进
- RIFD由读卡器和标签组成，读卡器只能读取标签上的数据。NFC既可以做读卡器，也能提供标签服务，还能实现P2P点对点传输数据。
- 修改，RFID是只读的，NFC上的数据可以修改，例如公交卡
-

10.5. 资产投资与份额持有

传统艺术品投资门槛非常高，一是用户不知道从哪些渠道可以投资，二是艺术品价值过高，三是艺术品鉴定难。这导致了投资艺术品门槛过高。 Token 能实现份额化，实现人人参与，人人持有，P2P交易。

例如某机构上链一件艺术品，用户可以投资该艺术品的一定份额，可以转让他持有的权益。且交易去中心化，不受任何机构，管理者的制约。

下面的合约可以展示如何分割艺术品份额，最终达到链上资产的份额分割和持有与交易。

```
pragma solidity ^0.4.25;

/**
 * @title SafeMath
 * @dev Math operations with safety checks that revert on error
 */
library SafeMath {

  /**
  * @dev Multiplies two numbers, reverts on overflow.
  */
```

```
  function mul(uint256 a, uint256 b) internal pure returns
(uint256) {
    // Gas optimization: this is cheaper than requiring 'a' not
being zero, but the
    // benefit is lost if 'b' is also tested.
    // See: https://github.com/OpenZeppelin/openzeppelin-
solidity/pull/522
    if (a == 0) {
      return 0;
    }

    uint256 c = a * b;
    require(c / a == b);

    return c;
  }

  /**
  * @dev Integer division of two numbers truncating the
quotient, reverts on division by zero.
  */
  function div(uint256 a, uint256 b) internal pure returns
(uint256) {
    require(b > 0); // Solidity only automatically asserts when
dividing by 0
    uint256 c = a / b;
    // assert(a == b * c + a % b); // There is no case in which
this doesn't hold

    return c;
  }

  /**
  * @dev Subtracts two numbers, reverts on overflow (i.e. if
subtrahend is greater than minuend).
  */
  function sub(uint256 a, uint256 b) internal pure returns
(uint256) {
    require(b <= a);
    uint256 c = a - b;

    return c;
  }

  /**
  * @dev Adds two numbers, reverts on overflow.
  */
  function add(uint256 a, uint256 b) internal pure returns
```

```
(uint256) {
    uint256 c = a + b;
    require(c >= a);

    return c;
  }

  /**
  * @dev Divides two numbers and returns the remainder
(unsigned integer modulo),
  * reverts when dividing by zero.
  */
  function mod(uint256 a, uint256 b) internal pure returns
(uint256) {
    require(b != 0);
    return a % b;
  }
}

contract Ownable {

    address public owner;

    event OwnershipTransferred(address indexed previousOwner,
address indexed newOwner);

    constructor() public {
        owner = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == owner);
        _;
    }

    function transferOwnership(address newOwner) public
onlyOwner {
        require(newOwner != address(0));
        emit OwnershipTransferred(owner, newOwner);
        owner = newOwner;
    }

}

contract NetkillerAssetsToken is Ownable {
    using SafeMath for uint256;

    string public name;
```

```
    string public symbol;
    uint public decimals;
    uint256 public totalSupply;

    mapping(address => mapping(string => uint256)) internal
balances;
    mapping(string => address) internal tokens;

    event Transfer(address indexed _from, address indexed _to,
string indexed _tokenId);
    event Burn(address indexed from, string _tokenId);

    constructor(
        string tokenName,
        string tokenSymbol,
        uint decimalUnits
    ) public {
        owner = msg.sender;
        name = tokenName;
        symbol = tokenSymbol;
        decimals = decimalUnits;
        totalSupply = 0;
    }

    function add(address _owner, string _tokenId) onlyOwner
returns(bool status){
        balances[_owner][_tokenId] = 100 * 10 **
uint256(decimals);
        tokens[_tokenId] = _owner;
        totalSupply = totalSupply.add(1);
        return true;
    }

    function balanceOf(address _owner, string _tokenId)
constant returns(uint balance){
        return balances[_owner][_tokenId];
    }

    function ownerOf(string _tokenId) constant returns (address
owner) {
        return tokens[_tokenId];
    }

    function transfer(address _to, string _tokenId){

        address _from = msg.sender;
        uint256 amount = balances[_from][_tokenId];
        transfer(_to, amount, _tokenId);
```

```
    }
    function transfer(address _to, uint256 _value, string
_tokenId){
        require(msg.sender == ownerOf(_tokenId));
        require(msg.sender != _to);
        require(_to != address(0));

        address _from = msg.sender;
        uint256 amount = balances[_from][_tokenId];
        require(amount >= _value);

        balances[_from][_tokenId] = balances[_from]
[_tokenId].sub(_value);
        balances[_to][_tokenId] = balances[_to]
[_tokenId].add(_value);
        tokens[_tokenId] = _to;

        emit Transfer(_from, _to, _tokenId);
    }

    function burn(address _owner, string _tokenId) onlyOwner
public returns (bool success) {
        require(balances[_owner][_tokenId] > 0 &&
balances[_owner][_tokenId] == 100 * 10 ** uint256(decimals));

        balances[_owner][_tokenId] = 0;
        tokens[_tokenId] = address(0);

        totalSupply = totalSupply.sub(1);
        emit Burn(msg.sender, _tokenId);
        return true;
    }

}
```

由于 ERC721 不太符合我的需求，所以我结合 ERC20 和 ERC721 写出了我的合约。合约尽量保持了ERC20的使用习惯，函数定义尽量兼容ERC20。

我们来看下面的构造方法，每个种类的物品一个合约，例如字画，陶瓷，青铜器。

```
    constructor(
        string tokenName,
        string tokenSymbol,
        uint decimalUnits
    ) public {
        owner = msg.sender;
        name = tokenName;
        symbol = tokenSymbol;
        decimals = decimalUnits;
        totalSupply = 0;
    }
```

通过下面函数，添加资产到 Token，使链上资产与Token绑定。

```
    function add(address _owner, string _tokenId) onlyOwner
returns(bool status){
        balances[_owner][_tokenId] = 100 * 10 **
uint256(decimals);
        tokens[_tokenId] = _owner;
        totalSupply = totalSupply.add(1);
        return true;
    }
```

balances[_owner][_tokenId] = 100 * 10 ** uint256(decimals); 初始化份额是 100 表示 100%

totalSupply = totalSupply.add(1); 物品件数加一。可以用于统计链上资产的数量。

下面函数是查询资产的持有人

```
function ownerOf(string _tokenId) constant returns (address
owner) {
        return tokens[_tokenId];
```

```
    }
```

下面函数是，权益转让和权益份额转让。

```
    function transfer(address _to, string _tokenId){

        address _from = msg.sender;
        uint256 amount = balances[_from][_tokenId];
        transfer(_to, amount, _tokenId);
    }
    function transfer(address _to, uint256 _value, string
_tokenId){
        require(msg.sender == ownerOf(_tokenId));
        require(msg.sender != _to);
        require(_to != address(0));

        address _from = msg.sender;
        uint256 amount = balances[_from][_tokenId];
        require(amount >= _value);

        balances[_from][_tokenId] = balances[_from]
[_tokenId].sub(_value);
        balances[_to][_tokenId] = balances[_to]
[_tokenId].add(_value);
        tokens[_tokenId] = _to;

        emit Transfer(_from, _to, _tokenId);
    }
```

接下来，我们就是可以开发 Dapp 钱包了，在钱包中实现资产的转移交易。

这个合约可以移植到 EOS 上，Hyperledger Fabric 不可以，因为 Fabric 没有锁的机制，会导致计算出错。

10.6. 资产上链的

我们希望资产上链适用于任何领域，后面也方便将业务拓展。所以我实现了一个万能合约。以不变应万变，Hyperledger Fabic 链码如下。

```
package main

import (
        "fmt"
        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)

type SmartContract struct {}

func (s *SmartContract) Init(stub shim.ChaincodeStubInterface)
pb.Response {
        return shim.Success(nil)
}

func (s *SmartContract) Query(stub shim.ChaincodeStubInterface)
pb.Response {
        return shim.Success(nil)
}

func (s *SmartContract) Invoke(stub
shim.ChaincodeStubInterface) pb.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to
interact with the ledger appropriately
        if function == "create" {
                return s.create(stub, args)
        } else if function == "find" {
                return s.find(stub, args)
        } else if function == "update" {
                return s.update(stub, args)
        } else if function == "delete" {
                return s.delete(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}
```

```
func (s *SmartContract) create(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _key    := args[0]
        _data   := args[1]

        if(_data == ""){
                return shim.Error("Incorrect string of data")
        }

        existAsBytes,err := stub.GetState(_key)
        if string(existAsBytes) != "" {
                fmt.Println("Failed to create account,
Duplicate key.")
                return shim.Error("Failed to create account,
Duplicate key.")
        }

        err = stub.PutState(_key, []byte(_data))
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("create %s %s \n", _key, string(_data))

        return shim.Success(nil)
}

func (s *SmartContract) find(stub shim.ChaincodeStubInterface,
args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        _key    := args[0]
        _data, err := stub.GetState(_key)
        if err != nil {
                return shim.Error(err.Error())
        }
        if string(_data) == "" {
                return shim.Error("The key isn't exist.")
```

```
        }else{
                fmt.Printf("query %s %s \n", _key,
string(_data))
        }

        return shim.Success(_data)
}

func (s *SmartContract) update(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _key    := args[0]
        _data   := args[1]

        if(_data == ""){
                return shim.Error("Incorrect string of data")
        }

        err := stub.PutState(_key, []byte(_data))
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("update %s %s \n", _key,
string(_data))
        }

        return shim.Success(nil)
}

// Deletes an entity from state
func (t *SmartContract) delete(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        _key := args[0]

        // Delete the key from the state in ledger
        err := stub.DelState(_key)
        if err != nil {
                return shim.Error("Failed to delete state")
```

```
        }

        return shim.Success(nil)
}

func main() {

        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
}
```

链码有四个函数，分别是创建，查找，更新，删除。

```
        if function == "create" {
                return s.create(stub, args)             // 创建
        } else if function == "find" {
                return s.find(stub, args)               // 查找
        } else if function == "update" {
                return s.update(stub, args)             // 更新
        } else if function == "delete" {
                return s.delete(stub, args)             // 删除
        }
```

上链使用 create 方法，函数有两个参数，一个是 key, 另一个是数据。

key 使用 UUID 存储再数据库和链上，同时 UUID 对应通证的

data 是序列化 byte 数据。例如可以使用 json, hession, msgpack 等序列化后的数据。

```
err = stub.PutState(_key, []byte(_data))
```

这个链码考虑到前期产品上市，不确定性因素很多，需要更新和删除等等。后期我们可以在数据中设置一个 status 变量，当 status = false 就不在允许数据的删除和更新。

10.7. 原型设计

10.7.1. 注册与登录

注册

- 安装APP后，点击注册链接，进入注册界面。
- 用户类型选择选择 “机构”或者”鉴定师”。对于前台只有用户一种角色。
- 输入手机号码，然后获取验证码
- 输入密码
- 点击注册按钮，完成注册

登录,平台提供多重登录方式

- 密码登录
- 验证码登录
- 微信登录
- 指纹登录
- 面部识别登录
- 助记词登录
- 私钥登录
- 闪付卡登录(类似银行卡的 QuickPass 闪付，可以在 Post 机上做 Token 付款，转账等操作)

10.7.2. 用户角色

防伪查询

查看链上资产

资产交易

评论

分享

用户是权益持有人,当用户（用户端）委托机构负责自己的资产上链。权益持有人一次性获得次产权益100%份额，机构可以查看的机构名下的权益持有人。

10.7.3. 鉴定师角色

鉴定师是管理机构认证并颁发资质证书人员，艺术品溯源区块链领域，他主要的职责事艺术品鉴定。鉴定师可以挂靠到机构。

如何注册成为鉴定师?

首先进入APP -> 点击『注册 』->阅读条款->点击同意按钮。输入手机号码，发送验证码，输入密码，重复输入密码，选择『鉴定师』，提交。

注册成功会自动登录，进入完善资料页面，鉴定师需要实名认证，上传身份证信息，鉴定师证书，等等资质文件。鉴定师需要仔细填写每一项，并保证资料的真实性。

提交资料后，等待管理机构审批，管理机构会仔细核对每项数据。如果被拒，需要鉴定师重新填写，再提交。

审批通过前，只能看到鉴定师信息页面。 审批通过后，可以看到，鉴定师信息，已签名资产，未处理资产。

怎样鉴定物品?

鉴定师登录APP后，进入『我的』可以看淡未处理资产菜单，进入菜单可以看到任务列表。点开一件物品，将鉴定结果提交上去，完成鉴定。鉴定师可以在[已签名资产]中查看自己鉴定过的物品。

10.7.4. 机构角色

机构主要负责资产上链，审查，资产托管

如何注册成为机构

- 首先进入APP -> 点击『注册 』->阅读条款->点击同意按钮。
- 输入手机号码，发送验证码，输入密码，重复输入密码，选择『机构』，提交数据。
- 注册成功后自动登录APP，进入完善机构信息页面，填写机构信息同时上传资质文件，选择区块链应用领域（目前只开通了艺术品领域）
- 提交资料后，等待管理员审批，管理员会仔细核对每项数据。如果被拒，需要鉴定师重新填写，再提交。

审批通过前：机构只能看到机构信息页面

审批通过后：机构拥有机构权限，机构信息，钱包，地址管理，安全，区块链属性配置，资产管理，权益持有人，评论审核，申请溯源标签

10.7.4.1. 地址管理

地址为收货地址，管理员会邮寄AI智能标签给机构。

进入“我的” -> “设置” -> “地址管理” -> 添加地址

10.7.4.2. 申请溯源标签

首先机构需要完善收货地址，至少添加一个收货地址。

然后进入『申请溯源标签』，进入“我的” -> “机构” -> “申请溯源标签”。输入数量，选择类型，选择收货地址，系统会自动计算费用，提交后从钱包中扣取。

标签有以下几种类型

1. 易碎纸
2. PVC
3. 捆扎带
4. 玻璃管标签
5. 可以根据机构的需求定制AI防伪溯源标签的大小，形状等等。

注意：暂时只能线上申请，线下付款。待钱包功能上线后同意采用 Token 结算。提交信息后，管理员收到信息会主动联系你，完成付款后将标签邮寄给机构的收货地址。

10.7.4.3. 数字资产上链

数字资产上链是将企业数字化资产上传到区块链上，相比数据库区块链是分布式共识，可以防止信息篡改。

准备工作

- 检查你的App是否是最新版本
- 链接WiFI活4G（需要信号稳定）
- 开启手机或者设备的NFC功能，GPS定位功能，授权摄像头，指纹认证

- 准备好物品信息
- 将溯源标签粘贴到艺术品上
- 信息录入可以使用蓝牙键盘链接手机或者管理员指定专用设备上，以便加快信息录入
- 有些输入可以提供语音输入，或者扫码录入。

资产录入

- 录入物品信息，下一步
- 上传多媒体资料，例如图片，视频，声音等等
- 如果物品有历史事件资料，可以追加这些信息到历史记录中。
- 绑定AI只能溯源防伪标签
- 绑定二维码，方便作品调出
- 设置定位信息 （可选）
- 指定一个或多个鉴定师鉴定物品。
- 指定权益持有人（暂时不可用，权益持有人来自用户端）
- 鉴定师鉴定物品，输入鉴定结果。
- 机构提交信息。
- 管理员审批
- 资产上链。

经过上面几个步骤完成数字资产上链。

注意：管理员没有审批前可以修改资产信息，管理员审批后链上数据无法修改。

10.7.4.4. 机构成员管理

在整个区块链溯源防伪的过程中，上链的工作量是最大的，所以非一人所为，必须团队完成。 机构可以添加自己的员工，为员工分配令牌，通过令牌登录后可以一同完成资产上链的数据录入。

10.7.4.5. 资产审核

分支机构可以对员工添加的资产信息逐一核对，核对后点击『提交』按钮，信息将提交至管理员。待管理员在此审批通过，数据便会上链。

无论是机构拒绝还是管理员拒绝，信息都需要重新填写。

信息的审核责任主要在机构，管理员审核通常是看物品是否符合国家法律，法规，政策等等方面。

10.7.4.6. 鉴定师隶属于机构

当鉴定师为机构鉴定物品后即成为该机构的鉴定师，机构可以查看的机构名下的鉴定师，可以理解为鉴定师挂靠该机构，该机构负责管理鉴定师。

鉴定师负责物品鉴定，并给出鉴定结论。一个物品可以有多名鉴定师同时鉴定。

用户也可以邀请鉴定师鉴定某肩艺术品。

10.7.4.7. 评论

用户（用户端）可以评论上链的数字资产，增加用户与机构，用户与用于的互动粘合。

防止用户随意提交无意义的信息，所以机构需要审核每一条评论信息。

10.7.4.8. 安全

10.7.4.8.1. 指纹认证

开启指纹后，添加物品，审批，等等需要确认的地方都需要指纹验证。

10.7.4.8.2. 艺术品跟踪

当发现查询作品的GPS坐标与作品所在位置的GPS坐标不在安全范围时，发送报警信息您的艺术品可能被盗，请尽快核实，如需协助，我方可以向公安机关提供:

1. 物品所在位置
2. 以及查询者的身份信息

10.7.4.8.3. 令牌管理

为员工指派令牌，机构注册的账号，只能登录一部设备，如果需要多人录入资产信息，就需要使用令牌登录。

管理员会收取一部分费用（Token）

10.7.5. 钱包

用户数字资产的交易

部分 I. EOS

第 3 章 EOS

https://eos.io

EOS是一种全新的区块链架构，旨在实现分布式应用的性能拓展。EOS 项目的目标是实现一个类似操作系统一样的支撑应用程序的区块链架构。该架构可以提供账户，身份认证，数据库，异步通信以及可在数以百计的 CPU 或群集上的程序调度。该技术的最终形式是一个区块链体系架构，该区块链每秒可以支持数百万个交易，同时普通用户无需支付使用费用。

1. EOS 资源

1.1. EOS 主网与投票状态

```
https://eosnation.io
https://eoscountdown.com
https://status.producer.vote
http://eos.host/mainnet
https://www.mytokenpocket.vip
http://eosportal.io/chain/12/producers
http://eosnetworkmonitor.io
https://eosauthority.com/voting
```

1.2. EOS 投票工具

```
https://github.com/EOSLaoMao/eos-local-voter
https://www.mytokenpocket.vip
https://github.com/EOSLaoMao/eos-local-voter-desktop
http://eosportal.io
http://vote.libertyblock.io
https://tokenika.github.io/secure-bp-voting
```

https://github.com/greymass/eos-voter

1.3. EOS 区块链浏览器

https://eospark.com
http://scaneos.io
http://eosflare.io
http://eostracker.io
https://eosblock.co
https://explorer.eoseco.com

1.4. EOS 钱包资源

https://www.mytokenpocket.vip
https://meet.one/pomelo
http://halowallet.io
https://scatter-eos.com
https://eoswalletpro.com
https://token.im
http://www.medishares.org/wallet/cn
http://bitpie.com
http://www.eosbixin.com
https://cybex.io
http://eostoken.im
https://oraclechain.io

https://github.com/EOSPortal
https://github.com/EOSEssentials/EOSWallet
https://github.com/espritblock/eos4j
https://github.com/eostoken/wallet
https://github.com/espritblock/react-native-eos
https://github.com/OracleChain/PocketEOS-IOS
https://github.com/OracleChain/PocketEOS-Android
https://github.com/plactal/EosCommander

第 4 章 EOS 安装

1. CentOS

```
yum install -y centos-release-scl
yum install -y devtoolset-7
yum install -y git

yum install -y gcc gcc-c++ make patch cmake automake autoconf \
libtool ocaml doxygen graphviz-devel libicu-devel bzip2-devel
gmp-devel python-devel gettext-devel


cd /usr/local/src/
git clone https://github.com/EOSIO/eos --recursive
cd eos/
# git submodule update --init --recursive

./eosio_build.sh
```

```
[root@iZj6c7cj14ulhfndlmeicbZ eos]# ./eosio_build.sh

        Beginning build version: 1.2
        Wed May  2 03:15:34 UTC 2018
        User: root
        git head id: f537bc50b21a7807ff0ee3af83d8f560ce09afa5
        Current branch: * master

        ARCHITECTURE: Linux

        OS name: CentOS Linux
        OS Version: 7
        CPU speed: 2494Mhz
        CPU cores: 4
        Physical Memory: 7822 Mgb
        Disk install: /dev/vda1
        Disk space total: 492G
        Disk space available: 138G
```

```
        Checking Yum installation
        Yum installation found at /usr/bin/yum.

        Checking installation of Centos Software Collections
Repository.

        The Centos Software Collections Repository, devtoolset-
7 and Python3 are required to install EOSIO.
        Do you wish to install and enable this repository,
devtoolset-7 and Python3 packages?
1) Yes
2) No
#? 1
```

输入 1 回车继续

```
Complete!

        YUM repository successfully updated.

        Checking YUM for installed dependencies.

        Package git found.
        Package autoconf found.
        Package automake found.
        Package libtool  NOT  found.
        Package ocaml.x86_64  NOT  found.
        Package doxygen  NOT  found.
        Package graphviz-devel.x86_64  NOT  found.
        Package libicu-devel.x86_64  NOT  found.
        Package bzip2-devel.x86_64  NOT  found.
        Package openssl-devel.x86_64  NOT  found.
        Package gmp-devel.x86_64  NOT  found.
        Package python-devel.x86_64  NOT  found.
        Package gettext-devel.x86_64  NOT  found.

        The following dependencies are required to install
EOSIO.

        1. libtool
        2. ocaml.x86_64
        3. doxygen
```

```
        4. graphviz-devel.x86_64
        5. libicu-devel.x86_64
        6. bzip2-devel.x86_64
        7. openssl-devel.x86_64
        8. gmp-devel.x86_64
        9. python-devel.x86_64
        10. gettext-devel.x86_64


        Do you wish to install these dependencies?
1) Yes
2) No
#?
```

输入 1 回车继续

```
         _______  _______  _______ _________ _______
        (  ____ \(  ___  )(  ____ \\__   __/(  ___  )
        | (    \/| (   ) || (    \/   ) (   | (   ) |
        | (__    | |   | || (_____    | |   | |   | |
        |  __)   | |   | |(_____  )   | |   | |   | |
        | (      | |   | |      ) |   | |   | |   | |
        | (____/\| (___) |/\____) |___) (___| (___) |
        (_______/(_______)\_______)\_______/(_______)

        EOSIO has been successfully built. 01:20:59

        To verify your installation run the following commands:

        /root/opt/mongodb/bin/mongod -f
/root/opt/mongodb/mongod.conf &
        source /opt/rh/python33/enable
        export PATH=${HOME}/opt/mongodb/bin:$PATH
        cd /usr/local/src/eos/build; make test

        For more information:
        EOSIO website: https://eos.io
        EOSIO Telegram channel @ https://t.me/EOSProject
        EOSIO resources: https://eos.io/resources/
        EOSIO Stack Exchange: https://eosio.stackexchange.com
        EOSIO wiki: https://github.com/EOSIO/eos/wiki
```

这里跳过 make test 直接安装

```
cd build
make install
```

默认配置文件 cofnig.ini

```
[root@netkiller config]# grep -v "^#" config.ini | grep -v "^$"
bnet-endpoint = 0.0.0.0:4321
bnet-follow-irreversible = 0
bnet-no-trx = false
bnet-peer-log-format = ["${_name}" ${_ip}:${_port}]
blocks-dir = "blocks"
chain-state-db-size-mb = 1024
reversible-blocks-db-size-mb = 340
contracts-console = false
https-client-validate-peers = 1
http-server-address = 127.0.0.1:8888
access-control-allow-credentials = false
max-body-size = 1048576
verbose-http-errors = false
p2p-listen-endpoint = 0.0.0.0:9876
p2p-max-nodes-per-host = 1
agent-name = "EOS Test Agent"
allowed-connection = any
max-clients = 25
connection-cleanup-period = 30
network-version-match = 0
sync-fetch-span = 100
max-implicit-request = 1500
use-socket-read-watermark = 0
peer-log-format = ["${_name}" ${_ip}:${_port}]
enable-stale-production = false
pause-on-startup = false
max-transaction-time = 30
max-irreversible-block-age = -1
signature-provider =
```

```
EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV=KEY:5KQwr
PbwdL6PhXujxW37FSSQZ1JiwsST4cqQzDeyXtP79zkvFD3
keosd-provider-timeout = 5
txn-reference-block-lag = 0
wallet-dir = "."
unlock-timeout = 900
```

2. Mac

```
git clone https://github.com/eosio/eos --recursive

cd eos
cmake .
make
```

3. Docker 开发环境

4. 主网

```
git clone https://github.com/EOS-Mainnet/eos -b mainnet-1.0.7
cd eos
git submodule update --init --recursive
./eosio_build.sh -s EOS

cd build
make install
cd ~
wget https://eosnodes.privex.io/static/genesis.json
nodeos --genesis-json genesis.json
# 输入 ctrl-c 退出

vim ~/.local/share/eosio/nodeos/config/config.ini
# 配置节点从这里获取 https://eosnodes.privex.io/?config=1

# 最后启动 eosio

nodeos
```

5. 启动 EOS 节点

5.1. EOS 本地网

5.1.1. 单节点私链

这种模式一般用于合约开发，学习和测试

```
nodeos -e -p eosio --plugin eosio::chain_api_plugin --plugin
eosio::history_api_plugin --plugin eosio::wallet_api_plugin
```

区块数据保存在 ~/.local/share/eosio/nodeos/data

```
[root@netkiller ~]# find ~/.local/share/eosio/nodeos/data
/root/.local/share/eosio/nodeos/data
/root/.local/share/eosio/nodeos/data/blocks
/root/.local/share/eosio/nodeos/data/blocks/blocks.index
/root/.local/share/eosio/nodeos/data/blocks/reversible
/root/.local/share/eosio/nodeos/data/blocks/reversible/shared_memory.meta
/root/.local/share/eosio/nodeos/data/blocks/reversible/shared_memory.bin
/root/.local/share/eosio/nodeos/data/blocks/blocks.log
/root/.local/share/eosio/nodeos/data/state
/root/.local/share/eosio/nodeos/data/state/shared_memory.meta
/root/.local/share/eosio/nodeos/data/state/shared_memory.bin
```

本地私链会提示 warning: transaction executed locally, but may not be confirmed by the network yet.

5.1.2. 单机多节点

https://github.com/EOSIO/eos/wiki/Testnet-Single-Host-Multinode

EOS 有三个模块组成分别是nodeos，keosd，和cleos。nodeos 是节点进程，keosd 是钱包服务，cleos 是命令行接口。实现单机多节点主要工作在于怎样配置 nodeos 进程，使其端口不冲突。

5.1.3. 多机多节点

多级多节点配置只需要将其他节点的IP加入到 p2p-peer-address 池中即可，配置项涉及一下：

```
[root@iZj6c39y62jl5b1wmfv6u8Z config]# grep -v "^#"
~/.local/share/eosio/nodeos/config/config.ini | grep -v "^$"

p2p-peer-address = node1:9876
...
...
p2p-peer-address = node(n):9876
agent-name：改成你自己的标识，域名或其他.
producer-name：节点账户名(12位[12345a-z]字符串).
signature-provider：你的密钥对.
```

准备几台服务器或者云主机

```
Node1 172.16.0.10
Node2 172.16.0.20
Node3 172.16.0.30
```

创世区块 genesis.json

```
{
  "initial_timestamp": "2018-06-01T00:00:00.000",
  "initial_key": "EOS69EZcBVwgRz3AbHheR3ZpeHtaoHAPyLXfvmsiqYMAtazN3WdiL",
  "initial_configuration": {
    "max_block_net_usage": 1048576,
    "target_block_net_usage_pct": 1000,
    "max_transaction_net_usage": 524288,
    "base_per_transaction_net_usage": 12,
    "net_usage_leeway": 500,
    "context_free_discount_net_usage_num": 20,
    "context_free_discount_net_usage_den": 100,
    "max_block_cpu_usage": 100000,
    "target_block_cpu_usage_pct": 500,
    "max_transaction_cpu_usage": 50000,
    "min_transaction_cpu_usage": 100,
    "max_transaction_lifetime": 3600,
    "deferred_trx_expiration_window": 600,
    "max_transaction_delay": 3888000,
    "max_inline_action_size": 4096,
    "max_inline_action_depth": 4,
    "max_authority_depth": 6,
    "max_generated_transaction_count": 16
  },
```

```
  "initial_chain_id":
"0000000000000000000000000000000000000000000000000000000000000000"
}
```

5.1.3.1. 节点一

编辑配置文件 ~/.local/share/eosio/nodeos/config/config.ini

```
p2p-peer-address = 172.16.0.20:9876
p2p-peer-address = 172.16.0.30:9876
agent-name = "EOS Test Agent - Node 1"
producer-name = eosio
signature-provider =
EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV=KEY:5KQwrPbwdL6PhXujxW37FS
SQZ1JiwsST4cqQzDeyXtP79zkvFD3
```

启动 nodeos 进程

```
nodeos --genesis-json genesis.json
```

5.1.3.2. 节点二

编辑配置文件 ~/.local/share/eosio/nodeos/config/config.ini

```
p2p-peer-address = 172.16.0.10:9876
p2p-peer-address = 172.16.0.30:9876
agent-name = "EOS Test Agent - Node 2"
producer-name = eosio
signature-provider =
EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV=KEY:5KQwrPbwdL6PhXujxW37FS
SQZ1JiwsST4cqQzDeyXtP79zkvFD3
```

启动 nodeos 进程

5.1.3.3. 节点三

编辑配置文件 ~/.local/share/eosio/nodeos/config/config.ini

```
p2p-peer-address = 172.16.0.10:9876
p2p-peer-address = 172.16.0.20:9876
agent-name = "EOS Test Agent - Node 3"
producer-name = eosio
signature-provider =
EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV=KEY:5KQwrPbwdL6PhXujxW37FS
SQZ1JiwsST4cqQzDeyXtP79zkvFD3
```

启动 nodeos 进程

5.1.3.4. 进入 Node 1 创建钱包和部署合约

```
cleos wallet create
cleos wallet unlock

# 导入eosio账户的私钥到钱包，signature-provider 中的 KEY:是私钥
cleos wallet import 5KQwrPbwdL6PhXujxW37FSSQZ1JiwsST4cqQzDeyXtP79zkvFD3

cleos create key
cleos create account eosio eosio.token <公钥> <公钥>

cleos set contract eosio /usr/local/src/eos/build/contracts/eosio.bios -p eosio

# 发布eosio.token合约
cleos set contract eosio.token  /usr/local/src/eos/build/contracts/eosio.token

# 创建和发布代币
cleos  push action eosio.token create '["eosio","1000000000.0000 EOS",0,0,0]' -p
eosio.token
cleos  push action eosio.token issue '["eosio","1000000000.0000 EOS","issue"]' -
p eosio

# 查询代币数量
cleos get currency balance eosio.token eosio

# 部署eosio.system合约
cleos set contract eosio /usr/local/src/eos/build/contracts/eosio.system

# 创建账号
cleos create key
cleos system newaccount eosio test <公钥> <公钥> --stake-net '50.00 EOS' --stake-
cpu '50.00 EOS'

# 登录 Node 2 注册成为 bp
cleos  system regproducer test <公钥> http://127.0.0.1:8888

# 给创建的账户转账
cleos push action eosio.token transfer '["eosio", "test","100000000.0000
EOS","vote"]' -p eosio
```

```
# 锁定代币
cleos system delegatebw test test '25000000.0000 EOS' '25000000.0000 EOS' --
transfer

# 给出块节点投票
cleos system voteproducer prods test test
```

5.2. 测试网

5.2.1. Public Testnet Endpoints（公共测试网络的接入点）

5.2.1.1. testnet1.eos.io

```
HTTP Endpoint: testnet1.eos.io
P2P Endpoint: p2p-testnet1.eos.io:9876
Web Wallet Endpoint: t1wallet.eos.io, t1api.eos.io, t1readonly.eos.io

$ curl testnet1.eos.io/v1/chain/get_info
```

5.2.1.2. http://testnet.eoswtz.com

http://www.eoswtz.com

5.2.2. 本地连接到测试网

修改config.ini文件

```
p2p-peer-address = p2p-testnet1.eos.io
block-interval-seconds = 2
```

5.2.3. EOS (testnet) Explorer (Dawn 2.0)

http://eosmonitor.info

5.2.4. EOS Jungle Testnet Monitor (Dawn 4.0)

http://dev.cryptolions.io

5.3. 主网

5.3.1. 创世区块

```
https://github.com/EOS-Mainnet/eos/blob/mainnet-1.0.7/mainnet-genesis.json
```

5.3.2. eosnodes.privex.io

https://eosnodes.privex.io

```
[root@netkiller build]# cleos -u http://api.eosnewyork.io get info
{
  "server_version": "1509de21",
  "chain_id":
"aca376f206b8fc25a6ed44dbdc66547c36c6c33e3a119ffbeaef943642f0e906",
  "head_block_num": 3185805,
  "last_irreversible_block_num": 3185470,
  "last_irreversible_block_id":
"00309b3e25263b1f6cef6a87a304284afcc06bacd1290e6904760395e07d2321",
  "head_block_id":
"00309c8d9f01ba0ac66bd577cdfdd4e85162ba1d68e18f5660356bc8207615b1",
  "head_block_time": "2018-06-29T02:46:41.000",
  "head_block_producer": "eosisgravity",
  "virtual_block_cpu_limit": 200000000,
  "virtual_block_net_limit": 1048576000,
  "block_cpu_limit": 198989,
  "block_net_limit": 1048304
}
```

5.3.2.1. 创世区块

https://eosnodes.privex.io/static/genesis.json

```
{
  "initial_timestamp": "2018-06-08T08:08:08.888",
  "initial_key": "EOS7EarnUhcyYqmdnPon8rm7mBCTnBoot6o7fE2WzjvEX2TdggbL3",
  "initial_configuration": {
    "max_block_net_usage": 1048576,
    "target_block_net_usage_pct": 1000,
    "max_transaction_net_usage": 524288,
    "base_per_transaction_net_usage": 12,
    "net_usage_leeway": 500,
```

```
    "context_free_discount_net_usage_num": 20,
    "context_free_discount_net_usage_den": 100,
    "max_block_cpu_usage": 200000,
    "target_block_cpu_usage_pct": 1000,
    "max_transaction_cpu_usage": 150000,
    "min_transaction_cpu_usage": 100,
    "max_transaction_lifetime": 3600,
    "deferred_trx_expiration_window": 600,
    "max_transaction_delay": 3888000,
    "max_inline_action_size": 4096,
    "max_inline_action_depth": 4,
    "max_authority_depth": 6
  }
}
```

检查创世区块配置是否正确

```
[root@netkiller build]# cleos -u http://api.eosnewyork.io get info
{
  "server_version": "1509de21",
  "chain_id":
"aca376f206b8fc25a6ed44dbdc66547c36c6c33e3a119ffbeaef943642f0e906",
  "head_block_num": 3185805,
  "last_irreversible_block_num": 3185470,
  "last_irreversible_block_id":
"00309b3e25263b1f6cef6a87a304284afcc06bacd1290e6904760395e07d2321",
  "head_block_id":
"00309c8d9f01ba0ac66bd577cdfdd4e85162ba1d68e18f5660356bc8207615b1",
  "head_block_time": "2018-06-29T02:46:41.000",
  "head_block_producer": "eosisgravity",
  "virtual_block_cpu_limit": 200000000,
  "virtual_block_net_limit": 1048576000,
  "block_cpu_limit": 198989,
  "block_net_limit": 1048304
}
[root@netkiller build]# cleos get info
{
  "server_version": "90fefdd1",
  "chain_id":
"cf057bbfb72640471fd910bcb67639c22df9f92470936cddc1ade0e2f2e7dc4f",
  "head_block_num": 26364,
  "last_irreversible_block_num": 26363,
  "last_irreversible_block_id":
"000066fbb919e9e7613765190b7e2e6639588d36e834da77ab63f7cc43d04ec2",
  "head_block_id":
"000066fc54c27265bb243a3ef19923a54b52ab1cb568e067fb2f83ebcf9d1d26",
  "head_block_time": "2018-06-29T02:46:56.000",
  "head_block_producer": "eosio",
  "virtual_block_cpu_limit": 200000000,
  "virtual_block_net_limit": 1048576000,
  "block_cpu_limit": 199900,
  "block_net_limit": 1048576
}
```

对比两次 chain_id 是否一致，一致表示配置正确。

5.3.2.2. 主网超级节点

https://eosnodes.privex.io/?config=1

```
p2p-peer-address = 106.10.42.238:9876
p2p-peer-address = 123.59.116.52:49876
p2p-peer-address = 13.230.91.225:9865
p2p-peer-address = 130.211.59.178:9876
p2p-peer-address = 159.65.214.150:9876
p2p-peer-address = 173.242.25.101:7115
p2p-peer-address = 178.49.174.48:9876
p2p-peer-address = 18.191.33.148:59876
p2p-peer-address = 185.253.188.1:19876
p2p-peer-address = 185.253.188.1:19877
p2p-peer-address = 195.43.95.98:9876
p2p-peer-address = 34.252.209.121:5556
p2p-peer-address = 35.197.190.234:19878
p2p-peer-address = 40.114.68.16:9876
p2p-peer-address = 54.153.59.31:9999
p2p-peer-address = 807534da.eosnodeone.io:19872
p2p-peer-address = 94.130.250.22:9806
p2p-peer-address = api-full1.eoseoul.io:9876
p2p-peer-address = api-full2.eoseoul.io:9876
p2p-peer-address = api.eosuk.io:12000
p2p-peer-address = boot.eostitan.com:9876
p2p-peer-address = bp.antpool.com:443
p2p-peer-address = bp.cryptolions.io:9876
p2p-peer-address = bp.eosbeijing.one:8080
p2p-peer-address = bp.eosmedi.com:9877
p2p-peer-address = bp.libertyblock.io:9800
p2p-peer-address = dc1.eosemerge.io:9876
p2p-peer-address = dns1-p2p.oraclechain.io:49876
p2p-peer-address = eno.eosvan.io:19866
p2p-peer-address = eos-seed-de.privex.io:9876
p2p-peer-address = eos.infinitystones.io:9876
p2p-peer-address = eos.nodepacific.com:9876
p2p-peer-address = eos.staked.us:9870
p2p-peer-address = eosapi.blockmatrix.network:13546
p2p-peer-address = eosboot.chainrift.com:9876
p2p-peer-address = eosbp.buildteam.io:8532
p2p-peer-address = eosbp.eosvillage.io:8181
p2p-peer-address = eosnode.fi:9888
p2p-peer-address = eu-west-nl.eosamsterdam.net:9876
p2p-peer-address = eu1.eosdac.io:49876
p2p-peer-address = fn001.eossv.org:443
p2p-peer-address = fullnode.acroeos.one:9876
p2p-peer-address = fullnode.eoslaomao.com:443
p2p-peer-address = mainnet-eos.wancloud.cloud:55576
p2p-peer-address = mainnet.eos.ren:9376
p2p-peer-address = mainnet.eosarabia.org:3571
p2p-peer-address = mainnet.eoscalgary.io:5222
p2p-peer-address = mainnet.eoseco.com:10010
p2p-peer-address = mainnet.eospay.host:19876
p2p-peer-address = mainnet.eoswz.com:8866
```

```
p2p-peer-address = mainnet2.eostaxrelief.com:9876
p2p-peer-address = mars.fnp2p.eosbixin.com:443
p2p-peer-address = node.eos.lawyer:9876
p2p-peer-address = node.eosflare.io:1883
p2p-peer-address = node.eosio.lt:9878
p2p-peer-address = node.eosmeso.io:9876
p2p-peer-address = node1.eosjapan.co.jp:9876
p2p-peer-address = node1.eosnewyork.io:6987
p2p-peer-address = node2.blockeos.io:9987
p2p-peer-address = node2.eosarmy.io:3330
p2p-peer-address = node2.eosnewyork.io:6987
p2p-peer-address = node2.eosphere.io:9876
p2p-peer-address = p.jeda.one:3322
p2p-peer-address = p2p.eos.bitspace.no:9876
p2p-peer-address = p2p.eosdetroit.io:3018
p2p-peer-address = p2p.eosholding.ca:9876
p2p-peer-address = p2p.eosio.cr:1976
p2p-peer-address = p2p.eosio.cr:5418
p2p-peer-address = p2p.genereos.io:9876
p2p-peer-address = p2p.mainnet.eosgermany.online:9876
p2p-peer-address = p2p.mainnet.eospace.io:88
p2p-peer-address = p2p.meet.one:9876
p2p-peer-address = p2p.one.eosdublin.io:9876
p2p-peer-address = p2p.saltblock.io:19876
p2p-peer-address = p2p.two.eosdublin.io:9876
p2p-peer-address = peer.blockgenicbp.com:9876
p2p-peer-address = peer.eosio.sg:9876
p2p-peer-address = peer.eosjrr.io:9876
p2p-peer-address = peer.eosn.io:9876
p2p-peer-address = peer.main.alohaeos.com:9876
p2p-peer-address = peer1.eos.csx.io:9806
p2p-peer-address = peer1.eosthu.com:8080
p2p-peer-address = peer1.mainnet.eos.store:80
p2p-peer-address = peer1.mainnet.helloeos.com.cn:80
p2p-peer-address = peer2.eos.csx.io:9806
p2p-peer-address = peer2.eosthu.com:8080
p2p-peer-address = peer2.mainnet.helloeos.com.cn:80
p2p-peer-address = peering.dutcheos.io:9876
p2p-peer-address = peering.mainnet.eoscanada.com:9876
p2p-peer-address = peering1.mainnet.eosasia.one:80
p2p-peer-address = peering2.mainnet.eosasia.one:80
p2p-peer-address = pub0.eosys.io:6637
p2p-peer-address = pub1.eostheworld.io:9876
p2p-peer-address = pub1.eosys.io:6637
p2p-peer-address = pub2.eostheworld.io:9876
p2p-peer-address = publicnode.cypherglass.com:9876
p2p-peer-address = seed1.greymass.com:9876
p2p-peer-address = seed2.greymass.com:9876
```

5.3.3. mainnet.genereos.io

```
[root@netkiller ~]# cleos --url=http://mainnet.genereos.io get info
{
  "server_version": "b195012b",
  "chain_id":
```

```
"aca376f206b8fc25a6ed44dbdc66547c36c6c33e3a119ffbeaef943642f0e906",
  "head_block_num": 3218632,
  "last_irreversible_block_num": 3218299,
  "last_irreversible_block_id":
"00311b7b6a24585e4a1fd806f619ca075fe9c72fa0310b6ca465e91aad33999c",
  "head_block_id":
"00311cc8bb55e1c6d2ec9c1455908545ecee293efa556c1dce444814cf891611",
  "head_block_time": "2018-06-29T07:20:16.000",
  "head_block_producer": "teamgreymass",
  "virtual_block_cpu_limit": 200000000,
  "virtual_block_net_limit": 1048576000,
  "block_cpu_limit": 199900,
  "block_net_limit": 1048576
}
```


5.3.4. mainnet.eoswz.com


```
[root@netkiller ~]# cleos -u http://mainnet.eoswz.com get info
{
  "server_version": "aa351733",
  "chain_id":
"aca376f206b8fc25a6ed44dbdc66547c36c6c33e3a119ffbeaef943642f0e906",
  "head_block_num": 3221149,
  "last_irreversible_block_num": 3220819,
  "last_irreversible_block_id":
"003125538662bf4cdc367336daba8a8609d6b4f5edb6e8bf0189a27fa306fbde",
  "head_block_id":
"0031269d3ee1f2b8dbc9419a9d8ac0be7024c3a072d9f178ce297dbea6cbb58e",
  "head_block_time": "2018-06-29T07:41:14.500",
  "head_block_producer": "teamgreymass",
  "virtual_block_cpu_limit": 200000000,
  "virtual_block_net_limit": 1048576000,
  "block_cpu_limit": 199900,
  "block_net_limit": 1048576
}
```

6. nodeos 命令

6.1.

6.1.1. --contracts-console

--contracts-console 开启合约调试模式，合约中的 eosio:print() 将输出到控制台。

6.2. config.ini 配置文件

6.2.1. 插件配置

```
plugin = eosio::producer_plugin
plugin = eosio::wallet_api_plugin
plugin = eosio::chain_api_plugin
plugin = eosio::http_plugin
```

```
plugin = eosio::chain_plugin
```

第 5 章 CLEOS

1. 钱包

1.1. 创建钱包

创建默认钱包

```
$ cleos wallet create
```

演示

```
[root@netkiller ~]# cleos wallet list
"/usr/local/bin/keosd" launched
Wallets:
[]

[root@netkiller ~]# cleos wallet create
Creating wallet: default
Save password to use in the future to unlock this wallet.
Without password imported keys will not be retrievable.
"PW5Hu6VtABuC75RmjSaPv6BcwofA5DQMJ9xHFeFeefmZGNsdknAKQ"

[root@netkiller ~]# cleos wallet list
Wallets:
[
  "default *"
]
```

创建指定名称的钱包

```
$ cleos wallet create -n netkiller
```

操作演示

```
[root@netkiller ~]# cleos wallet create -n netkiller
Creating wallet: netkiller
Save password to use in the future to unlock this wallet.
Without password imported keys will not be retrievable.
"PW5J8qAhMPotrUQAswbPabXZPJq85YVGuxofhGVxo19xcynAfZcqx"

[root@netkiller ~]# cleos wallet list
Wallets:
[
  "default *",
  "netkiller *"
]
```

1.2. 钱包列表

```
$ cleos wallet list
```

1.3. 钱包锁

上锁

```
[root@netkiller ~]# cleos wallet lock
Locked: default
```

```
$ cleos wallet lock -n netkiller
```

解锁

```
[root@netkiller ~]# cleos wallet unlock
password: Unlocked: default

$ cleos wallet unlock -n netkiller
```

2. 账号

2.1. 创建公钥和私钥

```
$ cleos create key
```

```
[root@netkiller etc]# cleos create key
Private key: 5JXxZEQZNjyxNKSGHcdiAwE4uALykxwvgtAyLRxEygQJP9eULkH
Public key: EOS69EZcBVwgRz3AbHheR3ZpeHtaoHAPyLXfvmsiqYMAtazN3WdiL
```

2.2. 导入私钥

```
$ cleos wallet import 5K8apwojp2U4mcv1xAAjP541QFUEhkRWxskYbL3ZzCq1VoBwuSX
```

2.3. 查看私钥

```
$ cleos wallet private_keys --password ${your_wallet_password}
```

2.4. 创建账号

创建秘钥对

```
[root@netkiller ~]# cleos create key
Private key: 5JFMTVk4EjWW54xk73AMRPf5JbpFV2Cm7vtgt1jr9zVaPgLmaLQ
Public key: EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
```

导入私钥

```
[root@netkiller ~]# cleos wallet import
5JFMTVk4EjWW54xk73AMRPf5JbpFV2Cm7vtgt1jr9zVaPgLmaLQ
imported private key for: EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr

[root@netkiller ~]# cleos wallet keys
[
  "EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV",
  "EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr"
```

]

创建账号 neo

```
[root@netkiller ~]# cleos wallet unlock
[root@netkiller ~]# cleos create account eosio neo
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
executed transaction: e138b1e7557d76b3560b898942db942eb23b43f8387c60083741ab4d0680e139
200 bytes  311 us
#         eosio <= eosio::newaccount            {"creator":"eosio","name":"neo","owner":
{"threshold":1,"keys":[{"key":"EOS7fcRYssRt5SXVnsPpRNzj86E9h...
warning: transaction executed locally, but may not be confirmed by the network yet
```

3. set 命令

3.1. abi

当你的abi文件有变动需要更新时使用下面方法更新abi文件。

```
[root@netkiller eos]# cleos set abi contract.art art/art.abi
Setting ABI...
executed transaction:
32b9b9dc55e52eb424f7165b02bd1ca904b8e3aa42d8df7d4fa4a372291b1ea
b  240 bytes  409 us
#         eosio <= eosio::setabi
"90af0119999b274583020e656f73696f3a3a6162692f312e30000406637265
617465000304757365726572046e616d650574697...
warning: transaction executed locally, but may not be confirmed
by the network yet
```

4. 区块信息

4.1. 获得当前区块链信息

```
[root@netkiller ~]# cleos get info
{
  "server_version": "90fefdd1",
  "chain_id":
"cf057bbfb72640471fd910bcb67639c22df9f92470936cddc1ade0e2f2e7dc4
f",
  "head_block_num": 1063,
  "last_irreversible_block_num": 1062,
  "last_irreversible_block_id":
"00000426ca6002e4097bb85032e357bceac77d90075e0020f45b26d397c2183
b",
  "head_block_id":
"00000427049a6f175fd5c13660651e7fe36ef8199e316bed0349a178c33f525
b",
  "head_block_time": "2018-06-28T08:39:57.000",
  "head_block_producer": "eosio",
  "virtual_block_cpu_limit": 577784,
  "virtual_block_net_limit": 3033320,
  "block_cpu_limit": 199900,
  "block_net_limit": 1048576
}
```

4.2. 获取指定区块数据

```
[root@netkiller ~]# cleos get block 1063
{
  "timestamp": "2018-06-28T08:39:57.000",
  "producer": "eosio",
  "confirmed": 0,
  "previous":
"00000426ca6002e4097bb85032e357bceac77d90075e0020f45b26d397c2183
b",
  "transaction_mroot":
"0000000000000000000000000000000000000000000000000000000000000000
```

```
0",
  "action_mroot":
"b472502694c9f3fa5684f44edc4c34742708b2400690a49bb00a297b3d20145
6",
  "schedule_version": 0,
  "new_producers": null,
  "header_extensions": [],
  "producer_signature":
"SIG_K1_Jzx3cvL6pDxEsxhFbqPasqBymxKhodiiWjVmgtifFEDzThdYfBTvVvvm
TNTxaBLwBZ1AJxyuW1uR3J5nvKDwc3xnAgRuWk",
  "transactions": [],
  "block_extensions": [],
  "id":
"00000427049a6f175fd5c13660651e7fe36ef8199e316bed0349a178c33f525
b",
  "block_num": 1063,
  "ref_block_prefix": 918672735
}
```


4.3. 从区块链获取交易信息


```
[root@netkiller ~]# cleos get transaction
cf057bbfb72640471fd910bcb67639c22df9f92470936cddc1ade0e2f2e7dc4f
{
  "id":
"cf057bbfb72640471fd910bcb67639c22df9f92470936cddc1ade0e2f2e7dc4
f",
  "trx": null,
  "block_time": "2000-01-01T00:00:00.000",
  "block_num": 0,
  "last_irreversible_block": 1777,
  "traces": []
}
```


4.4. 获得账号信息


```
[root@netkiller ~]# cleos get account neo
permissions:
```

```
     owner     1:    1
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
        active     1:    1
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
memory:
     quota:       unlimited  used:      2.66 KiB

net bandwidth:
     used:               unlimited
     available:          unlimited
     limit:              unlimited

cpu bandwidth:
     used:               unlimited
     available:          unlimited
     limit:              unlimited
```

```
[root@netkiller ~]# cleos get accounts
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
{
  "account_names": [
    "neo"
  ]
}

[root@netkiller ~]# cleos get accounts
EOS5xfs4X3wNB5cdhzUAebBmaVN3eTsQspGpKrr7FgvEMcZV5kvn3
{
  "account_names": [
    "contract.cms"
  ]
}
```

4.5. 从区块链上获取 abi 文件

```
[root@netkiller eos]# cleos get code contract.art  -a art.abi
code hash:
37636dd268e72e75d0559f3c8acdce07feae20dd682bc4b859e7f62517f4bc5f
saving abi to art.abi
```

```
[root@netkiller eos]# cat art.abi
{
  "version": "eosio::abi/1.0",
  "types": [],
  "structs": [{
      "name": "create",
      "base": "",
      "fields": [{
          "name": "user",
          "type": "name"
        },{
          "name": "title",
          "type": "string"
        },{
          "name": "content",
          "type": "string"
        }
      ]
    },{
      "name": "change",
      "base": "",
      "fields": [{
          "name": "user",
          "type": "name"
        },{
          "name": "post_id",
          "type": "uint64"
        },{
          "name": "title",
          "type": "string"
        },{
          "name": "content",
          "type": "string"
        }
      ]
    },{
      "name": "remove",
      "base": "",
      "fields": [{
          "name": "user",
          "type": "name"
        },{
          "name": "post_id",
```

```
            "type": "uint64"
          }
        ]
      },{
        "name": "find",
        "base": "",
        "fields": [{
            "name": "post_id",
            "type": "uint64"
          },{
            "name": "user",
            "type": "name"
          }
        ]
      }
    ],
    "actions": [{
        "name": "create",
        "type": "create",
        "ricardian_contract": ""
      },{
        "name": "change",
        "type": "change",
        "ricardian_contract": ""
      },{
        "name": "remove",
        "type": "remove",
        "ricardian_contract": ""
      },{
        "name": "find",
        "type": "find",
        "ricardian_contract": ""
      }
    ],
    "tables": [],
    "ricardian_clauses": [],
    "error_messages": [],
    "abi_extensions": []
}
```

5. 智能合约 - EOS 代币

5.1. 编译智能合约

编译 eosio.bios 合约

```
cd /usr/local/src/eos/build/contracts/eosio.bios

[root@netkiller eosio.bios]# make
[  4%] Built target libc++
[  4%] Built target wasm
[  4%] Built target ast
[  4%] Built target asmjs
[  4%] Built target cfg
[ 10%] Built target passes
[ 12%] Built target support
[ 14%] Built target eosio-s2wasm
[ 17%] Built target Platform
[ 17%] Built target Logging
[ 17%] Built target IR
[ 17%] Built target WASM
[ 17%] Built target WAST
[ 17%] Built target eosio-wast2wasm
[ 19%] Built target eosiolib
[100%] Built target libc
[100%] Built target eosio.bios
```

编译 eosio.token 合约

```
cd /usr/local/src/eos/build/contracts/eosio.token

[root@netkiller eosio.token]# pwd
/usr/local/src/eos/build/contracts/eosio.token
[root@netkiller eosio.token]# make
[  4%] Built target libc++
[  4%] Built target wasm
[  4%] Built target ast
[  4%] Built target asmjs
[  4%] Built target cfg
[ 10%] Built target passes
[ 12%] Built target support
[ 14%] Built target eosio-s2wasm
[ 17%] Built target Platform
```

```
[ 17%] Built target Logging
[ 17%] Built target IR
[ 17%] Built target WASM
[ 17%] Built target WAST
[ 17%] Built target eosio-wast2wasm
[ 19%] Built target eosiolib
[100%] Built target libc
[100%] Built target eosio.token
```

5.2. 设置初始化账号 eosio

从配置文件 ~/.local/share/eosio/nodeos/config/config.ini 中查找 signature-provider

```
[root@netkiller ~]# grep "^signature-provider"
~/.local/share/eosio/nodeos/config/config.ini
signature-provider =
EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV=KEY:5KQwrPbwdL6PhX
ujxW37FSSQZ1JiwsST4cqQzDeyXtP79zkvFD3
```

找到 signature-provider 配置项，复制秘钥
5KQwrPbwdL6PhXujxW37FSSQZ1JiwsST4cqQzDeyXtP79zkvFD3

```
[root@netkiller ~]# cleos wallet import
5KQwrPbwdL6PhXujxW37FSSQZ1JiwsST4cqQzDeyXtP79zkvFD3
imported private key for:
EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV
```

导入 eosio 账号私钥到 default 钱包

```
[root@netkiller ~]# cleos wallet keys
[
  "EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV"
]
```

5.3. 创建账号

创建秘钥对

```
[root@netkiller ~]# cleos create key
Private key: 5JFMTVk4EjWW54xk73AMRPf5JbpFV2Cm7vtgt1jr9zVaPgLmaLQ
Public key: EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
```

导入私钥

```
[root@netkiller ~]# cleos wallet import
5JFMTVk4EjWW54xk73AMRPf5JbpFV2Cm7vtgt1jr9zVaPgLmaLQ
imported private key for:
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr

[root@netkiller ~]# cleos wallet keys
[
  "EOS6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV",
  "EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr"
]
```

创建账号 neo

```
[root@netkiller ~]# cleos wallet unlock
[root@netkiller ~]# cleos create account eosio neo
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
executed transaction:
e138b1e7557d76b3560b898942db942eb23b43f8387c60083741ab4d0680e139  200
bytes  311 us
#         eosio <= eosio::newaccount
{"creator":"eosio","name":"neo","owner":{"threshold":1,"keys":
[{"key":"EOS7fcRYssRt5SXVnsPpRNzj86E9h...
warning: transaction executed locally, but may not be confirmed by the
network yet
```

5.4. 部署合约 eosio.bios

```
[root@netkiller ~]# cleos wallet unlock
```

```
[root@netkiller ~]# cleos set contract eosio
/usr/local/src/eos/build/contracts/eosio.bios -p eosio
```

```
[root@netkiller ~]# cleos set contract eosio
/usr/local/src/eos/build/contracts/eosio.bios -p eosio
Reading WAST/WASM from
/usr/local/src/eos/build/contracts/eosio.bios/eosio.bios.wasm...
Using already assembled WASM...
Publishing contract...
executed transaction:
c8589dc4ddb429765e86e78add1420461ce35a4edac7e08fe790e4b876a1ce29  3720
bytes  815 us
#         eosio <= eosio::setcode
{"account":"eosio","vmtype":0,"vmversion":0,"code":"0061736d010000000162
1260037f7e7f0060057f7e7e7e7e...
#         eosio <= eosio::setabi
{"account":"eosio","abi":"0e656f73696f3a3a6162692f312e30050c6163636f756e
745f6e616d65046e616d650f7065...
warning: transaction executed locally, but may not be confirmed by the
network yet
```

5.5. 创建账号 netkiller

创建账号 netkiller 重复上面步骤，这个账号用于创建代币智能合约。

```
[root@netkiller ~]# cleos create key
Private key: 5KVTLTRgLdKj4b5FkkFpYMhYdhimPip3dtdfnZAQVQxQBBV4oFq
Public key: EOS5NyaD49BuTCScNEY8FPBCZ9t6VXThMAmFvgMg72XqcNVPXuEWH

[root@netkiller ~]# cleos wallet import
5KVTLTRgLdKj4b5FkkFpYMhYdhimPip3dtdfnZAQVQxQBBV4oFq
imported private key for:
EOS5NyaD49BuTCScNEY8FPBCZ9t6VXThMAmFvgMg72XqcNVPXuEWH

[root@netkiller ~]# cleos create account eosio netkiller
EOS5NyaD49BuTCScNEY8FPBCZ9t6VXThMAmFvgMg72XqcNVPXuEWH
EOS5NyaD49BuTCScNEY8FPBCZ9t6VXThMAmFvgMg72XqcNVPXuEWH
executed transaction:
fc87fc5cb598a24b36bf3dc10c542d7425d319d33291029de1f0c412dadea233  200
bytes  301 us
#         eosio <= eosio::newaccount
{"creator":"eosio","name":"netkiller","owner":{"threshold":1,"keys":
[{"key":"EOS5NyaD49BuTCScNEY8FPB...
warning: transaction executed locally, but may not be confirmed by the
```

```
network yet
```

5.6. EOS 代币合约

```
cleos set contract netkiller
/usr/local/src/eos/build/contracts/eosio.token
```

操作演示

```
[root@netkiller ~]# cleos set contract netkiller
/usr/local/src/eos/build/contracts/eosio.token
Reading WAST/WASM from
/usr/local/src/eos/build/contracts/eosio.token/eosio.token.wasm...
Using already assembled WASM...
Publishing contract...
executed transaction:
a8bdeafdadd37b6a3b2bf1de908725028e51ae3d5f8a9e0f95e4d33b3b22b8be  8104
bytes  1411 us
#         eosio <= eosio::setcode
{"account":"netkiller","vmtype":0,"vmversion":0,"code":"0061736d01000000
017e1560037f7e7f0060057f7e7e...
#         eosio <= eosio::setabi
{"account":"netkiller","abi":"0e656f73696f3a3a6162692f312e30010c6163636f
756e745f6e616d65046e616d6505...
warning: transaction executed locally, but may not be confirmed by the
network yet
```

code hash 有值表示合约部署成功

```
[root@netkiller ~]# cleos get code netkiller
code hash:
641f336aa1d08526201599c3c0ddb7a646e5ac8f9fd2493f56414d0422a0f957
```

code hash 为 0 表示合约部署失败

```
[root@netkiller ~]# cleos get code netkiller
code hash:
0000000000000000000000000000000000000000000000000000000000000000
```

5.7. 创建代币

```
[root@netkiller eosio.token]# cleos wallet unlock
password: Unlocked: default

[root@netkiller eosio.token]# cleos push action netkiller create
'["netkiller","1000 EOS",0,0,0]' -p netkiller
executed transaction:
7266dda0b3fde45bd03fc0ab4fdece35e66cf6e616165f113434f63394f3fa33  120
bytes  488 us
#     netkiller <= netkiller::create
{"issuer":"netkiller","maximum_supply":"1000 EOS"}
warning: transaction executed locally, but may not be confirmed by the
network yet

[root@netkiller ~]# cleos push action netkiller create
'["netkiller","10000 NRC",0,0,0]' -p netkiller
executed transaction:
cbb45fcca9e140f24cd90fb6c0fbb781afcc3a1d6da4dc912a7fde79b26b7dd0  120
bytes  521 us
#     netkiller <= netkiller::create
{"issuer":"netkiller","maximum_supply":"10000 NRC"}
warning: transaction executed locally, but may not be confirmed by the
network yet
```

查看合约信息

```
[root@netkiller ~]# cleos get currency stats netkiller EOS
{
  "EOS": {
    "supply": "1000 EOS",
    "max_supply": "1000 EOS",
    "issuer": "netkiller"
  }
}
```

另一个方式

```
cleos push action netkiller create '{"issuer":"netkiller",
"maximum_supply": "10.0000 EOS", "can_freeze": 1, "can_recall": 1,
"can_whitelist": 1}' -p netkiller@active
```

5.8. 发放代币

```
cleos push action netkiller issue '["neo","1000 EOS","issue"]' -p
netkiller
```

给 neo 账号发放 1000 个 EOS 币

```
[root@netkiller ~]# cleos push action netkiller issue '["neo","1000
EOS","issue"]' -p netkiller
executed transaction:
c60760dfbdad2face6917ff28015555f1cfc293d71eb7556fc2f7ec78591229b  128
bytes  1339 us
#     netkiller <= netkiller::issue
{"to":"neo","quantity":"1000 EOS","memo":"issue"}
#     netkiller <= netkiller::transfer
{"from":"netkiller","to":"neo","quantity":"1000 EOS","memo":"issue"}
#           neo <= netkiller::transfer
{"from":"netkiller","to":"neo","quantity":"1000 EOS","memo":"issue"}
warning: transaction executed locally, but may not be confirmed by the
network yet
```

```
cleos push action netkiller issue '{"to":"neo","quantity":"10.0000
EOS","memo":"备注信息"}' --permission netkiller@active
```

5.9. 查看代币余额

```
[root@netkiller ~]# cleos get table netkiller neo accounts
{
  "rows": [{
```

```
      "balance": "1000 EOS"
    }
  ],
  "more": false
}
```

```
[root@netkiller ~]# cleos get currency balance netkiller neo
1000 EOS

[root@netkiller ~]# cleos get currency balance netkiller neo EOS
1000 EOS
```

5.10. 转账

```
cleos push action eosio transfer '["eosio","netkiller","100 EOS",""]' -p
eosio
cleos push action contract transfer
'{"from":"from_address","to":"to_address","quantity":"1.0000
EOS","memo":"测试"}' --permission neo@active
```

操作演示

```
[root@netkiller ~]# cleos get currency balance netkiller netkiller EOS

[root@netkiller ~]# cleos push action netkiller transfer
'["neo","netkiller","10 EOS","memo"]' -p neo
executed transaction:
0e23837bd8a3a7876b2463cbde1d47a25d2ac2178bb42ddbccd3037416cc9e43  136
bytes  745 us
#     netkiller <= netkiller::transfer
{"from":"neo","to":"netkiller","quantity":"10 EOS","memo":"memo"}
#           neo <= netkiller::transfer
{"from":"neo","to":"netkiller","quantity":"10 EOS","memo":"memo"}
warning: transaction executed locally, but may not be confirmed by the
```

network yet

```
[root@netkiller ~]# cleos get currency balance netkiller netkiller EOS
10 EOS
```

第 6 章 智能合约开发

1. WebAssembly

EOS 的块链使用的是WebAssembly(http://webassembly.org/) 技术，编译后的 (WASM) 执行用户编写的智能合约。WASM是一种新兴的Web标准，广泛支持于谷歌、微软、苹果等。WASM标准的智能合约使用C/C++语言编写，使用clang/llvm(https://clang.llvm.org/) 编译。

2. 只能合约文件

创建智能合约

```
$ eosiocpp -n ${contract}
```

运行上面的命令会在./${project}目录下创建一个空的项目，它包含3个文件。

```
${contract}.abi ${contract}.hpp ${contract}.cpp
```

有些情况我们发现没有 hpp 文件，所以 hpp 是可有可无的。

2.1. hpp 头文件

${contract}.hpp 这是合约的头文件，可以包含一些变量，常量和函数的声明。

2.2. cpp 合约代码文件

${contract}.cpp 这是合约的源码文件，包含合约的具体实现。

2.3. abi 文件

作用类似以太坊的 ABI 文件。 ABI（ Application Binary Interface）文件是一个JSON格式的描述文件，说明了如何在他们的JSON和二进制之间转化用户的action。

ABI文件也同时说明了如何转换数据库的状态。一旦你用了ABI描述了你的合约，开发人员就和用户就可以和你的合约通过JSON进行交互。

ABI文件可以通过eosiocpp命令使用.hpp文件生成。

```
$ eosiocpp -g ${contract}.abi ${contract}.hpp
```

3. eosiocpp 命令

3.1. 创建新合约

```
[root@iZj6c39y62jl5b1wmfv6u8Z test]# eosiocpp -n hello
created hello from skeleton

[root@iZj6c39y62jl5b1wmfv6u8Z test]# find hello/
hello/
hello/hello.hpp
hello/hello.cpp
```

3.2. 编译 WAST 文件

```
eosiocpp -o output.wast contract.cpp
```

3.3. 编译 ABI 文件

```
eosiocpp -g contract.abi types.hpp
```

4. eosio.token 合约详解

4.1. token::create 方法

```
void token::create( account_name issuer,
                    asset        maximum_supply,
                    uint8_t      issuer_can_freeze,
                    uint8_t      issuer_can_recall,
                    uint8_t      issuer_can_whitelist )
```

4.2. token::issue 方法

```
void token::issue( account_name to, asset quantity, string memo )
```

4.3. token::transfer 转账方法

```
void token::transfer( account_name from,
                      account_name to,
                      asset        quantity,
                      string       memo )
```

5. 编译运行 hello 智能合约

hello 智能合约是官方提供的一个智能合约例子

找到config.ini中的配置项contracts-console = false 改为 true

```
[root@netkiller ~]# vim
~/.local/share/eosio/nodeos/config/config.ini

# print contract's output to console (eosio::chain_plugin)
contracts-console = true
```

源码

```
[root@netkiller hello]# cat
/usr/local/src/eos/contracts/hello/hello.cpp
#include <eosiolib/eosio.hpp>
using namespace eosio;

class hello : public eosio::contract {
  public:
      using contract::contract;

      /// @abi action
      void hi( account_name user ) {
         print( "Hello, ", name{user} );
      }
};

EOSIO_ABI( hello, (hi) )
```

编译智能合约

```
cd /usr/local/src/eos/build/contracts/hello

[root@netkiller hello]# make
[  4%] Built target libc++
[  4%] Built target wasm
[  4%] Built target ast
[  4%] Built target asmjs
[  4%] Built target cfg
[ 10%] Built target passes
[ 12%] Built target support
[ 14%] Built target eosio-s2wasm
[ 16%] Built target Platform
[ 16%] Built target Logging
[ 16%] Built target IR
[ 16%] Built target WASM
[ 16%] Built target WAST
[ 16%] Built target eosio-wast2wasm
[ 18%] Built target eosiolib
[ 97%] Built target libc
[100%] Built target hello




[root@netkiller hello]# cleos wallet unlock
password: Unlocked: default

[root@netkiller hello]# cleos set contract contract.hello
/usr/local/src/eos/build/contracts/hello -p eosio
Reading WAST/WASM from
/usr/local/src/eos/build/contracts/hello/hello.wasm...
Using already assembled WASM...
Publishing contract...
executed transaction:
f5695465f35b153d65c36cb0e07443fd3d8ccadde9c1daf8c472b0a7e84196b
0  4160 bytes  1040 us
#         eosio <= eosio::setcode
"0000000000ea30550000e2170061736d01000000013b0c60027f7e00600001
7e60027e7e0060027f7f006000017f60027f7...
#         eosio <= eosio::setabi
"0000000000ea3055912b0e656f73696f3a3a6162692f312e30000102686900
010475736572046e616d65010000000000000008...
warning: transaction executed locally, but may not be confirmed
by the network yet
```

```
[root@netkiller hello]# cleos push action eosio hi '["neo"]' -p
eosio
executed transaction:
476fa2416d227ffe078285714d10d2d726b8e9cc18b9f0ba672bfc1ef93efbd
5  104 bytes  284 us
#         eosio <= eosio::hi                    {"user":"neo"}
warning: transaction executed locally, but may not be confirmed
by the network yet
```

6. dice

7. 智能合约数据库操作 CURD

为了方便调试合约

找到config.ini中的配置项contracts-console = false 改为 true

```
[root@netkiller ~]# vim ~/.local/share/eosio/nodeos/config/config.ini

# print contract's output to console (eosio::chain_plugin)
contracts-console = true
```

这样 eosio::print() 输出的内容才会显示在控制台上。

7.1. 创建一个新项目

```
eosiocpp -n project
```

7.2. 创建结构体

例如我们需要这样一个数据结构

```
{
        id,
        description,
        completed
}
```

结构体定义如下

```
struct todo {
        uint64_t id;
        std::string description;
```

```
        uint64_t completed;

        uint64_t primary_key() const { return id; }
        EOSLIB_SERIALIZE(todo, (id)(description)(completed))
};
```

primary_key() 相当与数据中的主键。

定义一个表

```
typedef eosio::multi_index<N(todos), todo> todo_table;
todo_table todos;
```

7.3. 插入数据操作

```
void create(account_name author, const uint32_t id, const std::string&
description) {
        todos.emplace(author, [&](auto& new_todo) {
                new_todo.id  = id;
                new_todo.description = description;
                new_todo.completed = 0;
        });
```

7.4. 修改数据操作

```
void complete(account_name author, const uint32_t id) {
      auto todo_lookup = todos.find(id);
      eosio_assert(todo_lookup != todos.end(), "Todo does not exist");

      todos.modify(todo_lookup, author, [&](auto& modifiable_todo) {
        modifiable_todo.completed = 1;
      });

      eosio::print("todo#", id, " marked as complete");
    }
```

7.5. 删除数据操作

```
void destroy(account_name author, const uint32_t id) {
        auto todo_lookup = todos.find(id);
        todos.erase(todo_lookup);

        eosio::print("todo#", id, " destroyed");
}
```

7.6. 完整的合约例子

```
#include <eosiolib/eosio.hpp>
#include <string>

namespace eosio {
using std::string;
  class netkiller : public contract {
     public:
           netkiller( account_name self ):contract(self){}

           void create(account_name author, string title, string
content);
           void change(account_name author, uint64_t post_id, string
title, string content);
           void remove(account_name author, uint64_t post_id);
           void find(uint64_t post_id, account_name author);

     private:

           struct da {
                 uint64_t     post_id;
                 account_name poster;
                 string       title;
                 string       content;

                 uint64_t primary_key()const { return post_id; }
                 account_name get_poster() const { return poster; }

                 EOSLIB_SERIALIZE(da, (post_id)(poster)(title)(content))
           };
           typedef eosio::multi_index<N(data), da,
indexed_by<N(byposter), const_mem_fun<da, account_name,
&da::get_poster>> > article;
  };
}
```

```
#include "netkiller.hpp"

namespace eosio {

    void netkiller::create(account_name author, string title, string
content)
    {
        require_auth( author );
        article datable( _self, _self);
        datable.emplace(author, [&]( da & d){
            d.title = title;
            d.content = content;
            d.post_id = datable.available_primary_key();
            d.poster = author;
        });
    }

    void netkiller::change(account_name author, uint64_t post_id, string
title, string content)
    {
        require_auth(author);
        article datable( _self, author);
        auto post = datable.find(post_id);
        eosio_assert(post->poster == author, "netkiller");
        datable.modify(post, author, [&](auto& p){
            if (title != "")
                p.title = title;
            if (content != "")
                p.content = content;
        });
    }

    void netkiller::remove(account_name author, uint64_t post_id)
    {
        require_auth(author);
        article datable( _self, author);
        auto post = datable.find(post_id);
        eosio::print(post->title.c_str());

        eosio_assert(post->poster == author, "The author is invlide");
        datable.erase(post);
    }

    void netkiller::find(uint64_t post_id, account_name author){
        article datable(_self, _self);
        auto post_da = datable.find( post_id);
        eosio::print("Post_id: ", post_da->post_id, "  Post_Tile: ",
post_da->title.c_str(), " Content: ", post_da->content.c_str());
```

```
        auto poster_index = datable.template get_index<N(byposter)>();
        auto pos = poster_index.find( author );

        for (; pos != poster_index.end(); pos++)
        {
            eosio::print("content:", pos->content.c_str(), " post_id:",
pos->post_id, " title:", pos->title.c_str());
        }
    }

}
EOSIO_ABI(eosio::netkiller, (create)(change)(remove)(find))
```

7.6.1. 编译

```
eosiocpp -o cms.wast cms.cpp
eosiocpp -g cms.abi cms.cpp
```

7.6.2. 启动EOS私链开发环境

```
nodeos -e -p eosio --plugin eosio::chain_api_plugin --plugin
eosio::history_api_plugin --plugin eosio::wallet_api_plugin
```

7.6.3. 创建合约账号

这里我们创建一个账号，用这个账号部署合约，该账号是合约所有者。

创建秘钥对

```
[root@netkiller ~]# cleos wallet unlock
```

```
[root@netkiller ~]# cleos create key
Private key: 5HxCWNbTEADKbvdRBgeENXhxReHMQbVuPL5mumDqGCzmkPo5yy3
Public key: EOS7WEcxxHcmM7w7DHB56N6qQ2toMrdudYjeDTZb6LtL9g77MXzR4
```

导入私钥

```
[root@netkiller ~]# cleos wallet import
5HxCWNbTEADKbvdRBgeENXhxReHMQbVuPL5mumDqGCzmkPo5yy3
imported private key for:
EOS7WEcxxHcmM7w7DHB56N6qQ2toMrdudYjeDTZb6LtL9g77MXzR4

[root@netkiller ~]# cleos wallet keys | grep
EOS7WEcxxHcmM7w7DHB56N6qQ2toMrdudYjeDTZb6LtL9g77MXzR4
  "EOS7WEcxxHcmM7w7DHB56N6qQ2toMrdudYjeDTZb6LtL9g77MXzR4",
```

创建账号 neo

```
[root@netkiller ~]# cleos create account eosio contract.cms
EOS7WEcxxHcmM7w7DHB56N6qQ2toMrdudYjeDTZb6LtL9g77MXzR4
EOS7WEcxxHcmM7w7DHB56N6qQ2toMrdudYjeDTZb6LtL9g77MXzR4
executed transaction:
f04ba09f633dffbf97321c6d2e021192082383908fa690dc40032cd98a1bfd89  200
bytes  390 us
#         eosio <= eosio::newaccount
"0000000000ea305590af0119999b27450100000010003588ecdc868696f500c7782dbf
0da3b298830e67ea9b810469819d...
warning: transaction executed locally, but may not be confirmed by the
network yet
```

contract.art 就是我们合约账号，我们使用 contract 前缀来区分他是合约账号。

7.6.4. 部署合约

```
[root@netkiller eos]# cleos wallet unlock
password: Unlocked: default

[root@netkiller eos]# cleos set contract contract.cms cms
Reading WAST/WASM from cms/cms.wasm...
```

```
Using already assembled WASM...
Publishing contract...
executed transaction:
8a72e29389e170807daaf41e9c9e70ac4eff2f2f129ca22ef55ca9443768dedf  7176
bytes  1311 us
#         eosio <= eosio::setcode
"80250219999b27450000c3770061736d0100000001b2011b60037f7e7e0060027f7e006
0057f7e7e7f7f0060047f7e7f7f0...
#         eosio <= eosio::setabi
"80250219999b27459d020e656f73696f3a3a6162692f312e300005066372656174650000
306617574686f72046e616d65057...
warning: transaction executed locally, but may not be confirmed by the
network yet
```

7.6.5. 创建

```
cleos push action contract.cms create
'{"author":"contract.cms","title":"hello","content":"helloworld!!!"}' -p
contract.cms

[root@netkiller eos]# cleos push action contract.art create
'{"author":"contract.art","title":"hello","content":"helloworld!!!"}' -p
contract.art
executed transaction:
b6cab608fb4e7fa17a7f893848f3516e1bfd231769ad7d7226b0a099f309a771  120
bytes  899 us
#  contract.art <= contract.art::create
{"author":"contract.cms","title":"hello","content":"helloworld!!!"}
warning: transaction executed locally, but may not be confirmed by the
network yet
```

```
[root@netkiller eos]# cleos push action contract.cms create
'{"author":"neo","title":"hello","content":"helloworld!!!"}' -p neo
executed transaction:
90cb81b11386514b450e1a609f0e1e2633f6a4e40d453c127811e5cd33b46a5a  120
bytes  755 us
#  contract.art <= contract.art::create
{"author":"neo","title":"hello","content":"helloworld!!!"}
warning: transaction executed locally, but may not be confirmed by the
network yet
```

下面我们来查询一下刚刚插入的数据:

7.6.6. 查找

find

```
[root@netkiller eos]# cleos push action contract.cms find '{"id":0}' -p
contract.cms
executed transaction:
b3cba4d001fcb49a88926be208fa7bee59d557b0ed8a2bd12e65bdd3ff69c61e  104
bytes  486 us
#  contract.cms <= contract.cms::find           {"id":0}
>> id: 0 Tile: hello Content: helloworld!!!
```

query

```
[root@netkiller eos]# cleos push action contract.cms query
'{"author":"contract.cms", "id":0}' -p contract.cms
executed transaction:
c81a0d21634f4942f7d65dd49efcf5cf7cd739a049968b5f9b0eaad7de4c688c  112
bytes  583 us
#  contract.cms <= contract.cms::query
{"author":"contract.cms","id":0}
>> Post_id: 0  Post_Tile: hello Content:
helloworld!!!content:helloworld!!! id:0 title:hello
```

7.6.7. 修改

修改表中的数据

```
[root@netkiller eos]# cleos push action contract.cms change
'{"author":"contract.cms","id":0,"title":"word","content":"china"}' -p
contract.cms
executed transaction:
073205e4e3e30699a81394ee622aa84d568958919801b364d809ff34f4ca8412  120
```

```
bytes  553 us
#  contract.cms <= contract.cms::change
{"author":"contract.cms","id":0,"title":"word","content":"china"}
```

检查数据修改情况

```
[root@netkiller eos]# cleos push action contract.cms find '{"id":0}' -p
contract.cms
executed transaction:
300fe2c93cf2cfebcdd0524e0629a96b3011b0592be2119d001f38807c1c378b  104
bytes  498 us
#  contract.cms <= contract.cms::find           {"id":0}
>> id: 0 Tile: word Content: china
```

7.6.8. 删除

```
[root@netkiller eos]# cleos push action contract.cms remove
'{"author":"contract.cms","id":0}' -p contract.cms
executed transaction:
eeee8ff799c58d5e3a246ccec8d80c47599ad947d4581611d9a668abee53c0b5  112
bytes  770 us
#  contract.cms <= contract.cms::remove
{"author":"contract.cms","id":0}
>> word
```

检查被删除的数据，提示 Error 3070002: Runtime Error Processing WASM 表示找不到该记录。

```
[root@netkiller eos]# cleos push action contract.cms find '{"id":0}' -p
contract.cms
Error 3070002: Runtime Error Processing WASM
```

7.7. 序列主键

```
#include <eosiolib/eosio.hpp>

using namespace eosio;

class vehicle : public eosio::contract {
public:
    /// @abi table
    struct service_rec {
        uint64_t        pkey;
        account_name    customer;
        uint32_t        date;
        uint32_t        odometer;

        auto primary_key() const { return pkey; }

        account_name get_customer() const { return customer; }

        EOSLIB_SERIALIZE(service_rec, (pkey)(customer)(date)(odometer))
    };

    typedef multi_index<N(service), service_rec> service_table_type;

    using contract::contract;

    /// @abi action
    void exec(account_name owner, account_name customer) {

        service_table_type service_table(current_receiver(), owner);
        uint64_t pkeyf;
        service_table.emplace(owner, [&](auto &s_rec) {
            s_rec.pkey = service_table.available_primary_key(); // 主键自
增
            pkeyf = s_rec.pkey;
            print(pkeyf);// 打印主键内容
            s_rec.customer = customer;
            s_rec.date = 2000;
            s_rec.odometer = 100;
        });

        print("Hello, ", name{customer});
        service_rec result = service_table.get(pkeyf);
        print("_", result.pkey);
        print("_", result.customer);
        print("_", result.date);
        print("_", result.odometer);
    }

};

EOSIO_ABI(vehicle, (exec))
```

第 7 章 EOS Dapp 开发

```
[root@iZj6c39y62jl5b1wmfv6u8Z ~]# curl
http://127.0.0.1:8888/v1/chain/get_info
{"server_version":"90fefdd1","chain_id":"cf057bbfb72640471fd910
bcb67639c22df9f92470936cddc1ade0e2f2e7dc4f","head_block_num":22
122,"last_irreversible_block_num":22121,"last_irreversible_bloc
k_id":"000056694107636adfcaa75fc4879d48eafb6ce7ee9b108af074494b
aa77b0ea","head_block_id":"0000566ada1e3bd2d57b707c67d6a817fac3
41b70efb59506f973031142ef25c","head_block_time":"2018-06-
29T02:11:35.000","head_block_producer":"eosio","virtual_block_c
pu_limit":200000000,"virtual_block_net_limit":1048576000,"block
_cpu_limit":199900,"block_net_limit":1048576}
```

1. eosjs

1.1. 安装 eosjs

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/node.j
s/binrary/node-v10.5.0.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/node.j
s/binrary/profile.d.sh | bash

npm install eosjs
```

1.2. 实例演示

1.2.1. 智能合约

```
[root@netkiller ~]# mkdir echo
[root@netkiller ~]# cd echo/
```

```
[root@netkiller echo]# cat echo.cpp
// website: http://www.netkiller.cn
// author: netkiller@msn.com

#include <eosiolib/eosio.hpp>
#include <eosiolib/print.hpp>
#include <string>

using std::string;

class echo_test : public eosio::contract {
  public:
      using eosio::contract::contract;
      void echo(string tmp) {
         eosio::print(tmp);
      }
};

EOSIO_ABI( echo_test, (echo) )
```

```
[root@netkiller echo]# eosiocpp -o echo.wast echo.cpp
[root@netkiller echo]# eosiocpp -g echo.abi echo.cpp
760047ms thread-0   abi_generator.hpp:68
ricardian_contracts  ] Warning, no ricardian clauses found for
echo_test

760048ms thread-0   abi_generator.hpp:75
ricardian_contracts  ] Warning, no ricardian contract found for
echo

Generated echo.abi ...
```

```
[root@netkiller echo]# cleos wallet unlock
```

```
password: Unlocked: default

[root@netkiller echo]# cleos set contract neo ~/echo -p neo
Reading WAST/WASM from /root/echo/echo.wasm...
Using already assembled WASM...
Publishing contract...
executed transaction:
61a7cf6eaef1f46e0974369c3905f0fe3b5993c44ef0cd138172e260b3e35fe
e  2656 bytes  846 us
#         eosio <= eosio::setcode
"000000000000a89a0000bb250061736d0100000001320a60027f7f00600000
6000017e60027e7e006000017f60027f7f017...
#         eosio <= eosio::setabi
"000000000000a89a360e656f73696f3a3a6162692f312e300001046563686f
000103746d7006737472696e6701000000000...
warning: transaction executed locally, but may not be confirmed
by the network yet

[root@netkiller echo]# cleos push action neo echo
'["helloworld"]' -p neo
executed transaction:
0dfe1d9599e59a92e593e89fcfdd7eb7b069dda362c9c65a6f333b7959b1b8b
5  104 bytes  327 us
#           neo <= neo::echo
{"tmp":"helloworld"}
warning: transaction executed locally, but may not be confirmed
by the network yet
```

1.2.2. 通过 eosjs 访问智能合约

```
EOS = require('eosjs')
eos = EOS.Localnet({
    keyProvider:
['5JFMTVk4EjWW54xk73AMRPf5JbpFV2Cm7vtgt1jr9zVaPgLmaLQ'],
    httpEndpoint: 'http://127.0.0.1:8888'
})

eos.contract('neo').then((contract) => {
    contract.echo("helloworld", { authorization: ['neo']
}).then((res) => {
    console.log(res) })
})
```

运行结果

```
[root@iZj6c39y62jl5b1wmfv6u8Z test]# node test.js
deprecated, change Eos.Localnet(..) to just Eos(..)
{ broadcast: true,
  transaction:
   { compression: 'none',
     transaction:
      { expiration: '2018-07-02T09:41:21',
        ref_block_num: 4538,
        ref_block_prefix: 91102360,
        net_usage_words: 0,
        max_cpu_usage_ms: 0,
        delay_sec: 0,
        context_free_actions: [],
        actions: [Array],
        transaction_extensions: [] },
     signatures:
      [
'SIG_K1_K7kueHwDEYsX1xKrZrB1c1RZy2fD2iv8aeq74ww92ryGsmgYXA9qJXF
UM1UtEE867y5jNyyaw52GEnFKmHTWe7RFYm2gpD' ] },
  transaction_id:

'2643a8d5ac9d408822d7d20712518449e87d18e2164851a6164bfe19801a88
d8',
  processed:
   { id:

'2643a8d5ac9d408822d7d20712518449e87d18e2164851a6164bfe19801a88
d8',
     receipt:
      { status: 'executed', cpu_usage_us: 491, net_usage_words:
13 },
     elapsed: 491,
     net_usage: 104,
     scheduled: false,
     action_traces: [ [Object] ],
     except: null } }
```

第 8 章 FAQ

1. Error 3090003: provided keys, permissions, and delays do not satisfy declared authorizations

```
[root@netkiller ~]# cleos create account eosio netkiller
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
EOS7fcRYssRt5SXVnsPpRNzj86E9h5g62hBgKwr1NSzRmSpH9byZr
Error 3090003: provided keys, permissions, and delays do not
satisfy declared authorizations
Ensure that you have the related private keys inside your
wallet and your wallet is unlocked.
```

eosio 私钥没有导入到钱包

2. Error 3080006: transaction took too long

```
# cleos set contract eosio eosio.system

Reading WAST/WASM from eosio.system/eosio.system.wasm...
Using already assembled WASM...
Publishing contract...
Error 3080006: transaction took too long
Error Details:
deadline exceeded
```

nodeos程序启动时添加max-transaction-time即可解决这个问题

```
nodeos -e -p eosio --max-transaction-time=1000
```

3. 不显示合约中的 eosio::print() 输出

找到config.ini中的配置项contracts-console = false 改为 true

```
[root@netkiller ~]# vim
~/.local/share/eosio/nodeos/config/config.ini

# print contract's output to console (eosio::chain_plugin)
contracts-console = true
```

部分 II. Ethereum 以太坊

第 9 章 以太坊

1. 名词解释

DAPP 去中心化应用

离线的钱包就是冷钱包，
在线的钱包就是热钱包，

从安全性角度看：冷钱包>热钱包>平台
从便捷性看：平台>热钱包>冷钱包

bip32 = hd wallets, what they are how they work
bip39 = specific type of mnemonic, and the process for turning it into a bip32 seed
bip44 = a specific format of a bip32 wallet

2. IBAN (International Bank Account Number)

以太坊官网的说明： https://github.com/ethereum/wiki/wiki/ICAP:-Inter-exchange-Client-Address-Protocol

2.1. iban：国际银行账号

iban其英文全称为International Bank Account Number， 即国际银行帐号。iban的作用是为全球任意一家银行中的任意一个账户 生成一个全球唯一的账号，以便进行跨行交易。一个iban账号看起来像这样：

```
XE039RBH0XKV9FZMTH2701Q37FLX10NTWXU
```

iban地址最多可以包含34个字母和数字，其中的字母大小写不敏感。

iban 中包含以下信息

1. 国别码，用来标识国家，遵循ISO3166-1 alpha-2标准
2. 错误识别码，用来对地址进行校验，采用mod-97-10校验和协议，即ISO/IEC 7064:2003标准
3. 基本银行账号，即BBAN（Basic Bank Account Number），用来标识银行机构、网点及 客户在该机构内的账号，这三部分信息的编码方案依赖于前面提及的国别码

2.2. 以太坊iban：新的国别码和BBAN编码方案

以太坊引入了一个新的IBAN国别码：XE，其中E代表Ethereum，X代表非法币（non-jurisdictional currencies）。同时，以太坊提出了三种BBAN的编码格式：direct、basic和indirect。

direct编码方案中的BBAN为30个字母/数字，只有一个字段：账户编号。例如，以太坊 地址00c5496aee77c1ba1f0854206a26dda82a81d6d8转换为direct方案的BBAN账号，就 得到XE7338O073KYGTWWZN0F2WZ0R8PX5ZPPZS。

可以使用web3.js中的web3.eth.Iban.toIban() 方法来执行这一转换：

```
var Web3 = require('web3');
var web3 = new Web3();
web3.eth.Iban.toIban("0x00c5496aEe77C1bA1f0854206A26DdA82a81D6D
8");

'XE7338O073KYGTWWZN0F2WZ0R8PX5ZPPZS'
```

basic编码方案与direct方案的唯一区别在于，其BBAN长度为31个字母/数字，因此该方案 不兼容IBAN。

indrect编码方案中的BBAN长度为16个字母/数字，包含三个字段

1. 资产编号，由3个字母/数字组成
2. 机构编号，由4个字母/数字组成
3. 机构内客户编号，由9个字母/数字组成

例如，一个采用indrect编码方案的以太坊iban账号，看起来是这样：

```
XE81ETHXREGGAVOFYORK
```

XE81ETHXREGGAVOFYORK

1. ETH：在本例中，表示客户账户内的资产编号。目前ETH是唯一有效的资产编号
2. XREG：机构编号，XREG表示以太坊基本注册合约
3. GAVOFYORK：机构内客户的编号

2.3. iban账号与以太坊地址的转换

如前所述，使用web3.eth.Iban.toIban()方法，可以将一个以太坊地址 转换为direct编码方案的iban账号。与之对应的，可以使用 web3.eth.Iban.toAddress()方法， 将一个采用direct编码方案的iban账号，转换回以太坊地址。例如：

```
var Web3 = require('web3');
var web3 = new Web3();
web3.eth.Iban.toAddress("XE7338O073KYGTWWZN0F2WZ0R8PX5ZPPZS");

'0x00c5496aEe77C1bA1f0854206A26DdA82a81D6D8'
```

2.4. 检查iban账号的有效性

iban账号中的校验和用来帮助核验一个给定字符串是否为有效的iban账号。可以使用 web3.js中的web3.eth.Iban.isValid() 来进行执行校验。例如：

```
> web3.eth.Iban.isValid("XE81ETHXREGGAVOFYORK");
true

> var iban = new web3.eth.Iban("XE81ETHXREGGAVOFYORK");
undefined
> iban.isValid();
true

> web3.eth.Iban.isValid("XE82ETHXREGGAVOFYORK");
```

```
false
```

3. 如何计算Gas手续费

下面我们用实例讲解怎样计算以太坊在执行交易时花费的 gas 费用。

```
> miner.start(1)
null
```

准备两个账号

```
> eth.getBalance(eth.accounts[3])
1000000000000000000
> eth.getBalance(eth.accounts[5])
0
```

开始计算 gas 费用

```
> var estimateGas = eth.estimateGas({from:eth.accounts[1], to:
eth.accounts[2], value: web3.toWei(1)})
undefined
> console.log(estimateGas)
21000
undefined
>
> var cost = estimateGas * gasPrice
undefined
> console.log(cost)
378000000000000
undefined
> web3.fromWei(cost)
"0.000378"
```

gas 花费 0.000378 ETH

4. 转出账号中所有 ETH，Ethereum Wallet 中的 Send everything 实现方法

```
> personal.unlockAccount(eth.accounts[3], "12345678")
true

> eth.sendTransaction({from: eth.accounts[3], to:
eth.accounts[5], value: eth.getBalance(eth.accounts[3]) - cost,
gas: estimateGas})
"0x4e27a477e128b200239bc2ecd899077c6ae064da963a919fef41bcc7462a
ec8d"
```

查看交易细节

```
>
web3.eth.getTransaction("0x4e27a477e128b200239bc2ecd899077c6ae0
64da963a919fef41bcc7462aec8d")
{
  blockHash:
"0x59a9905831e7ae3cb9e7c6f125cf48e2688ef4b39317838f6f6b6c8837d0
1404",
  blockNumber: 4367,
  from: "0x8efb99ec55bcfbe2cfe47918f2d9e55fa732111f",
  gas: 21000,
  gasPrice: 18000000000,
  hash:
"0x4e27a477e128b200239bc2ecd899077c6ae064da963a919fef41bcc7462a
ec8d",
  input: "0x",
  nonce: 15,
  r:
"0xa297401df3a1fb0298cbc1dd609deebe9ded319fadc55934ecef4d525198
215",
  s:
"0x780d8c46bc8d1bb89ae9d78055307d9d68a4f89ba699ef86d3f8ba883831
39a6",
  to: "0xf0688330101d53bd0c6ede2ef04d33c2010e9a5d",
```

```
  transactionIndex: 0,
  v: "0x42",
  value: 999622000000000000
}
```

现在查看from账号，余额已经清零。

```
> eth.getBalance(eth.accounts[3])
0
```

5. （0/12 block confirmations)

```
web3.eth.blockNumber-
web3.eth.getTransaction(<txhash>).blockNumber
```

6. 太坊账户管理 keystore 文件

Post author: Martin Wang
Post link: http://stevenocean.github.io/2018/04/02/about-ethereum-keystore.html

6.1. 什么是 keystore 文件

以太坊的每个外部账户都是由一对密钥（一个公钥和一个私钥）定义的。账户以地址为索引，地址由公钥衍生而来，取公钥的最后 20个字节。每对私钥 /地址都编码在一个钥匙文件里，也就是我们说的keystore文件。该文件是 JSON 文本文件，可以用任何文本编辑器打开和浏览。钥匙文件的关键部分，账户私钥，通常用你创建帐户时设置的密码进行加密。也就是说 keystore 文件，就是你独有的、用于签署交易的以太坊私钥的加密文件。如果你丢失了这个文件，你就丢失了私钥，意味着你失去了签署交易的能力，意味着你的资金被永久的锁定在了你的账户里。

6.2. keystore 文件的内容

我们先看一下keystore文件都包含哪些数据:

```
neo@MacBook-Pro ~/Library/Ethereum/testnet/keystore % cat UTC--2018-04-01T09-30-44.943874000Z--
d5eeae04932dbc2e65b948a76a6cdfd44323a5dd
{       "address":"d5eeae04932dbc2e65b948a76a6cdfd44323a5dd",
        "crypto":{
                "cipher":"aes-128-ctr",
                "ciphertext":"16715298517abb35cb44e9a32d1f81f21ead63006c57eb0ff434318a6ea3ed3f",
                "cipherparams":{
                        "iv":"1ff6fea34e682158e7660ae67960ff76"
                },
                "kdf":"scrypt",
                "kdfparams":{
                        "dklen":32,
                        "n":262144,
                        "p":1,
                        "r":8,

"salt":"a99af42dac2363db631ef7c57a25705c7efdee73b19c11b27f9a91d41cd32d1c"
                },
                "mac":"dcd248fb996604dfcb69a86604af3992b4a9b8d20cc05e0c7608189dbbe66eda"
        },
        "id":"55edb869-1c86-4c68-924e-8247575a158b",
        "version":3
}
```

我们可以看到大部分内容都在 crypto 字段中，这里包括：

cipher：对称 AES 算法的名称;
cipherparams：上述 cipher 算法需要的参数;
ciphertext：你的以太坊私钥使用上述 cipher 算法进行加密;
kdf：密钥生成函数，用于让你用密码加密 keystore 文件;

kdfparams：上述 kdf 算法需要的参数;
Mac：用于验证密码的代码。

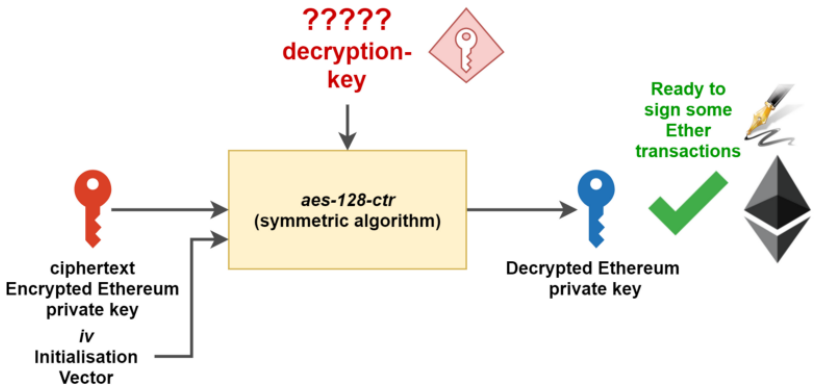
6.3. keystore文件如何工作的?

6.3.1. 加密你的私钥

就像之前提到的，一个以太坊账户就是用于加密签署交易的一个私钥-公钥对。为了确保你的私钥没有在文件中明文存储（即任何人只要能得到这个文件就能读），使用强对称算法（cipher）对其加密至关重要。

这些对称算法使用密钥来加密数据。加密后的数据可以使用相同的方法和同样的密钥来解密，因此算法命名为对称算法。在本文中，我们称这个对称密钥为解密密钥，因为它将用于对我们的以太坊私钥进行解密。

加密过程，如下图：



以下是 cipher，cipherparams 和 ciphertext 对应的概念：

1. cipher: 是用于加密以太坊私钥的对称加密算法。此处cipher用的是 aes-128-ctr 加密模式。
2. cipherparams: 是 aes-128-ctr 加密算法需要的参数。在这里，用到的唯一的参数 iv，是aes-128-ctr加密算法需要的初始化向量。
3. ciphertext: 密文是 aes-128-ctr 函数的加密输入。

在这里，你已经有了进行解密以太坊私钥计算所需要的一切。但是我们首先要取回解密密钥。

6.3.2. 用你的密码来保护它

要确保解锁你的账户很容易，你不需要记住你的每一个又长又非用户友好型的用于解密 ciphertext 密文解密密钥。相反，以太坊开发者选择了基于密码的保护，也就是说你只需要输入密码就能拿回解密密钥。

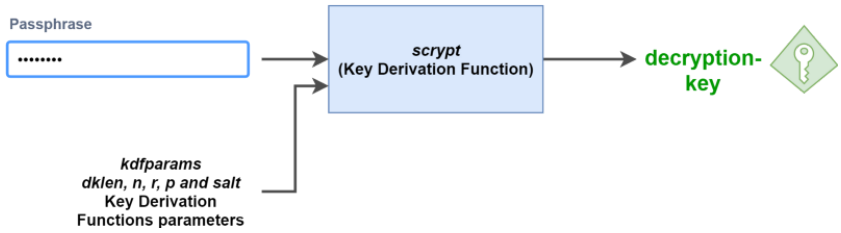
为了能做到这一点，以太坊用了一个密钥生成函数，输入密码和一系列参数就能计算解密密钥。

这就是 kdf 和 kdfparams 的用途：

1. kdf: 是一个密钥生成函数，根据你的密码计算（或者取回）解密密钥。在这里，kdf 用的是scrypt算法。

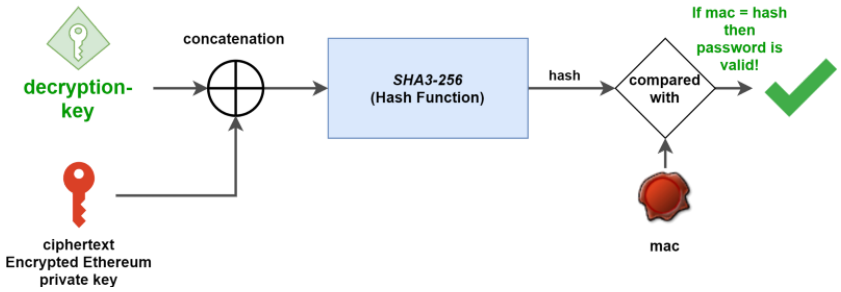
2. kdfparams: 是scrypt函数需要的参数。在这里，简单来说，dklen、n、r、p 和 salt 是 kdf 函数的参数。

在这里，用 kdfparams 参数对 scrypt 函数进行调整，反馈到我们的密码中，你就会得到解密密钥也就是密钥生成函数的输出。



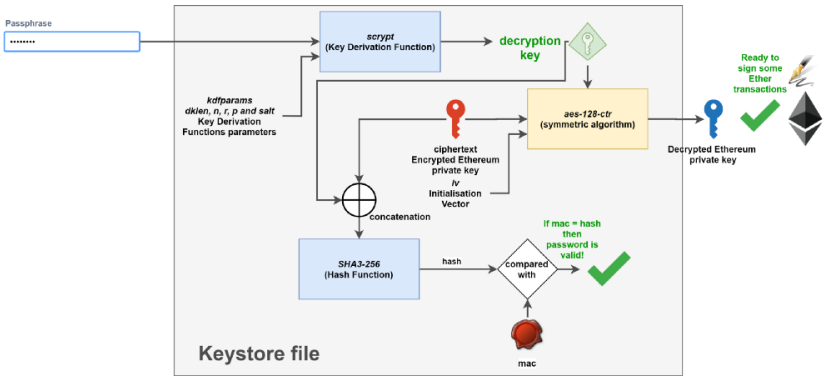
6.3.3. 确认输入的密码是正确的

这就是 keystore 文件中 mac 值起作用的地方。在密钥生成函数执行之后，它的输出（解密密钥）和 ciphertext 密文就被处理，并且和 mac（就像一种认可的印章）作比较。如果结果和 mac 相同，那么密码就是正确的，并且解密就可以开始了。



6.3.4. 将这三步结合起来

首先，你输入了密码，这个密码作为 kdf 密钥生成函数的输入，来计算解密密钥。然后，刚刚计算出的解密密钥和 ciphertext 密文连接并进行处理，和 mac 比较来确保密码是正确的。最后，通过 cipher 对称函数用解密密钥对 ciphertext 密文解密。



就像你从图中可以看到的，整个过程可以看做一个黑盒（不过，图中是个灰盒），你的密码是惟一的输入，你的以太坊私钥是惟一的输出。所需的其他信息都可以在你的以太坊账户创建时生成的keystore文件中获得。

7. 批量转账遇到的问题与解决方案

有时我们需要将Token发送到多个地址上去。通常的做法就是使用web.js写一个循环程序，但是实际使用过程中发现这种做法存在很多问题，常会发生转账失败情况。

经过分析造成发送失败原因是，频率太高，因为web.js 是异步操作，当前程序还未完成交易，交易尚未确认的情况下第二笔交易发送出去，这时就会出错。

```
web3.eth.getTransactionCount(from).then(function(nonce){
        // 问题就出在 nonce ， nonce 如同数据库中的序列主键，如果上一个交易没有完成，下一个交易取得 nonce + 1 后与上一个 pending 的交易相同，产生冲突
}
```

最好的解方案是在合约中实现批量转账功能，这样需要一个交易txhash 完成多比交易。

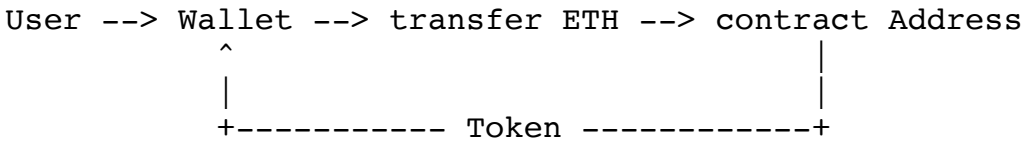
如果你的Token已经在使用了，且没有批量转账的功能怎么解决呢？我们可以在写一个Token 继承原来的Token，在新的 Token 中实现批量转账功能

```
Function: batchTransfer(address[] _to, uint256[] _value)

MethodID: 0x88d695b2
[0]:
0000000000000000000000000000000000000000000000000000000000000040
[1]:
00000000000000000000000000000000000000000000000000000000000000a0
```

[2]:
0000000000000000000000000000000000000000000000000000000000000002
[3]:
000000000000000000000000372967130b1a7cd386348bc59de905e628e36f3a
[4]:
000000000000000000000000372967130b1a7cd386348bc59de905e628e36f3b
[5]:
0000000000000000000000000000000000000000000000000000000000000002
[6]:
000000000000000000000000000000000000000000000000000000000000000c
[7]:
0000000000000000000000000000000000000000000000000000000000000022

8. 代币兑换

兑换代币通常是指使用 ETH 或者其他币兑换Token，我是这样实现的，使用智能合约，将 ETH 达到指定合约地址，合约会打回代币给用户。



这种方式不用人工参与，也不用开发程序。缺点不能时时汇率，需要人工设置汇率。

第 10 章 以太坊私链入门

本文所用软件版本

- geth v.1.8.13
- truffle v4.0.6
- web3.js v1.0.0
- Solidity Version: 0.4.24

1. 软件安装与配置

1.1. Ubuntu

1.1.1. 安装 geth

安装环境

Ubuntu 17.10

```
sudo apt upgrade -y
sudo apt install software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt update
sudo apt install ethereum
```

```
neo@netkiller ~ % geth version
Geth
Version: 1.8.10-stable
Git Commit: 4bb3c89d44e372e6a9ab85a8be0c9345265c763a
Architecture: amd64
Protocol Versions: [63 62]
Network Id: 1
```

```
Go Version: go1.9.5
Operating System: linux
GOPATH=
GOROOT=/usr/lib/go-1.9
```

1.1.2. 安装 solc

```
sudo apt install solc -y
```

```
neo@netkiller ~ % solc --version
solc, the solidity compiler commandline interface
Version: 0.4.19+commit.c4cbbb05.Linux.g++
```

1.1.3. Node.js

```
curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -
sudo apt-get install -y nodejs
```

1.2. CentOS 7

```
yum update -y
yum install git wget bzip2 -y
yum install golang -y

cd /usr/local/src
git clone https://github.com/ethereum/go-ethereum.git
cd go-ethereum/
gmake all
mv build /srv/go-ethereum

echo "export PATH=$PATH:$PWD/build/bin" >> /etc/profile
source /etc/profile
```

上面安装版本是 unstable 版本，如果是生产环境请使用 Release 版本
https://github.com/ethereum/go-ethereum/tags

```
wget https://github.com/ethereum/go-ethereum/archive/v1.8.10.tar.gz
tar zxvf v1.8.10.tar.gz
cd go-ethereum-1.8.10/
gmake all

mv build /srv/go-ethereum-1.8.10
```

1.3. Windows

访问 https://geth.ethereum.org/downloads/
下载并安装 Geth for Windows

1.4. Mac OS

```
brew update
brew upgrade
brew tap ethereum/ethereum
brew install ethereum
brew install solidity
```

1.4.1. 安装 Node

```
brew install node
brew install npm
npm config set registry https://registry.npm.taobao.org
```

1.4.2. Ethereum Wallet

下载安装以太坊钱包（大陆网络可能下载有问题，需要翻墙）

https://github.com/ethereum/mist/releases/download/v0.9.3/Ethereum-Wallet-macosx-0-9-3.dmg

1.5. 编译安装

```
git clone https://github.com/ethereum/go-ethereum
sudo apt-get install -y build-essential golang
cd go-ethereum
make geth
```

1.6. Netkiller OSCM 一键安装

Netkiller OSCM 是由 Netkiller 制作的一套自动化安装脚本。

适用于 CentOS 7

1.6.1. 1.8.10

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/ethereum/
centos/go-ethereum-1.8.10.sh | bash
```

安装完成后使用下面命令进入控制台

```
[root@localhost ~]# su - ethereum
Last login: Sat Feb  3 00:23:52 EST 2018 on pts/0

[ethereum@localhost ~]$ geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.10-stable/linux-amd64/go1.8.10
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0
rpc:1.0 txpool:1.0 web3:1.0

>
```

1.6.2. 1.8.1

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/ethereum/
centos/go-ethereum-1.8.1.sh | bash
```

1.6.3. 1.8.10

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/gcc/gcc.sh |
bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/golang/golang-
1.10.2.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/ethereum/
centos/go-ethereum-1.8.10.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/ethereum/
systemd/private.sh | bash

curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/node.js/binrary
/node-v10.1.0.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/node.js/binrary
/profile.d.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/ethereum/
truffle/truffle.sh | bash
```

1.7. 防止 geth 异常退出

在同步主网的过程中，我们发现经常出现geth崩溃退出，为了防止异常退出，我们写了这个脚本。

```
[ethereum@netkiller ~]$ cat run.sh
#!/bin/bash
for (( ; ; ))
do
        #geth --syncmode light --cache 2048 --maxpeers 200
        # geth --syncmode light --cache 2048 --maxpeers 200 --rpc --
rpcaddr 0.0.0.0 --rpcport 7545 --rpcapi web3,admin,eth,personal --port
30303 2> /tmp/geth.log
        geth --datadir private --networkid 44444 --port 30302 --mine --
rpc
        #geth --testnet --syncmode light --cache 2048 --maxpeers 200 --
rpc 2> /tmp/geth.log
        sleep 10

done &
```

2. 创世区块

```
cd ~
mkdir -p ethereum
cd ethereum
```

2.1. 初始化创世区块

创建文件 genesis.json

```
{
  "nonce": "0x0000000000000042",
  "difficulty": "0x020000",
  "mixhash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000000000000000000",
  "timestamp": "0x00",
  "parentHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "extraData":
"0x11bbe8db4e347b4e8c937c1c8370e4b5ed33adb3db69cbdb7a38e1e50b1b82fa",
  "gasLimit": "0x4c4b40",
  "config": {
      "chainId": 15,
      "homesteadBlock": 0,
      "eip155Block": 0,
      "eip158Block": 0
  },
  "alloc": { }
}
```

mixhash: 与nonce配合用于挖矿，由上一个区块的一部分生成的hash。注意他和nonce的设置需要满足以太坊的Yellow paper, 4.3.4. Block Header Validity, (44)章节所描述的条件。.

nonce: nonce就是一个64位随机数，用于挖矿，注意他和mixhash的设置需要满足

以太坊的Yellow paper, 4.3.4. Block Header Validity, (44)章节所描述的条件。

difficulty: 设置当前区块的难度，如果难度过大，cpu挖矿就很难，这里设置较小难度

alloc: 用来预置账号以及账号的以太币数量，因为私有链挖矿比较容易，所以我们不需要预置有币的账号，需要的时候自己创建即可以。

coinbase: 矿工的账号，随便填

timestamp: 设置创世块的时间戳

parentHash: 上一个区块的hash值，因为是创世块，所以这个值是0

extraData: 附加信息，随便填，可以填你的个性信息

gasLimit: 该值设置对GAS的消耗总量限制，用来限制区块能包含的交易信息总和，因为我们是私有链，所以填最大。

初始化创世区块

```
neo@netkiller ~/ethereum % geth init genesis.json
WARN [01-19|17:35:17] No etherbase set and no accounts found as default
INFO [01-19|17:35:17] Allocated cache and file handles
database=/home/neo/.ethereum/geth/chaindata cache=16 handles=16
INFO [01-19|17:35:17] Writing custom genesis block
INFO [01-19|17:35:17] Successfully wrote genesis state
database=chaindata                          hash=611596…424d04
INFO [01-19|17:35:17] Allocated cache and file handles
database=/home/neo/.ethereum/geth/lightchaindata cache=16 handles=16
INFO [01-19|17:35:18] Writing custom genesis block
INFO [01-19|17:35:18] Successfully wrote genesis state
database=lightchaindata                             hash=611596…424d04
```

默认目录是 /home/neo/.ethereum/ 你可以通过 --datadir 参数指定目录

```
neo@netkiller ~/ethereum % geth --datadir data init genesis.json
```

```
WARN [01-19|17:38:16] No etherbase set and no accounts found as default
INFO [01-19|17:38:16] Allocated cache and file handles
database=/home/neo/ethereum/data/geth/chaindata cache=16 handles=16
INFO [01-19|17:38:17] Writing custom genesis block
INFO [01-19|17:38:17] Successfully wrote genesis state
database=chaindata                                  hash=611596…424d04
INFO [01-19|17:38:17] Allocated cache and file handles
database=/home/neo/ethereum/data/geth/lightchaindata cache=16 handles=16
INFO [01-19|17:38:17] Writing custom genesis block
INFO [01-19|17:38:17] Successfully wrote genesis state
database=lightchaindata                             hash=611596…424d04

neo@netkiller ~/ethereum % find data
data
data/keystore
data/geth
data/geth/chaindata
data/geth/chaindata/LOCK
data/geth/chaindata/LOG
data/geth/chaindata/MANIFEST-000000
data/geth/chaindata/CURRENT
data/geth/chaindata/000001.log
data/geth/lightchaindata
data/geth/lightchaindata/LOCK
data/geth/lightchaindata/LOG
data/geth/lightchaindata/MANIFEST-000000
data/geth/lightchaindata/CURRENT
data/geth/lightchaindata/000001.log
```

目录结构

```
data
├── geth
│   ├── chaindata
│   │   ├── 000001.log
│   │   ├── CURRENT
│   │   ├── LOCK
│   │   ├── LOG
│   │   └── MANIFEST-000000
│   └── lightchaindata
│       ├── 000001.log
│       ├── CURRENT
│       ├── LOCK
│       ├── LOG
│       └── MANIFEST-000000
└── keystore
```

2.2. 创建主账号

```
neo@netkiller ~/ethereum % geth account new
Your new account is locked with a password. Please give a password. Do
not forget this password.
Passphrase:
Repeat passphrase:
Address: {39211a30bfe33d4b12fcbd786472c8a552b93389}
```

提示

密码可以直接回车，但是后面 Ethereum Wallet 遇到了麻烦， Ethereum Wallet 要求密码必须是8为数

2.3. 启动节点

```
neo@netkiller ~/ethereum % geth --networkid 123456 --rpc --rpccorsdomain
"*" --nodiscover console
WARN [01-19|17:47:06] No etherbase set and no accounts found as default
INFO [01-19|17:47:06] Starting peer-to-peer node
instance=Geth/v1.8.10-stable-4bb3c89d/linux-amd64/go1.9.5
INFO [01-19|17:47:06] Allocated cache and file handles
database=/home/neo/.ethereum/geth/chaindata cache=128 handles=1024
INFO [01-19|17:47:06] Initialised chain configuration          config="
{ChainID: 15 Homestead: 0 DAO: <nil> DAOSupport: false EIP150: <nil>
EIP155: 0 EIP158: 0 Byzantium: <nil> Engine: unknown}"
INFO [01-19|17:47:06] Disk storage enabled for ethash caches
dir=/home/neo/.ethereum/geth/ethash count=3
INFO [01-19|17:47:06] Disk storage enabled for ethash DAGs
dir=/home/neo/.ethash                count=2
INFO [01-19|17:47:06] Initialising Ethereum protocol
versions="[63 62]" network=123456
INFO [01-19|17:47:06] Loaded most recent local header          number=0
hash=611596…424d04 td=131072
INFO [01-19|17:47:06] Loaded most recent local full block      number=0
hash=611596…424d04 td=131072
INFO [01-19|17:47:06] Loaded most recent local fast block      number=0
hash=611596…424d04 td=131072
INFO [01-19|17:47:06] Loaded local transaction journal
transactions=0 dropped=0
INFO [01-19|17:47:06] Regenerated local transaction journal
transactions=0 accounts=0
INFO [01-19|17:47:06] Starting P2P networking
INFO [01-19|17:47:06] RLPx listener up
```

```
self="enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c46
93d2f470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5b6b17a8a4b40527@[
::]:30303?discport=0"
INFO [01-19|17:47:06] IPC endpoint opened: /home/neo/.ethereum/geth.ipc
INFO [01-19|17:47:06] HTTP endpoint opened: http://127.0.0.1:8545
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.10-stable-4bb3c89d/linux-amd64/go1.9.5
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0
rpc:1.0 txpool:1.0 web3:1.0

> INFO [01-19|17:47:09] Mapped network port
proto=tcp extport=30303 intport=30303 interface="UPNP IGDv1-IP1"
```

identity: 区块链的标示，随便填写，用于标示目前网络的名字

init: 指定创世块文件的位置，并创建初始块

datadir: 设置当前区块链网络数据存放的位置

port: 网络监听端口

rpc: 启动rpc通信，可以进行智能合约的部署和调试

rpcapi: 设置允许连接的rpc的客户端，一般为db,eth,net,web3

networkid: 设置当前区块链的网络ID，用于区分不同的网络，是一个数字

console: 启动命令行模式，可以在Geth中执行命令

2.4. 使用节点进行挖矿

2.4.1. 启动矿工开始挖矿

```
> miner.start(1)
```

这里的1表示只使用一个线程运行,第一次运行时将开始创建DAG文件，只需等待进度条到100，则将开始挖矿。 实际你看到的挖矿速度很快，这是因为我们已经在初始化创世区块时配置为:"nonce": "0x0000000000000042"。 “0x42”难度能让你在私有测试网链上快速挖以太币。

提示

挖矿时必然有矿工账户，而系统默认使用创建的第一个账号。

```
> miner.start(1)
INFO [01-19|21:06:43] Updated mining threads                       threads=1
INFO [01-19|21:06:43] Transaction pool price threshold updated
price=18000000000
INFO [01-19|21:06:43] Starting mining operation
null
> INFO [01-19|21:06:43] Commit new mining work
number=1 txs=0 uncles=0 elapsed=717.552µs
INFO [01-19|21:06:46] Generating ethash verification cache     epoch=0
percentage=91 elapsed=3.000s
INFO [01-19|21:06:46] Generated ethash verification cache      epoch=0
elapsed=3.273s
INFO [01-19|21:06:51] Generating DAG in progress               epoch=0
percentage=0  elapsed=5.056s
INFO [01-19|21:06:56] Generating DAG in progress               epoch=0
percentage=1  elapsed=10.140s
INFO [01-19|21:07:01] Generating DAG in progress               epoch=0
percentage=2  elapsed=15.119s
INFO [01-19|21:07:06] Generating DAG in progress               epoch=0
percentage=3  elapsed=19.924s
INFO [01-19|21:07:11] Generating DAG in progress               epoch=0
percentage=4  elapsed=24.739s
INFO [01-19|21:07:16] Generating DAG in progress               epoch=0
percentage=5  elapsed=29.473s
INFO [01-19|21:07:22] Generating DAG in progress               epoch=0
percentage=6  elapsed=35.641s
INFO [01-19|21:07:26] Generating DAG in progress               epoch=0
percentage=7  elapsed=40.374s
INFO [01-19|21:07:31] Generating DAG in progress               epoch=0
percentage=8  elapsed=45.134s
INFO [01-19|21:07:36] Generating DAG in progress               epoch=0
percentage=9  elapsed=49.908s
INFO [01-19|21:07:41] Generating DAG in progress               epoch=0
percentage=10 elapsed=54.633s
......
......
......
INFO [01-19|21:22:43] Generated ethash verification cache      epoch=0
elapsed=15m57.328s
```

```
INFO [01-19|21:22:47] Generating ethash verification cache     epoch=1
percentage=17 elapsed=3.031s
INFO [01-19|21:22:50] Generating ethash verification cache     epoch=1
percentage=34 elapsed=6.056s
INFO [01-19|21:22:53] Generating ethash verification cache     epoch=1
percentage=49 elapsed=9.562s
INFO [01-19|21:22:57] Generating ethash verification cache     epoch=1
percentage=70 elapsed=13.115s
INFO [01-19|21:23:00] Generating ethash verification cache     epoch=1
percentage=90 elapsed=16.123s
INFO [01-19|21:23:01] Generated ethash verification cache      epoch=1
elapsed=17.576s
INFO [01-19|21:23:19] Generating DAG in progress               epoch=1
percentage=0  elapsed=18.198s
INFO [01-19|21:23:32] Successfully sealed new block            number=1
hash=e2b5b9…9b1bfe
INFO [01-19|21:23:32] 🔨 mined potential block                  number=1
hash=e2b5b9…9b1bfe
INFO [01-19|21:23:32] Commit new mining work                   number=2
txs=0 uncles=0 elapsed=1.188ms
INFO [01-19|21:23:37] Generating DAG in progress               epoch=1
percentage=1  elapsed=35.913s
INFO [01-19|21:23:41] Successfully sealed new block            number=2
hash=62db3f…e27b50
INFO [01-19|21:23:41] 🔨 mined potential block                  number=2
hash=62db3f…e27b50
INFO [01-19|21:23:41] Commit new mining work                   number=3
txs=0 uncles=0 elapsed=772.239µs
INFO [01-19|21:23:43] Successfully sealed new block            number=3
hash=34384b…c387f2
INFO [01-19|21:23:43] 🔨 mined potential block                  number=3
hash=34384b…c387f2
INFO [01-19|21:23:43] Commit new mining work                   number=4
txs=0 uncles=0 elapsed=1.002ms
INFO [01-19|21:23:55] Generating DAG in progress               epoch=1
percentage=2  elapsed=53.757s
INFO [01-19|21:24:13] Generating DAG in progress               epoch=1
percentage=3  elapsed=1m11.561s
INFO [01-19|21:24:30] Generating DAG in progress               epoch=1
percentage=4  elapsed=1m28.986s
INFO [01-19|21:24:30] Successfully sealed new block            number=4
hash=681970…462135
INFO [01-19|21:24:30] 🔨 mined potential block                  number=4
hash=681970…462135
INFO [01-19|21:24:30] Commit new mining work                   number=5
txs=0 uncles=0 elapsed=833.629µs
INFO [01-19|21:24:36] Successfully sealed new block            number=5
hash=7b058b…d2f07a
INFO [01-19|21:24:36] 🔨 mined potential block                  number=5
hash=7b058b…d2f07a
INFO [01-19|21:24:36] Commit new mining work                   number=6
txs=0 uncles=0 elapsed=897.815µs
```

```
INFO [01-19|21:24:43] Successfully sealed new block            number=6
hash=a5fc3d…b1221e
INFO [01-19|21:24:43] 🔗 block reached canonical chain          number=1
hash=e2b5b9…9b1bfe
INFO [01-19|21:24:43] 🔨 mined potential block                  number=6
hash=a5fc3d…b1221e
INFO [01-19|21:24:43] Commit new mining work                   number=7
txs=0 uncles=0 elapsed=758.061µs
INFO [01-19|21:24:47] Successfully sealed new block            number=7
hash=003b02…e886fd
INFO [01-19|21:24:47] 🔗 block reached canonical chain          number=2
hash=62db3f…e27b50
INFO [01-19|21:24:47] 🔨 mined potential block                  number=7
hash=003b02…e886fd
INFO [01-19|21:24:47] Commit new mining work                   number=8
txs=0 uncles=0 elapsed=920.862µs
INFO [01-19|21:24:48] Generating DAG in progress               epoch=1
percentage=5  elapsed=1m46.827s
INFO [01-19|21:25:06] Generating DAG in progress               epoch=1
percentage=6  elapsed=2m4.338s
INFO [01-19|21:25:23] Successfully sealed new block            number=8
hash=fd23c9…361c65
INFO [01-19|21:25:23] 🔗 block reached canonical chain          number=3
hash=34384b…c387f2
INFO [01-19|21:25:23] 🔨 mined potential block                  number=8
hash=fd23c9…361c65
INFO [01-19|21:25:23] Commit new mining work                   number=9
txs=0 uncles=0 elapsed=825.737µs
INFO [01-19|21:25:23] Generating DAG in progress               epoch=1
percentage=7  elapsed=2m22.061s
```

2.4.2. 停止挖矿

```
> miner.stop()
true
>
```

2.4.3. 查看所挖金额

```
> eth.getBalance(eth.accounts[0])
```

```
70000000000000000000
```

2.5. 在创世链中制定矿工账号并为它充值

```
  "alloc": {
    "0xe8abf98484325fd6afc59b804ac15804b978e607": {
      "balance": "300000"
    },
    "0x013b5e735e1b48421dd3de3b931d6f03e769e22b": {
      "balance": "400000"
    }
  }
```

3. Blockchain explorer (区块链浏览器)

https://github.com/ethereumproject/explorer

4. 单机多实例演示

在没有条件安装虚拟机也没有多台服务器的情况下我们可以使用一台服务器运行多个实例的方法也可以实现多个节点运行环境。

```
cd ~
mkdir -p ethereum
cd ethereum
mkdir data{1,2}
```

创建文件 genesis.json

```
{
  "nonce": "0x0000000000000042",
  "difficulty": "0x020000",
  "mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000000000000000000",
  "timestamp": "0x00",
  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "extraData": "0x11bbe8db4e347b4e8c937c1c8370e4b5ed33adb3db69cbdb7a38e1e50b1b82fa",
  "gasLimit": "0x4c4b40",
  "config": {
      "chainId": 15,
      "homesteadBlock": 0,
      "eip155Block": 0,
      "eip158Block": 0
  },
  "alloc": { }
}
```

4.1. 实例一

```
geth --datadir ~/ethereum/data1 init genesis.json
geth --datadir="~/ethereum/data1" --networkid 123456 --port 30301 --rpc --rpcaddr="0.0.0.0" -
-rpccorsdomain "*" --rpcport 8541
```

启动后终端输出

```
neo@netkiller ~/ethereum % geth --datadir ~/ethereum/data1 init genesis.json
WARN [02-02|22:09:56] No etherbase set and no accounts found as default
INFO [02-02|22:09:56] Allocated cache and file handles
database=/home/neo/ethereum/data1/geth/chaindata cache=16 handles=16
INFO [02-02|22:09:56] Writing custom genesis block
INFO [02-02|22:09:56] Successfully wrote genesis state         database=chaindata
hash=611596…424d04
INFO [02-02|22:09:56] Allocated cache and file handles
database=/home/neo/ethereum/data1/geth/lightchaindata cache=16 handles=16
INFO [02-02|22:09:57] Writing custom genesis block
INFO [02-02|22:09:57] Successfully wrote genesis state         database=lightchaindata
```

```
hash=611596…424d04

neo@netkiller ~ % geth --datadir="~/ethereum/data1" --networkid 123456 --port 30301 --rpc --
rpcaddr="0.0.0.0" --rpccorsdomain "*" -rpcport 8541
WARN [02-02|22:36:02] No etherbase set and no accounts found as default
INFO [02-02|22:36:02] Starting peer-to-peer node               instance=Geth/v1.8.10-stable-
4bb3c89d/linux-amd64/go1.9.5
INFO [02-02|22:36:02] Allocated cache and file handles
database=/home/neo/ethereum/data1/geth/chaindata cache=128 handles=1024
INFO [02-02|22:36:02] Initialised chain configuration          config="{ChainID: 15
Homestead: 0 DAO: <nil> DAOSupport: false EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil>
Engine: unknown}"
INFO [02-02|22:36:02] Disk storage enabled for ethash caches
dir=/home/neo/ethereum/data1/geth/ethash count=3
INFO [02-02|22:36:02] Disk storage enabled for ethash DAGs     dir=/home/neo/.ethash
count=2
INFO [02-02|22:36:02] Initialising Ethereum protocol           versions="[63 62]"
network=123456
INFO [02-02|22:36:02] Loaded most recent local header          number=0 hash=611596…424d04
td=131072
INFO [02-02|22:36:02] Loaded most recent local full block      number=0 hash=611596…424d04
td=131072
INFO [02-02|22:36:02] Loaded most recent local fast block      number=0 hash=611596…424d04
td=131072
INFO [02-02|22:36:02] Loaded local transaction journal         transactions=0 dropped=0
INFO [02-02|22:36:02] Regenerated local transaction journal    transactions=0 accounts=0
INFO [02-02|22:36:02] Starting P2P networking
INFO [02-02|22:36:05] UDP listener up
self=enode://53433417f11d1d9a17f155cbaad2c4ec375af7b141e2989f049b572fc3f856d78f254e58fa82ed6e
ab48a16b7d625527214522ec0fd3e3af030b5b8dfdadc062@14.103.209.119:30301
INFO [02-02|22:36:05] HTTP endpoint opened: http://0.0.0.0:8541
INFO [02-02|22:36:05] IPC endpoint opened: /home/neo/ethereum/data1/geth.ipc
INFO [02-02|22:36:05] RLPx listener up
self=enode://53433417f11d1d9a17f155cbaad2c4ec375af7b141e2989f049b572fc3f856d78f254e58fa82ed6e
ab48a16b7d625527214522ec0fd3e3af030b5b8dfdadc062@14.103.209.119:30301
INFO [02-02|22:36:05] Mapped network port                      proto=udp extport=30301
intport=30301 interface="UPNP IGDv1-IP1"
INFO [02-02|22:36:07] Mapped network port                      proto=tcp extport=30301
intport=30301 interface="UPNP IGDv1-IP1"
```

4.2. 实例二

```
geth --datadir ~/ethereum/data2 init genesis.json
geth --datadir="~/ethereum/data2" --networkid 123456 --port 30302 --rpc --rpcaddr="0.0.0.0" -
-rpccorsdomain "*" --rpcport 8542
```

启动后控制台输出与实例一类似

4.3. 添加节点

开启一个新终端窗口，运行下面命令查看节点一的 enode 字符串

```
geth --exec 'admin.nodeInfo.enode' attach ethereum/data1/geth.ipc
"enode://53433417f11d1d9a17f155cbaad2c4ec375af7b141e2989f049b572fc3f856d78f254e58fa82ed6eab48
a16b7d625527214522ec0fd3e3af030b5b8dfdadc062@[::]:30301?discport=0"
```

进入节点二，并连接到节点一。

```
neo@netkiller ~ % geth attach ethereum/data2/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.10-stable-4bb3c89d/linux-amd64/go1.9.5
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
web3:1.0

>
admin.addPeer("enode://53433417f11d1d9a17f155cbaad2c4ec375af7b141e2989f049b572fc3f856d78f254e
58fa82ed6eab48a16b7d625527214522ec0fd3e3af030b5b8dfdadc062@[::]:30302")
true
>
admin.addPeer("enode://53433417f11d1d9a17f155cbaad2c4ec375af7b141e2989f049b572fc3f856d78f254e
58fa82ed6eab48a16b7d625527214522ec0fd3e3af030b5b8dfdadc062@[::]:30301")
true
```

查看节点

```
> admin.peers
[{
    caps: ["eth/63"],
    id:
"53433417f11d1d9a17f155cbaad2c4ec375af7b141e2989f049b572fc3f856d78f254e58fa82ed6eab48a16b7d62
5527214522ec0fd3e3af030b5b8dfdadc062",
    name: "Geth/v1.8.10-stable-4bb3c89d/linux-amd64/go1.9.5",
    network: {
      localAddress: "[::1]:51250",
      remoteAddress: "[::1]:30301"
    },
    protocols: {
      eth: {
        difficulty: 131072,
        head: "0x611596e7979cd4e7ca1531260fa706093a5492ecbdf58f20a39545397e424d04",
        version: 63
      }
    }
}]
```

至此，节点已经添加完毕。

```
> exit
```

退出

4.4. 节点测试

这里我们实现两个节点间的以太币转账。

现在两个节点上都没有任何账号

```
neo@netkiller ~ % geth --exec 'personal.listAccounts' attach ethereum/data1/geth.ipc
[]
neo@netkiller ~ % geth --exec 'personal.listAccounts' attach ethereum/data2/geth.ipc
[]
```

在两个节点上分别创建两个账号，一个是矿工账号，另一个是普通账号。

```
neo@netkiller ~ % geth --exec 'personal.newAccount("abc123")' attach ethereum/data1/geth.ipc
"0x5ad227e8d7e460713c78eebbe558473571edae72"

neo@netkiller ~ % geth --exec 'personal.newAccount("abc123")' attach ethereum/data1/geth.ipc
"0x3e822e05ee975e02be3f15f32b0fddced8d5bdd0"

neo@netkiller ~ % geth --exec 'personal.listAccounts' attach ethereum/data1/geth.ipc
["0x5ad227e8d7e460713c78eebbe558473571edae72", "0x3e822e05ee975e02be3f15f32b0fddced8d5bdd0"]
```

```
neo@netkiller ~ % geth --exec 'personal.newAccount("abc123")' attach ethereum/data2/geth.ipc
"0xa6df3e3c141e27726f4aeb21a5dab2e5c76c9565"

neo@netkiller ~ % geth --exec 'personal.newAccount("abc123")' attach ethereum/data2/geth.ipc
"0xa66c7b8b1c26856d284a0b962385babe02caa51d"

neo@netkiller ~ % geth --exec 'personal.listAccounts' attach ethereum/data2/geth.ipc
["0xa6df3e3c141e27726f4aeb21a5dab2e5c76c9565", "0xa66c7b8b1c26856d284a0b962385babe02caa51d"]
```

启动挖矿

```
geth --exec 'miner.start(1)' attach ethereum/data1/geth.ipc
geth --exec 'miner.start(1)' attach ethereum/data2/geth.ipc
```

如果正常运行，两个节点上的矿工账号都会有一定的以太币。而普通账号额度应该为零。

```
neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[0])' attach
ethereum/data1/geth.ipc
299438256000000000000

neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[1])' attach
ethereum/data1/geth.ipc
0

neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[1])' attach
ethereum/data1/geth.ipc
298000056000000000000

neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[1])' attach
ethereum/data2/geth.ipc
0
```

我们尝试从节点一矿工账号向节点二上的普通用户转账。

```
neo@netkiller ~ % geth attach ethereum/data1/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.10-stable-4bb3c89d/linux-amd64/go1.9.5
coinbase: 0x5ad227e8d7e460713c78eebbe558473571edae72
at block: 144 (Fri, 02 Feb 2018 23:24:35 HST)
 datadir: /home/neo/ethereum/data1
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
web3:1.0

> personal.listAccounts
["0x5ad227e8d7e460713c78eebbe558473571edae72"]

> personal.unlockAccount(eth.accounts[0], "abc123")
true

> eth.sendTransaction({from: "0x5ad227e8d7e460713c78eebbe558473571edae72", to:
"0xa66c7b8b1c26856d284a0b962385babe02caa51d", value: web3.toWei(1, "ether")})
"0x87c059d0769c8a74499ddd08c04a10f23b7681651615a28098d73ec63a943684"

> eth.pendingTransactions
[{
    blockHash: null,
    blockNumber: null,
    from: "0x5ad227e8d7e460713c78eebbe558473571edae72",
    gas: 90000,
    gasPrice: 18000000000,
    hash: "0x87c059d0769c8a74499ddd08c04a10f23b7681651615a28098d73ec63a943684",
    input: "0x",
    nonce: 2,
    r: "0xce004f964f268a00e90cadd4e8a685131aa34f37144f7e2e47dc7fe4ec784e55",
    s: "0x412209c18513a28422e62c4bdb85a223f190e133cf71990a87c570a3a53ae093",
    to: "0xa66c7b8b1c26856d284a0b962385babe02caa51d",
    transactionIndex: 0,
    v: "0x41",
    value: 1000000000000000000
}]
```

稍后一会，当使用 eth.pendingTransactions 查看挂起交易为空的时候，表示已经处理完毕。这时退出控制台。

```
> eth.pendingTransactions
[]
> exit
```

现在查看节点二上的第二个普通账号余额

```
neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[1])' attach
ethereum/data2/geth.ipc
1000000000000000000
```

转账成功

现在我们从节点二上的普通用户向节点一上的普通用户转账。

```
neo@netkiller ~ % geth attach ethereum/data2/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.10-stable-4bb3c89d/linux-amd64/go1.9.5
coinbase: 0xa6df3e3c141e27726f4aeb21a5dab2e5c76c9565
at block: 319 (Fri, 02 Feb 2018 23:50:07 HST)
 datadir: /home/neo/ethereum/data2
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
web3:1.0

> personal.unlockAccount(eth.accounts[1], "abc123")
true

> eth.sendTransaction({from: "0xa66c7b8b1c26856d284a0b962385babe02caa51d", to:
"0x3e822e05ee975e02be3f15f32b0fddced8d5bdd0", value: web3.toWei(0.1, "ether")})
"0x951bd161dfd000ff825379cb0644c4acd4afd4d3e1ac4f4c1c6009b3c2a1d366"

> eth.pendingTransactions
[]
> exit
```

查看两个普通账号的余额

```
neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[1])' attach
ethereum/data1/geth.ipc
100000000000000000
neo@netkiller ~ % geth --exec 'eth.getBalance(eth.accounts[1])' attach
ethereum/data2/geth.ipc
899622000000000000
```

5. 使用 pm2 启动以太坊

```
npm install -g pm2
```

创建 ~/geth.json 文件

```
[
  {
    "name"              : "geth",
    "cwd"               : "/usr/bin/",
    "script"            : "geth",
    "args"              : "--rpcapi eth,web3 --rpc --dev --
datadir /home/neo/ethereum",
    "log_date_format"   : "YYYY-MM-DD HH:mm Z",
    "out_file"      : "/home/neo/ethereum/log/geth_out.log",
    "error_file"    : "/home/neo/ethereum/log/geth_err.log",
    "log_file"      : "/home/neo/ethereum/log/geth_log.log",
    "merge_logs"        : false,
    "watch"             : false,
    "max_restarts"      : 10,
    "exec_interpreter"  : "none",
    "exec_mode"         : "fork_mode"
  }
]
```

启动以太坊

```
pm2 start geth.json
```

第 11 章 geth v1.8.16 命令详解

https://geth.ethereum.org

版本号

```
neo@MacBook-Pro ~/ethereum/web3 % geth version
Geth
Version: 1.8.16-stable
Architecture: amd64
Protocol Versions: [63 62]
Network Id: 1
Go Version: go1.10
Operating System: darwin
GOPATH=
GOROOT=/usr/local/opt/go/libexec
```

帮助信息

```
neo@MacBook-Pro ~/ethereum/web3 % geth --help
NAME:
   geth - the go-ethereum command line interface

   Copyright 2013-2017 The go-ethereum Authors

USAGE:
   geth [options] command [command options] [arguments...]

VERSION:
   1.8.16-stable

COMMANDS:
   account     Manage accounts
   attach      Start an interactive JavaScript environment
(connect to node)
   bug         opens a window to report a bug on the geth repo
   console     Start an interactive JavaScript environment
   copydb      Create a local chain from a target chaindata
```

```
folder
   dump        Dump a specific block from storage
   dumpconfig  Show configuration values
   export      Export blockchain into file
   import      Import a blockchain file
   init        Bootstrap and initialize a new genesis block
   js          Execute the specified JavaScript files
   license     Display license information
   makecache   Generate ethash verification cache (for testing)
   makedag     Generate ethash mining DAG (for testing)
   monitor     Monitor and visualize node metrics
   removedb    Remove blockchain and state databases
   version     Print version numbers
   wallet      Manage Ethereum presale wallets
   help, h     Shows a list of commands or help for one command

ETHEREUM OPTIONS:
  --config value                               TOML configuration
file
  --datadir "/Users/neo/Library/Ethereum"  Data directory for
the databases and keystore
  --keystore                                   Directory for the
keystore (default = inside the datadir)
  --nousb                                      Disables monitoring
for and managing USB hardware wallets
  --networkid value                            Network identifier
(integer, 1=Frontier, 2=Morden (disused), 3=Ropsten, 4=Rinkeby)
(default: 1)
  --testnet                                    Ropsten network: pre-
configured proof-of-work test network
  --rinkeby                                    Rinkeby network: pre-
configured proof-of-authority test network
  --syncmode "fast"                            Blockchain sync mode
("fast", "full", or "light")
  --gcmode value                               Blockchain garbage
collection mode ("full", "archive") (default: "full")
  --ethstats value                             Reporting URL of a
ethstats service (nodename:secret@host:port)
  --identity value                             Custom node name
  --lightserv value                            Maximum percentage of
time allowed for serving LES requests (0-90) (default: 0)
  --lightpeers value                           Maximum number of LES
client peers (default: 100)
  --lightkdf                                   Reduce key-derivation
RAM & CPU usage at some expense of KDF strength

DEVELOPER CHAIN OPTIONS:
  --dev               Ephemeral proof-of-authority network with
```

```
a pre-funded developer account, mining enabled
  --dev.period value  Block period to use in developer mode (0 =
mine only if transaction pending) (default: 0)

ETHASH OPTIONS:
  --ethash.cachedir                     Directory to store the
ethash verification caches (default = inside the datadir)
  --ethash.cachesinmem value            Number of recent ethash
caches to keep in memory (16MB each) (default: 2)
  --ethash.cachesondisk value           Number of recent ethash
caches to keep on disk (16MB each) (default: 3)
  --ethash.dagdir "/Users/neo/.ethash"  Directory to store the
ethash mining DAGs (default = inside home folder)
  --ethash.dagsinmem value              Number of recent ethash
mining DAGs to keep in memory (1+GB each) (default: 1)
  --ethash.dagsondisk value             Number of recent ethash
mining DAGs to keep on disk (1+GB each) (default: 2)

TRANSACTION POOL OPTIONS:
  --txpool.nolocals            Disables price exemptions for
locally submitted transactions
  --txpool.journal value       Disk journal for local
transaction to survive node restarts (default:
"transactions.rlp")
  --txpool.rejournal value     Time interval to regenerate the
local transaction journal (default: 1h0m0s)
  --txpool.pricelimit value    Minimum gas price limit to
enforce for acceptance into the pool (default: 1)
  --txpool.pricebump value     Price bump percentage to replace
an already existing transaction (default: 10)
  --txpool.accountslots value  Minimum number of executable
transaction slots guaranteed per account (default: 16)
  --txpool.globalslots value   Maximum number of executable
transaction slots for all accounts (default: 4096)
  --txpool.accountqueue value  Maximum number of non-executable
transaction slots permitted per account (default: 64)
  --txpool.globalqueue value   Maximum number of non-executable
transaction slots for all accounts (default: 1024)
  --txpool.lifetime value      Maximum amount of time non-
executable transaction are queued (default: 3h0m0s)

PERFORMANCE TUNING OPTIONS:
  --cache value            Megabytes of memory allocated to
internal caching (default: 1024)
  --cache.database value   Percentage of cache memory allowance
to use for database io (default: 75)
  --cache.gc value         Percentage of cache memory allowance
to use for trie pruning (default: 25)
```

```
  --trie-cache-gens value  Number of trie node generations to
keep in memory (default: 120)

ACCOUNT OPTIONS:
  --unlock value    Comma separated list of accounts to unlock
  --password value  Password file to use for non-interactive
password input

API AND CONSOLE OPTIONS:
  --rpc                  Enable the HTTP-RPC server
  --rpcaddr value        HTTP-RPC server listening interface
(default: "localhost")
  --rpcport value        HTTP-RPC server listening port
(default: 8545)
  --rpcapi value         API's offered over the HTTP-RPC
interface
  --ws                   Enable the WS-RPC server
  --wsaddr value         WS-RPC server listening interface
(default: "localhost")
  --wsport value         WS-RPC server listening port (default:
8546)
  --wsapi value          API's offered over the WS-RPC interface
  --wsorigins value      Origins from which to accept websockets
requests
  --ipcdisable           Disable the IPC-RPC server
  --ipcpath              Filename for IPC socket/pipe within the
datadir (explicit paths escape it)
  --rpccorsdomain value  Comma separated list of domains from
which to accept cross origin requests (browser enforced)
  --rpcvhosts value      Comma separated list of virtual
hostnames from which to accept requests (server enforced).
Accepts '*' wildcard. (default: "localhost")
  --jspath loadScript    JavaScript root path for loadScript
(default: ".")
  --exec value           Execute JavaScript statement
  --preload value        Comma separated list of JavaScript
files to preload into the console

NETWORKING OPTIONS:
  --bootnodes value     Comma separated enode URLs for P2P
discovery bootstrap (set v4+v5 instead for light servers)
  --bootnodesv4 value   Comma separated enode URLs for P2P v4
discovery bootstrap (light server, full nodes)
  --bootnodesv5 value   Comma separated enode URLs for P2P v5
discovery bootstrap (light server, light nodes)
  --port value          Network listening port (default: 30303)
  --maxpeers value      Maximum number of network peers (network
disabled if set to 0) (default: 25)
```

```
  --maxpendpeers value  Maximum number of pending connection
attempts (defaults used if set to 0) (default: 0)
  --nat value           NAT port mapping mechanism
(any|none|upnp|pmp|extip:<IP>) (default: "any")
  --nodiscover          Disables the peer discovery mechanism
(manual peer addition)
  --v5disc              Enables the experimental RLPx V5 (Topic
Discovery) mechanism
  --netrestrict value   Restricts network communication to the
given IP networks (CIDR masks)
  --nodekey value       P2P node key file
  --nodekeyhex value    P2P node key as hex (for testing)

MINER OPTIONS:
  --mine                    Enable mining
  --minerthreads value      Number of CPU threads to use for
mining (default: 8)
  --etherbase value         Public address for block mining
rewards (default = first account created) (default: "0")
  --targetgaslimit value    Target gas limit sets the artificial
target gas floor for the blocks to mine (default: 4712388)
  --gasprice "18000000000"  Minimal gas price to accept for
mining a transactions
  --extradata value         Block extra data set by the miner
(default = client version)

GAS PRICE ORACLE OPTIONS:
  --gpoblocks value      Number of recent blocks to check for
gas prices (default: 20)
  --gpopercentile value  Suggested gas price is the given
percentile of a set of recent transaction gas prices (default:
60)

VIRTUAL MACHINE OPTIONS:
  --vmdebug  Record information useful for VM and contract
debugging

LOGGING AND DEBUGGING OPTIONS:
  --metrics                 Enable metrics collection and
reporting
  --fakepow                 Disables proof-of-work verification
  --nocompaction            Disables db compaction after import
  --verbosity value         Logging verbosity: 0=silent,
1=error, 2=warn, 3=info, 4=debug, 5=detail (default: 3)
  --vmodule value           Per-module verbosity: comma-
separated list of <pattern>=<level> (e.g. eth/*=5,p2p=4)
  --backtrace value         Request a stack trace at a specific
logging statement (e.g. "block.go:271")
```

```
  --debug                    Prepends log messages with call-site
location (file and line number)
  --pprof                    Enable the pprof HTTP server
  --pprofaddr value          pprof HTTP server listening
interface (default: "127.0.0.1")
  --pprofport value          pprof HTTP server listening port
(default: 6060)
  --memprofilerate value     Turn on memory profiling with the
given rate (default: 524288)
  --blockprofilerate value   Turn on block profiling with the
given rate (default: 0)
  --cpuprofile value         Write CPU profile to the given file
  --trace value              Write execution trace to the given
file

WHISPER (EXPERIMENTAL) OPTIONS:
  --shh                        Enable Whisper
  --shh.maxmessagesize value   Max message size accepted
(default: 1048576)
  --shh.pow value              Minimum POW accepted (default:
0.2)

DEPRECATED OPTIONS:
  --fast   Enable fast syncing through state downloads
  --light  Enable light client mode

MISC OPTIONS:
  --help, -h  show help


COPYRIGHT:
   Copyright 2013-2017 The go-ethereum Authors
```

中文翻译

命令:

account　管理账户

attach　启动交互式JavaScript环境（连接到节点）

bug　　上报bug Issues

console　启动交互式JavaScript环境

copydb 从文件夹创建本地链
dump Dump（分析）一个特定的块存储
dumpconfig 显示配置值
export 导出区块链到文件
import 导入一个区块链文件
init 启动并初始化一个新的创世纪块
js 执行指定的JavaScript文件(多个)
license 显示许可信息
makecache 生成ethash验证缓存(用于测试)
makedag 生成ethash 挖矿DAG(用于测试)
monitor 监控和可视化节点指标
removedb 删除区块链和状态数据库
version 打印版本号
wallet 管理Ethereum预售钱包
help,h 显示一个命令或帮助一个命令列表
ETHEREUM选项:
--config value TOML 配置文件
--datadir “xxx” 数据库和keystore密钥的数据目录
--keystore keystore存放目录(默认在datadir内)
--nousb 禁用监控和管理USB硬件钱包
--networkid value 网络标识符(整型, 1=Frontier, 2=Morden (弃用), 3=Ropsten, 4=Rinkeby) (默认: 1)
--testnet Ropsten网络:预先配置的POW(proof-of-work)测试网络
--rinkeby Rinkeby网络: 预先配置的POA(proof-of-authority)测试网络
--syncmode "fast" 同步模式 ("fast", "full", or "light")
--ethstats value 上报ethstats service URL (nodename:secret@host:port)
--identity value 自定义节点名
--lightserv value 允许LES请求时间最大百分比(0 – 90)(默认值:0)
--lightpeers value 最大LES client peers数量(默认值:20)
--lightkdf 在KDF强度消费时降低key-derivation RAM&CPU使用

开发者（模式）选项:
--dev 使用POA共识网络，默认预分配一个开发者账户并且会自动开启挖矿。
--dev.period value 开发者模式下挖矿周期 (0 = 仅在交易时) (默认: 0)
ETHASH 选项:
--ethash.cachedir ethash验证缓存目录(默认 = datadir目录内)
--ethash.cachesinmem value 在内存保存的最近的ethash缓存个数 (每个缓存16MB) (默认: 2)
--ethash.cachesondisk value 在磁盘保存的最近的ethash缓存个数 (每个缓存16MB) (默认: 3)
--ethash.dagdir "" 存ethash DAGs目录 (默认 = 用户hom目录)
--ethash.dagsinmem value 在内存保存的最近的ethash DAGs 个数 (每个1GB以上) (默认: 1)
--ethash.dagsondisk value 在磁盘保存的最近的ethash DAGs 个数 (每个1GB以上) (默认: 2)
交易池选项:
--txpool.nolocals 为本地提交交易禁用价格豁免
--txpool.journal value 本地交易的磁盘日志：用于节点重启 (默认: "transactions.rlp")
--txpool.rejournal value 重新生成本地交易日志的时间间隔 (默认: 1小时)
--txpool.pricelimit value 加入交易池的最小的gas价格限制(默认: 1)
--txpool.pricebump value 价格波动百分比（相对之前已有交易） (默认: 10)
--txpool.accountslots value 每个帐户保证可执行的最少交易槽数量 (默认: 16)
--txpool.globalslots value 所有帐户可执行的最大交易槽数量 (默认: 4096)
--txpool.accountqueue value 每个帐户允许的最多非可执行交易槽数

量 (默认: 64)
--txpool.globalqueue value 所有帐户非可执行交易最大槽数量 (默认: 1024)
--txpool.lifetime value 非可执行交易最大入队时间(默认: 3小时)
性能调优的选项:
--cache value 分配给内部缓存的内存MB数量，缓存值(最低16 mb /数据库强制要求)(默认:128)
--trie-cache-gens value 保持在内存中产生的trie node数量(默认:120)
帐户选项:
--unlock value 需解锁账户用逗号分隔
--password value 用于非交互式密码输入的密码文件
API和控制台选项:
--rpc 启用HTTP-RPC服务器
--rpcaddr value HTTP-RPC服务器接口地址(默认值:“localhost”)
--rpcport value HTTP-RPC服务器监听端口(默认值:8545)
--rpcapi value 基于HTTP-RPC接口提供的API
--ws 启用WS-RPC服务器
--wsaddr value WS-RPC服务器监听接口地址(默认值:“localhost”)
--wsport value WS-RPC服务器监听端口(默认值:8546)
--wsapi value 基于WS-RPC的接口提供的API
--wsorigins value websockets请求允许的源
--ipcdisable 禁用IPC-RPC服务器
--ipcpath 包含在datadir里的IPC socket/pipe文件名(转义过的显式路径)
--rpccorsdomain value 允许跨域请求的域名列表(逗号分隔)(浏览器强制)
--jspath loadScript JavaScript加载脚本的根路径(默认值:“.”)
--exec value 执行JavaScript语句(只能结合console/attach使用)
--preload value 预加载到控制台的JavaScript文件列表(逗号分隔)
网络选项:
--bootnodes value 用于P2P发现引导的enode urls(逗号分隔)(对于

light servers用v4+v5代替)
--bootnodesv4 value 用于P2P v4发现引导的enode urls(逗号分隔) (light server, 全节点)
--bootnodesv5 value 用于P2P v5发现引导的enode urls(逗号分隔) (light server, 轻节点)
--port value 网卡监听端口(默认值:30303)
--maxpeers value 最大的网络节点数量(如果设置为0，网络将被禁用)(默认值:25)
--maxpendpeers value 最大尝试连接的数量(如果设置为0，则将使用默认值)(默认值:0)
--nat value NAT端口映射机制 (any|none|upnp|pmp|extip:<IP>) (默认: “any”)
--nodiscover 禁用节点发现机制(手动添加节点)
--v5disc 启用实验性的RLPx V5(Topic发现)机制
--nodekey value P2P节点密钥文件
--nodekeyhex value 十六进制的P2P节点密钥(用于测试)
矿工选项:
--mine 打开挖矿
--minerthreads value 挖矿使用的CPU线程数量(默认值:8)
--etherbase value 挖矿奖励地址(默认=第一个创建的帐户)(默认值: “0”)
--targetgaslimit value 目标gas限制： 设置最低gas限制（低于这个不会被挖？） (默认值:“4712388”)
--gasprice value 挖矿接受交易的最低gas价格
--extradata value 矿工设置的额外块数据(默认=client version)
GAS价格选项:
--gpoblocks value 用于检查gas价格的最近块的个数 (默认: 10)
--gpopercentile value 建议gas价参考最近交易的gas价的百分位数，(默认: 50)
虚拟机的选项:
--vmdebug 记录VM及合约调试信息

日志和调试选项:
--metrics 启用metrics收集和报告
--fakepow 禁用proof-of-work验证
--verbosity value 日志详细
度:0=silent, 1=error, 2=warn, 3=info, 4=debug, 5=detail (default: 3)
--vmodule value 每个模块详细度:以 <pattern>=<level>的逗号分隔列
表 (比如 eth/*=6,p2p=5)
--backtrace value 请求特定日志记录堆栈跟踪 (比如 “block.go:271”)
--debug 突出显示调用位置日志(文件名及行号)
--pprof 启用pprof HTTP服务器
--pprofaddr value pprof HTTP服务器监听接口(默认值:127.0.0.1)
--pprofport value pprof HTTP服务器监听端口(默认值:6060)
--memprofilerate value 按指定频率打开memory profiling (默
认:524288)
--blockprofilerate value 按指定频率打开block profiling (默认值:0)
--cpuprofile value 将CPU profile写入指定文件
--trace value 将execution trace写入指定文件
WHISPER实验选项:
--shh 启用Whisper
--shh.maxmessagesize value 可接受的最大的消息大小 (默认
值: 1048576)
--shh.pow value 可接受的最小的POW (默认值: 0.2)
弃用选项:
--fast 开启快速同步
--light 启用轻客户端模式
其他选项:
–help, -h 显示帮助

1. api 相关参数

rpcapi 启动后允许连接到系统的API协议

```
geth --networkid 100000 --rpc --rpcapi "db,eth,net,web3" --
rpccorsdomain "*" --datadir "/app/chain" --port "30303" console
```

系统默认监听 127.0.0.1 如果希望外部访问本机，需要通过--rpcaddr指定监听地址。

```
geth --networkid 123456 --rpc --rpcaddr="0.0.0.0" --
rpccorsdomain "*" --nodiscover
```

1.1. rpcapi

--rpcapi 可以控制访问内容

```
$ geth --rpc --rpcapi personal,db,eth,net,web3 --rinkeby
```

1.2. rpcaddr

默认是 127.0.0.1

HTTP endpoint closed: http://127.0.0.1:8545

通过 --rpcaddr="0.0.0.0" 指定监听地址

HTTP endpoint opened: http://0.0.0.0:8545

```
neo@netkiller ~/ethereum % geth --networkid 123456 --rpc --
rpcaddr="0.0.0.0" --rpccorsdomain "*" --nodiscover console
INFO [01-20|01:41:33] Starting peer-to-peer node
instance=Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1
INFO [01-20|01:41:33] Allocated cache and file handles
database=/home/neo/.ethereum/geth/chaindata cache=128
handles=1024
INFO [01-20|01:41:34] Initialised chain configuration
config="{ChainID: 15 Homestead: 0 DAO: <nil> DAOSupport: false
EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil> Engine:
unknown}"
INFO [01-20|01:41:34] Disk storage enabled for ethash caches
dir=/home/neo/.ethereum/geth/ethash count=3
INFO [01-20|01:41:34] Disk storage enabled for ethash DAGs
dir=/home/neo/.ethash                count=2
INFO [01-20|01:41:34] Initialising Ethereum protocol
versions="[63 62]" network=123456
INFO [01-20|01:41:34] Loaded most recent local header
number=531 hash=1a2707…3a27bc td=79083846
INFO [01-20|01:41:34] Loaded most recent local full block
number=531 hash=1a2707…3a27bc td=79083846
INFO [01-20|01:41:34] Loaded most recent local fast block
number=531 hash=1a2707…3a27bc td=79083846
INFO [01-20|01:41:34] Loaded local transaction journal
transactions=0 dropped=0
INFO [01-20|01:41:34] Regenerated local transaction journal
transactions=0 accounts=0
WARN [01-20|01:41:34] Blockchain not empty, fast sync disabled
INFO [01-20|01:41:34] Starting P2P networking
INFO [01-20|01:41:34] RLPx listener up
self="enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029
cf4a6c4693d2f470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5b
6b17a8a4b40527@[::]:30303?discport=0"
INFO [01-20|01:41:34] IPC endpoint opened:
/home/neo/.ethereum/geth.ipc
INFO [01-20|01:41:34] HTTP endpoint opened: http://0.0.0.0:8545
Welcome to the Geth JavaScript console!

instance: Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1
coinbase: 0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
at block: 531 (Tue, 14 Nov 2017 17:36:05 HST)
 datadir: /home/neo/.ethereum
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0
personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> INFO [01-20|01:41:40] Mapped network port
proto=tcp extport=30303 intport=30303 interface="UPNP IGDv1-IP1"
```

2. 启动 Websocket 端口

```
geth --syncmode light --rpc --rpcaddr 0.0.0.0 --rpcapi web3,eth
--ws --wsaddr 0.0.0.0 --wsapi web3,eth --wsorigins '*'
```

安装 websocket 测试工具 wscat

```
npm install -g wscat
```

测试 Websocket

```
wscat -c ws://127.0.0.1:8546
```

3. 日志

--verbosity 日志输出级别控制

```
geth --verbosity 0 console
```

日志输出到文件中。

```
geth --networkid 123456 --rpc --rpcaddr="0.0.0.0" --
rpccorsdomain "*"  2>> /tmp/geth.log
```

后台运行

```
neo@netkiller ~ % geth --networkid 123456 --rpc --
rpcaddr="0.0.0.0" --rpccorsdomain "*" 2>> /tmp/geth.log &
[1] 30075

neo@netkiller ~ % tail -f /tmp/geth.log
INFO [02-02|21:18:59] Got interrupt, shutting down...
INFO [02-02|21:18:59] HTTP endpoint closed: http://0.0.0.0:8545
INFO [02-02|21:18:59] IPC endpoint closed:
/home/neo/.ethereum/geth.ipc
INFO [02-02|21:18:59] Blockchain manager stopped
INFO [02-02|21:18:59] Stopping Ethereum protocol
INFO [02-02|21:18:59] Ethereum protocol stopped
INFO [02-02|21:18:59] Transaction pool stopped
INFO [02-02|21:18:59] Database closed
database=/home/neo/.ethereum/geth/chaindata
INFO [02-02|21:18:59] Mapped network port
proto=tcp extport=30303 intport=30303 interface="UPNP IGDv1-IP1"
WARN [02-02|21:19:00] Already shutting down, interrupt more to
panic. times=9
```

4. 控制台

Geth JavaScript console

```
neo@netkiller ~/ethereum % geth  --networkid 123456 console
INFO [01-19|22:14:52] Starting peer-to-peer node
instance=Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1
INFO [01-19|22:14:52] Allocated cache and file handles
database=/home/neo/.ethereum/geth/chaindata cache=128
handles=1024
INFO [01-19|22:14:52] Initialised chain configuration
config="{ChainID: 15 Homestead: 0 DAO: <nil> DAOSupport: false
EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil> Engine:
unknown}"
INFO [01-19|22:14:52] Disk storage enabled for ethash caches
dir=/home/neo/.ethereum/geth/ethash count=3
INFO [01-19|22:14:52] Disk storage enabled for ethash DAGs
dir=/home/neo/.ethash                count=2
INFO [01-19|22:14:52] Initialising Ethereum protocol
versions="[63 62]" network=123456
INFO [01-19|22:14:52] Loaded most recent local header
number=14 hash=70d7f1…45850a td=1966848
INFO [01-19|22:14:52] Loaded most recent local full block
number=14 hash=70d7f1…45850a td=1966848
INFO [01-19|22:14:52] Loaded most recent local fast block
number=14 hash=70d7f1…45850a td=1966848
INFO [01-19|22:14:52] Loaded local transaction journal
transactions=0 dropped=0
INFO [01-19|22:14:52] Regenerated local transaction journal
transactions=0 accounts=0
WARN [01-19|22:14:52] Blockchain not empty, fast sync disabled
INFO [01-19|22:14:52] Starting P2P networking

INFO [01-19|22:14:56] UDP listener up
self=enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029
cf4a6c4693d2f470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5
b6b17a8a4b40527@101.232.64.12:30303
INFO [01-19|22:14:56] RLPx listener up
self=enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029
cf4a6c4693d2f470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5
b6b17a8a4b40527@101.232.64.12:30303
INFO [01-19|22:14:56] IPC endpoint opened:
```

```
/home/neo/.ethereum/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1
coinbase: 0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
at block: 14 (Fri, 19 Jan 2018 21:27:16 HST)
 datadir: /home/neo/.ethereum
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0
personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

>
> INFO [01-19|22:14:56] Mapped network port
proto=udp extport=30303 intport=30303 interface="UPNP IGDv1-
IP1"
```

5. 连接控制台

一般测试启动我们使用 console，如果是正式启动无需使用 console。同事我们使用&符号使其进入后台运行。

```
neo@netkiller ~/ethereum % geth --networkid 123456 --rpc --
rpcaddr="0.0.0.0" --rpccorsdomain "*" --nodiscover &
```

进入控制台

```
neo@netkiller ~/ethereum % geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1
coinbase: 0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
at block: 531 (Tue, 14 Nov 2017 17:36:05 HST)
 datadir: /home/neo/.ethereum
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0
rpc:1.0 txpool:1.0 web3:1.0
```

退出控制台

```
> exit
```

5.1. 指定 geth.ipc 文件位置

```
geth --ipcpath ~/.ethereum/geth.ipc attach
```

5.2. IPC 方式连接

```
neo@netkiller ~ % geth attach ethereum/data1/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0
rpc:1.0 txpool:1.0 web3:1.0

>
```

5.3. TCP 连接控制台

连接远程控制台

```
neo@netkiller ~/ethereum % geth --exec 'eth.coinbase' attach
http://172.16.0.10:8545
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

5.4. WebSocket 方式

```
$ geth attach ws://191.168.1.1:8546
```

6. 账号管理

6.1. 新建账号

查看账号

```
neo@netkiller ~/ethereum % geth account list
Account #0: {83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6}
keystore:///home/neo/.ethereum/keystore/UTC--2018-01-20T04-04-06.786586541Z-
-83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
```

创建账号

```
neo@netkiller ~/ethereum % geth account new
Your new account is locked with a password. Please give a password. Do not
forget this password.
Passphrase:
Repeat passphrase:
Address: {e8abf98484325fd6afc59b804ac15804b978e607}
```

指定密码

```
$ echo "abc123" > password
$ geth --password /path/to/password account new
```

账号创建在 ~/.ethereum/keystore 目录下

```
Mac: ~/Library/Ethereum/keystore
Linux: ~/.ethereum/keystore
Windows: %APPDATA%/Ethereum/keystore
```

```
neo@netkiller ~/.ethereum/keystore % ll
total 8.0K
-rw------- 1 neo neo 491 Jan 19 18:04 UTC--2018-01-20T04-04-06.786586541Z-
-83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
-rw------- 1 neo neo 491 Jan 19 20:11 UTC--2018-01-20T06-11-23.608902164Z--
```

```
e8abf98484325fd6afc59b804ac15804b978e607
```

6.2. 查看账号

```
neo@netkiller ~/ethereum % geth account list
Account #0: {83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6}
keystore:///home/neo/.ethereum/keystore/UTC--2018-01-20T04-04-06.786586541Z-
-83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
Account #1: {e8abf98484325fd6afc59b804ac15804b978e607}
keystore:///home/neo/.ethereum/keystore/UTC--2018-01-20T06-11-23.608902164Z--
e8abf98484325fd6afc59b804ac15804b978e607
```

6.3. 从私钥导入以太坊地址

首先将私钥存储到文件中，然后导入，导入后需要输入密码。

```
[ethereum@netkiller ~]$ echo
"f592b7bf06ca9fd7696ba95d6ed8e357de6a2379b6d5fe1ffd53c6b4b063cd4a" > privatekey
[ethereum@netkiller ~]$ geth account import privatekey
INFO [04-27|17:25:38] Maximum peer count                       ETH=25 LES=0
total=25
Your new account is locked with a password. Please give a password. Do not
forget this password.
Passphrase:
Repeat passphrase:
Address: {372fda02e8a1eca513f2ee5901dc55b8b5dd7411}
```

查看导入文件

```
[ethereum@netkiller ~]$ ls
.ethereum/keystore/*372fda02e8a1eca513f2ee5901dc55b8b5dd7411
.ethereum/keystore/UTC--2018-04-27T09-25-48.189741023Z-
-372fda02e8a1eca513f2ee5901dc55b8b5dd7411

[ethereum@netkiller ~]$ cat .ethereum/keystore/UTC--2018-04-27T09-25-
48.189741023Z--372fda02e8a1eca513f2ee5901dc55b8b5dd7411
{"address":"372fda02e8a1eca513f2ee5901dc55b8b5dd7411","crypto":{"cipher":"aes-
128-
ctr","ciphertext":"8e23a0274a4d91bea61273ee3ce2951c292698df13b2214e8afab5c9595a7
2c4","cipherparams":
{"iv":"c5f4fd65e6b6da4f83d30f8d327905d1"},"kdf":"scrypt","kdfparams":
{"dklen":32,"n":262144,"p":1,"r":8,"salt":"593eb041c4425723af7ef58a55c03fc875749
cd0f7174ce011047e41205e6cdd"},"mac":"e1f3bebd85632db39aaa6ff03fa7641109c1fe46cfc
c4251c32d27ad002343eb"},"id":"ca863875-0702-49a8-b5c5-c46a50b95375","version":3}
```

最后一定记得删除私钥问题，因为这个私钥没有加密过，一旦泄露后果不堪设想。

```
[ethereum@netkiller ~]$ rm -rf privatekey
```

如果不存储文件可以单行执行，然后删除 history 历史文件

```
[ethereum@netkiller ~]$ geth account import <(echo
f592b7bf06ca9fd7696ba95d6ed8e357de6a2379b6d5fe1ffd53c6b4b063cd4a)
[ethereum@netkiller ~]$ >.bash_history
```

7. 配置自动解锁账号

创建password文件，在里面输入密码，每个账号一行密码如：

```
123456
123456
123456
```

启动参数

```
geth --rpc --rpcaddr="0.0.0.0" --rpccorsdomain="*" --unlock
'0,1,2' --password ~/.ethereum/password   --nodiscover --
maxpeers '5' --networkid '12345' --datadir '~/.ethereum'
console
```

```
neo@MacBook-Pro ~/ethereum % geth --networkid 123456 --rpc --
rpcaddr="0.0.0.0" --rpccorsdomain "*" --mine --minerthreads 1 -
-unlock 0 --password ./password
INFO [02-28|22:51:25] Maximum peer count
ETH=25 LES=0 total=25
INFO [02-28|22:51:25] Starting peer-to-peer node
instance=Geth/v1.8.1-stable/darwin-amd64/go1.10
INFO [02-28|22:51:25] Allocated cache and file handles
database=/Users/neo/Library/Ethereum/geth/chaindata cache=768
handles=128
INFO [02-28|22:51:25] Initialised chain configuration
config="{ChainID: 15 Homestead: 0 DAO: <nil> DAOSupport: false
EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil> Engine:
unknown}"
INFO [02-28|22:51:25] Disk storage enabled for ethash caches
dir=/Users/neo/Library/Ethereum/geth/ethash count=3
INFO [02-28|22:51:25] Disk storage enabled for ethash DAGs
dir=/Users/neo/.ethash                          count=2
INFO [02-28|22:51:25] Initialising Ethereum protocol
```

```
versions="[63 62]" network=123456
WARN [02-28|22:51:25] Head state missing, repairing chain
number=5373 hash=6a2b1b…9e3c38
INFO [02-28|22:51:25] Rewound blockchain to past state
number=5372 hash=f6900d…59872d
INFO [02-28|22:51:25] Loaded most recent local header
number=5373 hash=6a2b1b…9e3c38 td=2246785401
INFO [02-28|22:51:25] Loaded most recent local full block
number=5372 hash=f6900d…59872d td=2246064913
INFO [02-28|22:51:25] Loaded most recent local fast block
number=5373 hash=6a2b1b…9e3c38 td=2246785401
INFO [02-28|22:51:25] Loaded local transaction journal
transactions=0 dropped=0
INFO [02-28|22:51:25] Regenerated local transaction journal
transactions=0 accounts=0
WARN [02-28|22:51:25] Blockchain not empty, fast sync disabled
INFO [02-28|22:51:25] Starting P2P networking
INFO [02-28|22:51:28] UDP listener up
self=enode://f4e5cee0be36c8823423d3115c488f29f82b5f95c02a796cd8
0c4f47078726e0b2dc5b95407469275fb52a24ff0fb95b2a32f38d6eb95f417
a2f65268d4e3fa8@[::]:30303
INFO [02-28|22:51:28] RLPx listener up
self=enode://f4e5cee0be36c8823423d3115c488f29f82b5f95c02a796cd8
0c4f47078726e0b2dc5b95407469275fb52a24ff0fb95b2a32f38d6eb95f417
a2f65268d4e3fa8@[::]:30303
INFO [02-28|22:51:28] IPC endpoint opened
url=/Users/neo/Library/Ethereum/geth.ipc
INFO [02-28|22:51:28] HTTP endpoint opened
url=http://0.0.0.0:8545                 cors=*
vhosts=localhost
WARN [02-28|22:51:28] ------------------------------------------
--------------------------
WARN [02-28|22:51:28] Referring to accounts by order in the
keystore folder is dangerous!
WARN [02-28|22:51:28] This functionality is deprecated and will
be removed in the future!
WARN [02-28|22:51:28] Please use explicit addresses! (can
search via `geth account list`)
WARN [02-28|22:51:28] ------------------------------------------
--------------------------
INFO [02-28|22:51:29] Unlocked account
address=0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7
INFO [02-28|22:51:29] Transaction pool price threshold updated
price=18000000000
INFO [02-28|22:51:29] Etherbase automatically configured
address=0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7
INFO [02-28|22:51:29] Starting mining operation
INFO [02-28|22:51:29] Commit new mining work
```

```
number=5373 txs=0 uncles=0 elapsed=403.127µs
INFO [02-28|22:51:38] Successfully sealed new block
number=5373 hash=f70a5d…fe9b7b
INFO [02-28|22:51:38] 🔨 mined potential block
number=5373 hash=f70a5d…fe9b7b
```

8. 运行JS

```
neo@netkiller ~/ethereum % geth --exec "eth.blockNumber" attach
531
```

```
$ geth --exec 'loadScript("/tmp/checkbalances.js")' attach
http://123.123.123.123:8545
$ geth --jspath "/tmp" --exec 'loadScript("checkbalances.js")'
attach http://123.123.123.123:8545
```

9. 节点管理

```
geth --bootnodes
enode://pubkey1@ip1:port1,enode://pubkey2@ip2:port2,enode://pub
key3@ip3:port3
```

10. 启动挖矿

```
geth --mine
```

10.1. 挖矿线程数

默认为 2

```
geth --mine --minerthreads=16
```

10.2. 指定旷工账号

默认是第一个账号

```
geth --mine --minerthreads=16 --
etherbase=0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6
```

11. 运行智能合约

```
pragma solidity ^0.4.18;

contract Netkiller {
    string name;
    int num;
    function Netkiller() public{
        name = "default";
        num = 1;
    }
    function setName(string _name) public{
        name = _name;
    }
    function getName() public view returns(string){
        return name;
    }
    function setNum(int n) public{
        num = n;
    }
    function addNum(int m) public view returns(int res){
        res = m + num;
    }
}
```

```
solc --bin --abi --optimize -o output Netkiller.sol
```

var abi = JSON.parse('内容来自 Netkiller.abi')

var bytecode = "0x"+"内容来自 Netkiller.bin"

```
var abi = JSON.parse('[{"constant":true,"inputs":
[],"name":"getName","outputs":
```

```
[{"name":"","type":"string"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"n","type":"int256"}],"name":"setNum","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"funct
ion"},{"constant":false,"inputs":
[{"name":"_name","type":"string"}],"name":"setName","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"funct
ion"},{"constant":true,"inputs":
[{"name":"m","type":"int256"}],"name":"addNum","outputs":
[{"name":"res","type":"int256"}],"payable":false,"stateMutabili
ty":"view","type":"function"},{"inputs":
[],"payable":false,"stateMutability":"nonpayable","type":"const
ructor"}]');
var bytecode =
"0x6060604052341561000f57600080fd5b604080519081016040528060078
526020017f64656661756c74000000000000000000000000000000000000
000000000081525060009080519060200190610005a929190610067565b5060
01808190555061010c565b828054600181600116156101000203166002900049
060005260206002090601f016020900481019282601f106100a857805160ff
191683800011785556100d6565b828001600101855582156100d6579182015b8
28111156100d5578251825591602001919060010190610 0ba565b5b50905061
00e391906100e7565b5090565b61010991905b808211156101055760008160
0905550600101610 0ed565b5090565b90565b61036b8061011b6000396000f3
006060604052600436106100625760003 57c010000000000000000000000
0000000000000000000000000000000000900463ffffffff16806317d7de7c1461
00675780636a5aa5ec146100f5578063c47f002714610118578063cc13962a1
4610175575b600080fd5b34156100725760 0080fd5b61007a6101ac565b6040
518080602001828103825283818151815260200191508051906020019080838
360005b838110156100ba578082015181840152602081019050610 09f565b50
505050905090810190601f1680156100e7578082038051600183602003610 10
00a031916815260200191505b509250505060405180910390f35b3415610100
57600080fd5b6101166004808035906020019091905050610254565b005b341
561012357600080fd5b610173600480803590602001908201803590602001 90
8080601f016020809104026020016040519081016040528093929190818152 6
020018383808284378201915050505050509190505061025e565b005b341561
018057600080fd5b6101966004808035906020019091905050610278565b604
051808281526020019150506040518091039 0f35b6101b4610286565b600080
546001816001161561010002031660029004806 01f016020809104026020016
040519081016040528093929190818152602001828054600181600116156101 00
020316600290048015610 24a5780601f1061021f576101008083540402835 29
16020019161024a565b820191906000526020600020905b815481529060010 1
906020018083116102 2d57829003601f168201915b505050505090509056 5b8
060018190555050565b806000908051906020019061027492919061029a565b
5050565b600060015482019050919050565b602060405190810160405280600
081525090565b828054600181600116156101000203166002900490600052 60
2060002090601f016020900481019282601f106102db57805160ff191683800
1178555610309565b828001600101855582156103095791820 15b8281111561
030857825182559160200191906 00101906102ed565b5b5090506103169 1906
```

```
1031a565b5090565b61033c91905b8082111561033857600081600090555060
0101610320565b5090565b905600a165627a7a7230582002b42864089a104d4
d80ba0190924e09e15f3c0d0aa107463f49e2689e159b480029";
var gas = web3.eth.estimateGas({data: bytecode})
var deploy = {from:eth.coinbase, data: bytecode, gas: gas};
var address  = eth.coinbase;

personal.unlockAccount(eth.accounts[0],"chen1980")

var mycontract = web3.eth.contract(abi)

var instance = mycontract.new({from:eth.coinbase,
data:bytecode, gas:300000}, function(e, contract){
    if(!e) {

      if(!contract.address) {
        console.log("Contract transaction send:
TransactionHash: " + contract.transactionHash + " waiting to be
mined...");

      } else {
        console.log("Contract mined! Address: " +
contract.address);
        console.log(contract);
      }

    }
});

var test = mycontract.at(eth.coinbase)

test.addNum(567);
```

12. Ropsten测试网络

```
$ nohup geth --datadir $DATADIR \
             --unlock 0 \
             --password <(echo -n "SECRETPASSWORD") \
             --testnet \
             --fast \
             2>> $HOME/Desktop/ethereum/testnet/geth.log &
```

13. 静态节点

```
$ vim ~/.ethereum/static-nodes.json
```

```
[
        //由EthFans维护的长期节点，位置：广州

"enode://6427b7e7446bb05f22fe7ce9ea175ec05858953d75a5a6e4f99a6a
ec0779a8bd6276f1959a42fe5948acbe14bcd0652082dc546d3b37ae8f2aea4
1eba4eca43b@121.201.14.181:30303",
        //由秘猿维护的长期节点，位置：杭州

"enode://91922b12115c067005c574844c6bbdb114eb262f90b6355cec89e1
3b483c3e4669c6d63ec66b6e3ca7a3a462d28edb3c659e9fa05ed4c7234524e
582a8816743@120.27.164.92:13333",
        //由云币网维护的分别位于亚太，广东和北京的长期节点

"enode://3dde41a994b3b99f938f75ddf6d48318c78ddd869c70b48d00b922
190bb434fc5474f6250c143723f4387273d123e02f6a38f07d0311f240d2915
f6140e09850@207.226.141.212:30303",

"enode://7ab8fa90b204f2146c00939b8474549c544caa3598a0894fa639a5
cdbd992cbc6135fd776f8bcf97ae95fdaa3afbfa2d107fea71549119afd7ea5
7356b899be5@121.201.24.236:30303",

"enode://db81152a8296089b04a21ad9bf347df3ff0450ffc8215d9f50c400
ccf8d18963118010cacf03c4b71981cf9cac5394438cab3039e98db4d2aae58
59ab7d1793e@139.198.1.244:30303",
        //由ethfans社区成员点典维护的节点，位置：深圳

"enode://68dd1360f0a4ac362b41124692e31652ffe26f6f06a284ca11f3b5
14b3968594ac1f4320d1aa1ca343b06327c18a2e40eded74edfb3086e1baaa2
7ca24226b21@113.106.85.172:30303",
        //由银链科技维护的两个节点，分别位于香港、广州

"enode://58f6b6908286cefe43c166cfc4fed033c750caa1bc3f6e1e1e1507
752c0b91248addb3122f8557c5f8912e702285a160ab3a10203ae1eff3807ed
a25d6ed6478@45.113.71.186:30303",
```

"enode://87190a01c02cafb97e7f49672b4c3be2937cf79c3969e0b8e7b35c
ac28cebfbda52a13d56fd2113c726a1dd359c9476ccf7e60651439cef56e3a7
1039f6a4f5e@119.29.207.90:30303",
        //由中钞研究院维护的节点，位于杭州

"enode://d1fdd05a62fd9544eeb455e4f4d4bd8bb574138d82d8f909f3041d
0792e3401f8695133d39ad0a3aa5d217e3c5bed0511b531505a67b03607a909
ae9096720d2@120.26.129.121:30303",
        //由ethfans社区成员fsword维护的节点，位于上海

"enode://a1e9cf99eca94590ae776c8dd5c6c043a8c1f0375e9e391c9fb551
33385bf453ac3d3fb3ead8e63415b2ef99d54a19e2a7bc830cd1fdbbb283818
e3bcb0ea31e@182.254.209.254:30303",
        //云象科技维护的节点

"enode://562796b19d43d79dfb6160abd2d7bb78a2f2efd9501a0a767c0067
7e0fb3a4407235f813c3003682c2a421a58709c52f595827bc15708cc5f534f
55d0f8d03ad@121.40.199.54:30303",
        //币网维护的节点

"enode://fa2c17dcc83a6e2825668210abf7480452de4b13d8bdea8f301c3b
603701918bc4dade9e68d119d7a8214e90e7ea10a2782041c98951385d97bee
73358fb08f4@120.26.124.58:30303",
        //朝夕网络维护的节点

"enode://0b331b27e2976d797aed1d1464ac483a7f262860334cb5737a01a0
188da08d79226a6973adc5f2a2c1a20192b399161eee23a0d56ecf472cbe405
8d010ecc89f@47.89.49.61:30303",

"enode://0639f20fdb5af1fecd2f2bc0ddb648885483a5945686530e6b0466
78635d3435dd7b92fe34209f81ec6f003482aa78e407e5e6eb1b10be4773a2a
dbcf1fc1ba6@118.192.161.147:30303",
        //布鲁克Brockex广东亚太

"enode://fd2a5d30e4f3917ee640876cc57d72a8bf5ecf049e9106c95e60cf
306dd7a5dd68d1a295f3718af44a7083252686926d6e8a402f1abe6f805e10e
7281967db28@121.201.29.82:30303",

"enode://0d1b9eed7afe2d5878d5d8a4c2066b600a3bcac2e5730586421af2
24e93a58cd03cac75bf0b2a62fd8049cd3692a085758cc1e407c8b2c94bb069
814a5e8d0f0@209.9.106.245:30303",
        //成都以太云维护的节点

"enode://ca087a651571d04953187753af969f7deb1582af2a06a3048b90ad
b3f87d4c41973aac4b5e80449efc97154dac769a5ea447b123c3aaf7a2c2382

5a1558804dc@[::]:30303",

"enode://9b53b9d41d964f71db60d2198cfa9013fc7808d707c5e0a32da1e2
2d3cacd6adbae46901df6506a752d9d4e3791df29171315fbb86f7b09331a25
458158fe65b@[::]:30303"
]

14. JavaScript Console

14.1. personal 管理

14.1.1. 创建账号

```
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

指定密码创建用户

```
personal.newAccount("123")
```

14.1.2. 列出账号

列出所有账号

```
> personal.listAccounts
["0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6", "0xe8abf98484325fd6afc59b804ac15804b978e607"]
```

列出指定账号

```
> personal.listAccounts[1]
"0xe8abf98484325fd6afc59b804ac15804b978e607"

> personal.listAccounts[0]
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

14.1.3. 解锁账号

```
 > personal.unlockAccount(eth.accounts[0],"password")
```

指定过期时间，单位是毫秒，下面例子是 20 分钟

```
> personal.unlockAccount(eth.accounts[0],"password", 1000*60*20)
```

14.2. eth 管理

14.2.1. 矿工账号

查看默认旷工账号，系统中的第一个账号。

```
> eth.coinbase
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

查看账号列表

```
> eth.accounts
["0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6", "0xe8abf98484325fd6afc59b804ac15804b978e607"]

> eth.accounts[0]
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

14.2.2. 余额

```
> eth.getBalance(eth.accounts[0])
70000000000000000000
```

14.2.2.1. 单位转换

eth.getBalance()返回的余额是以太币的最小面额wei，将wei转换为以太币ether。

```
primary = eth.accounts[0]
balance = web3.fromWei(eth.getBalance(primary), "ether");
```

演示

```
> primary = eth.accounts[0]
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
> balance = web3.fromWei(eth.getBalance(primary), "ether");
70
```

14.2.2.2. 一次检查所有账号余额

定义函数

```
function checkAllBalances() {
```

```
  web3.eth.getAccounts(function(err, accounts) {
   accounts.forEach(function(id) {
    web3.eth.getBalance(id, function(err, balance) {
     console.log("" + id + ":\tbalance: " + web3.fromWei(balance, "ether") + " ether");
   });
  });
 });
};
```

运行函数

```
checkAllBalances()
```

输出演示

```
> function checkAllBalances() {
...   web3.eth.getAccounts(function(err, accounts) {
.........    accounts.forEach(function(id) {
.............     web3.eth.getBalance(id, function(err, balance) {
.....................      console.log("" + id + ":\tbalance: " + web3.fromWei(balance, "ether")
+ " ether");
.....................    });
.............   });
.........  });
... };
undefined
> checkAllBalances()
0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6:     balance: 929 ether
0xe8abf98484325fd6afc59b804ac15804b978e607:     balance: 11 ether
undefined
```

14.2.3. 解锁账号

```
> personal.unlockAccount(eth.accounts[0],"password",50000)
true
```

14.2.4. 转账

```
personal.unlockAccount("0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6", "", 300)
eth.sendTransaction({from: '0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6', to:
'0xe8abf98484325fd6afc59b804ac15804b978e607', value: web3.toWei(1, "ether")})
```

默认矿工账号

```
> eth.coinbase
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

查看系统中的账号，如果没有请参考上面章节创建

```
> eth.accounts
["0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6", "0xe8abf98484325fd6afc59b804ac15804b978e607"]
```

转出账号中又 285 个以太币

```
> web3.fromWei(eth.getBalance(eth.accounts[0]))
285
```

转入账号目前是 0

```
> web3.fromWei(eth.getBalance(eth.accounts[1]))
0
```

解锁传出账号，否则不能转出。personal.unlockAccount(账号, 密码, 300)

```
> personal.unlockAccount("0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6", "", 300)
true
```

转账操作

```
> eth.sendTransaction({from: '0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6', to:
'0xe8abf98484325fd6afc59b804ac15804b978e607', value: web3.toWei(10, "ether")})
"0xb0674a7fee52555d8712f3a1f0f30fbbbf67ff7b5b4b53ab5d131262613215c6"
```

如果你现在查看转入账号，你会发现余额仍然是 0 ，交易还未成功写进区块，写进区块的方式是挖矿，所以你必须执行挖矿

```
> miner.start(1)
null
```

稍后几分钟，再次查看转入账号，将会看到有10个以太币入账。传出账号会减少10个以太币，同时仍然继续挖矿中。

```
> web3.fromWei(eth.getBalance(eth.accounts[1]))
10
```

从账号0想账号1转账

```
> personal.unlockAccount(eth.accounts[0],"password",50000)
true
> eth.sendTransaction({from: eth.accounts[0], to: eth.accounts[1], value: web3.toWei(1,
"ether")})
```

14.2.5. 查看挂起的交易

```
> eth.pendingTransactions
[{
    blockHash: null,
    blockNumber: null,
    from: "0x5fba50fce50baf0b8a7314200ba46336958ac97e",
    gas: 90000,
    gasPrice: 20000000000,
    hash: "0x51a75422f79fa96e70a0c1481851bc9f827868c44203b68d74f9815ffb367d5f",
    input: "0x",
    nonce: 0,
    r: "0x5632a8ade4a767dbd949ba1042cb33f98dd0722ab999ba18e1454d19d8bd1f6d",
    s: "0x515dcfa3de297f0c956ad9a061a5561f47cc9ccbb0a547cda59193c77fcbe3f7",
    to: "0x0a8c35653d8b229c16f0c9ce6f63cffb877cfdcf",
    transactionIndex: 0,
    v: "0x42",
```

```
    value: 1000000000000000000
}]
```

如果返回空值，表示交易全部完成。

```
> eth.pendingTransactions
[]
```

14.2.6. 查看当前区块总数

```
> eth.blockNumber
719
```

14.2.7. 查看当前Gas价格

```
> web3.eth.gasPrice
5000000000

> web3.fromWei(web3.eth.gasPrice.toString())
"0.000000018"
```

14.2.8. 评估执行花费的GAS

```
> eth.estimateGas({from:eth.accounts[1], to: eth.accounts[2], value:1234000000000000000})
21000
```

14.2.9. 查看区块信息

```
> eth.getBlock(1)
{
  difficulty: 131072,
  extraData: "0xd783010702846765746885676f312e398777696e646f7773",
  gasLimit: 4995119,
  gasUsed: 0,
  hash: "0x62bc2fc0fd647b43013ba75e65a3f16520cebb60cadc0995965c891d4266a88c",
  logsBloom:
"0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000",
  miner: "0xba1821227ebad7e88a51078f6219b392fd601822",
  mixHash: "0x80ad2e29d057e0f6bb9d03f12a252fdc65cb520248fa0023ae6624c997b5d5f2",
  nonce: "0x3d743ef28ba8c8c8",
```

```
  number: 1,
  parentHash: "0x611596e7979cd4e7ca1531260fa706093a5492ecbdf58f20a39545397e424d04",
  receiptsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
  size: 535,
  stateRoot: "0x418d9f66e33cbd41fd524b87c10c5fc75f8a6360896967d827a2d56a0c8aadda",
  timestamp: 1510715249,
  totalDifficulty: 262144,
  transactions: [],
  transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  uncles: []
}
> eth.getBlock(2)
{
  difficulty: 131072,
  extraData: "0xd783010702846765746885676f312e398777696e646f7773",
  gasLimit: 4990242,
  gasUsed: 0,
  hash: "0x3323c566fd3501c29e8b026c4a9d2e83a8dd62d153a32b5f879120e999013d07",
  logsBloom:
"0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000",
  miner: "0xba1821227ebad7e88a51078f6219b392fd601822",
  mixHash: "0xc415fc64f1bc13842455f8289d33842260c1578a32b5befded5b4293e7a92276",
  nonce: "0x4f4a0e65412663db",
  number: 2,
  parentHash: "0x62bc2fc0fd647b43013ba75e65a3f16520cebb60cadc0995965c891d4266a88c",
  receiptsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
  size: 535,
  stateRoot: "0x39ccb22ff5ca19a2340e137fa64acedb704d42827a753780b53a0aff8922e403",
  timestamp: 1510715341,
  totalDifficulty: 393216,
  transactions: [],
  transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  uncles: []
}
```

14.2.10. 返回交易信息

```
> web3.eth.getTransaction("0x4e27a477e128b200239bc2ecd899077c6ae064da963a919fef41bcc7462aec8d")
{
  blockHash: "0x59a9905831e7ae3cb9e7c6f125cf48e2688ef4b39317838f6f6b6c8837d01404",
  blockNumber: 4367,
  from: "0x8efb99ec55bcfbe2cfe47918f2d9e55fa732111f",
  gas: 21000,
  gasPrice: 18000000000,
  hash: "0x4e27a477e128b200239bc2ecd899077c6ae064da963a919fef41bcc7462aec8d",
  input: "0x",
  nonce: 15,
  r: "0xa297401df3a1fb0298cbc1dd609deebe9ded319fadc55934ecef4d525198215",
  s: "0x780d8c46bc8d1bb89ae9d78055307d9d68a4f89ba699ef86d3f8ba88383139a6",
  to: "0xf0688330101d53bd0c6ede2ef04d33c2010e9a5d",
  transactionIndex: 0,
  v: "0x42",
  value: 999622000000000000
}
```

14.2.11. 返回交易收据

```
>
web3.eth.getTransactionReceipt("0x4e27a477e128b200239bc2ecd899077c6ae064da963a919fef41bcc7462aec
8d")
{
  blockHash: "0x59a9905831e7ae3cb9e7c6f125cf48e2688ef4b39317838f6f6b6c8837d01404",
  blockNumber: 4367,
  contractAddress: null,
  cumulativeGasUsed: 21000,
  from: "0x8efb99ec55bcfbe2cfe47918f2d9e55fa732111f",
  gasUsed: 21000,
  logs: [],
  logsBloom:
"0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000",
  root: "0x69d278e7594e8a5affbebc84380516437f3e4b241b272e6eaebf752af9c76047",
  to: "0xf0688330101d53bd0c6ede2ef04d33c2010e9a5d",
  transactionHash: "0x4e27a477e128b200239bc2ecd899077c6ae064da963a919fef41bcc7462aec8d",
  transactionIndex: 0
}
```

返回值:

Object - 交易的收据对象，如果找不到返回null
blockHash: String - 32字节，这个交易所在区块的哈希。
blockNumber: Number - 交易所在区块的块号。
transactionHash: String - 32字节，交易的哈希值。
transactionIndex: Number - 交易在区块里面的序号，整数。
from: String - 20字节，交易发送者的地址。
to: String - 20字节，交易接收者的地址。如果是一个合约创建的交易，返回null。
cumulativeGasUsed: Number - 当前交易执行后累计花费的gas总值10。
gasUsed: Number - 执行当前这个交易单独花费的gas。
contractAddress: String - 20字节，创建的合约地址。如果是一个合约创建交易，返回合约地址，其它情况返回null。
logs: Array - 这个交易产生的日志对象数组。

14.2.12. eth.syncing 同步状态

```
> eth.syncing
{
  currentBlock: 645835,
  highestBlock: 5439559,
  knownStates: 1090968,
  pulledStates: 1080746,
  startingBlock: 0
}
```

显示百分比

```
console.log(parseInt(eth.syncing.currentBlock/eth.syncing.highestBlock*100,10)+'%')
```

剩余块数

```
eth.syncing.highestBlock - eth.syncing.currentBlock

setInterval(function(){
       console.log(eth.syncing.highestBlock - eth.syncing.currentBlock)
},5000);
```

进度监控

```
var lastPercentage = 0;var lastBlocksToGo = 0;var timeInterval = 10000;
setInterval(function(){
    var percentage = eth.syncing.currentBlock/eth.syncing.highestBlock*100;
    var percentagePerTime = percentage - lastPercentage;
    var blocksToGo = eth.syncing.highestBlock - eth.syncing.currentBlock;
    var bps = (lastBlocksToGo - blocksToGo) / (timeInterval / 1000)
    var etas = 100 / percentagePerTime * (timeInterval / 1000)

    var etaM = parseInt(etas/60,10);
    console.log(parseInt(percentage,10)+'% ETA: '+etaM+' minutes @ '+bps+'bps');

    lastPercentage = percentage;lastBlocksToGo = blocksToGo;
},timeInterval);
```

14.2.13. 查看智能合约编译器

```
> eth.compile
{
  lll: function(),
  serpent: function(),
  solidity: function()
}
```

14.3. web3

14.3.1. Ether币的基本单位

Ether币最小的单位是Wei，也是命令行默认的单位, 然后每1000个进一个单位，依次是

kwei (1000 Wei)
mwei (1000 KWei)
gwei (1000 mwei)
szabo (1000 gwei)
finney (1000 szabo)
ether (1000 finney)

如何进行ether 和 Wei之间的转换，简单地说就是就是1 以太币 = 1000000000000000000 Wei （这就是上一站章中为何我们转移0.01个以太币，结果却显示很长的原因）

单位转换

```
> web3.fromWei(10000000000000000,"ether")
"0.01"
```

14.3.2. web3.toWei

```
Ether—> Wei

> web3.toWei(1)
"1000000000000000000"
> web3.toWei(1.3423423)
"1342342300000000000"
> web3.toWei(0.00034)
"340000000000000"
```

14.3.3. web3.fromWei

```
Wei —> Ether

> web3.fromWei(10000000000000000)
"0.01"
> web3.fromWei(1000000000000000000)
"1"
>
```

14.4. admin 管理

```
> admin
{
  datadir: "/home/ethereum/.ethereum",
  nodeInfo: {
    enode:
"enode://89a7bfd4472dff266ccdccc76bc75586a9005144df54d4575a9e07d9b2aee8aae6c8004cc98c6212d5f0f4d
5e3a524617abd7e7d6f39d5df053f94b7250e5610@[::]:30303?discport=0",
    id:
"89a7bfd4472dff266ccdccc76bc75586a9005144df54d4575a9e07d9b2aee8aae6c8004cc98c6212d5f0f4d5e3a5246
17abd7e7d6f39d5df053f94b7250e5610",
    ip: "::",
```

```
    listenAddr: "[::]:30303",
    name: "Geth/v1.8.2-stable/linux-amd64/go1.8.3",
    ports: {
      discovery: 0,
      listener: 30303
    },
    protocols: {
      les: {
        config: {...},
        difficulty: 17179869184,
        genesis: "0xd4e56740f876aef8c010b86a40d5f56745a118d0906a34e69aec8c0db1cb8fa3",
        head: "0xd4e56740f876aef8c010b86a40d5f56745a118d0906a34e69aec8c0db1cb8fa3",
        network: 1
      }
    }
  },
  peers: [],
  addPeer: function(),
  clearHistory: function(),
  exportChain: function(),
  getDatadir: function(callback),
  getNodeInfo: function(callback),
  getPeers: function(callback),
  importChain: function(),
  removePeer: function(),
  sleep: function github.com/ethereum/go-ethereum/console.(*bridge).Sleep-fm(),
  sleepBlocks: function github.com/ethereum/go-ethereum/console.(*bridge).SleepBlocks-fm(),
  startRPC: function(),
  startWS: function(),
  stopRPC: function(),
  stopWS: function()
}
```

14.4.1. 看看 networkid

```
> admin.nodeInfo.protocols.eth.network
123456
```

14.4.2. 节点管理

可以通过admin.addPeer()方法连接到其他节点，两个节点要想联通，必须保证网络是相通的，并且要指定相同的networkid

提示

注意去掉 --nodiscover 参数

确保网络可用

```
> net.listening
true
```

14.4.2.1. 显示节点

显示当前节点信息

```
> admin.nodeInfo
{
  enode:
"enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f470b642fd2cc3507f7e851df
60aaeb730a1270b7a477f91ec5b6b17a8a4b40527@14.103.209.119:30303",
  id:
"9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f470b642fd2cc3507f7e851df60aaeb73
0a1270b7a477f91ec5b6b17a8a4b40527",
  ip: "14.103.209.119",
  listenAddr: "[::]:30303",
  name: "Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1",
  ports: {
    discovery: 30303,
    listener: 30303
  },
  protocols: {
    eth: {
      difficulty: 108754979,
      genesis: "0x611596e7979cd4e7ca1531260fa706093a5492ecbdf58f20a39545397e424d04",
      head: "0x61330b27cfbfaecbb36bb8666cbe0564c1e0bdecfdcd153622d8c2ca2b82786e",
      network: 123456
    }
  }
}
```

节点地址

```
> admin.nodeInfo.enode
"enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f470b642fd2cc3507f7e851df
60aaeb730a1270b7a477f91ec5b6b17a8a4b40527@[::]:30303?discport=0"
```

[::] 是 ipv6 地址，可以改为 ipv4 地址。

14.4.2.2. 添加节点

```
>
admin.addPeer('enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f470b642fd2
cc3507f7e851df60aaeb730a1270b7a477f91ec5b6b17a8a4b40527@172.16.0.1:30303')
```

```
admin.addPeer('enode://6427b7e7446bb05f22fe7ce9ea175ec05858953d75a5a6e4f99a6aec0779a8bd6276f1959
a42fe5948acbe14bcd0652082dc546d3b37ae8f2aea41eba4eca43b@121.201.14.181:30303');
admin.addPeer('enode://91922b12115c067005c574844c6bbdb114eb262f90b6355cec89e13b483c3e4669c6d63ec
66b6e3ca7a3a462d28edb3c659e9fa05ed4c7234524e582a8816743@120.27.164.92:13333');
admin.addPeer('enode://3dde41a994b3b99f938f75ddf6d48318c78ddd869c70b48d00b922190bb434fc5474f6250
c143723f4387273d123e02f6a38f07d0311f240d2915f6140e09850@207.226.141.212:30303');
admin.addPeer('enode://7ab8fa90b204f2146c00939b8474549c544caa3598a0894fa639a5cdbd992cbc6135fd776
f8bcf97ae95fdaa3afbfa2d107fea71549119afd7ea57356b899be5@121.201.24.236:30303');
admin.addPeer('enode://db81152a8296089b04a21ad9bf347df3ff0450ffc8215d9f50c400ccf8d18963118010cac
f03c4b71981cf9cac5394438cab3039e98db4d2aae5859ab7d1793e@139.198.1.244:30303');
admin.addPeer('enode://68dd1360f0a4ac362b41124692e31652ffe26f6f06a284ca11f3b514b3968594ac1f4320d
1aa1ca343b06327c18a2e40eded74edfb3086e1baaa27ca24226b21@113.106.85.172:30303');
admin.addPeer('enode://58f6b6908286cefe43c166cfc4fed033c750caa1bc3f6e1e1e1507752c0b91248addb3122
```

```
f8557c5f8912e702285a160ab3a10203ae1eff3807eda25d6ed6478@45.113.71.186:30303');
admin.addPeer('enode://87190a01c02cafb97e7f49672b4c3be2937cf79c3969e0b8e7b35cac28cebfbda52a13d56
fd2113c726a1dd359c9476ccf7e60651439cef56e3a71039f6a4f5e@119.29.207.90:30303');
admin.addPeer('enode://d1fdd05a62fd9544eeb455e4f4d4bd8bb574138d82d8f909f3041d0792e3401f8695133d3
9ad0a3aa5d217e3c5bed0511b531505a67b03607a909ae9096720d2@120.26.129.121:30303');
admin.addPeer('enode://a1e9cf99eca94590ae776c8dd5c6c043a8c1f0375e9e391c9fb55133385bf453ac3d3fb3e
ad8e63415b2ef99d54a19e2a7bc830cd1fdbbb283818e3bcb0ea31e@182.254.209.254:30303');
admin.addPeer('enode://562796b19d43d79dfb6160abd2d7bb78a2f2efd9501a0a767c00677e0fb3a4407235f813c
3003682c2a421a58709c52f595827bc15708cc5f534f55d0f8d03ad@121.40.199.54:30303');
admin.addPeer('enode://fa2c17dcc83a6e2825668210abf7480452de4b13d8bdea8f301c3b603701918bc4dade9e6
8d119d7a8214e90e7ea10a2782041c98951385d97bee73358fb08f4@120.26.124.58:30303');
admin.addPeer('enode://0b331b27e2976d797aed1d1464ac483a7f262860334cb5737a01a0188da08d79226a6973a
dc5f2a2c1a20192b399161eee23a0d56ecf472cbe4058d010ecc89f@47.89.49.61:30303');
admin.addPeer('enode://0639f20fdb5af1fecd2f2bc0ddb648885483a5945686530e6b046678635d3435dd7b92fe3
4209f81ec6f003482aa78e407e5e6eb1b10be4773a2adbcf1fc1ba6@118.192.161.147:30303');
admin.addPeer('enode://fd2a5d30e4f3917ee640876cc57d72a8bf5ecf049e9106c95e60cf306dd7a5dd68d1a295f
3718af44a7083252686926d6e8a402f1abe6f805e10e7281967db28@121.201.29.82:30303');
admin.addPeer('enode://0d1b9eed7afe2d5878d5d8a4c2066b600a3bcac2e5730586421af224e93a58cd03cac75bf
0b2a62fd8049cd3692a085758cc1e407c8b2c94bb069814a5e8d0f0@209.9.106.245:30303');
admin.addPeer('enode://ca087a651571d04953187753af969f7deb1582af2a06a3048b90adb3f87d4c41973aac4b5
e80449efc97154dac769a5ea447b123c3aaf7a2c23825a1558804dc@182.150.37.23:30303');
admin.addPeer('enode://9b53b9d41d964f71db60d2198cfa9013fc7808d707c5e0a32da1e22d3cacd6adbae46901d
f6506a752d9d4e3791df29171315fbb86f7b09331a25458158fe65b@182.150.37.24:30303');
```

14.4.2.3. 查看节点

查看节点数量

```
> net.peerCount
1
```

查看节点地址

```
> admin.peers
[{
    caps: ["eth/62", "eth/63", "par/1", "par/2", "pip/1"],
    id:
"a7bbd8fb72e02681b027908f14fd2dbd80e35a1477d7d9d4dc19ed34420be26fe9f991c83a83e4ab8aa371ffbb14949
4471f30216bc2f662d1ebc6d01811c7a2",
    name: "Parity/v1.7.12-stable-9b796e8-20180121/x86_64-linux-gnu/rustc1.21.0",
    network: {
      localAddress: "172.16.0.1:34092",
      remoteAddress: "52.67.171.152:30388"
    },
    protocols: {
      eth: "handshake"
    }
}]
```

列出节点IP地址

```
admin.peers.forEach(function(p) {console.log(p.network.remoteAddress);})
```

```
> admin.peers.forEach(function(p) {console.log(p.network.remoteAddress);})
52.90.40.206:30303
120.79.161.22:30303
13.229.198.234:30303
169.0.182.34:30303
23.111.151.158:30304
120.27.196.141:30303
118.193.93.174:30303
35.229.242.1:30303
159.203.32.64:30303
undefined
>
```

14.4.2.4. networkid

```
> admin.nodeInfo.protocols.les.network
1
```

14.5. miner 挖矿管理

14.5.1. 开始挖矿

```
> miner.start(2)
null
```

过几分钟后运行

```
> web3.fromWei(eth.getBalance(eth.coinbase), "ether")
30
```

这时我们已经看到已经产生了30个以太币。

14.5.2. 停止挖矿

```
> miner.stop()
true
>
```

14.5.3. 设置默认矿工账号

默认挖矿使用系统中的第一个账号，你可以使用 miner.setEtherbase() 指定账号。

```
> miner.setEtherbase("0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6")
true
```

```
> eth.accounts
["0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6", "0xe8abf98484325fd6afc59b804ac15804b978e607",
"0x013b5e735e1b48421dd3de3b931d6f03e769e22b"]

> eth.coinbase
"0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6"
```

```
> miner.setEtherbase("0xe8abf98484325fd6afc59b804ac15804b978e607")
true

> eth.coinbase
"0xe8abf98484325fd6afc59b804ac15804b978e607"
```

14.6. txpool 管理

14.6.1. txpool.status

查看状态

```
> txpool.status
{
  pending: 0,
  queued: 0
}
```

例如做一笔转账

```
> amount = web3.toWei(5,'ether')
"5000000000000000000"
> eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:amount})

> txpool.status
{
  pending: 1,
  queued: 0
}

> miner.start(1);admin.sleepBlocks(1);miner.stop();

> txpool.status
{
  pending: 0,
  queued: 0
}

> web3.fromWei(eth.getBalance(eth.accounts[1]),'ether')
5
```

14.7. net

14.7.1. 监听状态

```
> net.listening
true
```

第 12 章 Wallet

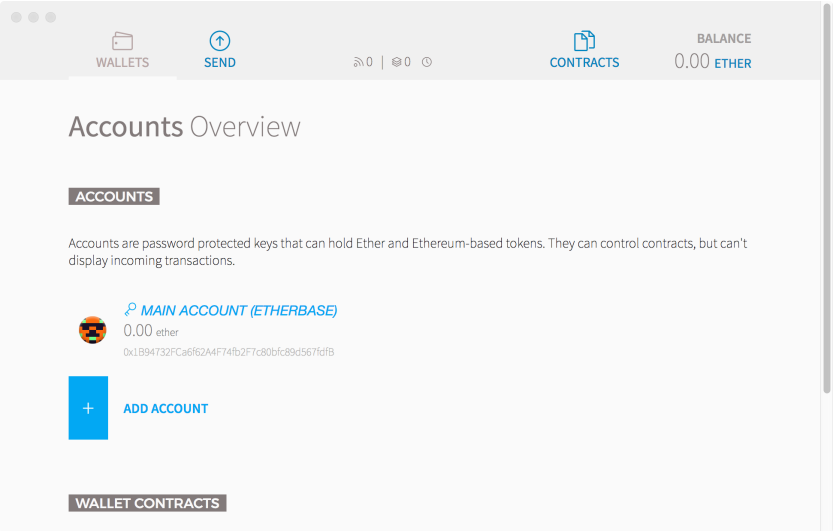
1. Ethereum Wallet(Mist)

开始学习以太坊时阅读大量文章常常会提到 Mist 一头雾水，后来才知道 Mist 就是 Ethereum Wallet。Ethereum Wallet 是软件名字， Mist 是项目名字。

Ethereum Wallet 可以在以太坊首页下载，如果你需要安装历史版本可以访问 https://github.com/ethereum/mist/releases

参数：

```
neo@MacBook-Pro ~ %  "/Applications/Ethereum Wallet.app/Contents/MacOS/Ethereum Wallet" --help
Usage: /Applications/Ethereum Wallet.app/Contents/MacOS/Ethereum Wallet --help
[Mist options] [Node options]

Mist options:
  --mode, -m              App UI mode: wallet, mist.[string] [default: "wallet"]
  --node                  Node to use: geth, eth        [string] [default: null]
  --network               Network to connect to: main, test
                                                        [string] [default: null]
  --rpc                   Path to node IPC socket file OR HTTP RPC hostport (if
                          IPC socket file then --node-ipcpath will be set with
                          this value).                                  [string]
  --swarmurl              URL serving the Swarm HTTP API. If null, Mist will
                          open a local node.
                                    [string] [default: "http://localhost:8500"]
  --gethpath              Path to Geth executable to use instead of default.
                                                                        [string]
  --ethpath               Path to Eth executable to use instead of default.
                                                                        [string]
  --ignore-gpu-blacklist  Ignores GPU blacklist (needed for some Linux
                          installations).                              [boolean]
  --reset-tabs            Reset Mist tabs to their default settings.   [boolean]
  --logfile               Logs will be written to this file in addition to the
                          console.                                      [string]
  --loglevel              Minimum logging threshold: info, debug, error, trace
                          (shows all logs, including possible passwords over
                          IPC!).                     [string] [default: "info"]
  --syncmode              Geth synchronization mode: [fast|light|full]  [string]
  --version, -v           Display Mist version.                        [boolean]
  --skiptimesynccheck     Disable checks for the presence of automatic time sync
                          on your OS.                                  [boolean]

Node options:
  -  To pass options to the underlying node (e.g. Geth) use the --node- prefix,
     e.g. --node-datadir

Options:
  -h, --help  Show help                                                [boolean]
```



1.1. Ethereum Wallet 工作原理

Ethereum Wallet 工作原理非常简单，启动 Ethereum Wallet 是，Ethereum Wallet 会首先启动 geth，然后在启动 Ethereum Wallet 应用程序。

geth 负责与以太坊网络连接，包括主网，Ropsten和Rinkeby两个测试网，Solo 本地开发模式。geth

例如主网的启动目录是 /Users/neo/Library/Ethereum/geth.ipc 参数是:

```
/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth --syncmode light --cache 1024
```

1.1.1. geth 启动 ropsten 测试网

```
/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth --testnet --syncmode light --cache 1024 --ipcpath /Users/neo/Library/Ethereum/geth.ipc
```

1.1.2. 连接到本地测试网络

首先启动 geth

1.1.2.1. IPC

钱包默认是连接到下面地址。

```
IPC endpoint opened: /Users/neo/Library/Ethereum/geth.ipc
```

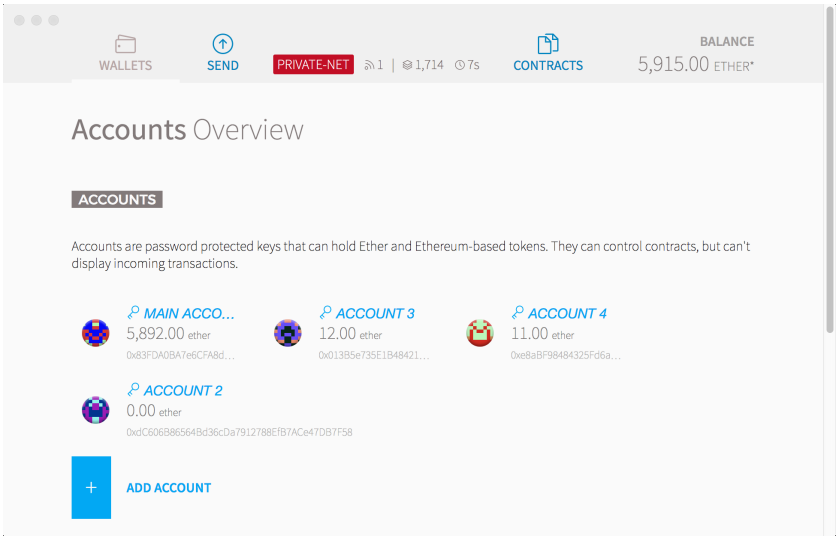
连接到其他ipc地址

```
"/Applications/Ethereum Wallet.app/Contents/MacOS/Ethereum Wallet" --rpc
/Users/other/Library/Ethereum/geth.ipc
```

1.1.2.2. TCP

如果需要连接到远程节点上，需要使用命令行，方法如下。启动钱包并连接到远程开发环境，localhost 改为你的IP地址即可。

```
"/Applications/Ethereum Wallet.app/Contents/MacOS/Ethereum Wallet" --rpc http://localhost:8545
```



1.1.3. 控制台

在 Ethereum Wallet 启动期间，随时可以进入Javascript控制台。

```
neo@MacBook-Pro ~/Library/Ethereum/geth % geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.1-stable-1e67410e/darwin-amd64/go1.9.4
 modules: admin:1.0 debug:1.0 eth:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> eth.accounts
["0xb94054c174995ae2a9e7fcf6c7924635fba8ecf7", "0xf56b81a2bcb964d2806071e9be4289a5559bb0fa",
"0x997e5ca600e19447d0b82afbf9c7f00de2b39b16"]
>
```

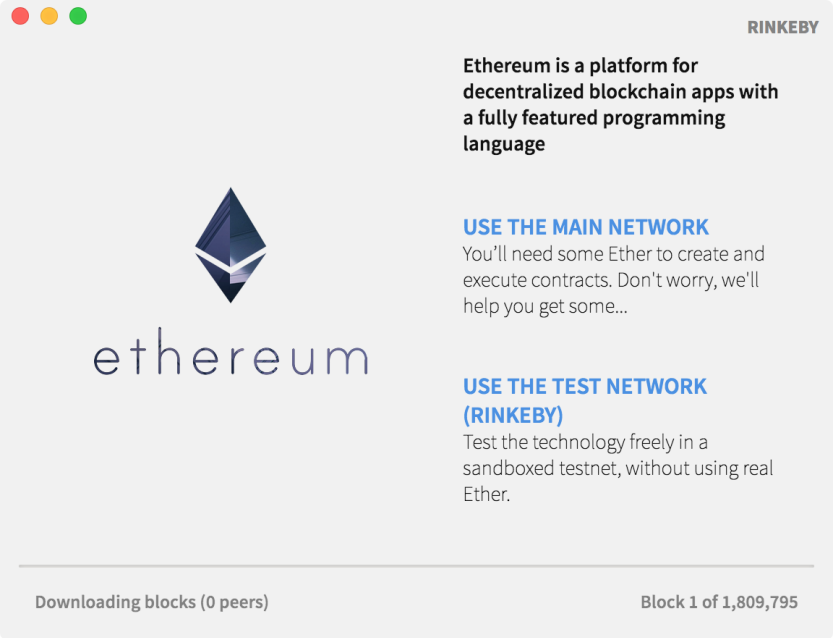
1.2. 主网络

1.2.1. 主网启动参数

```
/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth --syncmode
light --cache 1024
```

1.2.2. 进入主网

点击按钮 "USE THE MAIN NETWORK" 进入以太坊主网。



1.2.3. 以太坊区块浏览器

https://etherscan.io

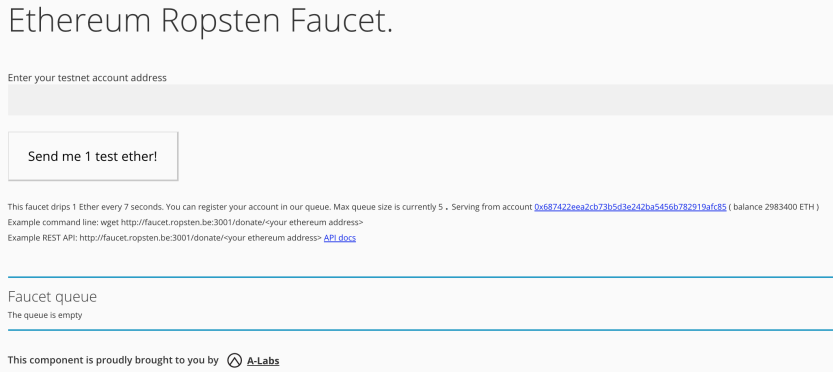
1.3. Ropsten 测试网络

1.3.1. 启动参数

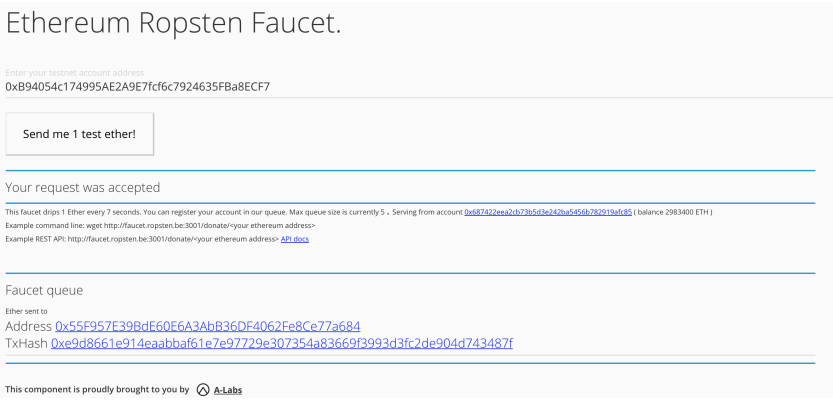
```
/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth --testnet --
syncmode light --cache 1024 --ipcpath /Users/neo/Library/Ethereum/geth.ipc
```

1.3.2. 获得测试币

http://faucet.ropsten.be:3001



输入账号然后点击 “Send me 1 test ether” 按钮



当现实 “Your request was accepted” 时，表示你的请求已经接受，接下来耐心等待。

1.3.3. Etherscan

https://ropsten.etherscan.io

1.4. Rinkeby 测试网络

https://www.rinkeby.io/

1.4.1. 测试网络

连接测试网络，启动 "Ethereum Wallet"，主菜单-Develop-Network，选择 "Rinkeby - Test network"。

如果这个菜单是灰色的，可以你在本地运行过 geth，解决方法如下：

```
mkdir ~/ethereum/rinkeby
geth --datadir ~/ethereum/rinkeby --rinkeby --rpc console
```

在另一个终端窗口执行

```
neo@MacBook-Pro ~/ethereum/rinkeby % "/Applications/Ethereum Wallet.app/Contents/MacOS/Ethereum
Wallet" ~/ethereum/rinkeby/geth.ipc
```

或者使用TCP方式连接

```
"/Applications/Ethereum Wallet.app/Contents/MacOS/Ethereum Wallet" --rpc http://localhost:8545
```

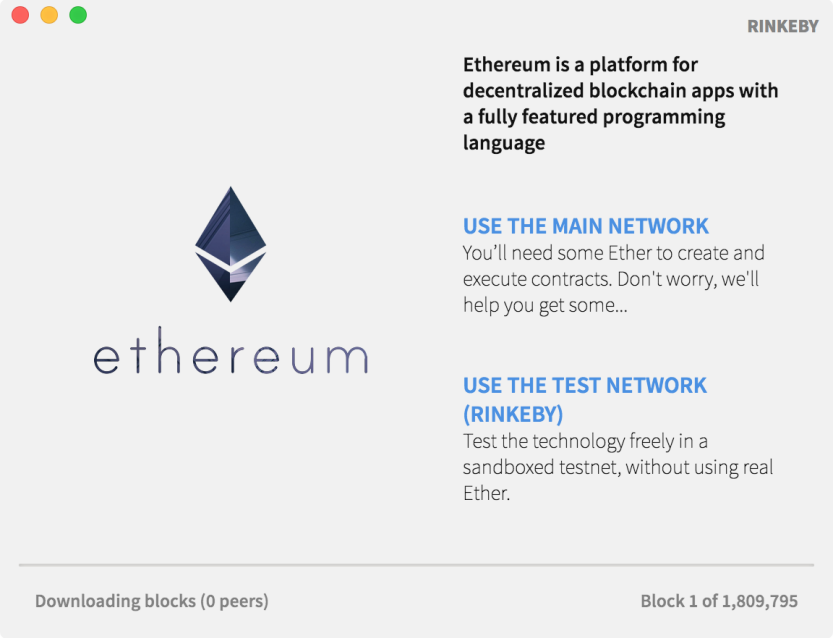
如果你没有安装 geth，可以使用下面的方法

```
neo@MacBook-Pro ~/ethereum/truffle % "/Users/neo/Library/Application Support/Ethereum
Wallet/binaries/Geth/unpacked/geth" --rinkeby --syncmode light --cache 1024 --ipcpath
/Users/neo/Library/Ethereum/geth.ipc
INFO [02-22|11:43:23] Maximum peer count                       ETH=0 LES=100 total=25
INFO [02-22|11:43:23] Starting peer-to-peer node               instance=Geth/v1.8.1-stable-
1e67410e/darwin-amd64/go1.9.4
INFO [02-22|11:43:23] Allocated cache and file handles
database=/Users/neo/Library/Ethereum/rinkeby/geth/lightchaindata cache=768 handles=128
INFO [02-22|11:43:27] Persisted trie from memory database      nodes=355 size=65.27kB
time=399.41µs gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [02-22|11:43:27] Initialised chain configuration          config="{ChainID: 4 Homestead: 1
DAO: <nil> DAOSupport: true EIP150: 2 EIP155: 3 EIP158: 3 Byzantium: 1035301 Engine: clique}"
INFO [02-22|11:43:27] Loaded most recent local header          number=1813389 hash=d58d6b…a489d9
td=3366103
INFO [02-22|11:43:27] Starting P2P networking
INFO [02-22|11:43:29] UDP listener up
net=enode://e84fbdb5d1b75d18fe45fdd13fdad9d5a8ff6d54b82cc8383525870054b91108010ccf8776ac8a50146c
9abf9d43c0117af2bbe0cc2668874d269c3817dec47e@[::]:30303
WARN [02-22|11:43:29] Light client mode is an experimental feature
INFO [02-22|11:43:29] RLPx listener up
self="enode://e84fbdb5d1b75d18fe45fdd13fdad9d5a8ff6d54b82cc8383525870054b91108010ccf8776ac8a5014
6c9abf9d43c0117af2bbe0cc2668874d269c3817dec47e@[::]:30303?discport=0"
INFO [02-22|11:43:29] IPC endpoint opened
url=/Users/neo/Library/Ethereum/geth.ipc
INFO [02-22|11:43:30] Mapped network port                      proto=udp extport=30303
intport=30303 interface="UPNP IGDv1-IP1"
INFO [02-22|11:43:31] Mapped network port                      proto=tcp extport=30303
intport=30303 interface="UPNP IGDv1-IP1"
INFO [02-22|11:43:32] Block synchronisation started
INFO [02-22|11:43:33] Imported new block headers               count=13 elapsed=245.507ms
number=1813402 hash=08317c…fd1806 ignored=0
INFO [02-22|11:43:48] Imported new block headers               count=1  elapsed=898.705µs
number=1813403 hash=8a7c1b…5e1652 ignored=0
INFO [02-22|11:44:03] Imported new block headers               count=1  elapsed=892.924µs
number=1813404 hash=511a30…561d32 ignored=0
```

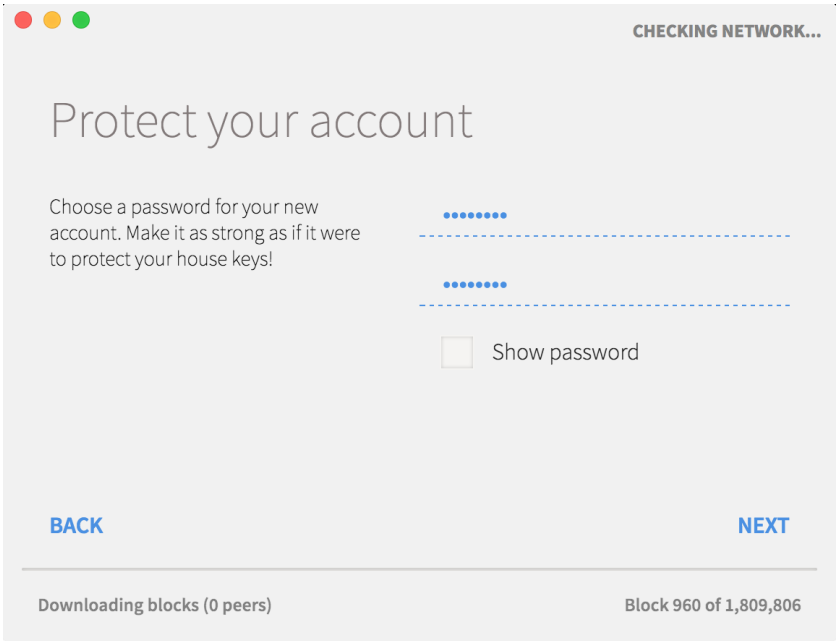
然后正常启动钱包程序 Launchpad - Ethereum Wallet

启动过程比较缓慢，启动后会同步区块信息

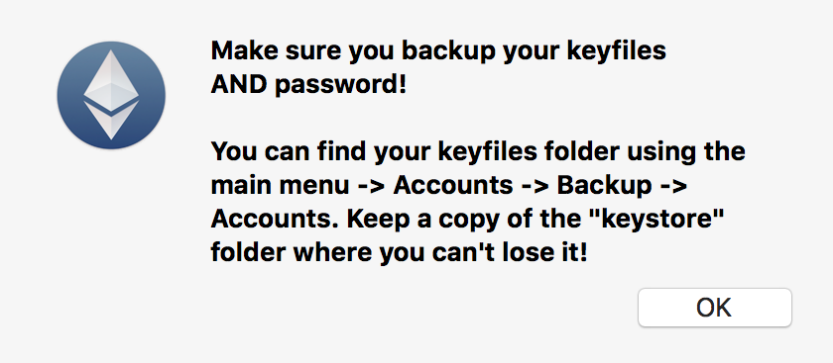
弹出第一个界面，让你选择网络，这里选择 Rinkeby 测试网络



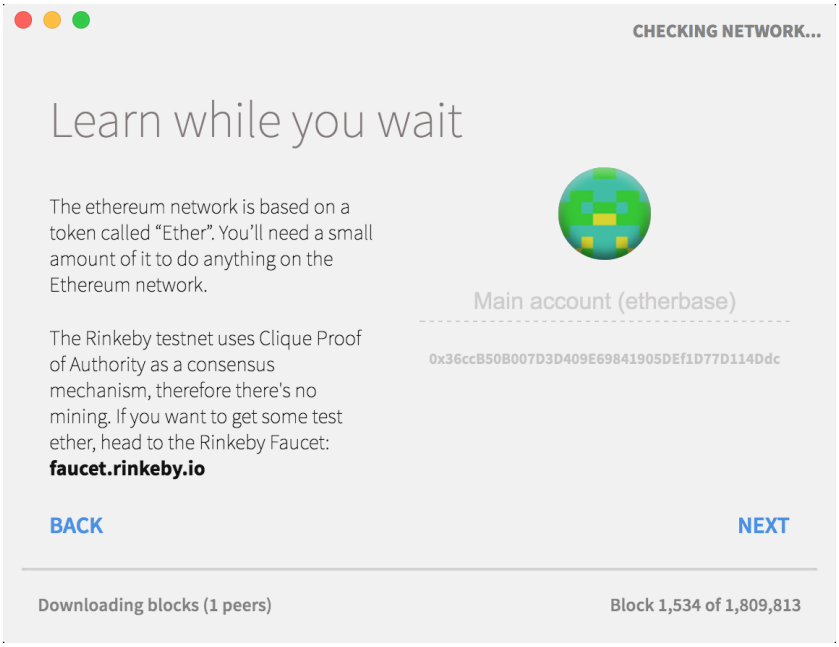
输入密码，要求八位字母和数字组合



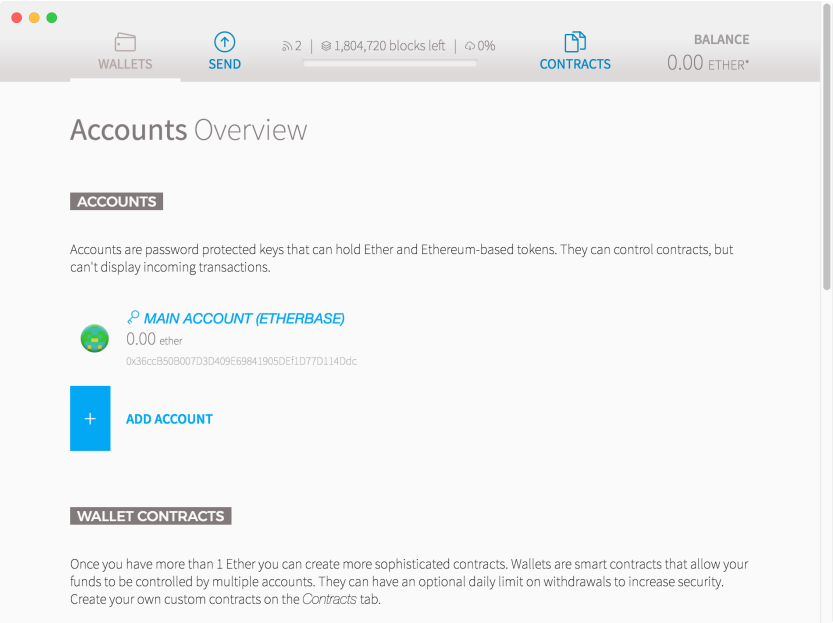
提示你备份钱包



生成账号 0x36ccB50B007D3D409E69841905DEf1D77D114Ddc



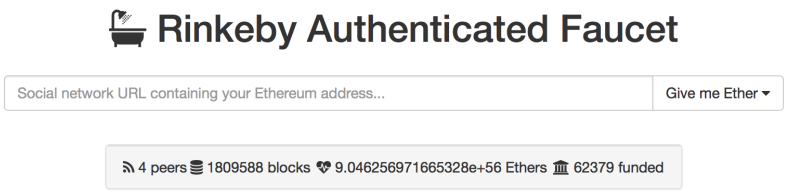
这时你会看到右下角的数字不断地变化，表示正在同步区块信息。同步过程比较漫长，请耐心等待。



1.4.2. 获取测试网络上的以太币

想获取Rinkeby测试网络中的以太币，需要去申请，地址：

https://faucet.rinkeby.io



输入你的钱包地址，并点击“Give me Ether”，有三种选项，前面是获得的以太币数量，后面是冷却时间，在冷却时间过后才能进行下一次以太币申请。例如第一项是生成3个以太币，8小时后才能再次申请。

如果一切顺利，你会看到你的钱包地址已经多出了申请数量的以太币。

注意：如果申请的人数很多，需要排队等待

1.4.3. 连接节点（Light node）

下载创世区块

```
mkdir ~/.rinkeby
cd ~/.rinkeby
wget https://www.rinkeby.io/rinkeby.json
```

初始化区块

```
geth --datadir=$HOME/.rinkeby init rinkeby.json
```

启动以太坊，这里采用Light node模式，仅仅下载区块信息，不会运行交易，速度比较快。

```
geth --networkid=4 --datadir=$HOME/.rinkeby --syncmode=light --ethstats='yournode:Respect my
authoritah!@stats.rinkeby.io' --
bootnodes=enode://a24ac7c5484ef4ed0c5eb2d36620ba4e4aa13b8c84684e1b4aab0cebea2ae45cb4d375b77eab56
516d34bfbd3c1a833fc51296ff084b770b94fb9028c4d25ccf@52.169.42.101:30303
```

1.4.4. 区块链浏览器

https://rinkeby.etherscan.io

1.5. Solo Network

solo 实际上就是geth 的 --dev 参数

开发模式会创建一个账号，同时启动挖矿，ETH多的你用不完。

```
/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth --dev --
minerthreads 1 --ipcpath /Users/neo/Library/Ethereum/geth.ipc
```

1.6. 私网

```
"/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth" --
datadir=$HOME/ethereum/private init $HOME/ethereum/genesis.json
"/Users/neo/Library/Application Support/Ethereum Wallet/binaries/Geth/unpacked/geth" --
networkid=4444 --datadir=$HOME/ethereum/private --ipcpath /Users/neo/Library/Ethereum/geth.ipc
```

1.7. 删除废弃的合约

在 Ethereum Wallet 钱包上会显示所有自己创建或者Watch的合约，有些合约仅仅是测试用的，用过即废弃，合约列表依然会显示在那里，只是变成灰色并且无法选择

怎样把无用的合约从列表删除呢?

```
1. Ethereum Wallet上打开开发者工具（从窗口选择开发->切换开发工具->钱包界面，或者输入快捷键Alt+Ctrl+I），显示
出控制台界面
2. 查询Watch Contract列表 - 在控制台输入命令：CustomContracts.find().fetch();
3. 在合约列表中找到需要删除的合约，并且复制ContractID
4. 删除合约，在控制台输入命令：CustomContracts.remove('<ContractID>')
```

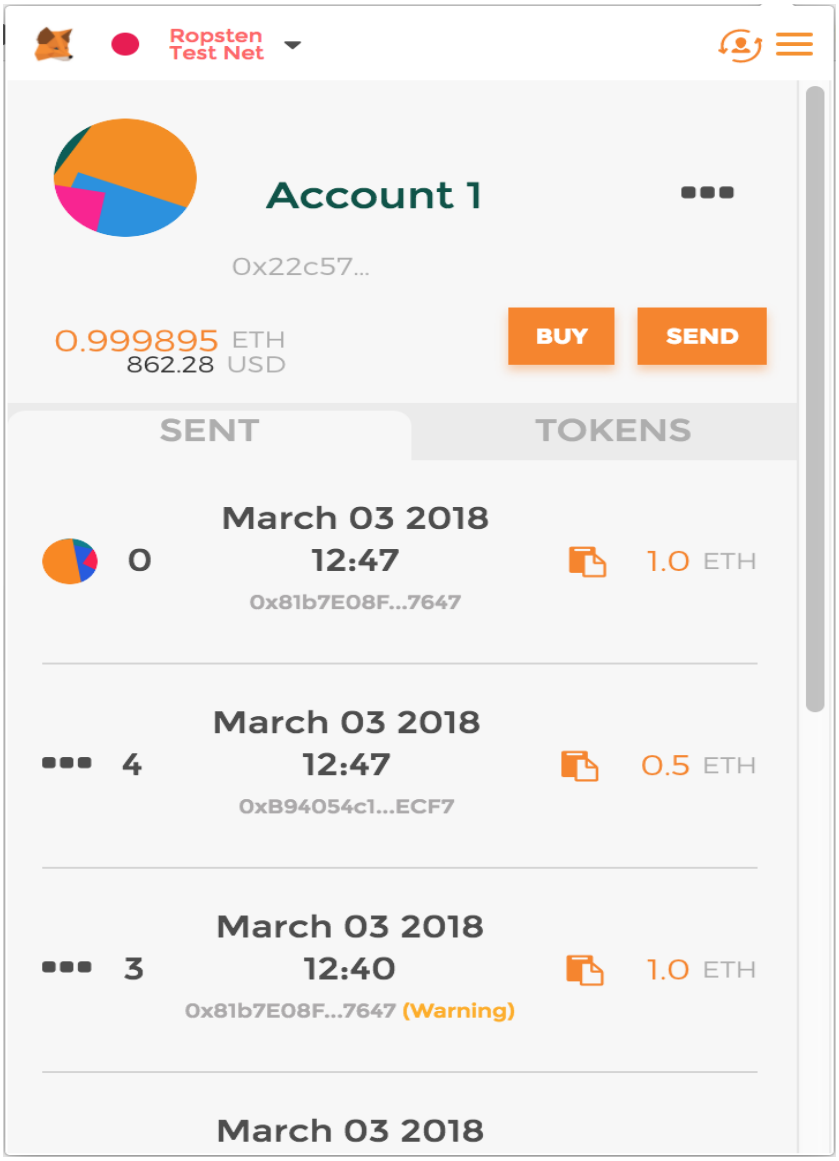
1.8. 免安装，在线使用

https://wallet.ethereum.org

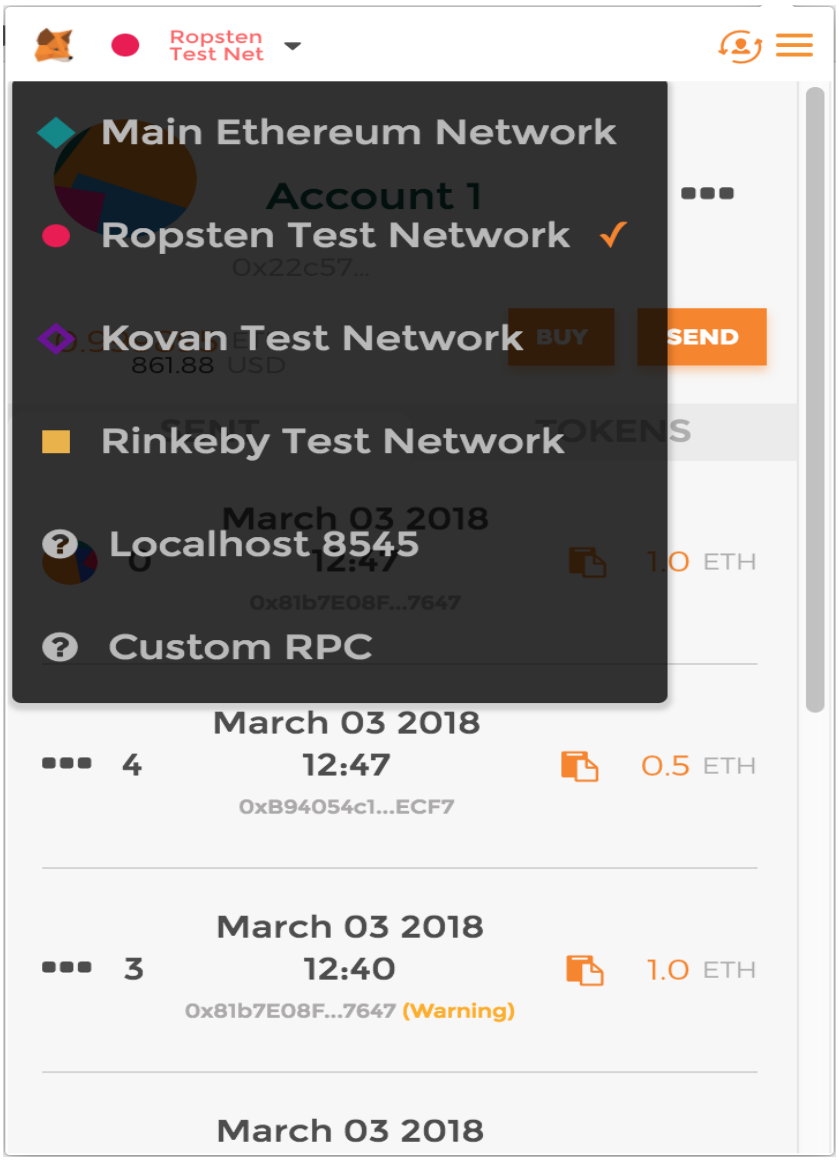
1.9. 获得空投币

2. MetaMask

MetaMask 是 Chrome 浏览器插件 https://metamask.io

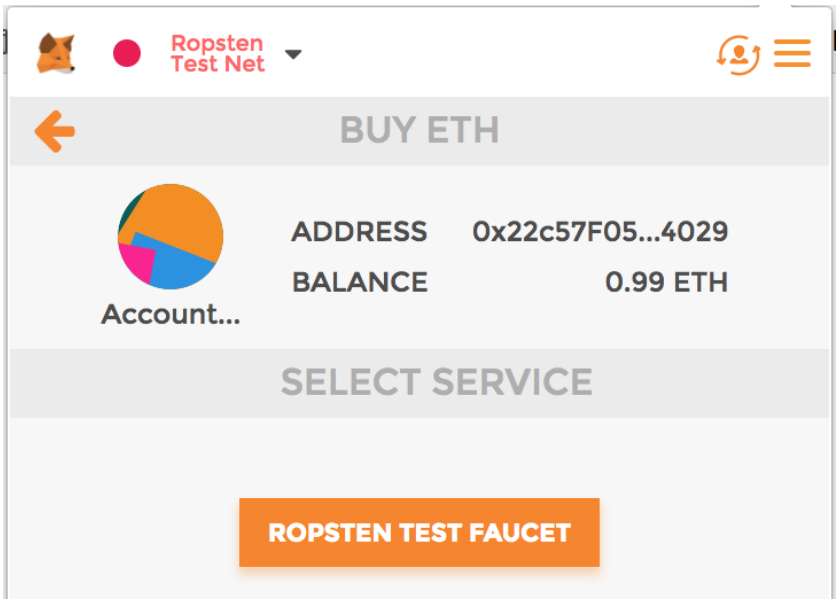


2.1. 测试网络

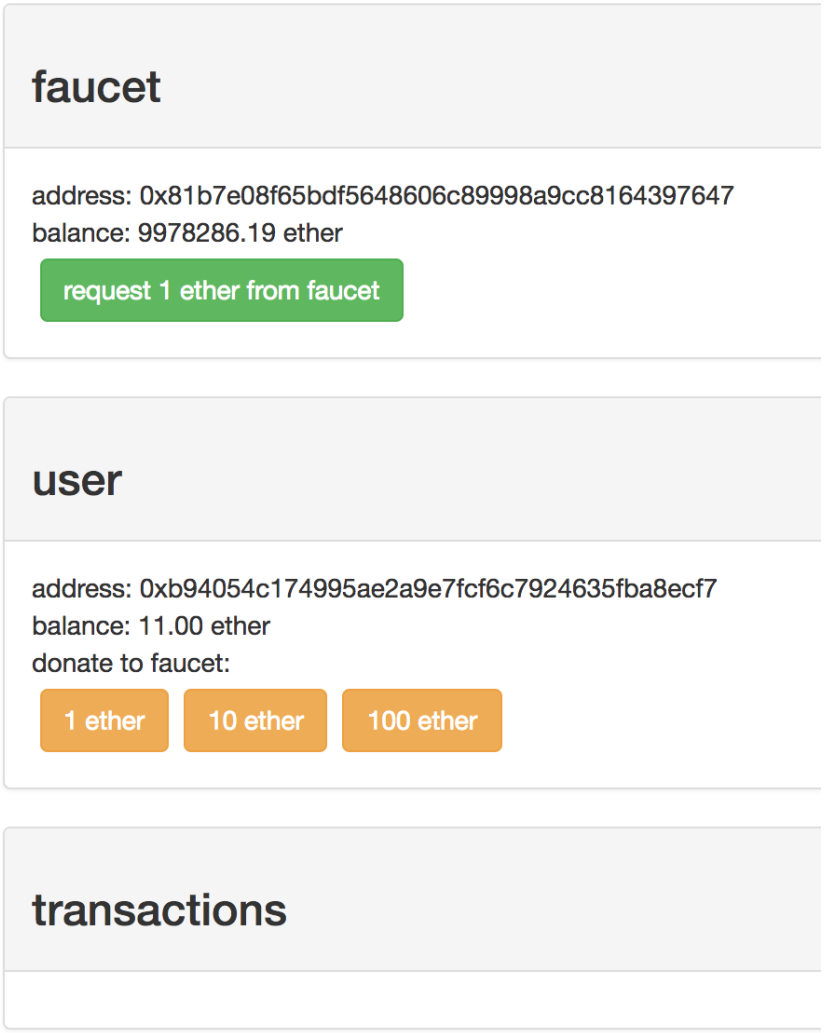


2.1.1. 获取测试网络上的以太币

点击 Buy 按钮，然后再点击 ROPSTEN TEST FAUCET 按钮。

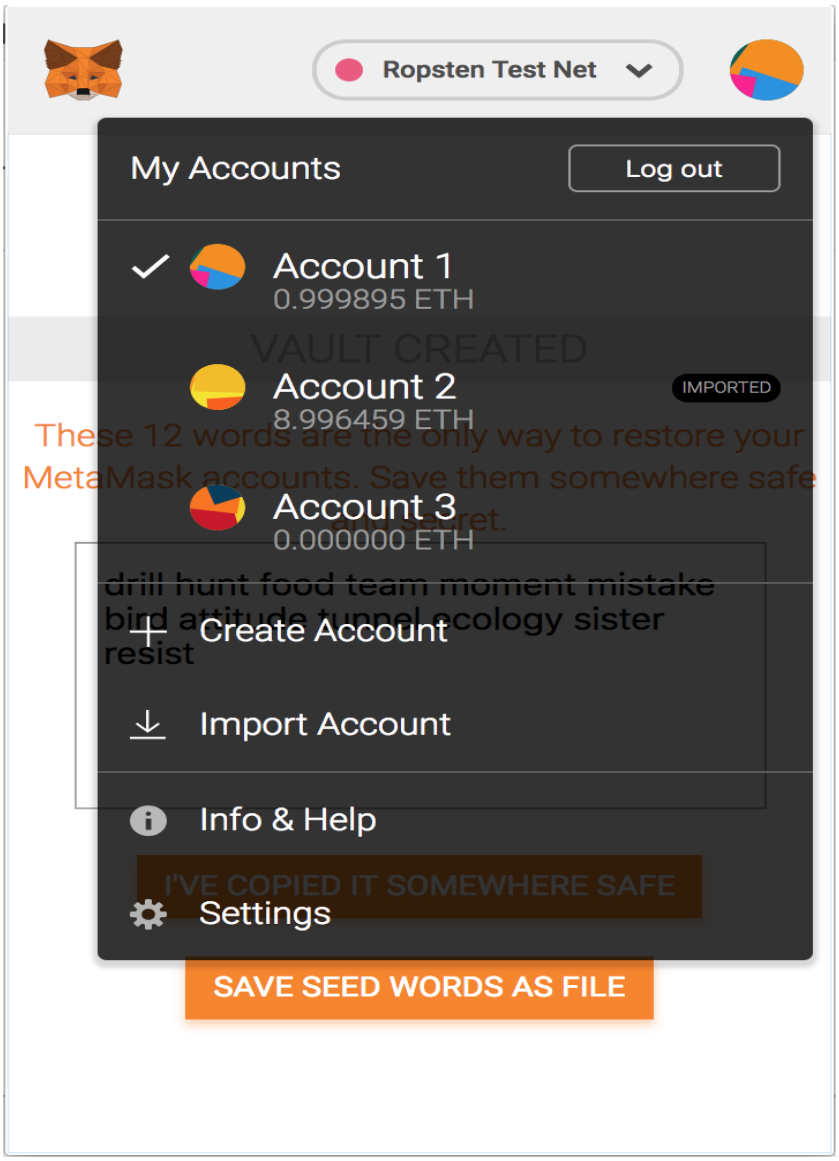


点击 request 1 ether from faucet

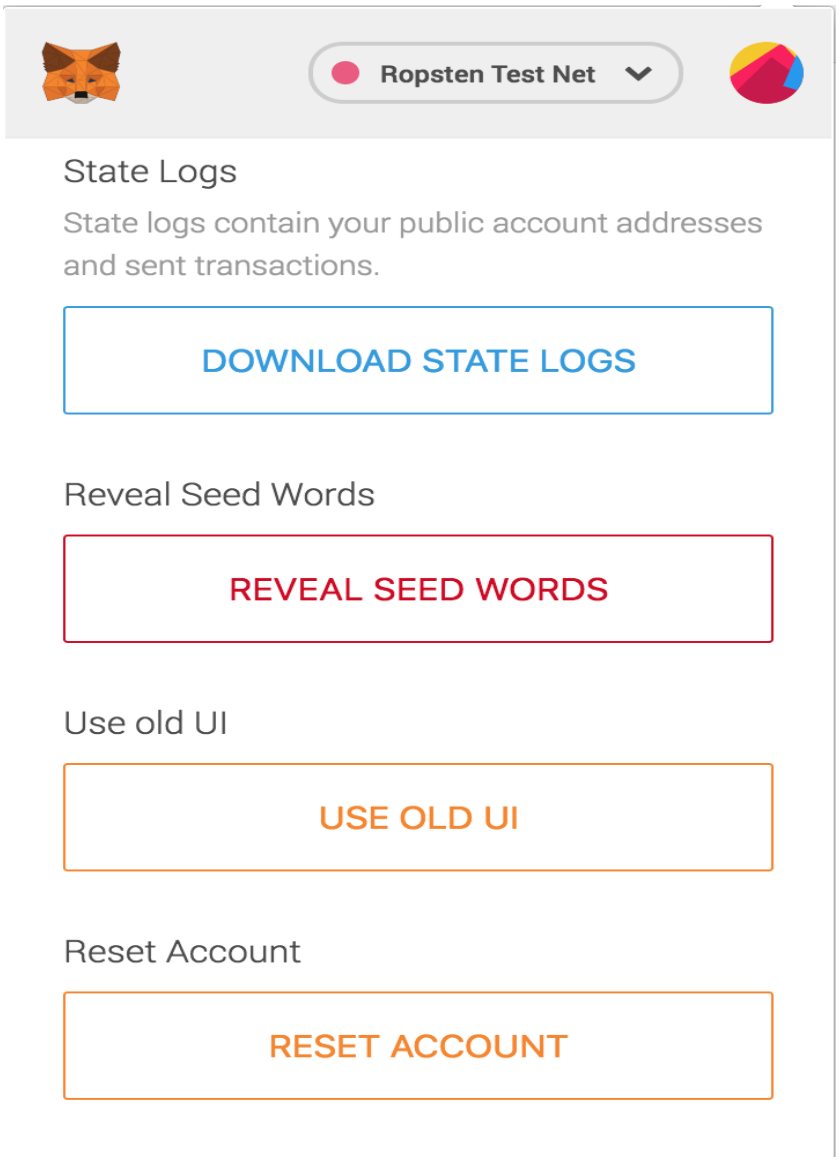


2.2. mnemonic - Reveal seed words

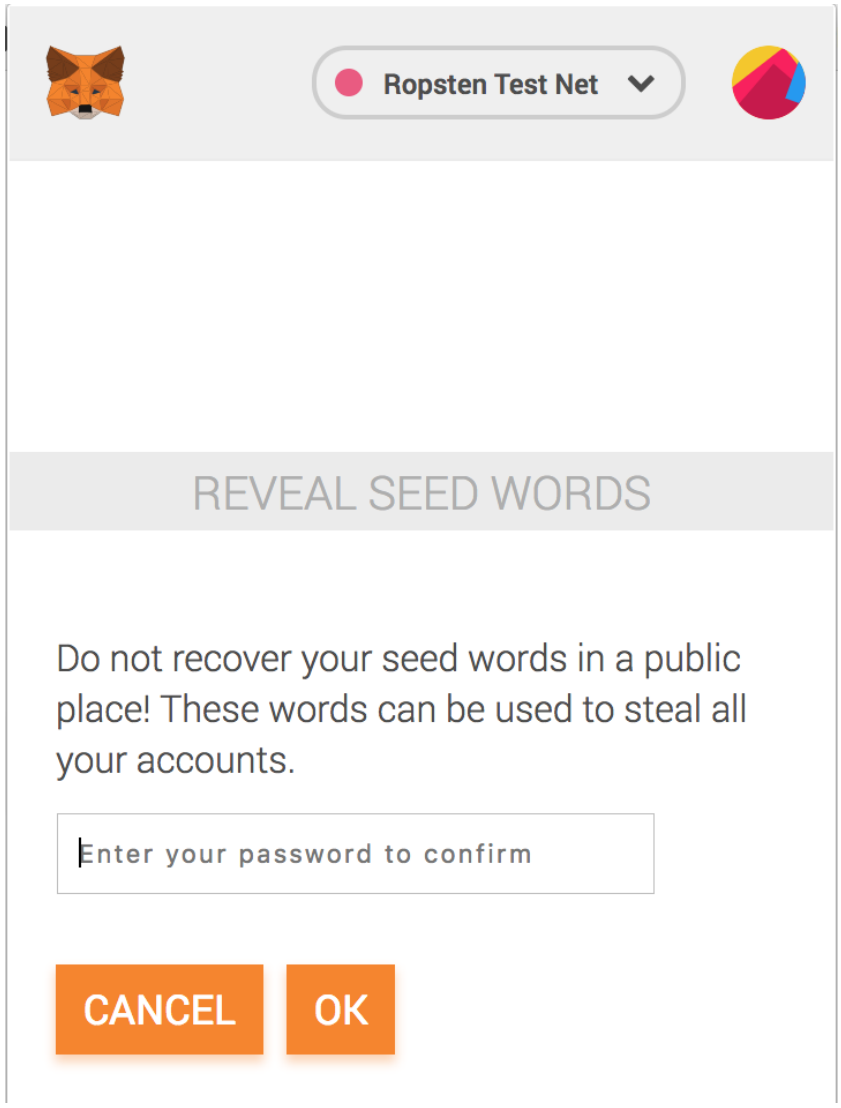
点击头像进入 Settings 菜单



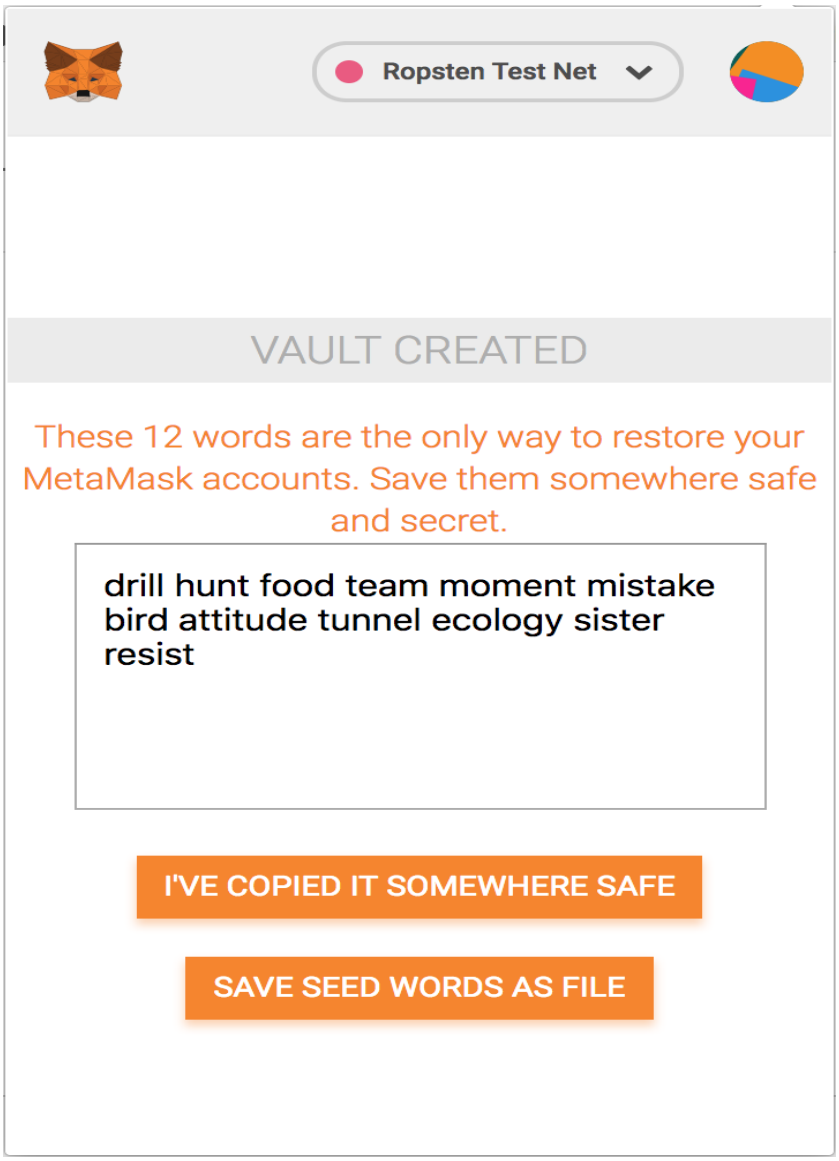
单机 Reveal seed words 按钮


Ropsten Test Net
State Logs
State logs contain your public account addresses and sent transactions.
DOWNLOAD STATE LOGS
Reveal Seed Words
REVEAL SEED WORDS
Use old UI
USE OLD UI
Reset Account
RESET ACCOUNT

输入账号密码解锁



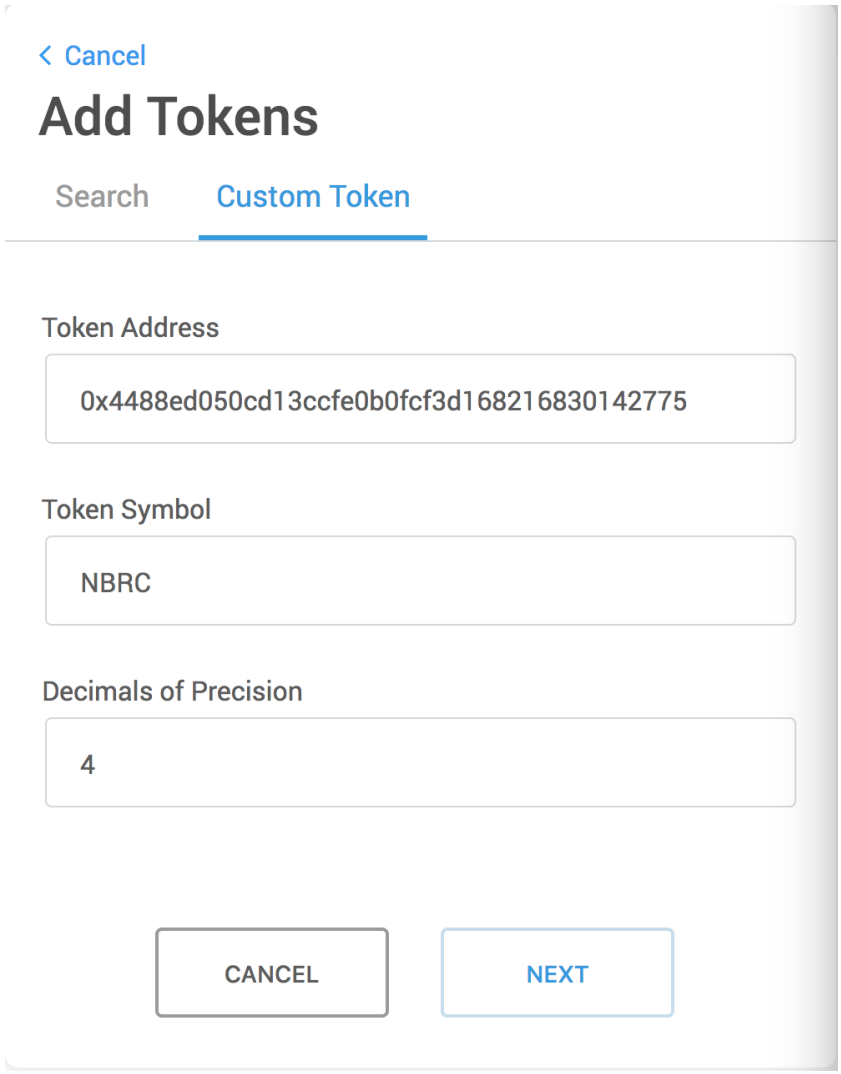
查看 mnemonic 字符串



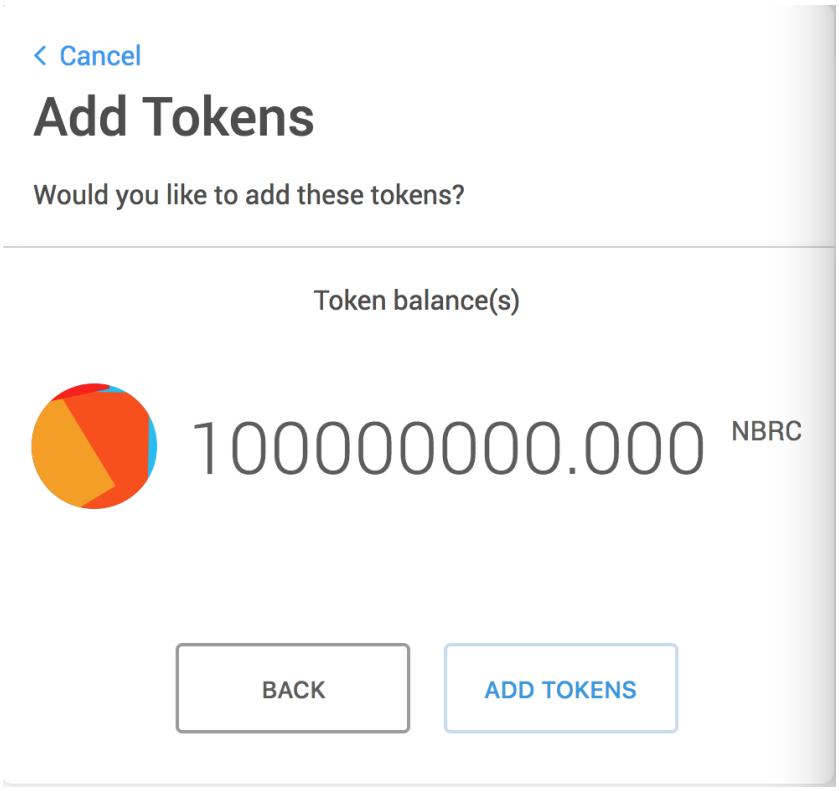
2.3. 添加 Token 币种

有时我们需要添加新币种，还有一个目的是为了接收空投币。

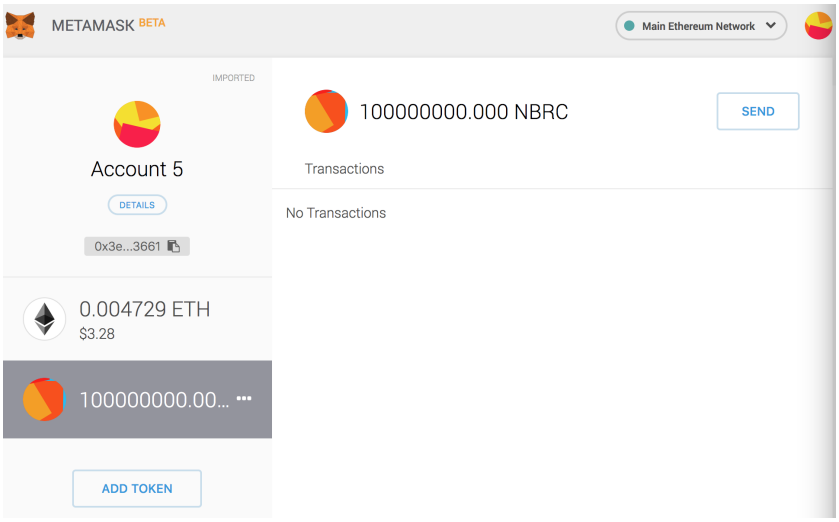
只需输入合约地址即可，其他选项会自动填充。


Cancel
Add Tokens
Search
Custom Token
Token Address
0x4488ed050cd13ccfe0b0fcf3d168216830142775
Token Symbol
NBRC
Decimals of Precision
4
CANCEL
NEXT

显示代币数量



完成代币添加



2.4. MetaMask Vault Decryptor

MetaMask 将钱包数据存储在浏览中 Local Storage 中，这是Chrome 谷歌浏览器特有的 Key,Value数据库。

Key 的名字是 metamask-config 将它复制到 https://metamask.github.io/vault-decryptor/ 中可以解密。

解密程序的 Github 地址 https://github.com/MetaMask/vault-decryptor

2.5. 部署合约

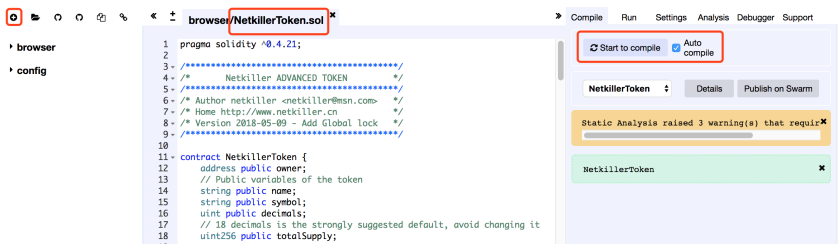
这里假设你已经安装 MetaMask 钱包，并且创建了账号，账号有足够的 ETH 用来创建合约。

我们准备一个合约

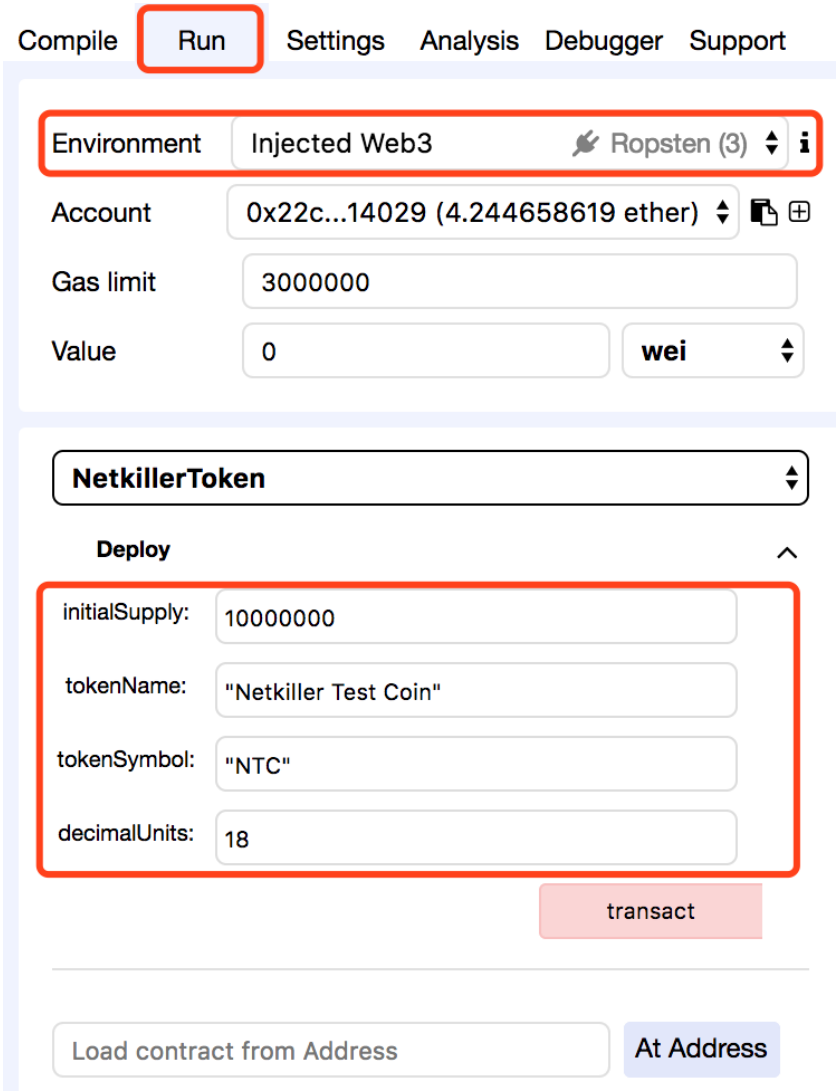
```
pragma solidity ^0.4.24;

/******************************************/
/*       Netkiller ADVANCED TOKEN         */
/******************************************/
/* Author netkiller <netkiller@msn.com>   */
/* Home http://www.netkiller.cn           */
/* Version 2018-05-23                     */
/******************************************/

contract NetkillerMiniToken {
    address public owner;
    // Public variables of the token
    string public name;
    string public symbol;
    uint public decimals;
    // 18 decimals is the strongly suggested default, avoid changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public allowance;

    // This generates a public event on the blockchain that will notify clients
    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the creator of the contract
     */
    constructor(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol,
        uint decimalUnits
    ) public {
        owner = msg.sender;
        name = tokenName;                                   // Set the name for display purposes
        symbol = tokenSymbol;
        decimals = decimalUnits;
        totalSupply = initialSupply * 10 ** uint256(decimals);  // Update total supply with the
decimal amount
        balanceOf[msg.sender] = totalSupply;                // Give the creator all initial
token
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    function transferOwnership(address newOwner) onlyOwner public {
        if (newOwner != address(0)) {
            owner = newOwner;
        }
    }

    /* Internal transfer, only can be called by this contract */
    function _transfer(address _from, address _to, uint _value) internal {
```

```
        require (_to != 0x0);                               // Prevent transfer to 0x0 address.
Use burn() instead
        require (balanceOf[_from] >= _value);               // Check if the sender has enough
        require (balanceOf[_to] + _value > balanceOf[_to]); // Check for overflows
        balanceOf[_from] -= _value;                         // Subtract from the sender
        balanceOf[_to] += _value;                           // Add the same to the recipient
        emit Transfer(_from, _to, _value);
    }

    /**
     * Transfer tokens
     *
     * Send `_value` tokens to `_to` from your account
     *
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transfer(address _to, uint256 _value) public {
        _transfer(msg.sender, _to, _value);
    }

    /**
     * Transfer tokens from other address
     *
     * Send `_value` tokens to `_to` in behalf of `_from`
     *
     * @param _from The address of the sender
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transferFrom(address _from, address _to, uint256 _value) public returns (bool
success) {
        require(_value <= allowance[_from][msg.sender]);     // Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    /**
     * Set allowance for other address
     *
     * Allows `_spender` to spend no more than `_value` tokens in your behalf
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     */
    function approve(address _spender, uint256 _value) public returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        emit Approval(msg.sender, _spender, _value);
        return true;
    }
}
```

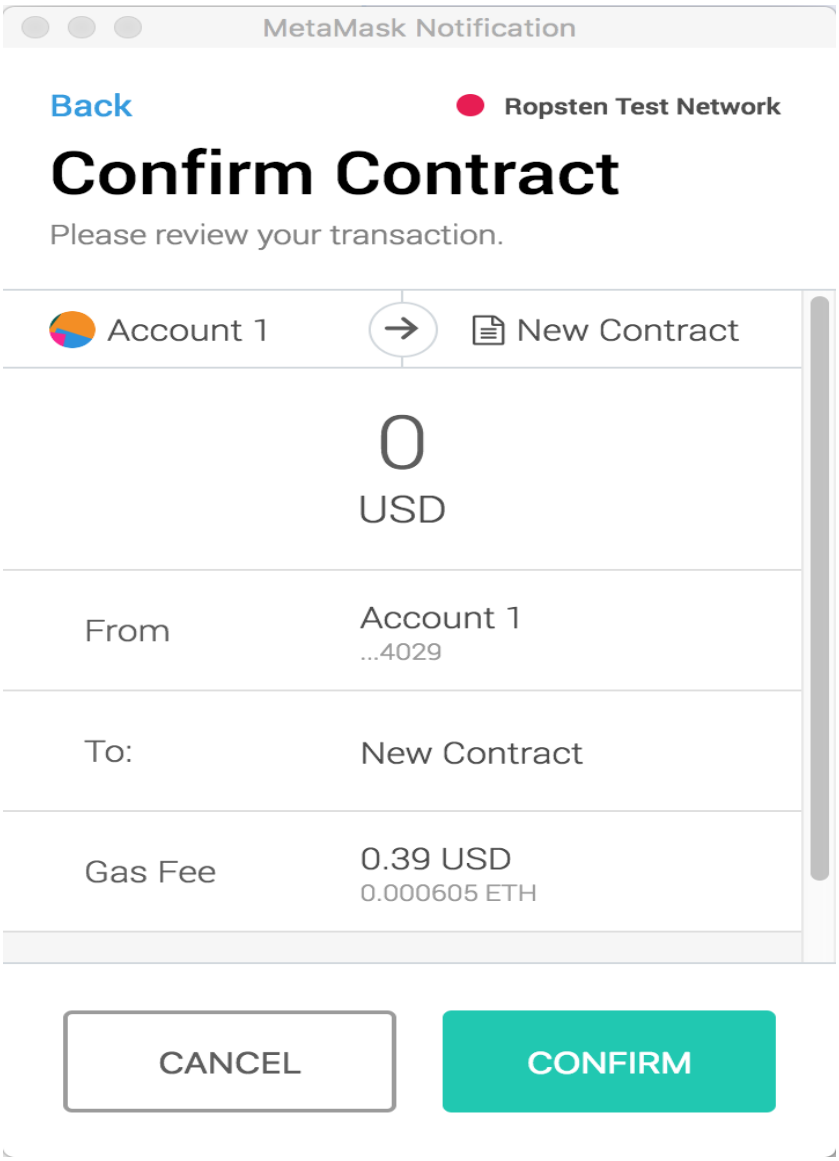
现在打开 https://remix.ethereum.org/



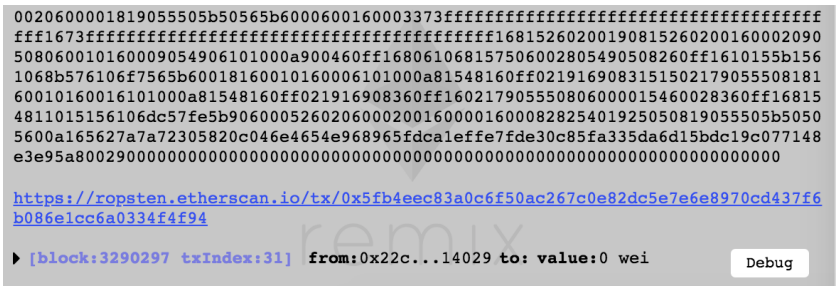
点击“+”加号创建合约，合约名称NetkillerToken, 编译合约。进入 “Run” 选项卡运行合约



Environment 设置为 Injected Web3 这时可以看到 MetaMask 中的账号以及余额, 然后设置构造方法，最后点击“transact”按钮开始部署合约。



确认手续费



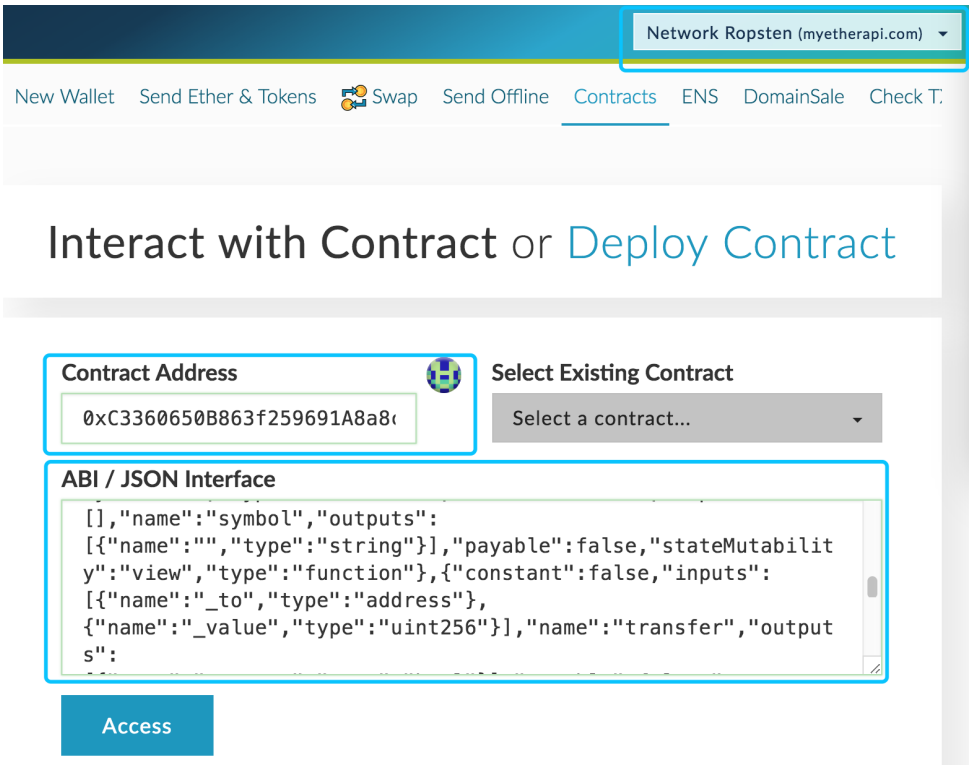
remix 显示出TX Hash 表示完成。

3. MyEtherWallet

https://www.myetherwallet.com/

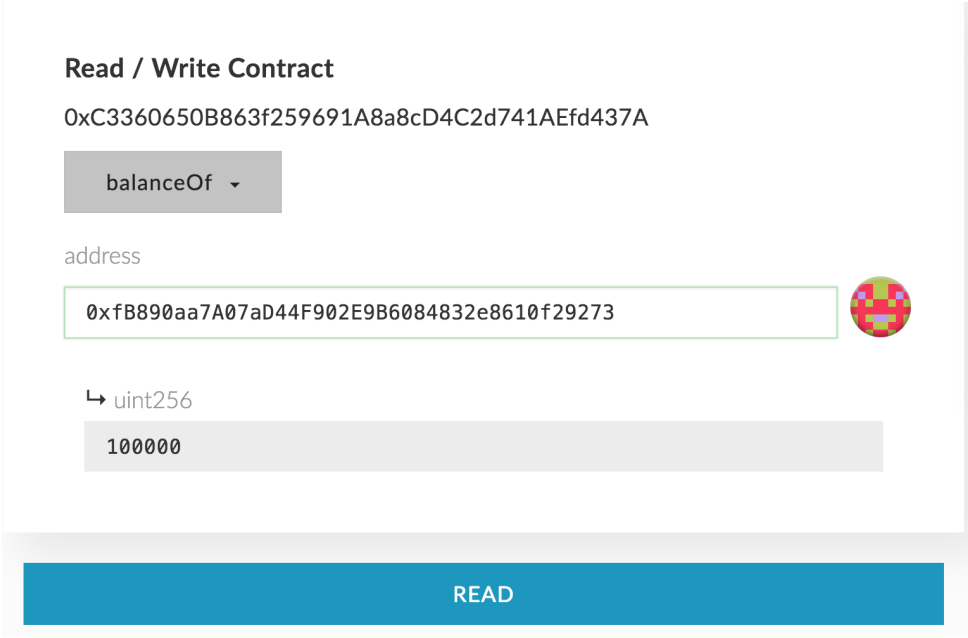
3.1. 执行ERC20智能合约函数

首先选择网络，例如主网，测试网，然后填写合约地址，粘贴ABI，点击ACCESS按钮



3.1.1. 查询余额

选择合约方法，输入参数，执行后可以看到结果，这里仅仅演示查询余额的方法。



3.1.2. 销毁代币

从合约的 Owner 账号中销毁一定数额的代币

从指定账号中销毁一定数额的代币

3.1.3. 冻结账号

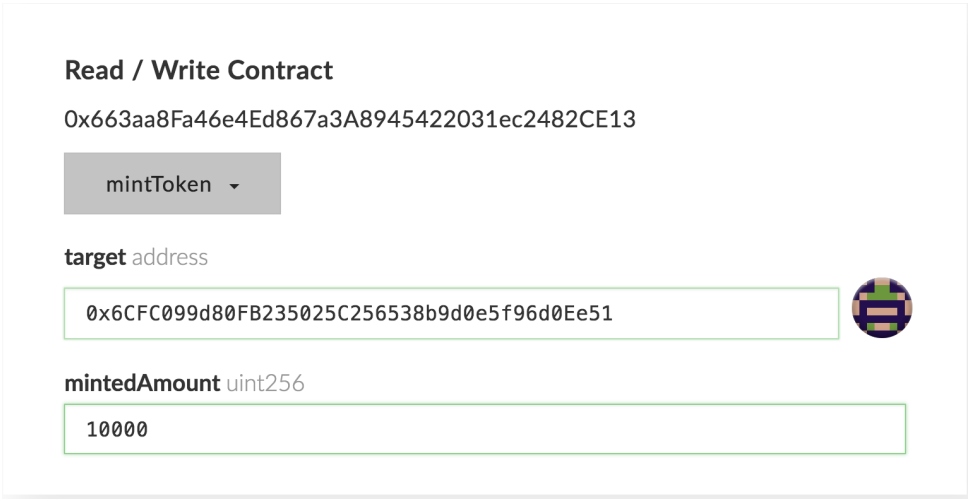
冻结账号,输入地址，然后选择 True 最后点击按钮 Write

解冻账号，重复上述步骤，选择 False 然后点击 Write 按钮。

查询账号状态，输入地址，点击 Write 按钮，返回 True 表示账号被冻结，返回 False 或 空值，表示没有冻结。

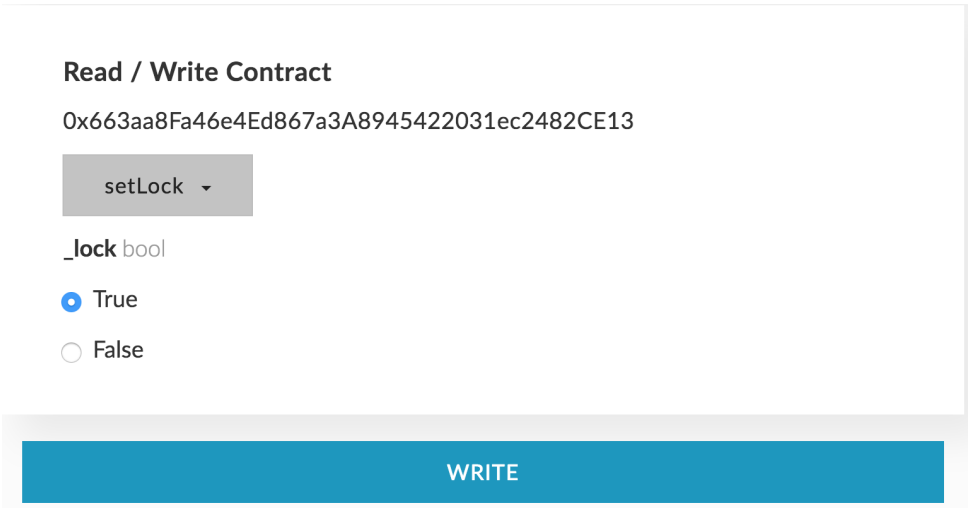
3.1.4. 增发代币

输入需要增发到那个地址，然后输入增发数量



3.1.5. 锁仓

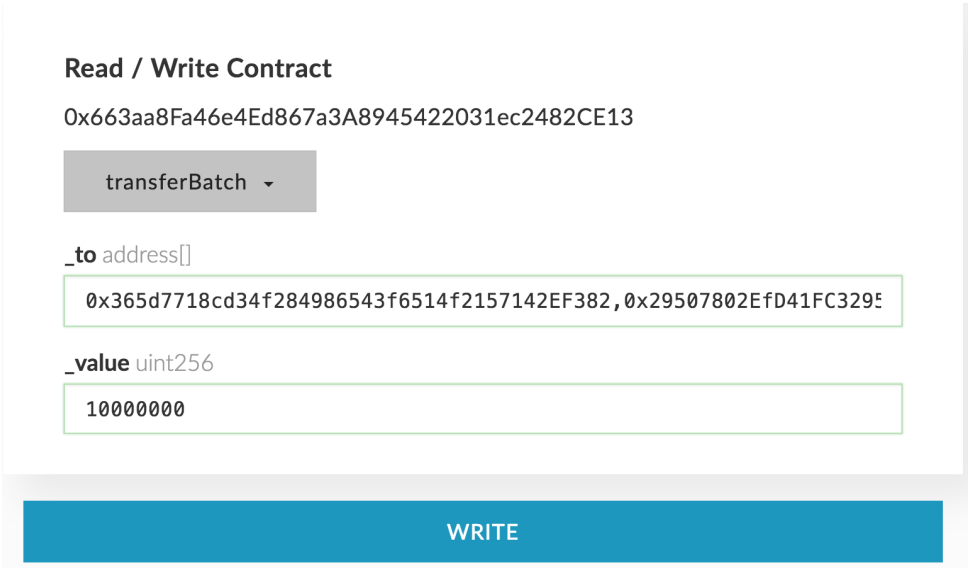
冻结是针对单个用户的，如果你需要停止一切交易（任何账号都不能转入，转出），那么可以采用锁仓方案。



True 表示开启锁仓， False 表示关闭锁仓。

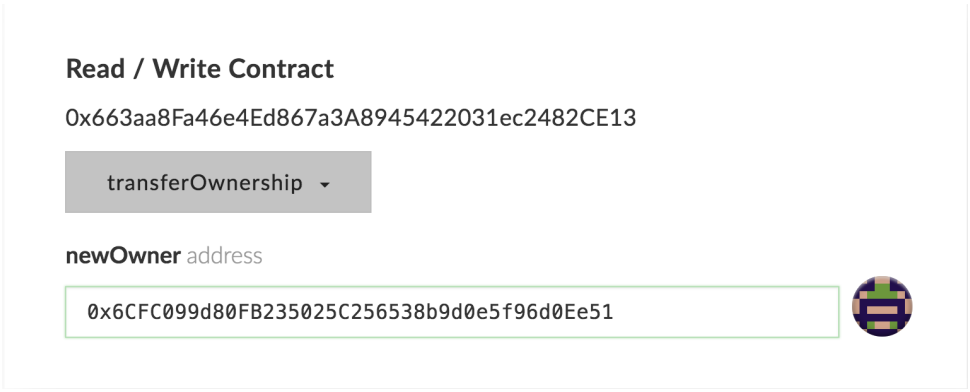
3.1.6. 批量打币

一次性对多个账号打币，输入账号使用逗号分隔（半角），然后输入数量。



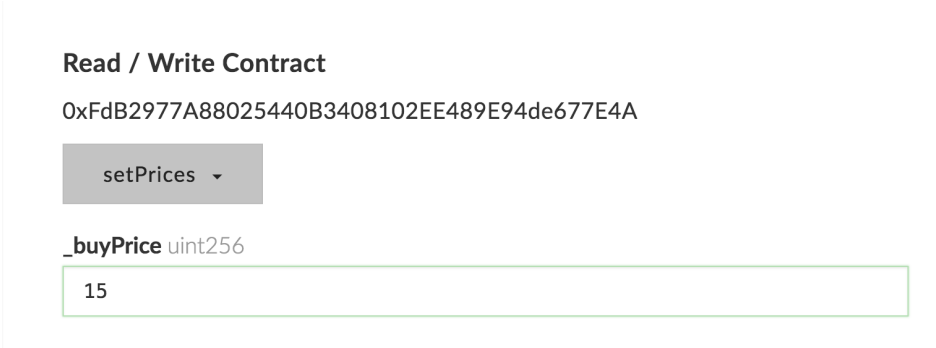
3.1.7. 修改合约管理者

合约的管理者拥有上面所有权限，如果你想转移给其他账号，使用下面方法操作即可。



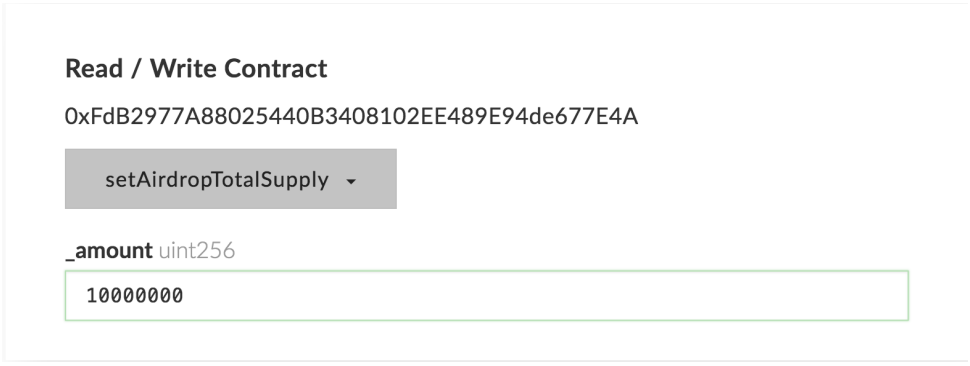
3.1.8. 设置兑换比例

将 ETH 发送到合约知道，返回 ETH * buyPrice 数量的 Token



3.1.9. 空投设置

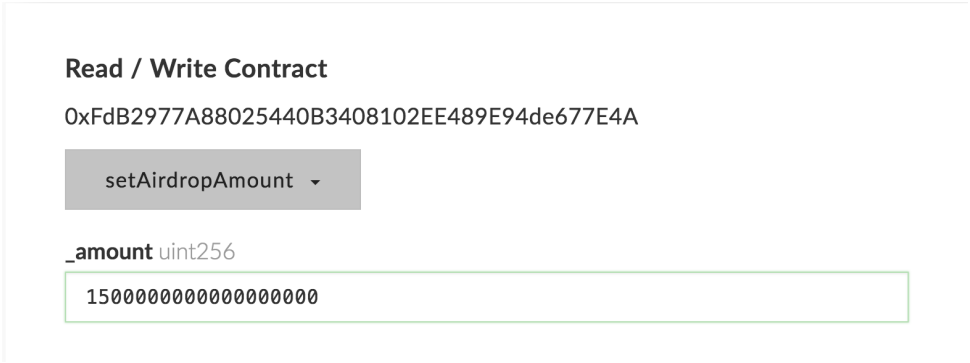
设置空投总量



换算 ETH 到 WEI https://etherconverter.online/ 例如你需要空投 1.5 个Token，请在下面填写 1500000000000000000

Wei	1500000000000000000
Kwei, Ada, Femtoether	1500000000000000
Mwei, Babbage, Picoether	1500000000000
Gwei, Shannon, Nanoether, Nano	1500000000
Szabo, Microether,Micro	1500000
Finney, Milliether,Milli	1500
Ether	1.5
Kether, Grand,Einstein	0.0015
Mether	0.0000015
Gether	0.0000000015
Tether	0.0000000000015
USD(at 362.051$ p/ ether)	543.077
EUR(at 312.519€ p/ ether)	468.779

设置空投数量



4. MyCrypto

https://www.mycrypto.com/

5. imToken

https://token.im

5.1. 添加 Token

具体教程请参考: https://github.com/consenlabs/token-profile/blob/master/README.md

创建文件 token-profile/erc20/0x4488ed050cd13ccfe0b0fcf3d168216830142775.json

```
{
  "symbol": "NBRC",
  "address": "0x4488ed050cd13ccfe0b0fcf3d168216830142775",
  "overview":{
        "en":"Netkiller eBook Reader Coin."
  },
  "email": "netkiller@msn.com",
  "website": "https://www.netkiller.cn/",
  "whitepaper": "https://www.netkiller.cn/file.pdf",
  "state": "LOCKED",
  "published_on": "2018-06-01",
  "initial_price":{
        "ETH":"0.00166666 ETH",
        "USD":"0.7 USD",
        "BTC":"0.00004137 BTC"
  },
  "links": {
    "blog": "https://my.oschina.net/neochen/blog",
    "telegram": "https://t.me/joinchat/netkiller",
    "github":
"https://github.com/netkiller/netkiller.github.io",
    "facebook": "https://www.facebook.com/bg7nyt",
  }
}
```

第 13 章 Token

ERC 代币标准

1. Ethereum Wallet 创建ERC20代币合约

https://ethereum.org/token

1.1. 合约文件

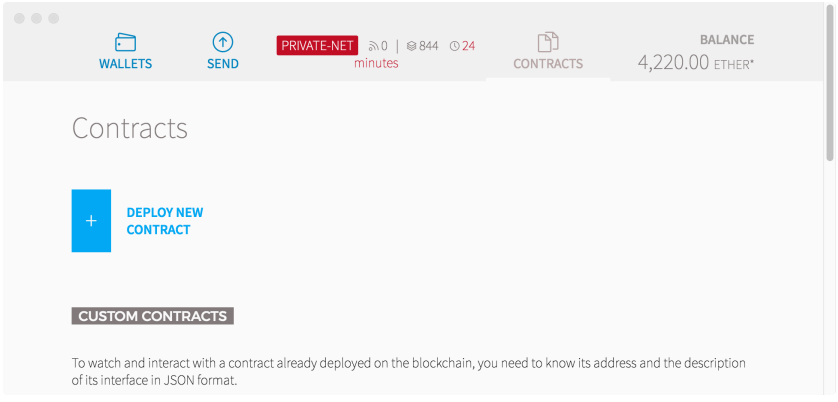
```
pragma solidity ^0.4.16;

interface tokenRecipient { function receiveApproval(address _from, uint256 _value, address
_token, bytes _extraData) public; }

contract TokenERC20 {
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public allowance;

    // This generates a public event on the blockchain that will notify clients
    event Transfer(address indexed from, address indexed to, uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the creator of the contract
     */
    function TokenERC20(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        totalSupply = initialSupply * 10 ** uint256(decimals);  // Update total supply with the
decimal amount
        balanceOf[msg.sender] = totalSupply;                // Give the creator all initial
tokens
        name = tokenName;                                   // Set the name for display purposes
        symbol = tokenSymbol;                               // Set the symbol for display
purposes
    }

    /**
     * Internal transfer, only can be called by this contract
     */
    function _transfer(address _from, address _to, uint _value) internal {
        // Prevent transfer to 0x0 address. Use burn() instead
        require(_to != 0x0);
        // Check if the sender has enough
        require(balanceOf[_from] >= _value);
        // Check for overflows
        require(balanceOf[_to] + _value > balanceOf[_to]);
        // Save this for an assertion in the future
        uint previousBalances = balanceOf[_from] + balanceOf[_to];
```

```
        // Subtract from the sender
        balanceOf[_from] -= _value;
        // Add the same to the recipient
        balanceOf[_to] += _value;
        Transfer(_from, _to, _value);
        // Asserts are used to use static analysis to find bugs in your code. They should never
fail
        assert(balanceOf[_from] + balanceOf[_to] == previousBalances);
    }

    /**
     * Transfer tokens
     *
     * Send `_value` tokens to `_to` from your account
     *
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transfer(address _to, uint256 _value) public {
        _transfer(msg.sender, _to, _value);
    }

    /**
     * Transfer tokens from other address
     *
     * Send `_value` tokens to `_to` on behalf of `_from`
     *
     * @param _from The address of the sender
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transferFrom(address _from, address _to, uint256 _value) public returns (bool
success) {
        require(_value <= allowance[_from][msg.sender]);     // Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    /**
     * Set allowance for other address
     *
     * Allows `_spender` to spend no more than `_value` tokens on your behalf
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     */
    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        return true;
    }

    /**
     * Set allowance for other address and notify
     *
     * Allows `_spender` to spend no more than `_value` tokens on your behalf, and then ping the
contract about it
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     * @param _extraData some extra information to send to the approved contract
     */
    function approveAndCall(address _spender, uint256 _value, bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
        if (approve(_spender, _value)) {
            spender.receiveApproval(msg.sender, _value, this, _extraData);
            return true;
        }
    }
```

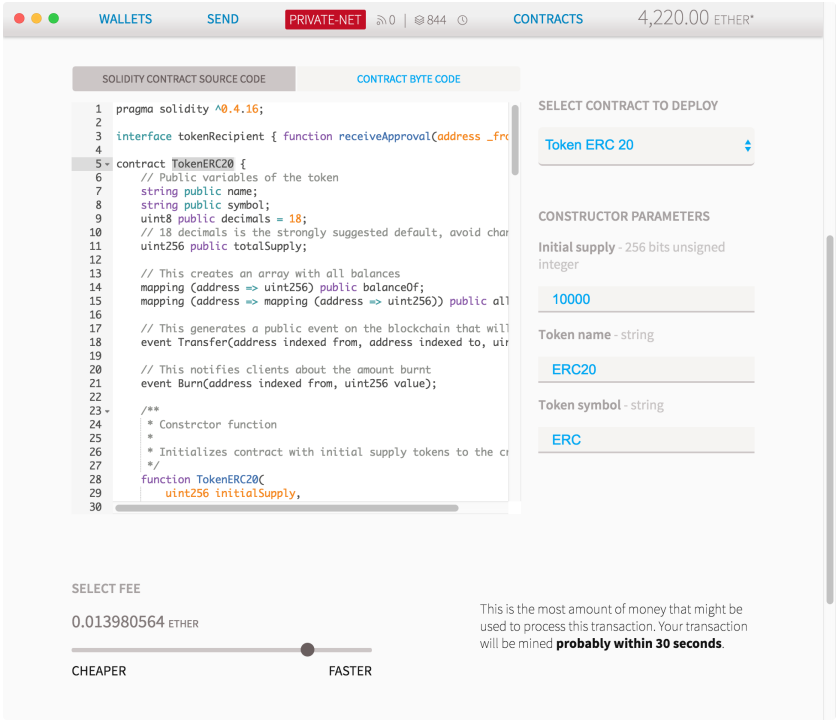
```
    /**
     * Destroy tokens
     *
     * Remove `_value` tokens from the system irreversibly
     *
     * @param _value the amount of money to burn
     */
    function burn(uint256 _value) public returns (bool success) {
        require(balanceOf[msg.sender] >= _value);   // Check if the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract from the sender
        totalSupply -= _value;                      // Updates totalSupply
        Burn(msg.sender, _value);
        return true;
    }

    /**
     * Destroy tokens from other account
     *
     * Remove `_value` tokens from the system irreversibly on behalf of `_from`.
     *
     * @param _from the address of the sender
     * @param _value the amount of money to burn
     */
    function burnFrom(address _from, uint256 _value) public returns (bool success) {
        require(balanceOf[_from] >= _value);                // Check if the targeted balance is
enough
        require(_value <= allowance[_from][msg.sender]);    // Check allowance
        balanceOf[_from] -= _value;                         // Subtract from the targeted
balance
        allowance[_from][msg.sender] -= _value;             // Subtract from the sender's
allowance
        totalSupply -= _value;                              // Update totalSupply
        Burn(_from, _value);
        return true;
    }
}
```

1.2. 部署合约

启动 Ethereum Wallet，点击 CONTRACTS 按钮，进入合约管理界面



点击 DEPLOY NEW CONTRACT 按钮，部署一个新合约



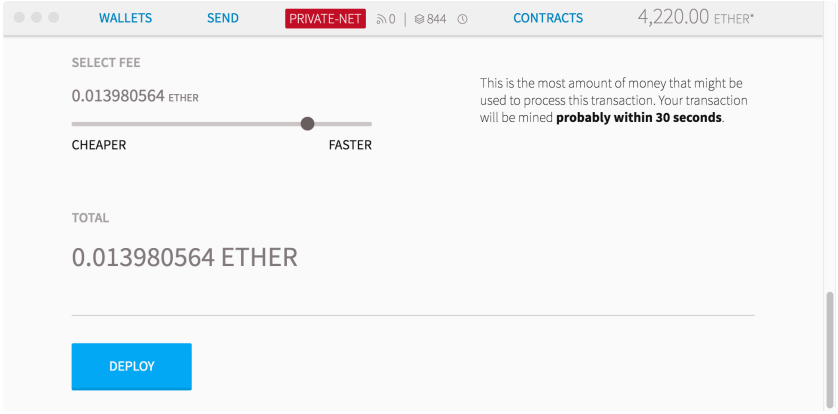
复制粘贴合约文件到 SOLIDITY CONTRACT SOURCE CODE 下方

SELECT CONTRACT TO DEPLOY 列表选择 “Token ERC 20”

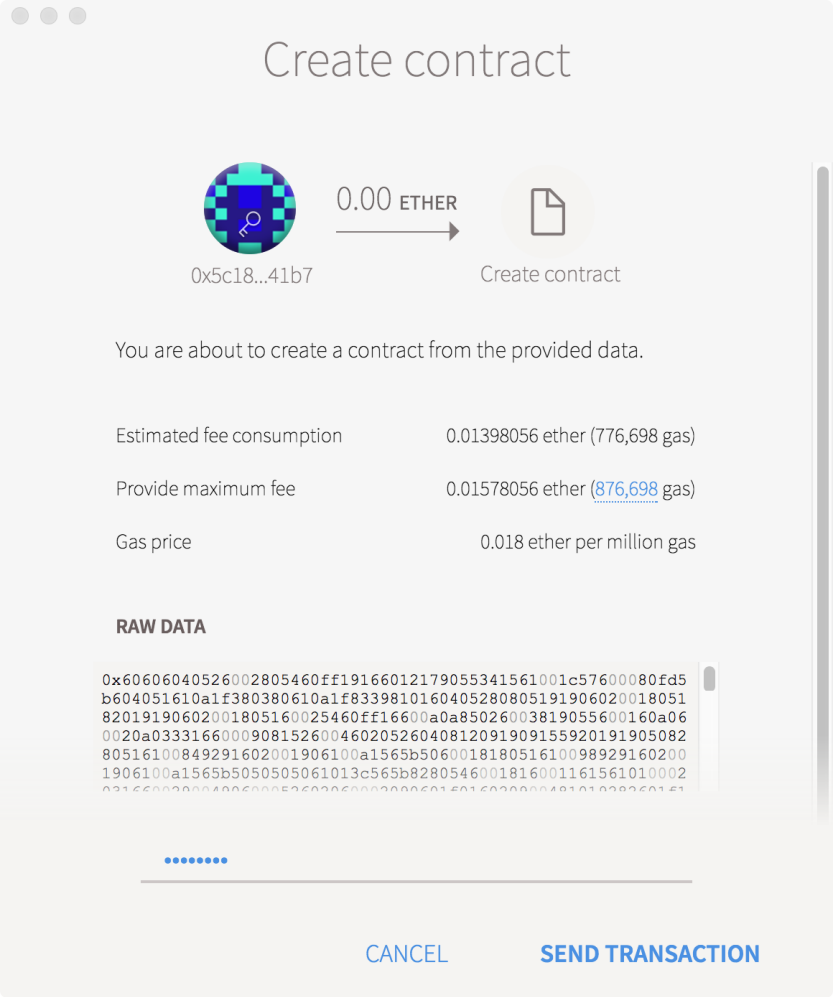
Initial supply 是初始发行货币量

Token name 是代币名称

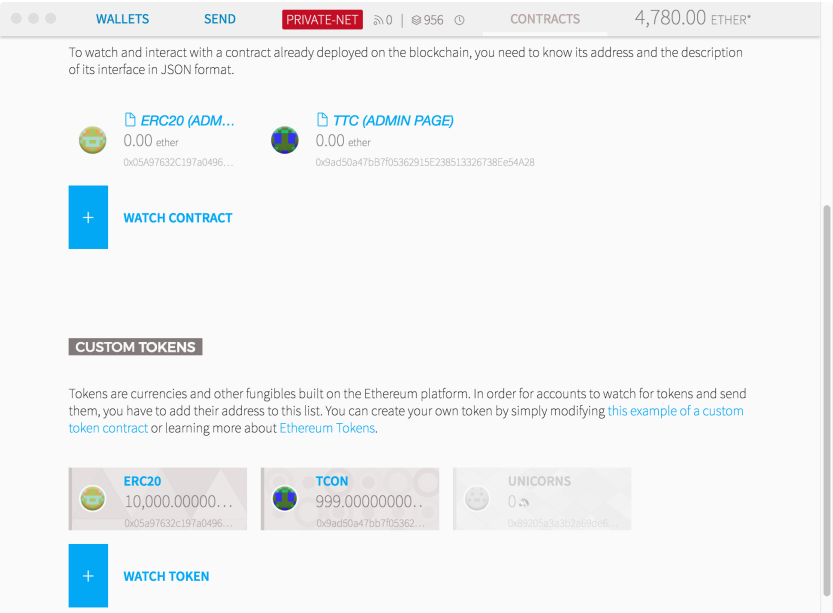
Token symbol 是代币符号



拉动滚动调，找到下方 “DEPLOY”按钮，点击该按钮。



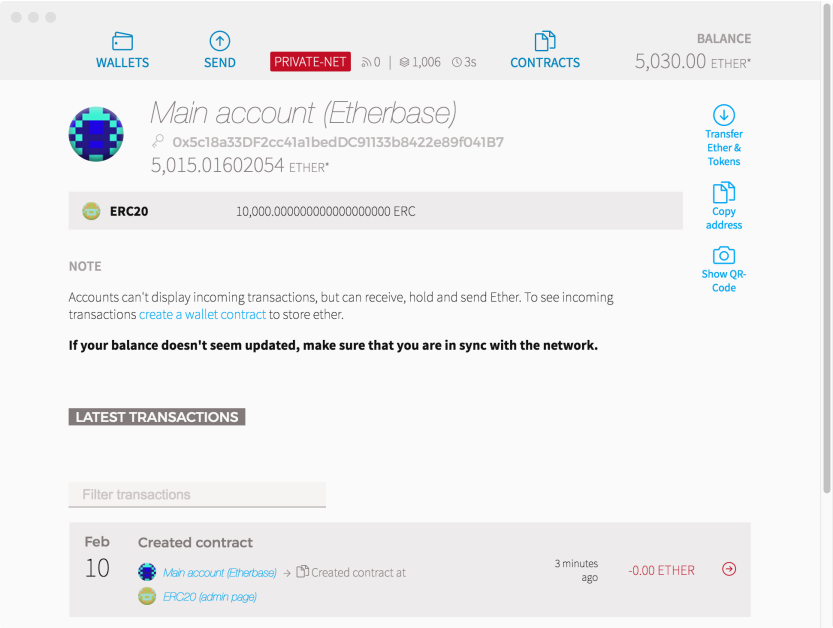
输入账号密码，并点击“SEND TRANSACTION” 按钮。



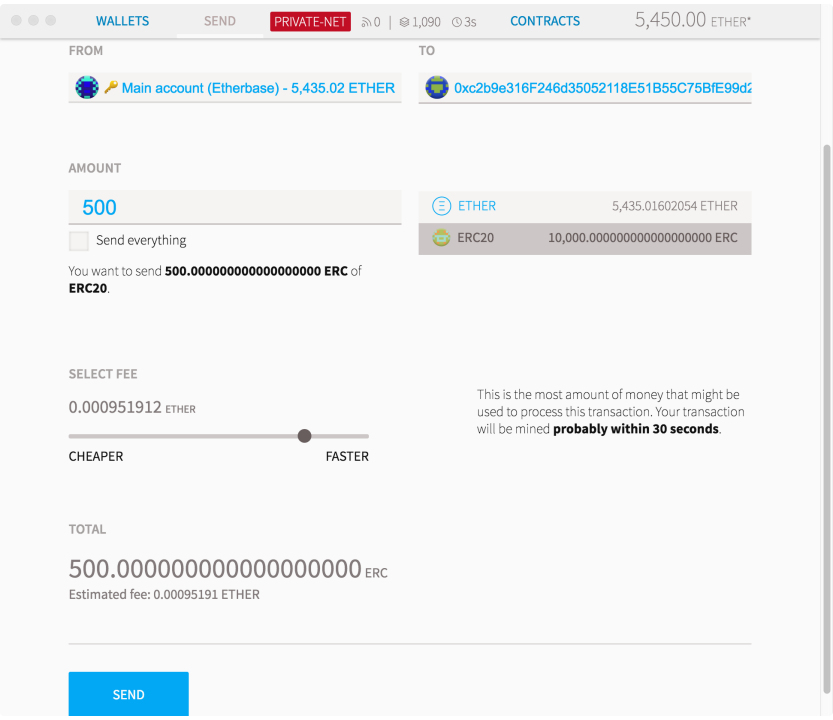
ERC20代币创建完成

1.3. 代币转账

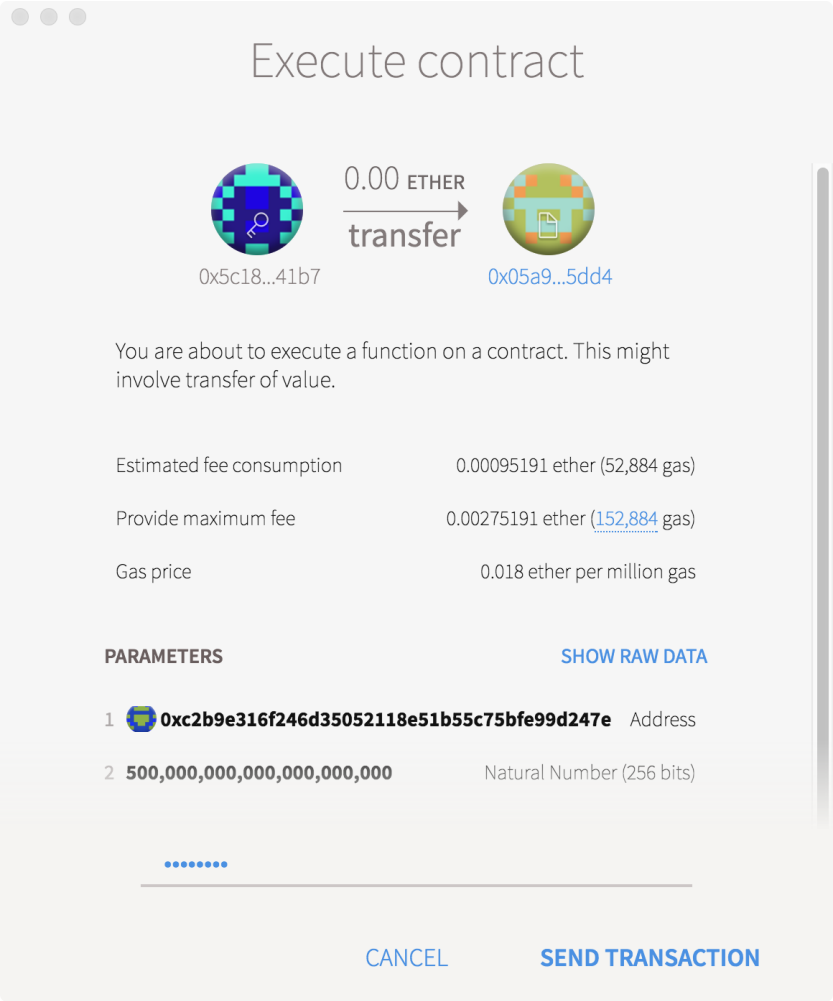
进入钱包可以看到当前账号的以太币数量，在下方还能看到 ERC20 代币。



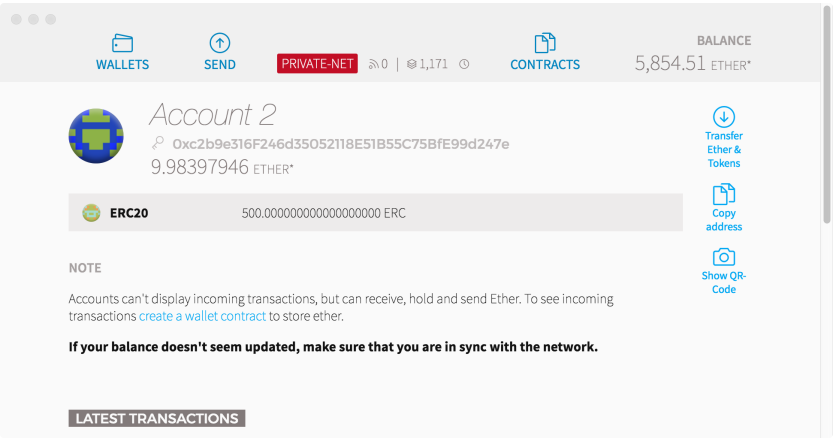
点击 SEND 按钮



填写 TO 地址 和 代币 500 个，点击 SEND 按钮

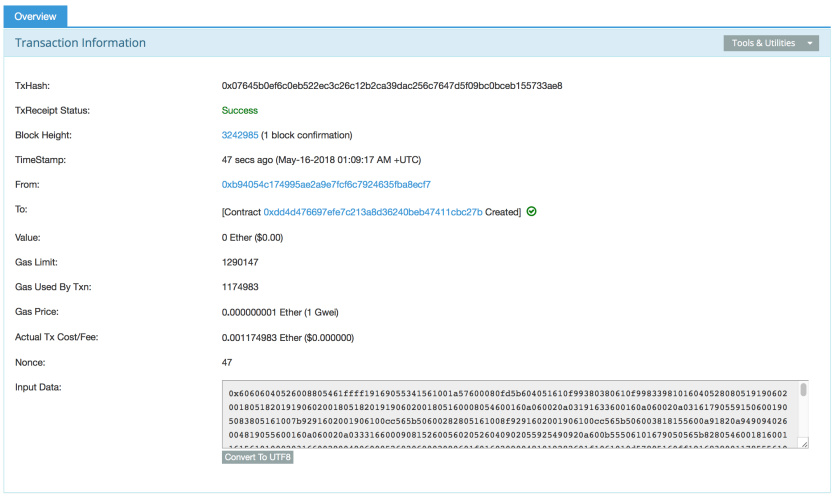


进入目标账号查看余额。

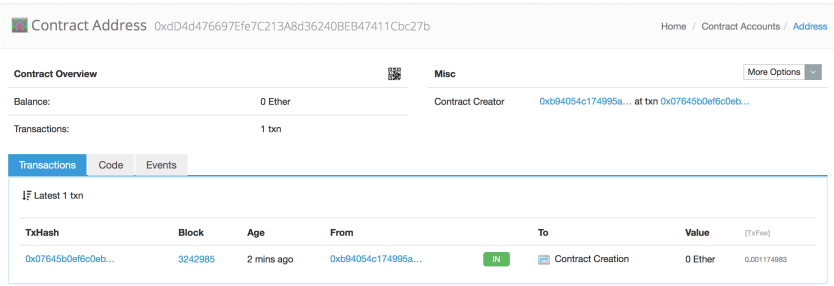


1.4. Verify And Publish

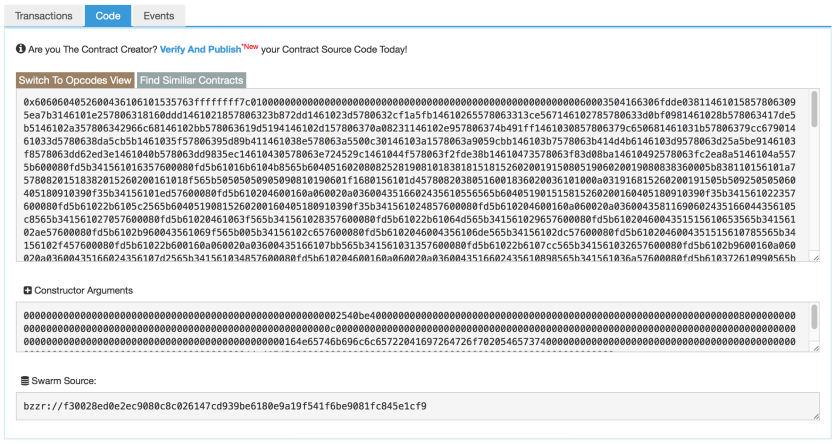
查看合约执行状态等待执行完成 TxReceipt Status:Success



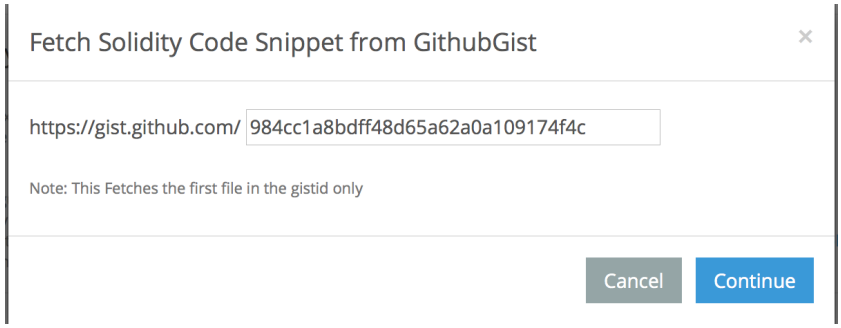
点击合约地址，查看合约



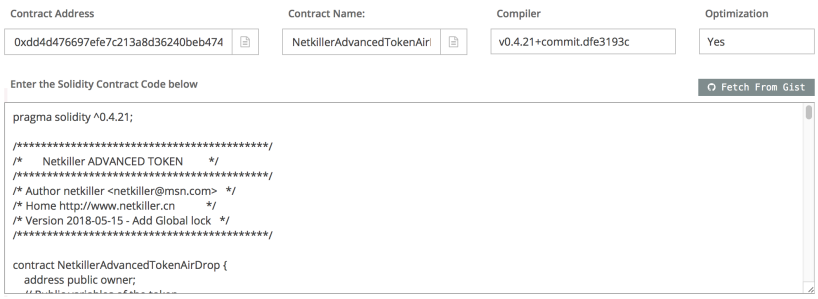
进入 Code 选项卡，点击 Verify And Publish 按钮



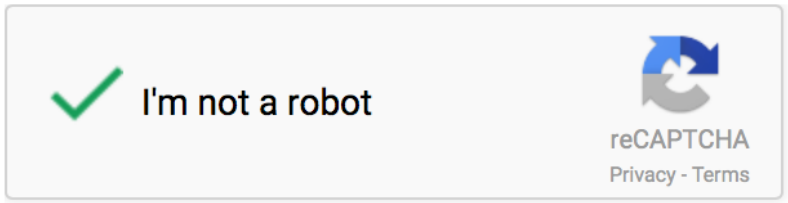
输入合约名称 Contract Name，选择编译器版本 Compiler，点击 Fetch From Gist 按钮输入 Gist 上的合约地址。



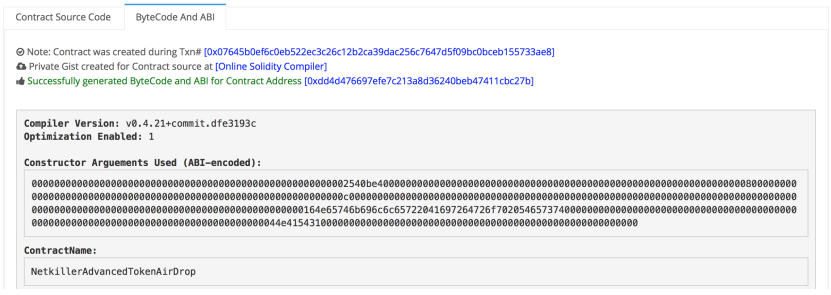
Optimization 选择 Yes



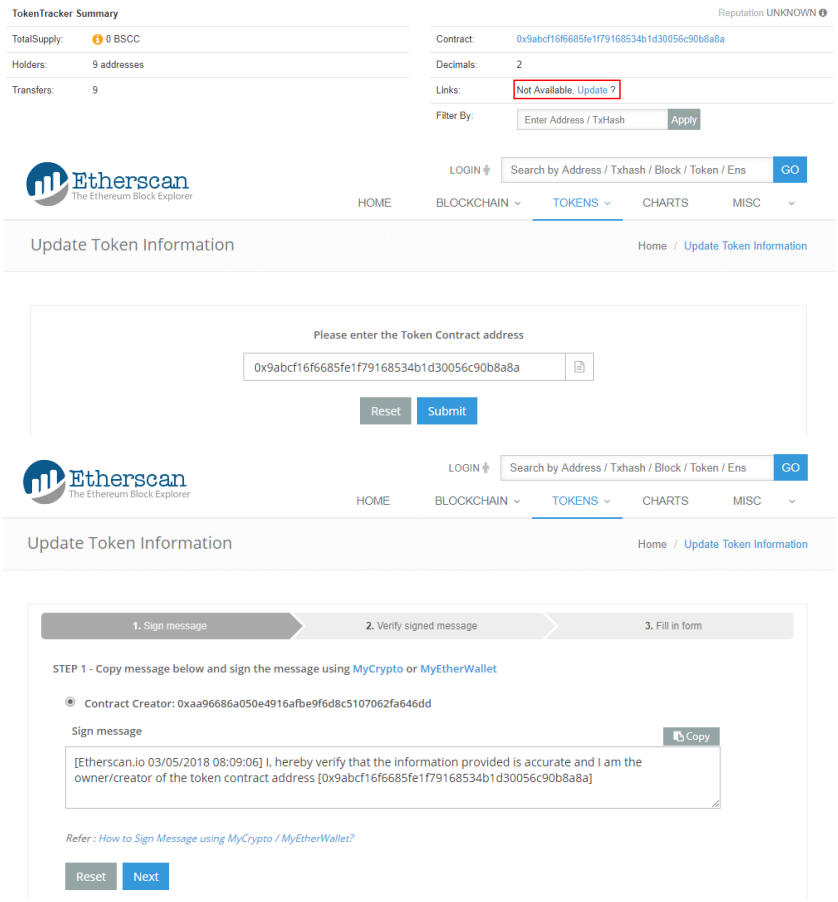
勾选验证码, 然后点击 Verify And Publish 按钮



合约校验成功



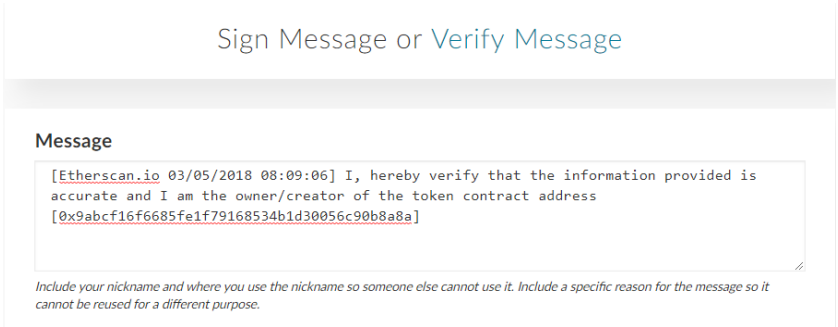
1.5. Links 链接更新



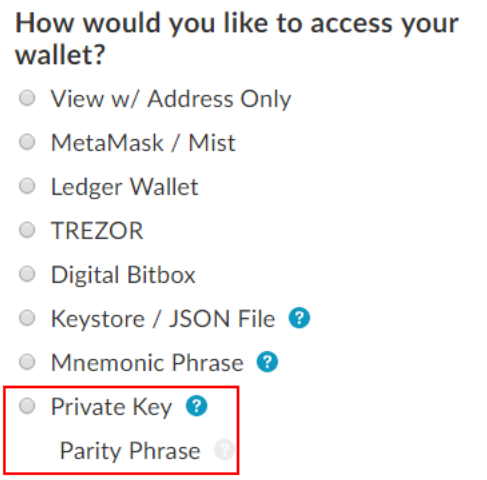
复制签名信息

进入 MyEtherWallet 签名页面 https://www.myetherwallet.com/signmsg.html

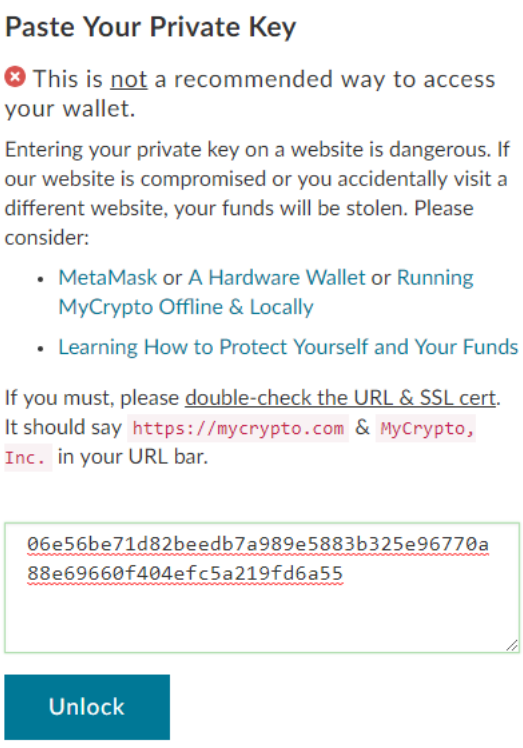
粘贴签名信息



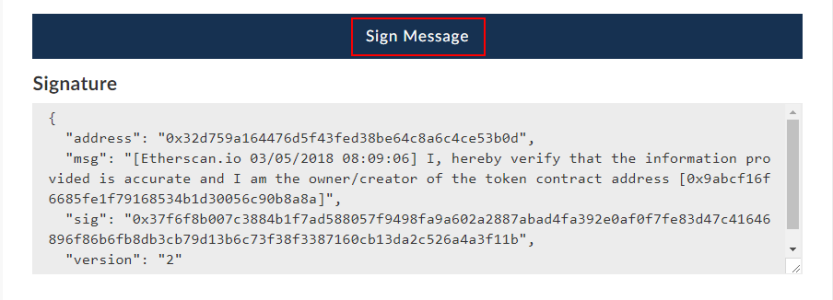
选择私钥签名



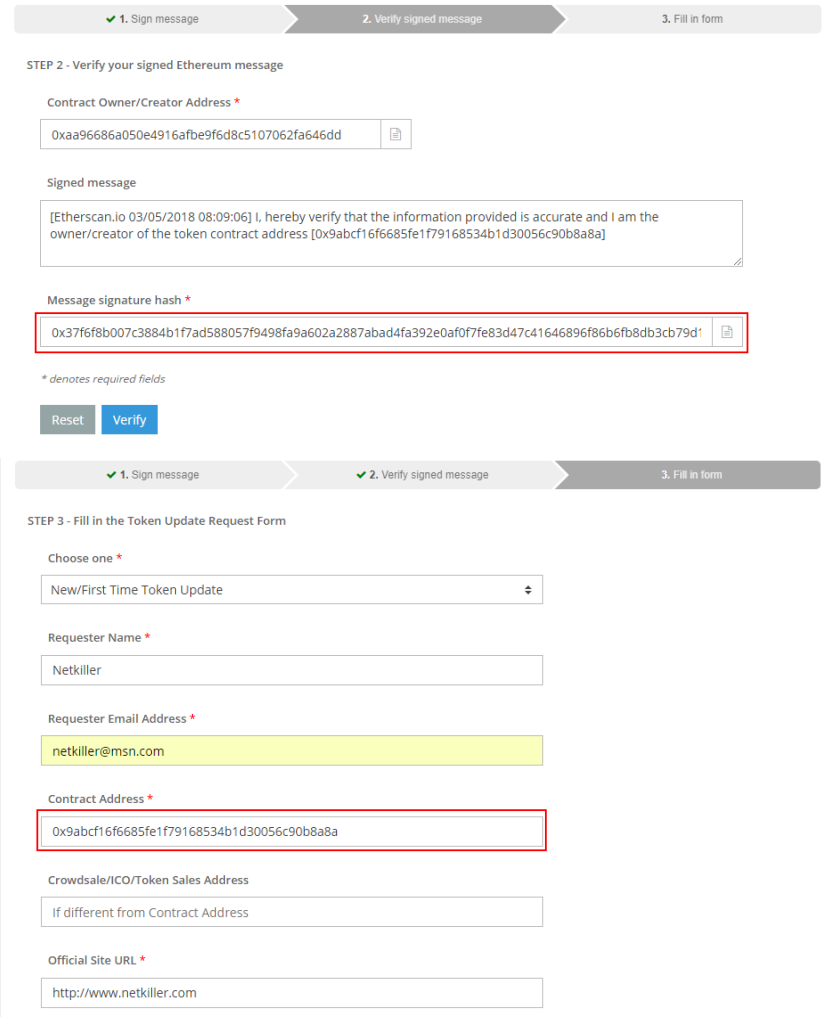
解锁账号



签名，复制 sig 的值



粘贴 sig，然后点击 Verify 按钮



信息填写完成后点击“Send Message” 保存

2. ERC20 Token Solidity 0.4.24

https://theethereum.wiki/w/index.php/ERC20_Token_Standard

ERC20 “描述了实现代币合约的标准功能”，ERC20 是各个代币的标准接口。ERC20 代币仅仅是以太坊代币的子集。为了充分兼容 ERC20，开发者需要将一组特定的函数集成到他们的智能合约中，以便在高层面能够执行以下操作：

ERC20 提供的方法

- 获得代币总供应量
- 获得账户余额
- 转让代币
- 批准花费代币

2.1. 构造方法

原合约

```
pragma solidity ^0.4.21;

contract NetkillerToken {
  address public owner;
  string public name;
  string public symbol;
  uint public decimals;
  uint256 public totalSupply;

  event Transfer(address indexed from, address indexed to,
uint256 value);
```

```
  /* This creates an array with all balances */
  mapping (address => uint256) public balanceOf;

  function NetkillerToken(uint256 initialSupply, string
tokenName, string tokenSymbol, uint decimalUnits) public {
    owner = msg.sender;
    name = tokenName;
    symbol = tokenSymbol;
    decimals = decimalUnits;
    totalSupply = initialSupply * 10 ** uint256(decimals);
    balanceOf[msg.sender] = totalSupply;
  }

  /* Send coins */
  function transfer(address _to, uint256 _value) public {
    /* Check if the sender has balance and for overflows */
    require(balanceOf[msg.sender] >= _value && balanceOf[_to] +
_value >= balanceOf[_to]);

    /* Add and subtract new balances */
    balanceOf[msg.sender] -= _value;
    balanceOf[_to] += _value;

    /* Notify anyone listening that this transfer took place */
    emit Transfer(msg.sender, _to, _value);
  }
}
```

新版合约

```
pragma solidity ^0.4.24;

contract NetkillerToken {
  address public owner;
  string public name;
  string public symbol;
  uint public decimals;
  uint256 public totalSupply;

  event Transfer(address indexed from, address indexed to,
uint256 value);

  /* This creates an array with all balances */
```

```
  mapping (address => uint256) public balanceOf;

  constructor(uint256 initialSupply, string tokenName, string
tokenSymbol, uint decimalUnits) public {
    owner = msg.sender;
    name = tokenName;
    symbol = tokenSymbol;
    decimals = decimalUnits;
    totalSupply = initialSupply * 10 ** uint256(decimals);
    balanceOf[msg.sender] = totalSupply;
  }

  /* Send coins */
  function transfer(address _to, uint256 _value) public {
    /* Check if the sender has balance and for overflows */
    require(balanceOf[msg.sender] >= _value && balanceOf[_to] +
_value >= balanceOf[_to]);

    /* Add and subtract new balances */
    balanceOf[msg.sender] -= _value;
    balanceOf[_to] += _value;

    /* Notify anyone listening that this transfer took place */
    emit Transfer(msg.sender, _to, _value);
  }
}
```

2.2. 官方规定 Method 方法

所有的ERC20代币都是按照下面这些方法来定义的。下面我们讲解一下每个方法的作用。

2.2.1. name

function name() view public returns (string name)

返回string类型的ERC20代币的名字，例如：Netkiller Reader Coin

2.2.2. symbol

function symbol() view public returns (string symbol)

返回string类型的ERC20代币的符号，也就是代币的简称，例如：NRC。

2.2.3. decimals

function decimals() view public returns (uint decimals)

支持几位小数点后几位。如果设置为3。也就是支持0.001表示。

2.2.4. totalSupply

function totalSupply() view public returns (uint256 totalSupply)

发行代币的总量，可以通过这个函数来获取。所有智能合约发行的代币总量是一定的，totalSupply必须设置初始值。

2.2.5. balanceOf

function balanceOf(address _owner) public returns (uint256 balance)

输入地址，可以获取该地址代币的余额。

2.2.6. transfer

function transfer(address _to, uint256 _value) public returns (bool success)

调用transfer函数将自己的token转账给_to地址，_value为转账金额

2.2.7. approve

function approve(address _spender, uint256 _value) public returns (bool success)

批准_spender账户从自己的账户转移_value个token。可以分多次转移。

2.2.8. transferFrom

function transferFrom(address _from, address _to, uint256 _value) public returns (bool success)

与approve搭配使用，approve批准之后，调用transferFrom函数来转移token。

2.2.9. allowance

function allowance(address _owner, address _spender) public returns (uint256 remaining)

返回_spender还能提取token的个数。

approve、transferFrom及allowance解释：账户A有1000个代币，想允许B账户随意调用100个代币。A账户按照以下形式调用approve函数approve(B,100)。当B账户想用这100个代币中的10个代币给C账户时，则调用transferFrom(A, C, 10)。这时调用allowance(A, B)可以查看B账户还能够调用A账户多少个token。

2.3. 事件

2.3.1. Transfer

event Transfer(address indexed _from, address indexed _to, uint256 _value)

当成功转移token时，一定要触发Transfer事件

2.3.2. Approval

event Approval(address indexed _owner, address indexed _spender, uint256 _value)

当调用approval函数成功时，一定要触发Approval事件

3. Netkiller Crowdsale Contract

3.1. Solidity 0.4.24

```
pragma solidity ^0.4.24;

/******************************************/
/*      Netkiller Crowdsale Contract      */
/******************************************/
/* Author netkiller <netkiller@msn.com>   */
/* Home http://www.netkiller.cn           */
/* Version 2018-06-07 - Solc ver: 0.4.24  */
/******************************************/

interface token {
    function transfer(address receiver, uint amount) external;
}

contract Crowdsale {
    address public beneficiary;
    uint public fundingGoal;
    uint public amountRaised;
    uint public deadline;
    uint public price;
    token public tokenReward;
    mapping(address => uint256) public balanceOf;
    bool fundingGoalReached = false;
    bool crowdsaleClosed = false;

    event GoalReached(address recipient, uint
totalAmountRaised);
    event FundTransfer(address backer, uint amount, bool
isContribution);

    /**
     * Constructor function
     *
     * Setup the owner
     */
    constructor(
        address ifSuccessfulSendTo,
        uint fundingGoalInEthers,
```

```
        uint durationInMinutes,
        uint etherCostOfEachToken,
        address addressOfTokenUsedAsReward
    ) public {
        beneficiary = ifSuccessfulSendTo;
        fundingGoal = fundingGoalInEthers * 1 ether;
        deadline = now + durationInMinutes * 1 minutes;
        price = etherCostOfEachToken * 1 ether;
        tokenReward = token(addressOfTokenUsedAsReward);
    }

    /**
     * Fallback function
     *
     * The function without name is the default function that
is called whenever anyone sends funds to a contract
     */
    function () payable public{
        require(!crowdsaleClosed);
        uint amount = msg.value;
        balanceOf[msg.sender] += amount;
        amountRaised += amount;
        tokenReward.transfer(msg.sender, amount / price);
        emit FundTransfer(msg.sender, amount, true);
    }

    modifier afterDeadline() { if (now >= deadline) _; }

    /**
     * Check if goal was reached
     *
     * Checks if the goal or time limit has been reached and
ends the campaign
     */
    function checkGoalReached() afterDeadline public{
        if (amountRaised >= fundingGoal){
            fundingGoalReached = true;
            emit GoalReached(beneficiary, amountRaised);
        }
        crowdsaleClosed = true;
    }


    /**
     * Withdraw the funds
     *
     * Checks to see if goal or time limit has been reached,
and if so, and the funding goal was reached,
```

```
     * sends the entire amount to the beneficiary. If goal was
not reached, each contributor can withdraw
     * the amount they contributed.
     */
    function safeWithdrawal() afterDeadline public{
        if (!fundingGoalReached) {
            uint amount = balanceOf[msg.sender];
            balanceOf[msg.sender] = 0;
            if (amount > 0) {
                if (msg.sender.send(amount)) {
                    emit FundTransfer(msg.sender, amount,
false);
                } else {
                    balanceOf[msg.sender] = amount;
                }
            }
        }

        if (fundingGoalReached && beneficiary == msg.sender) {
            if (beneficiary.send(amountRaised)) {
                emit FundTransfer(beneficiary, amountRaised,
false);
            } else {
                //If we fail to send the funds to beneficiary,
unlock funders balance
                fundingGoalReached = false;
            }
        }
    }
}
```

3.2. Solidity 0.4.21

```
pragma solidity ^0.4.21;

interface token {
    function transfer(address receiver, uint amount) external;
}

contract Crowdsale {
    address public beneficiary; // 募资成功后的收款地址
    uint public fundingGoal;    // 募资额度
```

```
    uint public amountRaised;   // 参与数量
    uint public deadline;       // 募资截止期

    uint public price;          // token 与以太坊的汇率, token卖多
少钱
    token public tokenReward;   // 要卖的token

    mapping(address => uint256) public balanceOf;

    bool fundingGoalReached = false;    // 众筹是否达到目标
    bool crowdsaleClosed = false;       // 众筹是否结束

    /**
    * 事件可以用来跟踪信息
    **/
    event GoalReached(address recipient, uint
totalAmountRaised);
    event FundTransfer(address backer, uint amount, bool
isContribution);

    /**
     * 构造函数, 设置相关属性
     */
    function Crowdsale(
        address ifSuccessfulSendTo,
        uint fundingGoalInEthers,
        uint durationInMinutes,
        uint finneyCostOfEachToken,
        address addressOfTokenUsedAsReward) public {
            beneficiary = ifSuccessfulSendTo;
//募资成功后的收款账号
            fundingGoal = fundingGoalInEthers * 1 ether;
//募资额度
            deadline = now + durationInMinutes * 1 minutes;
//募资时间
            price = finneyCostOfEachToken * 1 finney;
//每个代币的价格, 这里为了方便使用了单位finney及值为: 1 (1 ether =
1000 finney)
            tokenReward = token(addressOfTokenUsedAsReward);
// 代币合约地址。传入已发布的 token 合约的地址来创建实例
    }

    /**
     * 无函数名的Fallback函数,
```

```
     * 在向合约转账时，这个函数会被调用
     */
    function () payable public {
        require(!crowdsaleClosed);
        uint amount = msg.value;
        balanceOf[msg.sender] += amount;
        amountRaised += amount;
        tokenReward.transfer(msg.sender, amount / price);
        emit FundTransfer(msg.sender, amount, true);
    }

    /**
    *  定义函数修改器modifier
    * 用于在函数执行前检查某种前置条件（判断通过之后才会继续执行该方法）
    * _ 表示继续执行之后的代码
    **/
    modifier afterDeadline() { if (now >= deadline) _; }

    /**
     * 判断众筹是否完成融资目标， 这个方法使用了afterDeadline函数修改器
     *
     */
    function checkGoalReached() afterDeadline public{
        if (amountRaised >= fundingGoal) {
            fundingGoalReached = true;
            emit GoalReached(beneficiary, amountRaised);
        }
        crowdsaleClosed = true;
    }


    /**
     * 完成融资目标时，融资款发送到收款方
     * 未完成融资目标时，执行退款
     *
     */
    function safeWithdrawal() afterDeadline public{
        if (!fundingGoalReached) {
            uint amount = balanceOf[msg.sender];
            balanceOf[msg.sender] = 0;
            if (amount > 0) {
                if (msg.sender.send(amount)) {
                    emit FundTransfer(msg.sender, amount,
false);
                } else {
                    balanceOf[msg.sender] = amount;
```

```
                }
            }
        }

        if (fundingGoalReached && beneficiary == msg.sender) {
            if (beneficiary.send(amountRaised)) {
                emit FundTransfer(beneficiary, amountRaised,
false);
            } else {
                //If we fail to send the funds to beneficiary,
unlock funders balance
                fundingGoalReached = false;
            }
        }
    }
}
```

4. ERC721 - Non-Fungible Tokens

4.1.

```
interface ERC721 {

    event Transfer(address indexed _from, address indexed _to, uint256 indexed
_tokenId);
    event Approval(address indexed _owner, address indexed _approved, uint256
indexed _tokenId);
    event ApprovalForAll(address indexed _owner, address indexed _operator, bool
_approved);
    function balanceOf(address _owner) external view returns (uint256);
    function ownerOf(uint256 _tokenId) external view returns (address);
    function safeTransferFrom(address _from, address _to, uint256 _tokenId,
bytes data) external payable;
    function safeTransferFrom(address _from, address _to, uint256 _tokenId)
external payable;
    function transferFrom(address _from, address _to, uint256 _tokenId) external
payable;
    function approve(address _approved, uint256 _tokenId) external payable;
    function setApprovalForAll(address _operator, bool _approved) external;
    function getApproved(uint256 _tokenId) external view returns (address);
    function isApprovedForAll(address _owner, address _operator) external view
returns (bool);
}

interface ERC165 {
    function supportsInterface(bytes4 interfaceID) external view returns (bool);
}

interface ERC721TokenReceiver {
    function onERC721Received(address _operator, address _from, uint256
_tokenId, bytes _data) external returns(bytes4);
}
```

4.2. ERC721Metadata (可选)

可选实现接口，ERC721Metadata 接口用于提供合约的元数据：name , symbol 及 URI（NFT所对应的资源）。

其接口定义如下：

```
interface ERC721Metadata /* is ERC721 */ {
    function name() external pure returns (string _name);
    function symbol() external pure returns (string _symbol);
    function tokenURI(uint256 _tokenId) external view returns (string);
}
```

接口说明：
name()：返回合约名字，尽管是可选，但强烈建议实现，即便是返回空字符串。
symbol()：返回合约代币符号，尽管是可选，但强烈建议实现，即便是返回空字符串。
tokenURI()：返回_tokenId所对应的外部资源文件的URI（通常是IPFS或HTTP(S)路径）。外部资源文件需要包含名字、描述、图片，其格式的要求如下：

4.3. ERC721Enumerable （可选）

可选实现接口，ERC721Enumerable的主要目的是提高合约中NTF的可访问性，其接口定义如下：

```
interface ERC721Enumerable /* is ERC721 */ {
    function totalSupply() external view returns (uint256);
    function tokenByIndex(uint256 _index) external view returns (uint256);
    function tokenOfOwnerByIndex(address _owner, uint256 _index) external view
returns (uint256);
}
```

接口说明：

totalSupply()：返回NFT总量
tokenByIndex()：通过索引返回对应的tokenId。
tokenOfOwnerByIndex()：所有者可以一次拥有多个的NFT，此函数返回_owner拥有的NFT列表中对应索引的tokenId。

5. 经典参考案例

一下是作者发布的合约，供读者参考

5.1. Enterprise Token Ecosystem (ETE)

合约地址
https://etherscan.io/address/0x6333050c7a025027b51a8039cbafd2584933299d#code

5.2. 积分链 (PE Chain)

合约地址
https://etherscan.io/address/0x9bcf2ea3a9e18e550eba88c97a8bb6b7ea4f58b7#code

5.3. Global star league chain (GSLC)

合约地址
https://etherscan.io/token/0x7ffdcccc3e7e33c6163393195a947a6d45f25814

5.4. Kyber Network

合约地址
https://etherscan.io/address/0xC14f34233071543E979F6A79AA272b0AB1B4947D#code

该合约可以实现币币兑换

6. 代币合约官方文档

6.1. ERC20

https://github.com/ethereum/EIPs/issues/20

https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md

合约源码:
https://github.com/ConsenSys/Tokens/tree/master/contracts/eip20

https://theethereum.wiki/w/index.php/ERC20_Token_Standard

```
 // ----------------------------------------------------------------------
 // ERC20 Token Standard Interface
 // https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md
 // ----------------------------------------------------------------------
contract ERC20 {
      function name() constant returns (string name)
      function symbol() constant returns (string symbol)
      function decimals() constant returns (uint8 decimals)
      function totalSupply() constant returns (uint totalSupply);
      function balanceOf(address _owner) constant returns (uint balance);
      function transfer(address _to, uint _value) returns (bool success);
      function transferFrom(address _from, address _to, uint _value) returns (bool success);
      function approve(address _spender, uint _value) returns (bool success);
      function allowance(address _owner, address _spender) constant returns (uint remaining);
      event Transfer(address indexed _from, address indexed _to, uint _value);
      event Approval(address indexed _owner, address indexed _spender, uint _value);
```

```
}
```

6.1.1. 基本Token 官方提供的例子

这个例子仅仅用来初学使用，请不用使用在生产环境。

提供例子的地址： https://ethereum.org/token

这个例子中没有涉及合约管理者，所以任何人都可以操作这个合约。例如这个减持的函数burn() 人人都可以调用。所以这个例子只能用来学习ERC20合约开发，相当于Helloword 程序。

```
pragma solidity ^0.4.16;

interface tokenRecipient { function receiveApproval(address
_from, uint256 _value, address _token, bytes _extraData)
public; }

contract TokenERC20 {
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
allowance;

    // This generates a public event on the blockchain that
will notify clients
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);
```

```
    /**
     * Constructor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function TokenERC20(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        totalSupply = initialSupply * 10 ** uint256(decimals);
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
        symbol = tokenSymbol;                               //
Set the symbol for display purposes
    }

    /**
     * Internal transfer, only can be called by this contract
     */
    function _transfer(address _from, address _to, uint _value)
internal {
        // Prevent transfer to 0x0 address. Use burn() instead
        require(_to != 0x0);
        // Check if the sender has enough
        require(balanceOf[_from] >= _value);
        // Check for overflows
        require(balanceOf[_to] + _value > balanceOf[_to]);
        // Save this for an assertion in the future
        uint previousBalances = balanceOf[_from] +
balanceOf[_to];
        // Subtract from the sender
        balanceOf[_from] -= _value;
        // Add the same to the recipient
        balanceOf[_to] += _value;
        Transfer(_from, _to, _value);
        // Asserts are used to use static analysis to find bugs
in your code. They should never fail
        assert(balanceOf[_from] + balanceOf[_to] ==
previousBalances);
    }

    /**
```

```
     * Transfer tokens
     *
     * Send `_value` tokens to `_to` from your account
     *
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transfer(address _to, uint256 _value) public {
        _transfer(msg.sender, _to, _value);
    }

    /**
     * Transfer tokens from other address
     *
     * Send `_value` tokens to `_to` on behalf of `_from`
     *
     * @param _from The address of the sender
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(_value <= allowance[_from][msg.sender]);     //
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    /**
     * Set allowance for other address
     *
     * Allows `_spender` to spend no more than `_value` tokens
on your behalf
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     */
    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        return true;
    }

    /**
     * Set allowance for other address and notify
     *
     * Allows `_spender` to spend no more than `_value` tokens
```

```
on your behalf, and then ping the contract about it
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     * @param _extraData some extra information to send to the
approved contract
     */
    function approveAndCall(address _spender, uint256 _value,
bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
        if (approve(_spender, _value)) {
            spender.receiveApproval(msg.sender, _value, this,
_extraData);
            return true;
        }
    }

    /**
     * Destroy tokens
     *
     * Remove `_value` tokens from the system irreversibly
     *
     * @param _value the amount of money to burn
     */
    function burn(uint256 _value) public returns (bool success)
{
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        Burn(msg.sender, _value);
        return true;
    }

    /**
     * Destroy tokens from other account
     *
     * Remove `_value` tokens from the system irreversibly on
behalf of `_from`.
     *
     * @param _from the address of the sender
     * @param _value the amount of money to burn
     */
    function burnFrom(address _from, uint256 _value) public
```

```
returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
Update totalSupply
        Burn(_from, _value);
        return true;
    }
}
```

6.1.2. 官方提供的例子 ADVANCED TOKEN

提供例子的地址： https://ethereum.org/token 网页的下方

这个例子已经比较完善，但仍不能够用在生产环境，因为 function burn(uint256 _value) public returns (bool success) 仍然没有控制访问权限。

如果在生产环境使用这个合约你需要修改两处

```
function burn(uint256 _value) public returns (bool success)
改为
function burn(uint256 _value) onlyOwner public returns (bool
success)
```

```
function burnFrom(address _from, uint256 _value) public returns
(bool success)
改为
function burnFrom(address _from, uint256 _value) onlyOwner
public returns (bool success)
```

这样合约就很安全了。只能创建者可以减持代币。

```
pragma solidity ^0.4.16;

contract owned {
    address public owner;

    function owned() public {
        owner = msg.sender;
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    function transferOwnership(address newOwner) onlyOwner
public {
        owner = newOwner;
    }
}

interface tokenRecipient { function receiveApproval(address
_from, uint256 _value, address _token, bytes _extraData)
public; }

contract TokenERC20 {
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
allowance;

    // This generates a public event on the blockchain that
will notify clients
```

```
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function TokenERC20(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        totalSupply = initialSupply * 10 ** uint256(decimals);
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
        symbol = tokenSymbol;                               //
Set the symbol for display purposes
    }

    /**
     * Internal transfer, only can be called by this contract
     */
    function _transfer(address _from, address _to, uint _value)
internal {
        // Prevent transfer to 0x0 address. Use burn() instead
        require(_to != 0x0);
        // Check if the sender has enough
        require(balanceOf[_from] >= _value);
        // Check for overflows
        require(balanceOf[_to] + _value > balanceOf[_to]);
        // Save this for an assertion in the future
        uint previousBalances = balanceOf[_from] +
balanceOf[_to];
        // Subtract from the sender
        balanceOf[_from] -= _value;
        // Add the same to the recipient
        balanceOf[_to] += _value;
        Transfer(_from, _to, _value);
        // Asserts are used to use static analysis to find bugs
in your code. They should never fail
```

```
        assert(balanceOf[_from] + balanceOf[_to] ==
previousBalances);
    }

    /**
     * Transfer tokens
     *
     * Send `_value` tokens to `_to` from your account
     *
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transfer(address _to, uint256 _value) public {
        _transfer(msg.sender, _to, _value);
    }

    /**
     * Transfer tokens from other address
     *
     * Send `_value` tokens to `_to` in behalf of `_from`
     *
     * @param _from The address of the sender
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(_value <= allowance[_from][msg.sender]);     //
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    /**
     * Set allowance for other address
     *
     * Allows `_spender` to spend no more than `_value` tokens
in your behalf
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     */
    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        return true;
    }
```

```
    /**
     * Set allowance for other address and notify
     *
     * Allows `_spender` to spend no more than `_value` tokens
in your behalf, and then ping the contract about it
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     * @param _extraData some extra information to send to the
approved contract
     */
    function approveAndCall(address _spender, uint256 _value,
bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
        if (approve(_spender, _value)) {
            spender.receiveApproval(msg.sender, _value, this,
_extraData);
            return true;
        }
    }

    /**
     * Destroy tokens
     *
     * Remove `_value` tokens from the system irreversibly
     *
     * @param _value the amount of money to burn
     */
    function burn(uint256 _value) public returns (bool success)
{
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        Burn(msg.sender, _value);
        return true;
    }

    /**
     * Destroy tokens from other account
     *
     * Remove `_value` tokens from the system irreversibly on
behalf of `_from`.
```

```
     *
     * @param _from the address of the sender
     * @param _value the amount of money to burn
     */
    function burnFrom(address _from, uint256 _value) public
returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
Update totalSupply
        Burn(_from, _value);
        return true;
    }
}

/**********************************************/
/*       ADVANCED TOKEN STARTS HERE       */
/**********************************************/

contract MyAdvancedToken is owned, TokenERC20 {

    uint256 public sellPrice;
    uint256 public buyPrice;

    mapping (address => bool) public frozenAccount;

    /* This generates a public event on the blockchain that
will notify clients */
    event FrozenFunds(address target, bool frozen);

    /* Initializes contract with initial supply tokens to the
creator of the contract */
    function MyAdvancedToken(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) TokenERC20(initialSupply, tokenName, tokenSymbol) public
{}

    /* Internal transfer, only can be called by this contract
*/
    function _transfer(address _from, address _to, uint _value)
```

```
internal {
        require (_to != 0x0);                                   //
Prevent transfer to 0x0 address. Use burn() instead
        require (balanceOf[_from] >= _value);                   //
Check if the sender has enough
        require (balanceOf[_to] + _value > balanceOf[_to]); //
Check for overflows
        require(!frozenAccount[_from]);                         //
Check if sender is frozen
        require(!frozenAccount[_to]);                           //
Check if recipient is frozen
        balanceOf[_from] -= _value;                             //
Subtract from the sender
        balanceOf[_to] += _value;                               //
Add the same to the recipient
        Transfer(_from, _to, _value);
    }

    /// @notice Create `mintedAmount` tokens and send it to
`target`
    /// @param target Address to receive the tokens
    /// @param mintedAmount the amount of tokens it will
receive
    function mintToken(address target, uint256 mintedAmount)
onlyOwner public {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
        Transfer(0, this, mintedAmount);
        Transfer(this, target, mintedAmount);
    }

    /// @notice `freeze? Prevent | Allow` `target` from sending
& receiving tokens
    /// @param target Address to be frozen
    /// @param freeze either to freeze it or not
    function freezeAccount(address target, bool freeze)
onlyOwner public {
        frozenAccount[target] = freeze;
        FrozenFunds(target, freeze);
    }

    /// @notice Allow users to buy tokens for `newBuyPrice` eth
and sell tokens for `newSellPrice` eth
    /// @param newSellPrice Price the users can sell to the
contract
    /// @param newBuyPrice Price users can buy from the
contract
    function setPrices(uint256 newSellPrice, uint256
```

```
newBuyPrice) onlyOwner public {
        sellPrice = newSellPrice;
        buyPrice = newBuyPrice;
    }

    /// @notice Buy tokens from contract by sending ether
    function buy() payable public {
        uint amount = msg.value / buyPrice;               //
calculates the amount
        _transfer(this, msg.sender, amount);              //
makes the transfers
    }

    /// @notice Sell `amount` tokens to contract
    /// @param amount amount of tokens to be sold
    function sell(uint256 amount) public {
        require(this.balance >= amount * sellPrice);      //
checks if the contract has enough ether to buy
        _transfer(msg.sender, this, amount);              //
makes the transfers
        msg.sender.transfer(amount * sellPrice);          //
sends ether to the seller. It's important to do this last to
avoid recursion attacks
    }
}
```

下面就分析一下这个合约

在许多应用场景中，需要管理发行的代币，为了对代币进行管理，需要给合约添加一个管理者，为此创建了 owned 合约。

```
contract owned {
    address public owner;

    function owned() {
        owner = msg.sender;
    }

    modifier onlyOwner {
```

```
        require(msg.sender == owner);
        _;
    }

    // 实现所有权转移
    function transferOwnership(address newOwner) onlyOwner {
        owner = newOwner;
    }
}
```

这个合约重要的是加入了一个函数修改器（Function Modifiers）onlyOwner，函数修改器是一个合约属性，可以被继承，还能被重写。它用于在函数执行前检查某种前置条件。

代币增发,实现代币增发，代币增发就如同央行印钞票一样，想必很多人都需要这样的功能。给合约添加以下的方法：

```
function mintToken(address target, uint256 mintedAmount)
onlyOwner {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
        Transfer(0, owner, mintedAmount);
        Transfer(owner, target, mintedAmount);
    }
```

注意onlyOwner修改器添加在函数末尾，这表示只有owner才能调用这用函数。他的功能很简单，就是给指定的账户增加代币，同时增加总供应量。

资产冻结

有时为了监管的需要，需要实现冻结某些账户，冻结后，其资产仍在账户，但是不允许交易，之道解除冻结。

给合约添加以下的变量和方法（可以添加到合约的任何地方，但是建议把mapping加

到和其他mapping一起，event也是如此）：

```
mapping (address => bool) public frozenAccount;
event FrozenFunds(address target, bool frozen);

function freezeAccount(address target, bool freeze) onlyOwner {
    frozenAccount[target] = freeze;
    FrozenFunds(target, freeze);
}
```

单单以上的代码还无法冻结，需要把他加入到transfer函数中才能真正生效，因此修改transfer函数

```
function transfer(address _to, uint256 _value) {
        require(!frozenAccount[msg.sender]);
        ...
}
```

这样在转账前，对发起交易的账号做一次检查，只有不是被冻结的账号才能转账。

代币买卖（兑换）

可以自己的货币中实现代币与其他数字货币（ether 或其他tokens）的兑换机制。有了这个功能，我们的合约就可以在一买一卖中赚利润了。

先来设置下买卖价格

```
uint256 public sellPrice;
uint256 public buyPrice;

function setPrices(uint256 newSellPrice, uint256 newBuyPrice)
onlyOwner {
    sellPrice = newSellPrice;
    buyPrice = newBuyPrice;
}
```

setPrices()添加了onlyOwner修改器，注意买卖的价格单位是wei（最小的货币单位： 1 eth = 1000000000000000000 wei)

添加来添加买卖函数：

```
function buy() payable returns (uint amount){
    amount = msg.value / buyPrice;                    //
```

```
calculates the amount
    require(balanceOf[this] >= amount);               // checks
if it has enough to sell
    balanceOf[msg.sender] += amount;                  // adds
the amount to buyer's balance
    balanceOf[this] -= amount;                        //
subtracts amount from seller's balance
    Transfer(this, msg.sender, amount);               //
execute an event reflecting the change
    return amount;                                    // ends
function and returns
}

function sell(uint amount) returns (uint revenue){
    require(balanceOf[msg.sender] >= amount);         // checks
if the sender has enough to sell
    balanceOf[this] += amount;                        // adds
the amount to owner's balance
    balanceOf[msg.sender] -= amount;                  //
subtracts the amount from seller's balance
    revenue = amount * sellPrice;
    msg.sender.transfer(revenue);                     // sends
ether to the seller: it's important to do this last to prevent
recursion attacks
    Transfer(msg.sender, this, amount);               //
executes an event reflecting on the change
    return revenue;                                   // ends
function and returns
}
```

加入了买卖功能后，要求我们在创建合约时发送足够的以太币，以便合约有能力回购市面上的代币，否则合约将破产，用户没法先合约卖代币。

实现Gas的自动补充

以太坊中的交易时需要gas（支付给矿工的费用，费用以ether来支付）。而如果用户没有以太币，只有代币的情况（或者我们想向用户隐藏以太坊的细节），就需要自动补充gas的功能。这个功能将使我们代币更加好用。

自动补充的逻辑是这样了，在执行交易之前，我们判断用户的余额（用来支付矿工的费用），如果用户的余额非常少（低于某个阈值时）可能影响到交易进行，合约自动售出一部分代币来补充余额，以帮助用户顺利完成交易。

先来设定余额阈值：

```
uint minBalanceForAccounts;

    function setMinBalance(uint minimumBalanceInFinney)
onlyOwner {
         minBalanceForAccounts = minimumBalanceInFinney * 1
finney;
    }
```

finney 是货币单位 1 finney = 0.001eth
然后交易中加入对用户的余额的判断。

```
function transfer(address _to, uint256 _value) {
    ...
    if(msg.sender.balance < minBalanceForAccounts)
        sell((minBalanceForAccounts - msg.sender.balance) /
sellPrice);
    if(_to.balance<minBalanceForAccounts)    // 可选，让接受者也补
充余额，以便接受者使用代币。
        _to.send(sell((minBalanceForAccounts - _to.balance) /
sellPrice));
}
```

6.1.3. Netkiller Basic Token 的例子

下面是我一个合约例子仅供参考，为了部署方便我将所有内容都写入在一个文件中。

合约下载地址：
https://github.com/ibook/TokenERC20/blob/master/contracts/TokenERC20.sol

有些场景我们需要记录一些数据在区块链上，这个合约增加了 data 数据支持

```
  function transfer(address _to, uint256 _value, bytes _data)
```

```
public returns (bool) {
    require(_to != address(this));
    transfer(_to, _value);
    require(_to.call(_data));
    return true;
  }
```

完整的例子如下

```
pragma solidity ^0.4.20;

// Author: netkiller <netkiller@msn.com>
// Homepage: http://www.netkiller.cn

interface tokenRecipient { function receiveApproval(address
_from, uint256 _value, address _token, bytes _extraData)
public; }

contract TokenERC20 {
    address public owner;
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
allowance;

    // This generates a public event on the blockchain that
will notify clients
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);

    mapping (address => bool) public frozenAccount;
    event FrozenFunds(address target, bool frozen);
```

```
    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function TokenERC20(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        owner = msg.sender;

        totalSupply = initialSupply * 10 ** uint256(decimals);
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
        symbol = tokenSymbol;                               //
Set the symbol for display purposes
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    /**
     * Internal transfer, only can be called by this contract
     */
    function _transfer(address _from, address _to, uint _value)
internal {
        // Prevent transfer to 0x0 address. Use burn() instead
        require(_to != 0x0);
        // Check if the sender has enough
        require(balanceOf[_from] >= _value);
        // Check for overflows
        require(balanceOf[_to] + _value > balanceOf[_to]);
        // Save this for an assertion in the future
        uint previousBalances = balanceOf[_from] +
balanceOf[_to];
        // Subtract from the sender
        balanceOf[_from] -= _value;
        // Add the same to the recipient
        balanceOf[_to] += _value;
        Transfer(_from, _to, _value);
```

```
        // Asserts are used to use static analysis to find bugs
in your code. They should never fail
        assert(balanceOf[_from] + balanceOf[_to] ==
previousBalances);
    }

    function transfer(address _to, uint256 _value) public {
        require(!frozenAccount[msg.sender]);
        _transfer(msg.sender, _to, _value);
    }

    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(!frozenAccount[msg.sender]);
        require(_value <= allowance[_from][msg.sender]);     //
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        return true;
    }

    function approveAndCall(address _spender, uint256 _value,
bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
        if (approve(_spender, _value)) {
            spender.receiveApproval(msg.sender, _value, this,
_extraData);
            return true;
        }
    }

    function burn(uint256 _value) onlyOwner public returns
(bool success) {
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        Burn(msg.sender, _value);
```

```
        return true;
    }


    function burnFrom(address _from, uint256 _value) onlyOwner
public returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
Update totalSupply
        Burn(_from, _value);
        return true;
    }


  function transfer(address _to, uint256 _value, bytes _data)
public returns (bool) {
    require(_to != address(this));
    transfer(_to, _value);
    require(_to.call(_data));
    return true;
  }

  function transferFrom(address _from, address _to, uint256
_value, bytes _data) public returns (bool) {
    require(_to != address(this));

    transferFrom(_from, _to, _value);

    require(_to.call(_data));
    return true;
  }

  function approve(address _spender, uint256 _value, bytes
_data) public returns (bool) {
    require(_spender != address(this));

    approve(_spender, _value);

    require(_spender.call(_data));

    return true;
```

```
  }


    function transferOwnership(address _owner) onlyOwner public
{
        owner = _owner;
    }
    function mintToken(address target, uint256 mintedAmount)
public onlyOwner {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
        Transfer(0, owner, mintedAmount);
        Transfer(owner, target, mintedAmount);
    }

    function freezeAccount(address target, bool freeze) public
onlyOwner {
        frozenAccount[target] = freeze;
        FrozenFunds(target, freeze);
    }
}
```

6.1.4. Netkiller ADVANCED TOKEN

```
pragma solidity ^0.4.20;

/******************************************/
/*       Netkiller ADVANCED TOKEN         */
/******************************************/
/* Author netkiller <netkiller@msn.com>   */
/* Home http://www.netkiller.cn           */
/* Version 2018-03-05                     */
/******************************************/

interface tokenRecipient { function receiveApproval(address
_from, uint256 _value, address _token, bytes _extraData)
public; }

contract NetkillerAdvancedToken {
    address public owner;
    // Public variables of the token
    string public name;
```

```
    string public symbol;
    uint8 public decimals = 18;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    uint256 public sellPrice;
    uint256 public buyPrice;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
allowance;

    // This generates a public event on the blockchain that
will notify clients
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);
    event Approval(address indexed owner, address indexed
spender, uint256 value);

    mapping (address => bool) public frozenAccount;

    /* This generates a public event on the blockchain that
will notify clients */
    event FrozenFunds(address target, bool frozen);

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function NetkillerAdvancedToken(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        owner = msg.sender;
        totalSupply = initialSupply * 10 ** uint256(decimals);
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
```

```
        symbol = tokenSymbol;                                   //
Set the symbol for display purposes
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }
    function transferOwnership(address newOwner) onlyOwner
public {
        owner = newOwner;
    }

    /* Internal transfer, only can be called by this contract
*/
    function _transfer(address _from, address _to, uint _value)
internal {
        require (_to != 0x0);                                   //
Prevent transfer to 0x0 address. Use burn() instead
        require (balanceOf[_from] >= _value);                   //
Check if the sender has enough
        require (balanceOf[_to] + _value > balanceOf[_to]); //
Check for overflows
        require(!frozenAccount[_from]);                         //
Check if sender is frozen
        require(!frozenAccount[_to]);                           //
Check if recipient is frozen
        balanceOf[_from] -= _value;                             //
Subtract from the sender
        balanceOf[_to] += _value;                               //
Add the same to the recipient
        Transfer(_from, _to, _value);
    }

    /**
     * Transfer tokens
     *
     * Send `_value` tokens to `_to` from your account
     *
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transfer(address _to, uint256 _value) public {
        _transfer(msg.sender, _to, _value);
    }

    /**
     * Transfer tokens from other address
```

```
     *
     * Send `_value` tokens to `_to` in behalf of `_from`
     *
     * @param _from The address of the sender
     * @param _to The address of the recipient
     * @param _value the amount to send
     */
    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(_value <= allowance[_from][msg.sender]);     //
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    /**
     * Set allowance for other address
     *
     * Allows `_spender` to spend no more than `_value` tokens
in your behalf
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     */
    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        Approval(msg.sender, _spender, _value);
        return true;
    }

    /**
     * Set allowance for other address and notify
     *
     * Allows `_spender` to spend no more than `_value` tokens
in your behalf, and then ping the contract about it
     *
     * @param _spender The address authorized to spend
     * @param _value the max amount they can spend
     * @param _extraData some extra information to send to the
approved contract
     */
    function approveAndCall(address _spender, uint256 _value,
bytes _extraData)
        public
        returns (bool success) {
        tokenRecipient spender = tokenRecipient(_spender);
```

```
        if (approve(_spender, _value)) {
            spender.receiveApproval(msg.sender, _value, this,
_extraData);
            return true;
        }
    }

    /**
     * Destroy tokens
     *
     * Remove `_value` tokens from the system irreversibly
     *
     * @param _value the amount of money to burn
     */
    function burn(uint256 _value) onlyOwner public returns
(bool success) {
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        Burn(msg.sender, _value);
        return true;
    }

    /**
     * Destroy tokens from other account
     *
     * Remove `_value` tokens from the system irreversibly on
behalf of `_from`.
     *
     * @param _from the address of the sender
     * @param _value the amount of money to burn
     */
    function burnFrom(address _from, uint256 _value) onlyOwner
public returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
Update totalSupply
        Burn(_from, _value);
```

```
        return true;
    }

    /// @notice Create `mintedAmount` tokens and send it to
`target`
    /// @param target Address to receive the tokens
    /// @param mintedAmount the amount of tokens it will
receive
    function mintToken(address target, uint256 mintedAmount)
onlyOwner public {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
        Transfer(0, this, mintedAmount);
        Transfer(this, target, mintedAmount);
    }

    /// @notice `freeze? Prevent | Allow` `target` from sending
& receiving tokens
    /// @param target Address to be frozen
    /// @param freeze either to freeze it or not
    function freezeAccount(address target, bool freeze)
onlyOwner public {
        frozenAccount[target] = freeze;
        FrozenFunds(target, freeze);
    }

    /// @notice Allow users to buy tokens for `newBuyPrice` eth
and sell tokens for `newSellPrice` eth
    /// @param newSellPrice Price the users can sell to the
contract
    /// @param newBuyPrice Price users can buy from the
contract
    function setPrices(uint256 newSellPrice, uint256
newBuyPrice) onlyOwner public {
        sellPrice = newSellPrice;
        buyPrice = newBuyPrice;
    }

    /// @notice Buy tokens from contract by sending ether
    function buy() payable public {
        uint amount = msg.value / buyPrice;               //
calculates the amount
        _transfer(this, msg.sender, amount);              //
makes the transfers
    }

    /// @notice Sell `amount` tokens to contract
    /// @param amount amount of tokens to be sold
```

```
    function sell(uint256 amount) public {
        require(this.balance >= amount * sellPrice);      //
checks if the contract has enough ether to buy
        _transfer(msg.sender, this, amount);              //
makes the transfers
        msg.sender.transfer(amount * sellPrice);          //
sends ether to the seller. It's important to do this last to
avoid recursion attacks
    }

  function transfer(address _to, uint256 _value, bytes _data)
public returns (bool) {
    require(_to != address(this));
    transfer(_to, _value);
    require(_to.call(_data));
    return true;
  }

  function transferFrom(address _from, address _to, uint256
_value, bytes _data) public returns (bool) {
    require(_to != address(this));

    transferFrom(_from, _to, _value);

    require(_to.call(_data));
    return true;
  }

  function approve(address _spender, uint256 _value, bytes
_data) public returns (bool) {
    require(_spender != address(this));

    approve(_spender, _value);

    require(_spender.call(_data));

    return true;
  }

}
```

6.1.5. 空投代币

```
uint totalSupply = 100000000 ether;     // 总发行量
uint currentTotalAirdrop = 0;                    // 已经空投数量
uint airdrop = 1 ether;                 // 单个账户空投数量

// 存储是否空投过
mapping(address => bool) touched;

// 修改后的balanceOf方法
function balanceOf(address _owner) public view returns (uint256
balance) {
    if (!touched[_owner] && currentTotalAirdrop < totalSupply)
{
        touched[_owner] = true;
        currentTotalAirdrop += airdrop;
        balances[_owner] += airdrop;
    }
    return balances[_owner];
}
```

6.1.5.1. 案例一

```
pragma solidity ^0.4.8;

contract ERC20Interface {
    function totalSupply() public constant returns (uint256
supply);
    function balance() public constant returns (uint256);
    function balanceOf(address _owner) public constant returns
(uint256);
    function transfer(address _to, uint256 _value) public
returns (bool success);
    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success);
    function approve(address _spender, uint256 _value) public
returns (bool success);
    function allowance(address _owner, address _spender) public
constant returns (uint256 remaining);

    event Transfer(address indexed _from, address indexed _to,
```

```
uint256 _value);
    event Approval(address indexed _owner, address indexed
_spender, uint256 _value);
}

// penispenispenispenis
// YOU get a penis, and YOU get a penis, and YOU get a penis!
contract Penis is ERC20Interface {
    string public constant symbol = "PNS";
    string public constant name = "Penis";
    uint8 public constant decimals = 2;

    uint256 _totalSupply = 0;
    uint256 _airdropAmount = 8008135;
    uint256 _cutoff = _airdropAmount * 80085;

    mapping(address => uint256) balances;
    mapping(address => bool) initialized;

    // Penis accepts request to tip-touch another Penis
    mapping(address => mapping (address => uint256)) allowed;

    function Penis() {
        initialized[msg.sender] = true;
        balances[msg.sender] = _airdropAmount * 8008;
        _totalSupply = balances[msg.sender];
    }

    function totalSupply() constant returns (uint256 supply) {
        return _totalSupply;
    }

    // What's my girth?
    function balance() constant returns (uint256) {
        return getBalance(msg.sender);
    }

    // What is the length of a particular Penis?
    function balanceOf(address _address) constant returns
(uint256) {
        return getBalance(_address);
    }

    // Tenderly remove hand from Penis and place on another
Penis
    function transfer(address _to, uint256 _amount) returns
(bool success) {
        initialize(msg.sender);
```

```
        if (balances[msg.sender] >= _amount
            && _amount > 0) {
            initialize(_to);
            if (balances[_to] + _amount > balances[_to]) {

                balances[msg.sender] -= _amount;
                balances[_to] += _amount;

                Transfer(msg.sender, _to, _amount);

                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }

    // Perform the inevitable actions which cause release of
that which each Penis
    // is built to deliver. In EtherPenisLand there are only
Penises, so this
    // allows the transmission of one Penis's payload (or
partial payload but that
    // is not as much fun) INTO another Penis. This causes the
Penisae to change
    // form such that all may see the glory they each
represent. Erections.
    function transferFrom(address _from, address _to, uint256
_amount) returns (bool success) {
        initialize(_from);

        if (balances[_from] >= _amount
            && allowed[_from][msg.sender] >= _amount
            && _amount > 0) {
            initialize(_to);
            if (balances[_to] + _amount > balances[_to]) {

                balances[_from] -= _amount;
                allowed[_from][msg.sender] -= _amount;
                balances[_to] += _amount;

                Transfer(_from, _to, _amount);

                return true;
            } else {
```

```
                return false;
            }
        } else {
            return false;
        }
    }

    // Allow splooger to cause a payload release from your
Penis, multiple times, up to
    // the point at which no further release is possible..
    function approve(address _spender, uint256 _amount) returns
(bool success) {
        allowed[msg.sender][_spender] = _amount;
        Approval(msg.sender, _spender, _amount);
        return true;
    }

    function allowance(address _owner, address _spender)
constant returns (uint256 remaining) {
        return allowed[_owner][_spender];
    }

    // internal privats
    function initialize(address _address) internal returns
(bool success) {
        if (_totalSupply < _cutoff && !initialized[_address]) {
            initialized[_address] = true;
            balances[_address] = _airdropAmount;
            _totalSupply += _airdropAmount;
        }
        return true;
    }

    function getBalance(address _address) internal returns
(uint256) {
        if (_totalSupply < _cutoff && !initialized[_address]) {
            return balances[_address] + _airdropAmount;
        }
        else {
            return balances[_address];
        }
    }
}
```

6.1.5.2. 案例二

```
pragma solidity ^0.4.8;

contract ERC20Interface {
    function totalSupply() public constant returns (uint256
supply);
    function balance() public constant returns (uint256);
    function balanceOf(address _owner) public constant returns
(uint256);
    function transfer(address _to, uint256 _value) public
returns (bool success);
    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success);
    function approve(address _spender, uint256 _value) public
returns (bool success);
    function allowance(address _owner, address _spender) public
constant returns (uint256 remaining);

    event Transfer(address indexed _from, address indexed _to,
uint256 _value);
    event Approval(address indexed _owner, address indexed
_spender, uint256 _value);
}

contract Simoleon is ERC20Interface {
    string public constant symbol = "SIM";
    string public constant name = "Simoleon";
    uint8 public constant decimals = 2;

    uint256 _totalSupply = 0;
    uint256 _airdropAmount = 1000000;
    uint256 _cutoff = _airdropAmount * 10000;

    mapping(address => uint256) balances;
    mapping(address => bool) initialized;

    // Owner of account approves the transfer of an amount to
another account
    mapping(address => mapping (address => uint256)) allowed;

    function Simoleon() {
        initialized[msg.sender] = true;
        balances[msg.sender] = _airdropAmount * 1000;
        _totalSupply = balances[msg.sender];
    }

    function totalSupply() constant returns (uint256 supply) {
```

```
        return _totalSupply;
    }

    // What's my balance?
    function balance() constant returns (uint256) {
        return getBalance(msg.sender);
    }

    // What is the balance of a particular account?
    function balanceOf(address _address) constant returns
(uint256) {
        return getBalance(_address);
    }

    // Transfer the balance from owner's account to another
account
    function transfer(address _to, uint256 _amount) returns
(bool success) {
        initialize(msg.sender);

        if (balances[msg.sender] >= _amount
            && _amount > 0) {
            initialize(_to);
            if (balances[_to] + _amount > balances[_to]) {

                balances[msg.sender] -= _amount;
                balances[_to] += _amount;

                Transfer(msg.sender, _to, _amount);

                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }

    // Send _value amount of tokens from address _from to
address _to
    // The transferFrom method is used for a withdraw workflow,
allowing contracts to send
    // tokens on your behalf, for example to "deposit" to a
contract address and/or to charge
    // fees in sub-currencies; the command should fail unless
the _from account has
    // deliberately authorized the sender of the message via
```

```
some mechanism; we propose
    // these standardized APIs for approval:
    function transferFrom(address _from, address _to, uint256
_amount) returns (bool success) {
        initialize(_from);

        if (balances[_from] >= _amount
            && allowed[_from][msg.sender] >= _amount
            && _amount > 0) {
            initialize(_to);
            if (balances[_to] + _amount > balances[_to]) {

                balances[_from] -= _amount;
                allowed[_from][msg.sender] -= _amount;
                balances[_to] += _amount;

                Transfer(_from, _to, _amount);

                return true;
            } else {
                return false;
            }
        } else {
            return false;
        }
    }

    // Allow _spender to withdraw from your account, multiple
times, up to the _value amount.
    // If this function is called again it overwrites the
current allowance with _value.
    function approve(address _spender, uint256 _amount) returns
(bool success) {
        allowed[msg.sender][_spender] = _amount;
        Approval(msg.sender, _spender, _amount);
        return true;
    }

    function allowance(address _owner, address _spender)
constant returns (uint256 remaining) {
        return allowed[_owner][_spender];
    }

    // internal private functions
    function initialize(address _address) internal returns
(bool success) {
        if (_totalSupply < _cutoff && !initialized[_address]) {
            initialized[_address] = true;
```

```
            balances[_address] = _airdropAmount;
            _totalSupply += _airdropAmount;
        }
        return true;
    }

    function getBalance(address _address) internal returns
(uint256) {
        if (_totalSupply < _cutoff && !initialized[_address]) {
            return balances[_address] + _airdropAmount;
        }
        else {
            return balances[_address];
        }
    }
}
```

6.1.5.3. 案例三

```
pragma solidity ^0.4.18;


library SafeMath {
  function mul(uint256 a, uint256 b) internal pure returns
(uint256) {
    if (a == 0) {
      return 0;
    }
    uint256 c = a * b;
    require(c / a == b);
    return c;
  }

  function div(uint256 a, uint256 b) internal pure returns
(uint256) {
    uint256 c = a / b;
    return c;
  }

  function sub(uint256 a, uint256 b) internal pure returns
(uint256) {
    require(b <= a);
    return a - b;
```

```
  }

  function add(uint256 a, uint256 b) internal pure returns
(uint256) {
    uint256 c = a + b;
    require(c >= a);
    return c;
  }
}


contract Ownable {
  address public owner;

  event OwnershipTransferred(address indexed previousOwner,
address indexed newOwner);

  function Ownable() public {
    owner = msg.sender;
  }

  modifier onlyOwner() {
    require(msg.sender == owner);
    _;
  }

  function transferOwnership(address newOwner) public onlyOwner
{
    require(newOwner != address(0));
    OwnershipTransferred(owner, newOwner);
    owner = newOwner;
  }

}


contract SurpriseToken is Ownable{

    using SafeMath for uint256;

    string public constant name       = "SURPRISE";
    string public constant symbol     = "SPS";
    uint32 public constant decimals   = 18;
    uint256 public totalSupply        = 208932000 ether;
    uint256 public currentTotalSupply = 0;
    uint256 startBalance              = 276 ether;

    mapping(address => bool) touched;
```

```
    mapping(address => uint256) balances;
    mapping (address => mapping (address => uint256)) internal
allowed;

    event Transfer(address indexed from, address indexed to,
uint256 value);
    event Approval(address indexed owner, address indexed
spender, uint256 value);


    function transfer(address _to, uint256 _value) public
returns (bool) {
        require(_to != address(0));

        if( !touched[msg.sender] && currentTotalSupply <
totalSupply ){
            balances[msg.sender] = balances[msg.sender].add(
startBalance );
            touched[msg.sender] = true;
            currentTotalSupply = currentTotalSupply.add(
startBalance );
        }

        require(_value <= balances[msg.sender]);

        balances[msg.sender] =
balances[msg.sender].sub(_value);
        balances[_to] = balances[_to].add(_value);

        Transfer(msg.sender, _to, _value);
        return true;
    }


    function transferFrom(address _from, address _to, uint256
_value) public returns (bool) {
        require(_to != address(0));

        require(_value <= allowed[_from][msg.sender]);

        if( !touched[_from] && currentTotalSupply < totalSupply
){
            touched[_from] = true;
            balances[_from] = balances[_from].add( startBalance
);
            currentTotalSupply = currentTotalSupply.add(
startBalance );
        }
```

```
        require(_value <= balances[_from]);

        balances[_from] = balances[_from].sub(_value);
        balances[_to] = balances[_to].add(_value);
        allowed[_from][msg.sender] = allowed[_from]
[msg.sender].sub(_value);
        Transfer(_from, _to, _value);
        return true;
    }


    function approve(address _spender, uint256 _value) public
returns (bool) {
        allowed[msg.sender][_spender] = _value;
        Approval(msg.sender, _spender, _value);
        return true;
    }


    function allowance(address _owner, address _spender) public
view returns (uint256) {
        return allowed[_owner][_spender];
     }


    function increaseApproval(address _spender, uint
_addedValue) public returns (bool) {
        allowed[msg.sender][_spender] = allowed[msg.sender]
[_spender].add(_addedValue);
        Approval(msg.sender, _spender, allowed[msg.sender]
[_spender]);
        return true;
    }


    function decreaseApproval(address _spender, uint
_subtractedValue) public returns (bool) {
        uint oldValue = allowed[msg.sender][_spender];
        if (_subtractedValue > oldValue) {
          allowed[msg.sender][_spender] = 0;
        } else {
          allowed[msg.sender][_spender] =
oldValue.sub(_subtractedValue);
        }
        Approval(msg.sender, _spender, allowed[msg.sender]
[_spender]);
        return true;
```

```
     }


    function getBalance(address _a) internal constant
returns(uint256)
    {
        if( currentTotalSupply < totalSupply ){
            if( touched[_a] )
                return balances[_a];
            else
                return balances[_a].add( startBalance );
        } else {
            return balances[_a];
        }
    }


    function balanceOf(address _owner) public view returns
(uint256 balance) {
        return getBalance( _owner );
    }
}
```

6.2. ERC223 token standard reference implementation.

https://github.com/Dexaran/ERC223-token-standard

ERC223是以太坊上最新的代币(token)接口标准,主要是为了解决ERC220代币转账丢失问题

ERC220接口以transfer为例:

```
// @notice send `_value` token to `_to` from `msg.sender`
// @param _to The address of the recipient
// @param _value The amount of token to be transferred
// @return Whether the transfer was successful or not
function transfer(address _to, uint256 _value) public returns
(bool success);
```

transfer的功能非常简单，给一个指定地址转多少代币。

但是当初设计的时候没有考虑到的一个问题就是如果接收者是一个智能合约,那么合约是没法感知自己收到了多少token的.

ERC223 中的方法定义

```
function transfer(address _to, uint _value) public returns
(bool ok);
function transfer(address _to, uint _value, bytes _data) public
returns (bool ok);
function transfer(address _to, uint _value, bytes _data, string
_custom_fallback)
```

除此之外 ERC223 还提供了安全的数学运算方法。

```
pragma solidity ^0.4.11;


/**
 * Math operations with safety checks
 */
library SafeMath {
  function mul(uint a, uint b) internal returns (uint) {
    uint c = a * b;
    assert(a == 0 || c / a == b);
    return c;
  }

  function div(uint a, uint b) internal returns (uint) {
    // assert(b > 0); // Solidity automatically throws when
dividing by 0
    uint c = a / b;
    // assert(a == b * c + a % b); // There is no case in which
this doesn't hold
```

```
    return c;
  }

  function sub(uint a, uint b) internal returns (uint) {
    assert(b <= a);
    return a - b;
  }

  function add(uint a, uint b) internal returns (uint) {
    uint c = a + b;
    assert(c >= a);
    return c;
  }

  function max64(uint64 a, uint64 b) internal constant returns
(uint64) {
    return a >= b ? a : b;
  }

  function min64(uint64 a, uint64 b) internal constant returns
(uint64) {
    return a < b ? a : b;
  }

  function max256(uint256 a, uint256 b) internal constant
returns (uint256) {
    return a >= b ? a : b;
  }

  function min256(uint256 a, uint256 b) internal constant
returns (uint256) {
    return a < b ? a : b;
  }

  function assert(bool assertion) internal {
    if (!assertion) {
      throw;
    }
  }
}
```

ERC223 接口

```
pragma solidity ^0.4.11;

contract ERC223Interface {
    uint public totalSupply;
    function balanceOf(address who) constant returns (uint);
    function transfer(address to, uint value);
    function transfer(address to, uint value, bytes data);
    event Transfer(address indexed from, address indexed to,
uint value, bytes data);
}
```

```
pragma solidity ^0.4.11;

 /**
 * @title Contract that will work with ERC223 tokens.
 */

contract ERC223ReceivingContract {
/**
 * @dev Standard ERC223 function that will handle incoming
token transfers.
 *
 * @param _from  Token sender address.
 * @param _value Amount of tokens.
 * @param _data  Transaction metadata.
 */
    function tokenFallback(address _from, uint _value, bytes
_data);
}
```

```
pragma solidity ^0.4.11;

import './ERC223_interface.sol';
import './ERC223_receiving_contract.sol';
import '././SafeMath.sol';

/**
 * @title Reference implementation of the ERC223 standard
token.
 */
```

```
contract ERC223Token is ERC223Interface {
    using SafeMath for uint;

    mapping(address => uint) balances; // List of user
balances.

    /**
     * @dev Transfer the specified amount of tokens to the
specified address.
     *      Invokes the `tokenFallback` function if the
recipient is a contract.
     *      The token transfer fails if the recipient is a
contract
     *      but does not implement the `tokenFallback` function
     *      or the fallback function to receive funds.
     *
     * @param _to    Receiver address.
     * @param _value Amount of tokens that will be transferred.
     * @param _data  Transaction metadata.
     */
    function transfer(address _to, uint _value, bytes _data) {
        // Standard function transfer similar to ERC20 transfer
with no _data .
        // Added due to backwards compatibility reasons .
        uint codeLength;

        assembly {
            // Retrieve the size of the code on target address,
this needs assembly .
            codeLength := extcodesize(_to)
        }

        balances[msg.sender] =
balances[msg.sender].sub(_value);
        balances[_to] = balances[_to].add(_value);
        if(codeLength>0) {
            ERC223ReceivingContract receiver =
ERC223ReceivingContract(_to);
            receiver.tokenFallback(msg.sender, _value, _data);
        }
        Transfer(msg.sender, _to, _value, _data);
    }

    /**
     * @dev Transfer the specified amount of tokens to the
specified address.
     *      This function works the same with the previous one
     *      but doesn't contain `_data` param.
```

```
     *      Added due to backwards compatibility reasons.
     *
     * @param _to    Receiver address.
     * @param _value Amount of tokens that will be transferred.
     */
    function transfer(address _to, uint _value) {
        uint codeLength;
        bytes memory empty;

        assembly {
            // Retrieve the size of the code on target address,
this needs assembly .
            codeLength := extcodesize(_to)
        }

        balances[msg.sender] =
balances[msg.sender].sub(_value);
        balances[_to] = balances[_to].add(_value);
        if(codeLength>0) {
            ERC223ReceivingContract receiver =
ERC223ReceivingContract(_to);
            receiver.tokenFallback(msg.sender, _value, empty);
        }
        Transfer(msg.sender, _to, _value, empty);
    }


    /**
     * @dev Returns balance of the `_owner`.
     *
     * @param _owner   The address whose balance will be
returned.
     * @return balance Balance of the `_owner`.
     */
    function balanceOf(address _owner) constant returns (uint
balance) {
        return balances[_owner];
    }
}
```

6.3. ERC721 - Non-fungible Token Standard

https://eips.ethereum.org/EIPS/eip-721

https://github.com/OpenZeppelin/zeppelin-solidity/tree/master/contracts/token/ERC721

ERC721 是以太猫项目采用的合约标准。

6.4. ERC827 Token Standard (ERC20 Extension)

https://github.com/ethereum/EIPs/issues/827

https://github.com/OpenZeppelin/zeppelin-solidity/tree/master/contracts/token/ERC827

6.5. ERC875 for non fungible tokens and simple atomic swaps

https://github.com/ethereum/EIPs/issues/875

https://github.com/alpha-wallet/ERC875-Example

6.6. ERC: Standard URI scheme with metadata, value and byte code

https://gist.github.com/netkiller/427c196eb256ed43c7be08ac875d34a7

第 14 章 智能合约语言 Solidity v0.5.0

本文作者最近在找工作，有意向致电 13113668890

Solidity 是什么？Solidity是以太坊智能合约的编程语言。

1. Remix - browser-solidity

在线使用 browser-solidity

https://ethereum.github.io/browser-solidity/ https://remix.ethereum.org/

国内网络有时不给力，建议将 Remix 安装到本地目录。

1.1. 将 Remix(browser-solidity) 安装到本地

共享合约目录

```
npm install -g remixd
remixd -S "/home/ethereum/codebase/blocks/contracts"
```

安装 browser-solidity

```
git clone https://github.com/ethereum/browser-solidity
cd browser-solidity
npm install
npm run prepublish

sudo chown -R $USER:$(id -gn $USER) /home/neo/.config

npm start
```

启动后浏览器中输入 http://localhost:8080 可以看到 Remix 界面

Web3 Provider

Remix 提供三种运行环境,常用的有 JavaScript VM 和 Web3 Provider (连接到 --rpc --rpcaddr="0.0.0.0" --rpccorsdomain "*" --rpcport 8545)

Web3 Provider 方式需要解锁账号和启动挖矿

```
> personal.unlockAccount(eth.accounts[0],"");
> miner.start(2); admin.sleepBlocks(1); miner.stop();
```

1.2. 输入数组

```
        function mint(address[] _to, uint256 _value) public returns
(bool success) {
        for (uint i=0; i<_to.length; i++) {
            balanceOf[_to[i]] = _value;
        }
        return true;
    }
```

在Remix中输入数组的方法

```
["0x6F56648fbD2306f843442f8dC61d5C8861Fac7C9","0x81b7E08F65Bdf5648606c89
998A9CC8164397647"]
```

2. solc 命令

2.1. 使用 solc 编译 *.sol 代码

```
neo@netkiller ~/ethereum/solidity % solc --bin --abi --optimize
-o ./output helloworld.sol
neo@netkiller ~/ethereum/solidity % find output
output
output/HelloWorld.bin
output/HelloWorld.abi
```

3. 智能合约入门演示

这里我们先做一个 Helloword 演示，让你初步对智能合约有一个大概的认识。

提示

需要注意的是，你在网上会看到很多例子，对照这例子一步一步操作，始终无法成功，这根Solidity的版本有很大关系。

将下面代码粘贴到

```
pragma solidity ^0.4.25;

contract HelloWorld
{
    string tmp;

    function HelloWorld() public
    {

    }

    function get() public constant returns (string)
    {
        return tmp;
    }

    function set(string _tmp) public
    {
        tmp = _tmp;
    }

}
```

Compile - Details - WEB3DEPLOY

```
var helloworldContract =
web3.eth.contract([{"constant":false,"inputs":
[{"name":"_tmp","type":"string"}],"name":"set","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"funct
ion"},{"constant":true,"inputs":[],"name":"get","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability"
:"view","type":"function"},{"inputs":
[],"payable":false,"stateMutability":"nonpayable","type":"const
ructor"}]);
var helloworld = helloworldContract.new(
   {
     from: web3.eth.accounts[0],
     data:
'0x6060604052341561000f57600080fd5b6102e38061001e6000396000f300
6060604052600436106100c576000357c0100000000000000000000000000000
0000000000000000000000000000000900463ffffffff1680634ed3885e146100
515780636d4ce63c146100ae575b600080fd5b341561005c57600080fd5b610
0ac60048080359060200190820180359060200190808060 1f01602080910402
602001604051908101604052809392919081815260200183838082843782019
150505050505091905061013c565b005b34156100b957600080fd5b6100c1
610156565b604051808060200182810382528381815181526020019150805 19
060200190808383600005b838110156101015780820151818401526020810190
506100e6565b50505050905090810190601f16801561012e578082038051600
1836020036101000a031916815260200191505b509250505060405180910390
f35b806000908051906020019061015291906101fe565b5050565b61015e6
1027e565b60008054600181600116156101000203166002900480601f016020
809104026020016040519081016040528092919081815260200182805460018
160011615610100020316600290048015610 1f45780601f106101c957610100
808354040283529160200191610 1f4565b820191906000526020600020905b8
154815290600101906020018083116101d757829003601f168201915b505050
5050905090565b828054600181600116156101000203166002900490600052 6
02060002090601f016020900481019282601f1061023f57805160ff19168380
0117855561026d565b828001600101855582156102 6d579182015b828111156
1026c57825182559160200191906001019061025 1565b5b50905061027a9190
610292565b5090565b602060405190810160405280600081525090565b6102b
491905b808211156102b057600081600090555060010161029 8565b5090565b
905600a165627a7a72305820ea826c30d131f20a4d3a8e3fb059ffa95f4c222
a5b099029750e4c1937b46e5b0029',
     gas: '4700000'
   }, function (e, contract){
    console.log(e, contract);
    if (typeof contract.address !== 'undefined') {
         console.log('Contract mined! address: ' +
contract.address + ' transactionHash: ' +
contract.transactionHash);
    }
```

```
 })
```

部署智能合约需要消耗 gas 所以你要先解锁账号。

```
>
personal.unlockAccount("0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb
5a6", "", 300)
true
```

```
> var helloworldContract =
web3.eth.contract([{"constant":false,"inputs":
[{"name":"_tmp","type":"string"}],"name":"set","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"funct
ion"},{"constant":true,"inputs":[],"name":"get","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability"
:"view","type":"function"},{"inputs":
[],"payable":false,"stateMutability":"nonpayable","type":"const
ructor"}]);
undefined
> var helloworld = helloworldContract.new(
...    {
......      from: web3.eth.accounts[0],
......      data:
'0x6060604052341561000f57600080fd5b6102e38061001e6000396000f300
6060604052600436106100c576000357c0100000000000000000000000000000
0000000000000000000000000000000900463ffffffff1680634ed3885e146100
515780636d4ce63c146100ae575b600080fd5b341561005c57600080fd5b610
0ac60048080359060200190820180359060200190808060 1f01602080910402
602001604051908101604052809392919081815260200183838082843782019
150505050505091905050610 13c565b005b34156100b957600080fd5b6100c1
610156565b6040518080602001828103825283818151815260200191508051 9
060200190808383600 05b83811015610101578082015181840152602081019 0
506100e6565b505050509050908101906 01f16801561012e578082038051600
1836020036101000a031916815260200191505b509250505060405180910390
f35b806000908051906020019061015292 91906101fe565b5050565b61015e6
1027e565b60008054600181600116156101000203166002900480601f016020
8091040260200160405190810160405280929190818152602001828054600 18
160011615610100020316600290048015 6101f45780601f106101c957610100
808354040283529160200191 6101f4565b820191906000526020600020905b8
```

```
154815290600101906020018083116101d757829003601f168201915b505050
5050905090565b828054600181600116156101000203166002900490600052 6
02060002090601f016020900481019282601f1061023f57805160ff19168380
0117855561026d565b828001600101855821561026d579182015b828111156
1026c57825182559160200191906001019061025156 5b5b50905061027a9190
610292565b5090565b602060405190810160405280600081525090565b6102b
491905b808211156102b057600081600090555060010161029856 5b5090565b
905600a165627a7a72305820ea826c30d131f20a4d3a8e3fb059ffa95f4c222
a5b099029750e4c1937b46e5b0029',
......      gas: '4700000'
......    }, function (e, contract){
......     console.log(e, contract);
......     if (typeof contract.address !== 'undefined') {
.........          console.log('Contract mined! address: ' +
contract.address + ' transactionHash: ' +
contract.transactionHash);
.........     }
......  })
null [object Object]
undefined
```

helloworld 智能合约已经创建完毕

```
> helloworld
{
  abi: [{
      constant: false,
      inputs: [{...}],
      name: "set",
      outputs: [],
      payable: false,
      stateMutability: "nonpayable",
      type: "function"
  }, {
      constant: true,
      inputs: [],
      name: "get",
      outputs: [{...}],
      payable: false,
      stateMutability: "view",
      type: "function"
  }, {
      inputs: [],
```

```
      payable: false,
      stateMutability: "nonpayable",
      type: "constructor"
  }],
  address: undefined,
  transactionHash:
"0x466c9ad9db8f37ed5b65bc261210da92f51364ebab1dcbd3759bfc3e16ad
6502"
}
```

4. 数据类型

4.1. 数值型

int/uint：变长的有符号或无符号整型。变量支持的步长以8递增，支持从uint8到uint256，以及int8到int256。需要注意的是，uint和int默认代表的是uint256和int256。

有符号整型能够表示负数的代价是其能够存储正数的范围的缩小，因为其约一半的数值范围要用来表示负数。如：uint8的存储范围为0 ~ 255，而int8的范围为-127 ~ 127

支持的运算符:

比较：<=，<，==，!=，>=，>，返回值为bool类型。

位运算符：&，|，（^异或），（~非）。

数学运算：+，-，一元运算+，*，/，（%求余），（**次方），（<<左移），（>>右移）。

小数由"."组成，在他的左边或右边至少要包含一个数字。
如"1."，".1"，"1.3"均是有效的小数。

4.1.1. 加 +，减 -，乘 *，除 ／ 运算演示

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
```

```
contract Math {

  function mul(int a, int b) public pure returns (int) {

      int c = a * b;
      return c;
  }

  function div(int a, int b) public pure  returns (int) {

      int c = a / b;
      return c;
  }

  function sub(int a, int b) public pure  returns (int) {

      return a - b;
  }

  function add(int a, int b) public pure  returns (int) {

      int c = a + b;
      return c;
  }
}
```

4.1.2. 求余 % 运算演示

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract Math {

  function m(int a, int b) public pure returns (int) {

      int c = a % b;
      return c;
  }
}
```

4.1.3. 幂运算演示

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract Math {

  function m(uint a, uint b) public pure returns (uint) {

      uint c = a**b;
      return c;
  }

}
```

4.1.4. 与 &，| 或，非 ～，异或 ^ 演示

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract Math {

  function yu() public pure returns (uint) {

      uint a = 3; // 0b0011
      uint b = 4; // 0b0100

      uint c = a & b; // 0b0000
      return c; // 0
  }

  function huo() public pure returns (uint) {

      uint a = 3; // 0b0011
      uint b = 4; // 0b0100

      uint c = a | b; // 0b0111
      return c; // 7
  }
```

```
  function fei() public pure returns (uint8) {

      uint8 a = 3; // 0b00000011
      uint8 c = ~a; // 0b11111100
      return c; // 0
  }

  function yihuo() public pure returns (uint) {

      uint a = 3; // 0b0011
      uint b = 4; // 0b0100

      uint c = a ^ b; // 0b0111
      return c; // 252
  }
}
```

4.1.5. 位移演示

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn

contract Math {

  function leftShift() public pure returns (uint8) {

      uint8 a = 8; // 0b00001000
      uint8 c = a << 2; // 0b00100000
      return c; // 32
  }

  function rightShift() public pure returns (uint8) {

      uint8 a = 8; // 0b00001000
      uint8 c = a >> 2; // 0b00000010
      return c; // 2
  }

}
```

a << n 表示a的二进制位向左移动n位，在保证位数没有溢出的情况下等价于 a乘以2的n次方。
a >> n 表示a的二进制位向右移动n位，在保证位数没有溢出的情况下等价于 a除以2的n次方。

4.2. 字符串

string 字符串类型，字符串可以通过""或者''来定义字符串的值

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract StringTest {

    string name;

    function StringTest() public{
        name = "default";
    }
    function setName(string _name) public{
        name = _name;
    }
    function getName() public view returns(string){
        return name;
    }
}
```

4.2.1. 获取字符串长度

在 Solidity 中想获得字符串长度必须转成 bytes 类型然后使用 length 属性获得。bytes(string).length

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract StringTest {


    string public name = "http://www.netkiller.cn";


    function nameBytes() public constant returns (bytes) {

        return bytes(name);
    }

    function nameLength() public constant returns (uint) {

        return bytes(name).length;
    }

    function length(string _name) public pure returns (uint) {

        return bytes(_name).length;
    }

}
```

提示

注意：汉字采用UTF8编码，一个汉字等于3个字节，当你使用 length("景峯") 测试时会返回长度 6。

4.3. 布尔(Booleans)

bool: 可能的取值为常量值true和false。支持的运算符：

! 逻辑非

&& 逻辑与

|| 逻辑或

== 等于

!= 不等于

```
bool a = true;
bool b = !a;

// a == b -> false
// a != b -> true
// a || b -> true
// a && b -> false
```

4.4. 字节类型

```
bytes names = "netkiller"
bytes9 _names = "netkiller";
bytes(name)[0] = 0xFF;

bytes memory _tmp = new bytes(3);
_tmp[0] = 0x4e;
_tmp[1] = 0x65;
_tmp[2] = 0x6f;
```

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract BytesTest {

    bytes names = "netkiller";
```

```
    function get() public view returns (bytes) {

        return names;
    }
    function getBytes2() public pure returns (bytes2) {
        bytes9 _names = "netkiller";
        return bytes2(_names);
    }
    function bytesToString() public constant returns (string) {

        return string(names);
    }

    function copyBytes(bytes b) public pure returns (bytes) {

       bytes memory tmp = new bytes(b.length);

       for(uint i = 0; i < b.length; i++) {

           tmp[i] = b[i];
       }

       return tmp;
    }

    function bytesToString2() public pure returns (string) {
        bytes memory _tmp = new bytes(3);
        _tmp[0] = 0x4e;
        _tmp[1] = 0x65;
        _tmp[2] = 0x6f;
        return string(_tmp);
    }

}
```

.length可以动态修改字节数组的长度

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract BytesTest2 {
```

```
    // 初始化一个两个字节空间的字节数组
    bytes public array = new bytes(2);

    // 设置修改字节数组的长度
    function setLength(uint _len) public {
        array.length = _len;
    }

    // 返回字节数组的长度
    function getLength() constant public returns (uint) {
        return array.length;
    }

    // 往字节数组中添加字节
    function pushArray(byte _tmp) public{
        array.push(_tmp);
    }

}
```

4.5. 数组

```
        //创建一个memory的数组
        uint[] memory a = new uint[](7);

        uint[] x = [uint(1), 3, 4];

    bytes memory b = new bytes(10);
```

二维数组

```
uint [2][3] T = [[1,2],[3,4],[5,6]];
```

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract ArrayTest {

    uint [] array = [1,2,3,4,5];

    // 通过for循环计算数组内部的值的总和
    function sum() constant public returns (uint) {
        uint num = 0;
        for(uint i = 0; i < array.length; i++) {
            num = num + array[i];
        }
        return num;
    }

    function sumNumbers(uint[] _numbers) public pure returns
(uint) {
        uint num = 0;
        for(uint i = 0; i < _numbers.length; i++) {
            num = num + _numbers[i];
        }
        return num;
    }

}
```

4.5.1. length

.length 属性是活动数组的尺寸

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract ArrayLength {

    uint [] array = [1,2,3,4,5];

    function getLength() public constant returns (uint) {

        return array.length;
```

```
    }

}
```

4.5.2. push() 方法

通过 push 可以向数组中添加数据

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract ArrayLength {

    uint [] array = [1,2,3,4,5];

    function pushArray() public {

        array.push(6);
    }

    function getLength() public constant returns (uint) {

        return array.length;
    }

}
```

4.6. 枚举类型

State 就是一个自定义的整型，当枚举数不够多时，它默认的类型为uint8，当枚举数足够多时，它会自动变成uint16，枚举下标定义从左至右从零开始。

New=0, Pending=1, Done=2, Deleted=3

访问枚举方式 State.New 实际等于数字 0

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract EnumTest {
    enum State { New, Pending, Done, Deleted }
    State state = State.New;

    function set(State _state) public {
        state = _state;
    }

    function get() constant public returns (State) {
        return state;
    }

}
```

枚举用来定义状态

```
pragma solidity ^0.4.0;

contract Purchase {
    enum State { Created, Locked, Inactive } // Enum
}
```

4.7. 结构体

定义结构体

```
        struct Voter {
        uint weight; // weight is accumulated by delegation
        bool voted;  // if true, that person already voted
        address delegate; // person delegated to
        uint vote;   // index of the voted proposal
    }
```

```
    // This is a type for a single proposal.
    struct Proposal {
        bytes32 name;   // short name (up to 32 bytes)
        uint voteCount; // number of accumulated votes
    }
```

演示例子

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract Students {

    struct Person {
        string name;
        uint age;
        uint class;

    }

    Person person = Person("Neo",18,1);

    function getPerson() public view returns(string){
        return person.name;
    }
}
```

4.7.1. 函数返回Struct

Struct 不知直接返回，解决方法如下

```
pragma solidity ^0.4.19;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract Netkiller {
    struct JobStruct {
```

```
      uint a;
      uint b;
      uint c;
    }

    function getValues () public pure returns (uint, uint,
uint) {
      JobStruct memory js = JobStruct(1, 2, 3);
      return (js.a, js.b, js.c);
    }
}
```

4.8. address

```
address public minter;
```

下面是一个获得账号余额的例子。

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract AddressTest{

    function getBalance(address _addr) public constant returns
(uint){
        return _addr.balance;
    }

}
```

4.8.1. payable

4.8.2. .value()

4.8.3. .gas()

4.9. mapping

mapping 就是图数据结构，由 key 和 value 组成。

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract MappingExample {

    mapping(uint => string) map;

    function put(uint key, string value) public {
        map[key] = value;
    }

    function get(uint key) constant public returns (string) {
        return map[key];
    }
}
```

5. 单位

5.1. 货币单位（Ether Units）

kwei (1000 Wei)
mwei (1000 KWei)
gwei (1000 mwei)
szabo (1000 gwei)
finney (1000 szabo)
ether (1000 finney)

以太币单位其实是密码学家的名字，是以太坊创始人为了纪念他们在数字货币的领域的贡献。他们分别是:
wei: Wei Dai 戴伟 密码学家 ，发表 B-money
finney: Hal Finney 芬尼 密码学家、工作量证明机制（POW）提出
szabo: Nick Szabo 尼克萨博 密码学家、智能合约的提出者

```
pragma solidity ^0.4.25;
//Author: netkiller <netkiller@msn.com>
//Home: http://www.netkiller.cn
contract UnitTest {
    function tf() public pure returns (bool) {
      if (1 ether == 1000 finney){
          return true;
      }
      return false;
    }

    function ts() public pure returns (bool) {
      if (1 ether == 1000000 szabo){
          return true;
      }
      return false;
    }
```

```
    function tgw() public pure returns (bool) {
      if (1 ether == 1000000000000000000 wei){
          return true;
      }
      return false;
    }
}
```

5.2. 时间单位（Time Units）

时间单位: seconds, minutes, hours, days, weeks, years均可做为后缀，并进行相互转换，规则如下：

1 == 1 seconds (默认是seconds为单位)
1 minutes == 60 seconds
1 hours == 60 minutes
1 days == 24 hours
1 weeks = 7 days
1 years = 365 days

由于无法预测闰秒，必须由外部的预言（oracle）来更新从而得到一个精确的日历库。
所以使用这些单位进行日期计算需要特别小心，因为不是每年都是365天，且并不是每天都有24小时，因为还有闰秒。

6. 变量

```
address public minter;
string name;
int num;

    uint constant x = 32**22 + 8;
    string constant text = "abc";
    bytes32 constant myHash = keccak256("abc");

    uint256 ticket = 1 ether;
```

变量赋值

```
pragma solidity ^0.4.25;

contract C {
    uint[] data;

    function f() public view returns (uint, bool, uint) {
        return (7, true, 2);
    }

    function g() public {
        // 声明和分配变量。 明确指定类型是不可能的。
        var (x, b, y) = f();
        // 分配给一个预先存在的变量。
        (x, y) = (2, 7);
        // 互换值的常用技巧对于非价值存储类型不起作用。
        (x, y) = (y, x);
        // 组件可以省略（也可以用于变量声明）。
        // 如果元组以空组件结束，其余的值将被丢弃。
        (data.length,) = f(); // 设置长度为 7
        // 在左边也可以做同样的事情。
        (,data[3]) = f(); // Sets data[3] to 2
        // 组件只能在作业的左侧排除，但有一个例外：
```

```
        (x,) = (1,);
        // (1,) 是指定1元组元的唯一方法，因为（1）等于1。
    }
}
```

6.1. 全局变量

```
block.blockhash(uint blockNumber) returns (bytes32): hash of
the given block - only
works for 256 most recent blocks
block.coinbase (address): current block miner’s address
block.difficulty (uint): current block difficulty
block.gaslimit (uint): current block gaslimit
block.number (uint): current block number
block.timestamp (uint): current block timestamp
msg.data (bytes): complete calldata
msg.gas (uint): remaining gas
msg.sender (address): sender of the message (current call)
msg.value (uint): number of wei sent with the message
now (uint): current block timestamp (alias for block.timestamp)
tx.gasprice (uint): gas price of the transaction
6.4. Solidity in Depth 99Solidity Documentation, 0.4.10
tx.origin (address): sender of the transaction (full call
chain)
revert(): abort execution and revert state changes
keccak256(...) returns (bytes32): compute the Ethereum-SHA-3
(Keccak-256) hash of the
(tightly packed) arguments
sha3(...) returns (bytes32): an alias to keccak256()
sha256(...) returns (bytes32): compute the SHA-256 hash of the
(tightly packed) arguments
ripemd160(...) returns (bytes20): compute the RIPEMD-160 hash
of the (tightly packed) arguments
ecrecover(bytes32 hash, uint8 v, bytes32 r, bytes32 s) returns
(address):
recover address associated with the public key from elliptic
curve signature, return zero on error
addmod(uint x, uint y, uint k) returns (uint): compute (x + y)
% k where the addition is performed with arbitrary precision
and does not wrap around at 2**256
mulmod(uint x, uint y, uint k) returns (uint): compute (x * y)
% k where the multiplication is performed with arbitrary
```

```
precision and does not wrap around at 2**256
this (current contract's type): the current contract,
explicitly convertible to address
super: the contract one level higher in the inheritance
hierarchy
selfdestruct(address recipient): destroy the current contract,
sending its funds to the given address
.balance (uint256): balance of the Address in Wei
.send(uint256 amount) returns (bool): send given amount of Wei
to Address, returns false on failure
.transfer(uint256 amount): send given amount of Wei to Address,
throws on failure
```

block.blockhash(uint blockNumber) returns (bytes32): 某个区块的区块链hash值
block.coinbase (address): 当前区块的挖矿地址
block.difficulty (uint): 当前区块的难度
block.gaslimit (uint): 当前区块的gaslimit
block.number (uint): 当前区块编号
block.timestamp (uint): 当前区块时间戳
msg.data (bytes): 参数数据
msg.gas (uint): 剩余的gas
msg.sender (address): 当前发送消息的地址，执行合约的地址。
msg.sig (bytes4): 方法ID
msg.value (uint): 执行合约时，转账的eth数量，以wei为单位。
now (uint): 时间戳，等价于block.timestamp (uint)
tx.gasprice (uint): 交易的gas单价
tx.origin (address):交易发送地址

6.2. storage

使用 storage 这个关键字时，当前的函数必须是internal或者private类型。

6.3. memory

7. 函数

7.1. 构造方法

构造方法的定义是 contract 与 function 相同

```
pragma solidity ^0.4.18;

contract MyContractByNetkiller {
    /* Constructor */

    function MyContractByNetkiller() public{

    }
}
```

7.2. 定义函数

没有返回值

```
        function setName(string _name) public{
        name = _name;
    }
```

7.3. 函数返回值

有返回值

```
        function getName() public view returns(string){
        return name;
```

```
    }
```

7.4. 参数传递

除了 f(2,3) 这样传递参数，还可以使用类似字典或Map的方式 f({value: 2, key: 3});

```
pragma solidity ^0.4.0;

contract C {
    function f(uint key, uint value) {
        // ...
    }

    function g() {
        // named arguments
        f({value: 2, key: 3});
    }
}
```

7.5. 函数的例子

完整的例子

```
pragma solidity ^0.4.18;

contract MyContractByNetkiller {
    /* Constructor */
    string name;
    int num;
    function MyContractByNetkiller() public{
        name = "default";
        num = 1;
    }
    function setName(string _name) public{
        name = _name;
```

```
    }
    function getName() public view returns(string){
        return name;
    }
    function setNum(int n) public{
        num = n;
    }
    function addNum(int m) public view returns(int res){
        res = m + num;
    }
}
```

7.6. Fallback function

7.7. modifier

modifier 可以理解为 function 的触发器，或者理解为 hook。执行 function的时候会首先运行 modifier

_; 表示执行 modifier 完成所有命令后，继续运行 function 内的逻辑。

```
pragma solidity ^0.4.11;

contract owned {
    function owned() { owner = msg.sender; }
    address owner;
    uint price;
    mapping (address => bool) registeredAddresses;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }
    function changePrice(uint _price) onlyOwner {
```

```
        price = _price;
    }
    function close() onlyOwner {
        selfdestruct(owner);
    }
}
```

8. 事件

什么是 event 呢？ 在 Solidity 中 event 类似触发器，是合约与外部程序连接接口。

外部程序监听事件，智能合约中的事件一旦触发，就将数据交给监听程序处理。

换个角度，如果你懂得消息队列，那么只能合约中的 event 是消息发布者，外部的event 监听程序是消息的消费者。

使用 event 来创建,下面是 ERC20 的标准 event.

```
event Transfer(address indexed from, address indexed to, uint256
value);
```

事件名称你可以随意定义，开头字母大写即可，参数传递根据你的实际需要增加。

```
event Sent(address from, address to, uint amount);
```

9. 面向对象编程

9.1. 可见性修饰符

Solidity对函数和状态变量提供了四种可见性。分别是external,public,internal,private。其中函数默认是public。状态变量默认的可见性是internal。

internal - 状态变量默认为internal类型，函数只能通过内部访问（当前合约或者继承的合约），可在当前合约或继承合约中调用。类似于Java的protected
public - public标识的函数是合约接口的一部分。可以通过内部，或者消息来进行调用。与Java的public含义一致。
external - external标识的函数是合约接口的一部分。函数只能通过外部的方式调用。外部函数在接收大的数组时更有效。Java中无与此对应的关键字。
private - 只能在当前合约内访问，在继承合约中都不可访问。与Java中的private含义一致。

payable ：可支付的函数修饰符，没有该修饰符无法接受转账操作。

9.2. 错误处理

assert(bool condition)：不满足条件，将抛出异常

require(bool condition)：不满足条件，将抛出异常

revert() 抛出异常

```
if(msg.sender != owner) { revert(); }
assert(msg.sender == owner);
require(msg.sender == owner);
```

9.3. interface 接口

接口是抽象的合约，接口中不能实现方法。

接口:

- 不能继承其他合约或接口
- 不能定义构造方法
- 不能定义变量
- 不能定义结构体
- 不能定义枚举

```
pragma solidity ^0.4.11;

interface Token {
    function transfer(address recipient, uint amount) public;
}
```

9.4. library 库

定义 library

```
pragma solidity ^0.4.25;

// This is the same code as before, just without comments
library Set {
  struct Data { mapping(uint => bool) flags; }

  function insert(Data storage self, uint value)
      public
      returns (bool)
  {
      if (self.flags[value])
        return false; // already there
      self.flags[value] = true;
      return true;
  }

  function remove(Data storage self, uint value)
      public
      returns (bool)
  {
      if (!self.flags[value])
          return false; // not there
      self.flags[value] = false;
      return true;
  }

  function contains(Data storage self, uint value)
      public
      view
      returns (bool)
  {
      return self.flags[value];
  }
}
```

调用库中的函数

```
contract C {
    using Set for Set.Data; // this is the crucial change
    Set.Data knownValues;
```

```
    function register(uint value) public {
        // Here, all variables of type Set.Data have
        // corresponding member functions.
        // The following function call is identical to
        // `Set.insert(knownValues, value)`
        require(knownValues.insert(value));
    }
}
```

9.4.1. 使用库来扩展数据类型

```
pragma solidity ^0.4.25;

library Search {
    function indexOf(uint[] storage self, uint value)
        public
        view
        returns (uint)
    {
        for (uint i = 0; i < self.length; i++)
            if (self[i] == value) return i;
        return uint(-1);
    }
}

contract C {
    using Search for uint[];
    uint[] data;

    function append(uint value) public {
        data.push(value);
    }

    function replace(uint _old, uint _new) public {
        // This performs the library function call
        uint index = data.indexOf(_old);
        if (index == uint(-1))
            data.push(_new);
        else
            data[index] = _new;
    }
}
```

9.5. 继承

例子 mortal 继承 owned

```
pragma solidity ^0.4.11;

contract owned {
    function owned() { owner = msg.sender; }
    address owner;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }
}


contract mortal is owned {
    function close() onlyOwner {
        selfdestruct(owner);
    }
}
```

10. 合约调用

```
address token = 0xdC7c2ab64Bc6861852C0Cd60B79564164eD890CF;
token.call(bytes4(sha3("fun(uint256)")), a);
```

```
Function: transfer(address _to, uint256 _value)
MethodID: 0xa9059cbb

token.call('0xa9059cbb', _to, _value);
```

```
pragma solidity ^0.4.24;

contract Test{
    address public token;

    constructor(address _contractAddress) public {
        token = _contractAddress;
    }
    function transfer(address _to, uint256 _value) public
returns (bool success){
        if(token.call(bytes4(keccak256("fun(address,
uint256)")), _to, _value)){
            return false;
        }
        return true;
    }
}
```

11. 合约接收 ETH

首先你需要在智能合约中定义这个函数 function () payable public {}，这时这个合约地址就可以接收ETH了。

测试方法，向合约地址发送ETH即可。

11.1. 调用 selfdestruct(msg.sender); 取出合约中的 ETH

```
pragma solidity ^0.4.24;

contract NetkillerCashier {

    function () payable public {}

    function claim() public {
        selfdestruct(msg.sender);
    }
}
```

https://ropsten.etherscan.io/tx/0x6504df0e18416c3c319f1f11f84ffa40a752b47c257faee58a7ef2c8ef78cc45

```
 Contract 0x0896827f5e3d2683763321bdf780bde1824f6137
 TRANSFER  0.03 Ether from
0x0896827f5e3d2683763321bdf780bde1824f6137 to
0x22c57f0537414fd95b9f0f08f1e51d8b96f14029
 SELF-DESTRUCT Contract
0x0896827f5e3d2683763321bdf780bde1824f6137
```

查看 Code
https://ropsten.etherscan.io/address/0x0896827f5e3d2683763321bdf780bde1824f6137#code 显示

```
Contract SelfDestruct called at TxHash
0x6504df0e18416c3c319f1f11f84ffa40a752b47c257faee58a7ef2c8ef78cc45
```

11.2. 自动退款合约

本合约只收取 1 ETH 多余 ETH 将退给用户

```
pragma solidity ^0.4.24;

// Author: netkiller@msn.com
// Website: http://www.netkiller.cn

contract Refund {

    address owner = 0x0;

    uint256 ticket = 1 ether;

    constructor() public payable {
        owner = msg.sender;
    }

    function () public payable {
        require(msg.value >= ticket);
        if (msg.value > ticket) {
            uint refundFee = msg.value - ticket;
            msg.sender.transfer(refundFee);
        }
    }
}
```

11.3. 收款合约自动转账

合约收到ETH后自动转到 owner 账号中。

```
pragma solidity ^0.4.24;

contract NetkillerCashier {

    address public owner;

    constructor() public payable {
        owner = msg.sender;
    }
    function () payable public {
        owner.transfer(msg.value);
    }

}
```

11.4. 指定账号提取 ETH

```
pragma solidity ^0.4.24;

contract NetkillerCashier {

    address public owner;
    uint public amount;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    constructor() public {
        owner = msg.sender;
    }
```

```
    function () public payable {
        amount += msg.value;
    }

        function transferOwnership(address newOwner) onlyOwner
public {
        if (newOwner != address(0)) {
            owner = newOwner;
        }
    }

    function withdraw() onlyOwner public {
        msg.sender.transfer(amount);
        amount = 0;
    }

}
```

function transferOwnership(address newOwner) 可以修改指定账号提取 ETH

function withdraw()　　提取 ETH 的函数

https://ropsten.etherscan.io/tx/0xadad8c4cd7649d825fb8c362e97f80c4821b07c97d423050289986bd75703b78

```
 Contract 0xb31fb5297340a06e1af3e21c1780b7001db6890a
 TRANSFER  0.05 Ether from
0xb31fb5297340a06e1af3e21c1780b7001db6890a to
0x22c57f0537414fd95b9f0f08f1e51d8b96f14029
```

12. 合约中实例化一个接口

在合约中调用一个已经存在的合约。

```
pragma solidity ^0.4.24;

/****************************************/
/*      Netkiller Crowdsale Contract      */
/****************************************/
/* Author netkiller <netkiller@msn.com>   */
/* Home http://www.netkiller.cn           */
/* Version 2018-06-07 - Solc ver: 0.4.24  */
/****************************************/

interface token {
        function balanceOf(address _address) constant external
returns (uint256);
    function transfer(address receiver, uint amount) external;
}

contract Netkiller {

    token public tokenContract;

    constructor(address addressOfToken) public {
        tokenContract = token(addressOfToken);
    }
        function getBalance(address _address) view public
returns (uint256){
        return tokenContract.balanceOf(_address);
    }
    function transfer(address _to, uint256 _value) payable
public{
        tokenContract.transfer(_to, _value);
    }


}
```

13. 合约中实例化另一个合约

当合约已经部署到链上，如需扩展合约的功能可以采用此种方法。

```
pragma solidity ^0.4.24;

contract ERC20 {
    uint256 public totalSupply;
    uint public decimals;
    function balanceOf(address _address) constant public returns (uint256);
    function transfer(address _to, uint256 _value) public returns (bool
success);
}
contract NetkillerBatchTransfer {

    ERC20 public token;

    constructor(address _contractAddress) public {
        token = ERC20(_contractAddress);
    }
    function getBalance(address _address) view public returns (uint256){
        return token.balanceOf(_address);
    }
}
```

13.1. msg.sender 与 this 的区别

当一个合约访问另一个合约是，msg.sender 与 this 的区别是什么呢?

```
pragma solidity ^0.4.24;

/****************************************/
/*       Netkiller Mini TOKEN           */
/****************************************/
/* Author netkiller <netkiller@msn.com>   */
/* Home http://www.netkiller.cn           */
/* Version 2018-05-31 Fixed transfer bool */
/****************************************/

contract NetkillerMiniToken {

    constructor() public { }

    function test1() public view returns (address addr){
        return msg.sender;
    }
    function test2() public view returns (address addr){
        return this;
    }
```

```
}
```

例子

```
Contract A: 0xf328c11c4df88d18fcbd30ad38d8b4714f4b33bf
Contract B: 0xb9b7e0cb2edf5ea031c8b297a5a1fa20379b6a0a
```

Contract A 调用 Contract B

```
msg.sender      = 0xF328c11c4dF88d18FcBd30ad38d8B4714F4b33bF
this            = 0xB9B7e0cb2EDF5Ea031C8B297A5A1Fa20379b6A0a
```

13.2. 地址格式

在合约中出现地址，例如下面：

```
pragma solidity ^0.4.24;
contract ERC20 {
    function totalSupply() public constant returns (uint);
    function balanceOf(address tokenOwner) public constant returns (uint
balance);
    function allowance(address tokenOwner, address spender) public constant
returns (uint remaining);
    function transfer(address to, uint tokens) public returns (bool success);
    function approve(address spender, uint tokens) public returns (bool
success);
    function transferFrom(address from, address to, uint tokens) public returns
(bool success);

    event Transfer(address indexed from, address indexed to, uint tokens);
    event Approval(address indexed tokenOwner, address indexed spender, uint
tokens);
}

contract TestContract{
    ERC20 public token = ERC20(0xAeeD5A0C200efA0670330d0C7509C854c52AF859);

    function tot() public view returns (uint){
        return token.totalSupply();
    }

    function balance(address _address) public view returns (uint){
        return token.balanceOf(_address);
    }
```

```
}
```

合约中不支持全小写的地址。

正确的：0xAeeD5A0C200efA0670330d0C7509C854c52AF859
错误的：0xaeed5a0c200efa0670330d0c7509c854c52af859

14. Solidity 安全问题

14.1. 整型溢出

什么是整型溢出呢？在solidity编写合约时，定义整型一般是用uint8，uint256。一个变量如果定义为uint8表示的无符号的8位整型，即取值范围为0-255。当给这个变量赋值256时，即整型溢出变成了0，以此类推257变成了1。

```
pragma solidity ^0.4.24;

//author: netkiller <netkiller@msn.com>
//homepage: http://www.netkiller.cn

contract NetkillerOverflowTest{

    function add(uint8 a, uint8 b) pure public returns (uint8){
        uint8 result = a + b;
        return result;
    }
    function sub(uint8 a, uint8 b) pure public returns (uint8){
        uint8 result = a - b;
        return result;
    }
    function mul(uint8 a, uint8 b) pure public returns (uint8){
        uint8 result = a * b;
        return result;
    }
```

```
    function div(uint8 a, uint8 b) pure public returns (uint8){
        uint8 result = a / b;
        return result;
    }
}
```

调用上面合约，运行结果

```
254 + 1 = 255
254 + 2 = 0
254 + 3 = 1
```

减法运行结果

```
10 - 20 = 246
```

乘法运行结果

```
51 * 5 = 255
51 * 6 = 50
```

再来测试乘法

```
255 / 10 = 25
```

这有点想千年虫问题，即99年变成00年后，你无法区分1900年还是2000年。

现在测试一下uint256，uint256支持的取值范围是0到2^256-1

```
pragma solidity ^0.4.24;

//author: netkiller <netkiller@msn.com>
//homepage: http://www.netkiller.cn

contract TestUint256Overflow {
    // (2**256 — 1) + 1 = 0 向上溢出测试
    function overflow() pure public returns (uint256 _overflow)
{
        uint256 max = 2 ** 256 - 1;
        return max + 1;
    }

    // 0 — 1 = 2**256 — 1 向下溢出测试
    function underflow() pure public returns (uint256
_underflow) {
        uint256 min = 0;
        return min - 1;
    }
}
```

运行结果

```
_overflow : 0
_underflow :
115792089237316195423570985008687907853269984665640564039457584
007913129639935
```

第一个函数溢出为 0，第二个函数 0 - 1 = 115792089237316195423570985008687907853269984665640564039457584007913129639935

解决溢出问题使用SafeMath库

```
pragma solidity ^0.4.24;

//author: netkiller <netkiller@msn.com>
//homepage: http://www.netkiller.cn

library SafeMath {

  function mul(uint256 a, uint256 b) internal pure returns
(uint256 c) {
    if (a == 0) {
      return 0;
    }

    c = a * b;
    assert(c / a == b);
    return c;
  }

  function div(uint256 a, uint256 b) internal pure returns
(uint256) {
    return a / b;
  }

  function sub(uint256 a, uint256 b) internal pure returns
(uint256) {
    assert(b <= a);
    return a - b;
  }

  function add(uint256 a, uint256 b) internal pure returns
(uint256 c) {
    c = a + b;
    assert(c >= a);
    return c;
  }
}

contract NetkillerSafeMath {
```

```
    using SafeMath for uint256;

    function add(uint256 a, uint256 b) pure public returns
(uint256){
        uint256 result = a.add(b);
        return result;
    }
    function sub(uint256 a, uint256 b) pure public returns
(uint256){
        uint256 result = a.sub(b);
        return result;
    }
    function mul(uint256 a, uint256 b) pure public returns
(uint256){
        uint256 result = a.mul(b);
        return result;
    }
    function div(uint256 a, uint256 b) pure public returns
(uint256){
        uint256 result = a.div(b);
        return result;
    }
}
```

测试 SafeMath

```
add(115792089237316195423570985008687907853269984665640564039457
7584007913129639934,1)  =>
115792089237316195423570985008687907853269984665640564039457584
007913129639935
add(115792089237316195423570985008687907853269984665640564039457
7584007913129639935,1)  => 抛出异常
```

15. solidity example

15.1. Voting

```
pragma solidity ^0.4.25;
//author: netkiller <netkiller@msn.com>
//homepage: http://www.netkiller.cn
contract Voting {

  mapping (bytes32 => uint8) public votesReceived;

  // 存储候选人名字的数组
  bytes32[] public candidateList;


  // 构造函数 初始化候选人名单
  function Voting(bytes32[] candidateNames) public {

    candidateList = candidateNames;
  }

  // 查询某个候选人的总票数
  function totalVotesFor(bytes32 candidate) public constant
returns (uint8) {
    require(validCandidate(candidate) == true);
    // 或者
    // assert(validCandidate(candidate) == true);
    return votesReceived[candidate];
  }

  // 为某个候选人投票
  function voteForCandidate(bytes32 candidate) public{
    assert(validCandidate(candidate) == true);
    votesReceived[candidate] += 1;
  }

  // 检索投票的姓名是不是候选人的名字
  function validCandidate(bytes32 candidate) public constant
returns (bool) {
    for(uint i = 0; i < candidateList.length; i++) {
      if (candidateList[i] == candidate) {
```

```
        return true;
      }
    }
    return false;
  }

}
```

15.2. MetaCoin

```
pragma solidity ^0.4.25;
contract MetaCoin {
        mapping (address => uint) balances;
        event Transfer(address indexed _from, address indexed
_to, uint256 _value);
        function MetaCoin() public {
                balances[tx.origin] = 10000;
        }
        function sendCoin(address receiver, uint amount) public
returns(bool sufficient) {
                if (balances[msg.sender] < amount) return
false;
                balances[msg.sender] -= amount;
                balances[receiver] += amount;
                Transfer(msg.sender, receiver, amount);
                return true;
        }
        function getBalance(address addr) public view
returns(uint) {
                return balances[addr];
        }
}
```

15.3. Anonymous voting on Ethereum without a tally authority. Protocol from this paper

https://github.com/stonecoldpat/anonymousvoting

15.4. Ballot

```
pragma solidity ^0.4.0;
contract Ballot {

    struct Voter {
        uint weight;
        bool voted;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        uint voteCount;
    }

    address chairperson;
    mapping(address => Voter) voters;
    Proposal[] proposals;

    /// Create a new ballot with $(_numProposals) different
proposals.
    function Ballot(uint8 _numProposals) public {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
        proposals.length = _numProposals;
    }

    /// Give $(toVoter) the right to vote on this ballot.
    /// May only be called by $(chairperson).
    function giveRightToVote(address toVoter) public {
        if (msg.sender != chairperson || voters[toVoter].voted)
return;
        voters[toVoter].weight = 1;
    }

    /// Delegate your vote to the voter $(to).
    function delegate(address to) public {
        Voter storage sender = voters[msg.sender]; // assigns
reference
        if (sender.voted) return;
        while (voters[to].delegate != address(0) &&
voters[to].delegate != msg.sender)
            to = voters[to].delegate;
        if (to == msg.sender) return;
        sender.voted = true;
```

```
        sender.delegate = to;
        Voter storage delegateTo = voters[to];
        if (delegateTo.voted)
            proposals[delegateTo.vote].voteCount +=
sender.weight;
        else
            delegateTo.weight += sender.weight;
    }

    /// Give a single vote to proposal $(toProposal).
    function vote(uint8 toProposal) public {
        Voter storage sender = voters[msg.sender];
        if (sender.voted || toProposal >= proposals.length)
return;
        sender.voted = true;
        sender.vote = toProposal;
        proposals[toProposal].voteCount += sender.weight;
    }

    function winningProposal() public constant returns (uint8
_winningProposal) {
        uint256 winningVoteCount = 0;
        for (uint8 prop = 0; prop < proposals.length; prop++)
            if (proposals[prop].voteCount > winningVoteCount) {
                winningVoteCount = proposals[prop].voteCount;
                _winningProposal = prop;
            }
    }
}
```

15.5. Conference

```
pragma solidity ^0.4.25;

contract Conference {
  address public organizer;
  mapping (address => uint) public registrantsPaid;
  uint public numRegistrants;
  uint public quota;

  event Deposit(address _from, uint _amount);  // so you can
log these events
  event Refund(address _to, uint _amount);
```

```
  function Conference() public{ // Constructor
    organizer = msg.sender;
    quota = 500;
    numRegistrants = 0;
  }
  function buyTicket() public payable returns (bool success) {
    if (numRegistrants >= quota) { return false; }
    registrantsPaid[msg.sender] = msg.value;
    numRegistrants++;
    Deposit(msg.sender, msg.value);
    return true;
  }
  function changeQuota(uint newquota) public {
    if (msg.sender != organizer) { return; }
    quota = newquota;
  }
  function refundTicket(address recipient, uint amount) public
{
    if (msg.sender != organizer) { return; }
    if (registrantsPaid[recipient] == amount) {
      address myAddress = this;
      if (myAddress.balance >= amount) {
        recipient.transfer(amount);
        registrantsPaid[recipient] = 0;
        numRegistrants--;
        Refund(recipient, amount);
      }
    }
  }
  function destroy() public{ // so funds not locked in contract
forever
    if (msg.sender == organizer) {
      selfdestruct(organizer); // send funds to organizer
    }
  }
}
```

控制台调试

```
var contract;
Conference.deployed().then(function(instance)
{contract=instance;});
```

```
contract.buyTicket();
```

测试程序

```
neo@MacBook-Pro ~/ethereum/Conference % cat test/conference.js
var Conference = artifacts.require("./Conference.sol");

contract('Conference', function(accounts) {
        console.log(accounts);
        var owner_account = accounts[0];
    var sender_account = accounts[1];

  it("Initial conference settings should match", function(done)
{

        Conference.deployed({from:
owner_account}).then(function(conference) {
        conference.quota.call().then(function(quota) {
                assert.equal(quota, 100, "Quota doesn't
match!");
        }).then(function() {
                return conference.numRegistrants.call();
        }).then(function(num) {
                assert.equal(num, 0, "Registrants doesn't
match!");
                return conference.organizer.call();
        }).then(function(organizer) {
                assert.equal(organizer, owner_account, "Owner
doesn't match!");
                done();
        }).catch(done);
        }).catch(done);
  });

  it("Should update quota", function(done) {

        Conference.deployed({from:
owner_account}).then(function(conference) {
        conference.quota.call().then(
            function(quota) {
                assert.equal(quota, 100, "Quota doesn't
match!");
        }).then(
```

```
            function() {
                return conference.changeQuota(300);
        }).then(
            function() {
                return conference.quota.call()
        }).then(
            function(quota) {
                assert.equal(quota, 300, "New quota is not
correct!");
                done();
        }).catch(done);
        }).catch(done);
  });


  it("Should let you buy a ticket", function(done) {

        Conference.deployed({ from: accounts[0]
}).then(function(conference) {

        var ticketPrice = web3.toWei(.05, 'ether');
        var initialBalance =
web3.eth.getBalance(conference.address).toNumber();

                        conference.buyTicket({ from:
accounts[1], value: ticketPrice }).then(
          function() {
                                        var newBalance =
web3.eth.getBalance(conference.address).toNumber();
            var difference = newBalance - initialBalance;

assert.equal(difference, ticketPrice, "Difference should be
what was sent");
                                        return
conference.numRegistrants.call();
                        }).then(
                                function(num) {
                                        assert.equal(num, 1,
"there should be 1 registrant");
                                        return
conference.registrantsPaid.call(sender_account);
                        }).then(
                                function(amount) {

assert.equal(amount.toNumber(), ticketPrice, "Sender's paid but
is not listed as paying");
                                        return
web3.eth.getBalance(conference.address);
```

```
                        }).then(
                                function(bal) {
            assert.equal(bal.toNumber(), ticketPrice, "Final
balance mismatch");
                                    done();
                        }).catch(done);
        }).catch(done);
  });

  it("Should issue a refund by owner only", function(done) {

    Conference.deployed({ from: accounts[0]
}).then(function(conference) {

        var ticketPrice = web3.toWei(.05, 'ether');
        var initialBalance =
web3.eth.getBalance(conference.address).toNumber();

        conference.buyTicket({ from: accounts[1], value:
ticketPrice }).then(
          function() {
            var newBalance =
web3.eth.getBalance(conference.address).toNumber();
            var difference = newBalance - initialBalance;
            assert.equal(difference, ticketPrice, "Difference
should be what was sent");

            // Now try to issue refund as second user - should
fail
            return conference.refundTicket(accounts[1],
ticketPrice, {from: accounts[1]});
        }).then(
          function() {
            var balance =
web3.eth.getBalance(conference.address);
            assert.equal(balance, ticketPrice, "Balance should
be unchanged");
            // Now try to issue refund as organizer/owner
            return conference.refundTicket(accounts[1],
ticketPrice, {from: accounts[0]});
        }).then(
          function() {
            var postRefundBalance =
web3.eth.getBalance(conference.address).toNumber();
            assert.equal(postRefundBalance, initialBalance,
"Balance should be initial balance");
            done();
        }).catch(done);
```

```
      }).catch(done);
    });

});
```

第 15 章 Truffle v4.1.8 开发框架

Truffle 是 solidity 开发框架， http://truffleframework.com

1. 安装 Truffle

安装truffle

```
sudo npm install -g truffle
```

升级

```
npm update -g truffle
```

2. 开发环境

2.1. truffle develop

truffle develop 是 truffle 自带的开发环境。

2.2. Ganache

truffle develop 使用起来并不直观，还有另一个开发环境是 Ganache，Ganache有命令行版本和图形界面版本。下面是命令行版本安装方法：

```
sudo npm install -g ganache-cli
```

图形界面版本这里下载 http://truffleframework.com/ganache/

图形界面交互性相比命令行比较好，推荐使用。你能在界面上直接查看账号，区块，交易，日志等等。

2.3. testrpc

由于truffle 4.0 自带开发环境(truffle develop)，所以本章不在使用testrpc(不建议使用)，如果仍需要testrpc请参考下面安装方法：

```
# 安装testrpc
sudo npm install -g ethereumjs-testrpc
```

3. Truffle 快速入门

3.1. Ubuntu 环境

3.1.1. 启动开发环境

truffle 自带一个开发环境

```
neo@netkiller ~/ethereum/truffle-project %  truffle develop
Truffle Develop started at http://localhost:9545/

Accounts:
(0) 0x627306090abab3a6e1400e9345bc60c78a8bef57
(1) 0xf17f52151ebef6c7334fad080c5704d77216b732
(2) 0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef
(3) 0x821aea9a577a9b44299b9c15c88cf3087f3b5544
(4) 0x0d1d4e623d10f9fba5db95830f7d3839406c6af2
(5) 0x2932b7a2355d6fecc4b5c0b6bd44cc31df247a2e
(6) 0x2191ef87e392377ec08e7c08eb105ef5448eced5
(7) 0x0f4f2ac550a1b4e2280d04c21cea7ebd822934b5
(8) 0x6330a553fc93768f612722bb8c2ec78ac90b3bbc
(9) 0x5aeda56215b167893e80b4fe645ba6d5bab767de

Private Keys:
(0)
c87509a1c067bbde78beb793e6fa76530b6382a4c0241e5e4a9ec0a0f44dc0d3
(1)
ae6ae8e5ccbfb04590405997ee2d52d2b330726137b875053c36d94e974d162f
(2)
0dbbe8e4ae425a6d2687f1a7e3ba17bc98c673636790f1b8ad91193c05875ef1
(3)
c88b703fb08cbea894b6aeff5a544fb92e78a18e19814cd85da83b71f772aa6c
(4)
388c684f0ba1ef5017716adb5d21a053ea8e90277d0868337519f97bede61418
(5)
659cbb0e2411a44db63778987b1e22153c086a95eb6b18bdf89de078917abc63
(6)
82d052c865f5763aad42add438569276c00d3d88a2d062d36b2bae914d58b8c8
(7)
aa3680d5d48a8283413f7a108367c7299ca73f553735860a87b08f39395618b7
(8)
0f62d96d6675f32685bbdb8ac13cda7c23436f63efbb9d07700d8669ff12b7c4
```

```
(9)
8d5366123cb560bb606379f90a0bfd4769eecc0557f1b362dcae9012b548b1e5

Mnemonic: candy maple cake sugar pudding cream honey rich smooth
crumble sweet treat

truffle(develop)>
```

truffle develop 的作用于 testrpc 类似。

```
neo@netkiller ~/ethereum/truffle-project % testrpc
EthereumJS TestRPC v6.0.3 (ganache-core: 2.0.2)

Available Accounts
==================
(0) 0xb5fd43ee8fa5ce1db9a30a25ba385ee3bfc72966
(1) 0xf5a732345734e1f0f49cbadb145a20d1e1a44b95
(2) 0x834fcd8c55fdf21fd14c82e9a1ef5d3636a2fed6
(3) 0x5aa4d047d85727309d3ca653c83c3bb0ecd18903
(4) 0xb4db2dede86f4539e56ac4438f6e36f09c307e46
(5) 0x8da382b1a10ab2f1dc149e19fda228a07c78935c
(6) 0xb290297e89b52713548ff93e5fc23bc3c4183dde
(7) 0x546183289bd4d9d33a3aee0ee663c0729926e583
(8) 0xca58321e442533b7f827e6e8976e1905acd15214
(9) 0xe2c0b336bbb03564204e15a2cb7744564a53efcc

Private Keys
==================
(0)
ff32f7a06e2fb26b51a745c1e428c60df92c0f9bb3301b19a5b7e0cdfaae521a
(1)
cdbfe40321b6ade8a246748df1c48a738b8a531aee4d1f60a45bfd7f941e0064
(2)
7092117c2d7832980945e18645a60a1ed0e59261d040749f8b5202c2fc653d74
(3)
f329657c9ad808e9f794a7462a1a9c276266343d5ced263ab618b6a19d6857c1
(4)
1221766592618add3a57ab109f00efcc70867dd8a9b10a0f7ea75c2b619edfc3
(5)
c27005d6c3581193124c84766cc0b1cc318cb201b7d00b1035f4a4c7767ba790
(6)
6b7f43dca1dcc00203b751191096bb0602e17a9a94dcee8b846329efa703cea9
(7)
```

```
41be5971d71935bc88c3cf8aefd78ed3188c8721b7134cda3b25d353faf05d4f
(8)
bd4c9d512a4f2da2cdcd9e4f89c049e3e7ac81bf57a05369997c2f13e793bebb
(9)
04196d803f743cc1fd021e7d02d5a552f14ab9826ccd4d4b265ff96c45169d2c

HD Wallet
==================
Mnemonic:      confirm shift cable melody caught swing erode
language spend victory conduct van
Base HD Path:  m/44'/60'/0'/0/{account_index}

Listening on localhost:8545
```

3.1.2. 创建项目

```
cd ~/ethereum
mkdir truffle-project
cd truffle-project
truffle init
```

操作演示

```
neo@netkiller ~/ethereum/truffle-project % truffle init
Downloading...
Unpacking...
Setting up...
Unbox successful. Sweet!

Commands:

  Compile:        truffle compile
  Migrate:        truffle migrate
  Test contracts: truffle test

neo@netkiller ~/ethereum/truffle-project % tree
.
|-- contracts
```

```
|   `-- Migrations.sol
|-- migrations
|   `-- 1_initial_migration.js
|-- test
|-- truffle-config.js
`-- truffle.js

3 directories, 4 files
```

目录结构简单说明如下:

contract/ - Truffle默认的合约文件存放地址。
migrations/ - 存放发布脚本文件
test/ - 用来测试应用和合约的测试文件
truffle.js - Truffle的配置文件

app/ - 需要用户创建，应用文件运行的默认目录。

3.1.3. 创建合约

```
pragma solidity ^0.4.18;

contract Netkiller {
    string name;
    int num;
    function Netkiller() public{
        name = "default";
        num = 1;
    }
    function setName(string _name) public{
        name = _name;
    }
    function getName() public view returns(string){
        return name;
    }
    function setNum(int n) public{
```

```
        num = n;
    }
    function addNum(int m) public view returns(int res){
        res = m + num;
    }
}
```

```
neo@netkiller ~/ethereum/truffle-project % vim
migrations/2_initial_migration.js

var Netkiller = artifacts.require("./Netkiller.sol");

module.exports = function(deployer) {
  deployer.deploy(Netkiller);
};
```

3.1.4. 配置 Truffle

打开文件 truffle.js

```
module.exports = {
  // See
<http://truffleframework.com/docs/advanced/configuration>
  // to customize your Truffle configuration!
};
```

修改为

```
module.exports = {
  // See
<http://truffleframework.com/docs/advanced/configuration>
  // to customize your Truffle configuration!
```

```
  networks: {
    development: {
      host: "localhost",
      port: 9545,
      network_id: "*" // Match any network id
    }
  }

};
```

3.1.5. 编译智能合约

```
neo@netkiller ~/ethereum/truffle-project % truffle compile
Compiling ./contracts/Migrations.sol...
Compiling ./contracts/Netkiller.sol...
Writing artifacts to ./build/contracts
```

truffle默认只会编译最后一次修改过的合约文件, 这是为了减少比重复编译。"--all"选项,可以强制编译所有文件。

编译结果

```
neo@netkiller ~/ethereum/truffle-project % find build
build
build/contracts
build/contracts/Migrations.json
build/contracts/Netkiller.json
```

3.1.6. migrate

```
neo@netkiller ~/ethereum/truffle-project % truffle migrate
Using network 'development'.
```

```
Network up to date.
```

3.2. Mac 环境

Mac 环境

```
neo@MacBook-Pro ~/ethereum/truffle % node --version
v9.5.0

neo@MacBook-Pro ~/ethereum/truffle % npm version
{ npm: '5.6.0',
  ares: '1.13.0',
  cldr: '32.0.1',
  http_parser: '2.7.0',
  icu: '60.2',
  modules: '59',
  napi: '2',
  nghttp2: '1.29.0',
  node: '9.5.0',
  openssl: '1.0.2n',
  tz: '2017c',
  unicode: '10.0',
  uv: '1.19.1',
  v8: '6.2.414.46-node.18',
  zlib: '1.2.11' }

neo@MacBook-Pro ~/ethereum/truffle % truffle version
Truffle v4.0.6 (core: 4.0.6)
Solidity v0.4.19 (solc-js)
```

创建项目并初始化

```
mkdir -p ~/ethereum/truffle
cd      ethereum/truffle
truffle init
```

truffle 自带一个开发环境

```
neo@netkiller ~/ethereum/truffle-project %  truffle develop
Truffle Develop started at http://localhost:9545/

Accounts:
(0) 0x627306090abab3a6e1400e9345bc60c78a8bef57
(1) 0xf17f52151ebef6c7334fad080c5704d77216b732
(2) 0xc5fdf4076b8f3a5357c5e395ab970b5b54098fef
(3) 0x821aea9a577a9b44299b9c15c88cf3087f3b5544
(4) 0x0d1d4e623d10f9fba5db95830f7d3839406c6af2
(5) 0x2932b7a2355d6fecc4b5c0b6bd44cc31df247a2e
(6) 0x2191ef87e392377ec08e7c08eb105ef5448eced5
(7) 0x0f4f2ac550a1b4e2280d04c21cea7ebd822934b5
(8) 0x6330a553fc93768f612722bb8c2ec78ac90b3bbc
(9) 0x5aeda56215b167893e80b4fe645ba6d5bab767de

Private Keys:
(0)
c87509a1c067bbde78beb793e6fa76530b6382a4c0241e5e4a9ec0a0f44dc0d3
(1)
ae6ae8e5ccbfb04590405997ee2d52d2b330726137b875053c36d94e974d162f
(2)
0dbbe8e4ae425a6d2687f1a7e3ba17bc98c673636790f1b8ad91193c05875ef1
(3)
c88b703fb08cbea894b6aeff5a544fb92e78a18e19814cd85da83b71f772aa6c
(4)
388c684f0ba1ef5017716adb5d21a053ea8e90277d0868337519f97bede61418
(5)
659cbb0e2411a44db63778987b1e22153c086a95eb6b18bdf89de078917abc63
(6)
82d052c865f5763aad42add438569276c00d3d88a2d062d36b2bae914d58b8c8
(7)
aa3680d5d48a8283413f7a108367c7299ca73f553735860a87b08f39395618b7
(8)
0f62d96d6675f32685bbdb8ac13cda7c23436f63efbb9d07700d8669ff12b7c4
(9)
8d5366123cb560bb606379f90a0bfd4769eecc0557f1b362dcae9012b548b1e5

Mnemonic: candy maple cake sugar pudding cream honey rich smooth
crumble sweet treat

truffle(develop)>
```

创建合约文件 contracts/Greeter.sol

```
pragma solidity ^0.4.20;

contract Greeter
{
    address creator;
    string greeting;

    function Greeter() public
    {
        creator = msg.sender;
        greeting = "default";
    }

    function greet() constant public returns (string)
    {
        return greeting;
    }

    function setGreeting(string _newgreeting) public
    {
        greeting = _newgreeting;
    }

    function kill() public
    {
        if (msg.sender == creator)
            selfdestruct(creator);

    }

}
```

创建部署文件 migrations/2_initial_migration.js

```
var Greeter = artifacts.require("./Greeter.sol");

module.exports = function(deployer) {
  deployer.deploy(Greeter);
};
```

打开文件 truffle.js

```
module.exports = {
  // See
<http://truffleframework.com/docs/advanced/configuration>
  // to customize your Truffle configuration!
};
```

修改为

```
module.exports = {
  // See
<http://truffleframework.com/docs/advanced/configuration>
  // to customize your Truffle configuration!

  networks: {
    development: {
      host: "localhost",
      port: 9545,
      network_id: "*" // Match any network id
    }
  }

};
```

编译并部署合约

```
neo@MacBook-Pro ~/ethereum/truffle % truffle compile
Compiling ./contracts/Greeter.sol...
Writing artifacts to ./build/contracts

neo@MacBook-Pro ~/ethereum/truffle % truffle migrate --reset
Using network 'development'.
```

```
Running migration: 1_initial_migration.js
  Replacing Migrations...
  ...
0xddeac9a1c57772df50064f11227fcb5515e54a3e88e15843f5c0bc1b55a0dad
7
  Migrations: 0x2c2b9c9a4a25e24b174f26114e8926a9f2128fe4
Saving successful migration to network...
  ...
0x9b51540f5a7d75a8fc920e3e5e4ec66792ba31fd006bd176901f0e6347af2db
a
Saving artifacts...
Running migration: 2_initial_migration.js
  Deploying Greeter...
  ...
0x312ead931bbe4b288315317bdf6735ba4fe4f30c20382f085ca27be34581998
3
  Greeter: 0xfb88de099e13c3ed21f80a7a1e49f8caecf10df6
Saving successful migration to network...
  ...
0x69eaa7ed49cc72426706d54c4f52ba70b742ed6910f1223eb0df5f250b4b8ec
3
Saving artifacts...
```

测试脚本

```
Greeter.deployed().then(instance =>
console.log(instance.address))

var contract;
Greeter.deployed().then(function(instance){contract= instance;});
contract.greet();
contract.setGreeting("http://www.netkiller.cn")
contract.greet();
```

进入控制台，交互执行上面程序

```
neo@MacBook-Pro ~/ethereum/truffle % truffle console
```

```
truffle(development)>

truffle(development)> Greeter.deployed().then(instance =>
console.log(instance.address))
0x82d50ad3c1091866e258fd0f1a7cc9674609d254
undefined
truffle(development)> var contract;
undefined
truffle(development)> Greeter.deployed().then(function(instance)
{contract= instance;});
undefined
truffle(development)> contract.greet();
'default'
truffle(development)>
contract.setGreeting("http://www.netkiller.cn")
{ tx:
'0xa5cbfba78c84415517740a482c2bf2208da0c6b0ecabcd5c22db2c85749041
c8',
  receipt:
   { transactionHash:
'0xa5cbfba78c84415517740a482c2bf2208da0c6b0ecabcd5c22db2c85749041
c8',
     transactionIndex: 0,
     blockHash:
'0x557758e0d9b1ef81f41728cb92f43041d009751d5ce5e2e4424f7fb90f5204
1a',
     blockNumber: 16,
     gasUsed: 34206,
     cumulativeGasUsed: 34206,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.greet();
'http://www.netkiller.cn'
```

3.3. ERC20 代币部署

3.3.1. 合约文件

```
[ethereum@netkiller truffle]$ cat contracts/TokenERC20.sol
pragma solidity ^0.4.20;
```

```
contract TokenERC20 {
    address public owner;
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 2;
    // 18 decimals is the strongly suggested default, avoid
changing it
    uint256 public totalSupply;

    // This creates an array with all balances
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public
allowance;

    // This generates a public event on the blockchain that will
notify clients
    event Transfer(address indexed from, address indexed to,
uint256 value);

    // This notifies clients about the amount burnt
    event Burn(address indexed from, uint256 value);

    mapping (address => bool) public frozenAccount;
    event FrozenFunds(address target, bool frozen);

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the
creator of the contract
     */
    function TokenERC20(
        uint256 initialSupply,
        string tokenName,
        string tokenSymbol
    ) public {
        owner = msg.sender;

        totalSupply = initialSupply * 10 ** uint256(decimals);
// Update total supply with the decimal amount
        balanceOf[msg.sender] = totalSupply;                //
Give the creator all initial tokens
        name = tokenName;                                   //
Set the name for display purposes
        symbol = tokenSymbol;                               //
Set the symbol for display purposes
    }
```

```
    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    /**
     * Internal transfer, only can be called by this contract
     */
    function _transfer(address _from, address _to, uint _value)
internal {
        // Prevent transfer to 0x0 address. Use burn() instead
        require(_to != 0x0);
        // Check if the sender has enough
        require(balanceOf[_from] >= _value);
        // Check for overflows
        require(balanceOf[_to] + _value > balanceOf[_to]);
        // Save this for an assertion in the future
        uint previousBalances = balanceOf[_from] +
balanceOf[_to];
        // Subtract from the sender
        balanceOf[_from] -= _value;
        // Add the same to the recipient
        balanceOf[_to] += _value;
        // Asserts are used to use static analysis to find bugs
in your code. They should never fail
        assert(balanceOf[_from] + balanceOf[_to] ==
previousBalances);
    }

    function transfer(address _to, uint256 _value) public {
        require(!frozenAccount[msg.sender]);
        _transfer(msg.sender, _to, _value);
    }

    function transferFrom(address _from, address _to, uint256
_value) public returns (bool success) {
        require(!frozenAccount[msg.sender]);
        require(_value <= allowance[_from][msg.sender]);     //
Check allowance
        allowance[_from][msg.sender] -= _value;
        _transfer(_from, _to, _value);
        return true;
    }

    function approve(address _spender, uint256 _value) public
        returns (bool success) {
        allowance[msg.sender][_spender] = _value;
        return true;
```

```
    }

    function approveAndCall(address _spender, uint256 _value)
        public
        returns (bool success) {
        if (approve(_spender, _value)) {
            return true;
        }
    }

    function burn(uint256 _value) onlyOwner public returns (bool
success) {
        require(balanceOf[msg.sender] >= _value);   // Check if
the sender has enough
        balanceOf[msg.sender] -= _value;            // Subtract
from the sender
        totalSupply -= _value;                      // Updates
totalSupply
        return true;
    }


    function burnFrom(address _from, uint256 _value) onlyOwner
public returns (bool success) {
        require(balanceOf[_from] >= _value);                //
Check if the targeted balance is enough
        require(_value <= allowance[_from][msg.sender]);    //
Check allowance
        balanceOf[_from] -= _value;                         //
Subtract from the targeted balance
        allowance[_from][msg.sender] -= _value;             //
Subtract from the sender's allowance
        totalSupply -= _value;                              //
Update totalSupply
        return true;
    }


  function transfer(address _to, uint256 _value, bytes _data)
public returns (bool) {
    require(_to != address(this));
    transfer(_to, _value);
    require(_to.call(_data));
    return true;
  }

  function transferFrom(address _from, address _to, uint256
_value, bytes _data) public returns (bool) {
    require(_to != address(this));
```

```
    transferFrom(_from, _to, _value);
    require(_to.call(_data));
    return true;
  }

  function approve(address _spender, uint256 _value, bytes _data)
public returns (bool) {
    require(_spender != address(this));
    approve(_spender, _value);
    require(_spender.call(_data));
    return true;
  }


    function transferOwnership(address _owner) onlyOwner public {
        owner = _owner;
    }
    function mintToken(address target, uint256 mintedAmount)
public onlyOwner {
        balanceOf[target] += mintedAmount;
        totalSupply += mintedAmount;
    }

    function freezeAccount(address target, bool freeze) public
onlyOwner {
        frozenAccount[target] = freeze;
    }
}
```

3.3.2. 部署文件

```
[ethereum@netkiller truffle]$ cat migrations/2_initial_token.js
var TokenERC20 = artifacts.require("./TokenERC20.sol");

module.exports = function(deployer) {
  deployer.deploy(TokenERC20,1200000000,"Netkiller Coin","NKC");
};
```

3.3.3. 编译部署

```
[ethereum@netkiller truffle]$ truffle compile
Compiling ./contracts/Migrations.sol...
Compiling ./contracts/TokenERC20.sol...
Writing artifacts to ./build/contracts

[ethereum@netkiller truffle]$ truffle migrate
Using network 'development'.

Running migration: 1_initial_migration.js
  Deploying Migrations...
  ...
0x80746c040d6c6ed8178f90d74356a994fea81516bdd6579d003c433d7a0a011
6
  Migrations: 0x8cdaf0cd259887258bc13a92c0a6da92698644c0
Saving successful migration to network...
  ...
0xd7bc86d31bee32fa3988f1c1eabce403a1b5d570340a3a9cdba53a472ee8c95
6
Saving artifacts...
Running migration: 2_initial_token.js
  Deploying TokenERC20...
  ...
0xb375b52fe6df9f298828672f3ba3560d7a6f6806ce344e964439d0febc13cb9
7
  TokenERC20: 0x345ca3e014aaf5dca488057592ee47305d9b3e10
Saving successful migration to network...
  ...
0xf36163615f41ef7ed8f4a8f192149a0bf633fe1a2398ce001bf44c43dc7bdda
0
Saving artifacts...
```

3.3.4. 合约调用

```
[ethereum@netkiller truffle]$ truffle console
truffle(development)> var contract;
undefined
truffle(development)>
TokenERC20.deployed().then(function(instance)
{contract=instance;});
undefined
truffle(development)> contract.symbol.call().then(console.log);
```

```
NKC
truffle(development)> contract.name.call().then(console.log);
Netkiller Coin
```

3.4. 高级ERC20代币合约

3.4.1. 部署合约

解锁账号，查看 gas limit 价格

```
[ethereum@netkiller ~]$ geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.7-stable/linux-amd64/go1.10.2
coinbase: 0x8232ef29d29f46d3621350ab7097604247ed4830
at block: 2863 (Fri, 11 May 2018 17:16:01 CST)
 datadir: /home/ethereum/.ethereum
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0
personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> personal.unlockAccount(eth.accounts[0],"12345678",50000)
true
> web3.eth.getBlock("pending").gasLimit
4712388
> exit
```

配置 truffle.js

```
[ethereum@netkiller truffle]$ cat truffle.js
module.exports = {
  // See
<http://truffleframework.com/docs/advanced/configuration>
  // to customize your Truffle configuration!

  networks: {
    development: {
      host: "localhost",
```

```
      port: 8545,
          gas: 4712388,
      network_id: "*" // Match any network id
    }
  }

};
```

```
[ethereum@netkiller truffle]$ truffle compile
(node:23256) ExperimentalWarning: The fs.promises API is
experimental
Compiling ./contracts/NetkillerAdvancedToken.sol...
Writing artifacts to ./build/contracts
```

```
[ethereum@netkiller truffle]$ truffle migrate
(node:23456) ExperimentalWarning: The fs.promises API is
experimental
Using network 'development'.

Running migration: 1_initial_migration.js
  Deploying Migrations...
  ...
0xa5937808d9d42dba231738d79d5989e160a2bbc02aa7e8938d0ee71a11eab9a
7
  Migrations: 0x0f6d790c0ce6453161ead810246602c601f836e7
Saving successful migration to network...
  ...
0xabd2d76488caa48eac5b9ad5d662b34a8b41acacb5657ff4f78c5c4530913d2
b
Saving artifacts...
Running migration: 2_initial_token.js
  Deploying NetkillerAdvancedToken...
  ...
0x190239e85c54aa407ba8dd98357ebabbe2dcab65811f3c4aab7486af27436a0
9
  NetkillerAdvancedToken:
0xb45bd60c48ea18991a5f25a644682d8cf7572ccf
Saving successful migration to network...
  ...
0x97b9fffef368b2411341d9fed31bd771d0beebe02bde8c19c61a9e0dde0cc77
3
```

```
Saving artifacts...
```

3.4.2. 控制台检查合约

```
var contract;
NetkillerAdvancedTokenAirDrop.deployed().then(function(instance)
{contract=instance;});
contract.symbol.call().then(console.log);
contract.name.call().then(console.log);
contract.totalSupply.call().then(console.log);
contract.balanceOf.call(web3.eth.accounts[0]).then(console.log);
```

3.4.3. 测试转账

```
contract.transfer(web3.eth.accounts[1],100).then(function()
{contract.balanceOf.call(web3.eth.accounts[1]).then(console.log);
});
```

```
contract.transferFrom(web3.eth.accounts[0],web3.eth.accounts[1],1
00).then(function()
{contract.balanceOf.call(web3.eth.accounts[1]).then(console.log);
});
```

3.4.4. 锁仓

```
contract.setLock(true);
contract.transfer(web3.eth.accounts[2],100).then(function()
{contract.balanceOf.call(web3.eth.accounts[2]).then(console.log);
});
```

演示

```
truffle(development)> contract.setLock(true);
{ tx:
'0x6a603ca9456b574224aa97b8bf1d66280fc8509c94253c0037bc5bb71135a6
67',
  receipt:
   { transactionHash:
'0x6a603ca9456b574224aa97b8bf1d66280fc8509c94253c0037bc5bb71135a6
67',
     transactionIndex: 0,
     blockHash:
'0x4e0893b5ff6b1d85182dca0ceb3c0c2cb750183ff856b7ad2e323ad2a3b55f
dc',
     blockNumber: 13,
     gasUsed: 42424,
     cumulativeGasUsed: 42424,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)>
contract.transfer(web3.eth.accounts[2],100).then(function()
{contract.balanceOf.call(web3.eth.accounts[2]).then(console.log);
});
Error: VM Exception while processing transaction: revert
    at XMLHttpRequest._onHttpResponseEnd
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xh
r2.js:509:1)
    at XMLHttpRequest._setReadyState
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xh
r2.js:354:1)
    at XMLHttpRequestEventTarget.dispatchEvent
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xh
r2.js:64:1)
    at XMLHttpRequest.request.onreadystatechange
```

```
(/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/we
b3/httpprovider.js:128:1)
    at
/usr/local/lib/node_modules/truffle/build/webpack:/~/truffle-
provider/wrapper.js:134:1
    at
/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web
3/requestmanager.js:86:1
    at Object.InvalidResponse
(/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/we
b3/errors.js:38:1)
```

解锁后再测试

```
contract.setLock(false);
contract.transfer(web3.eth.accounts[2],100).then(function()
{contract.balanceOf.call(web3.eth.accounts[2]).then(console.log);
});
```

```
truffle(development)> contract.setLock(false);
{ tx:
'0x3d4b8d49626c1171ab7160274a9f573bb06999fa7bc33240704199c7905f62
fd',
  receipt:
   { transactionHash:
'0x3d4b8d49626c1171ab7160274a9f573bb06999fa7bc33240704199c7905f62
fd',
     transactionIndex: 0,
     blockHash:
'0x0c78b27d817449b6a3727a13041f1880cbf5466a8b81719f2541c08d59ca5f
b3',
     blockNumber: 15,
     gasUsed: 13680,
     cumulativeGasUsed: 13680,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)>
```

```
contract.transfer(web3.eth.accounts[2],100).then(function()
{contract.balanceOf.call(web3.eth.accounts[2]).then(console.log);
});
undefined
truffle(development)> BigNumber { s: 1, e: 2, c: [ 100 ] }
```

3.4.5. 测试空投

```
[ethereum@netkiller ~]$ geth account new
INFO [05-11|17:25:49] Maximum peer count
ETH=25 LES=0 total=25
Your new account is locked with a password. Please give a
password. Do not forget this password.
Passphrase:
Repeat passphrase:
Address: {3c1ba8b80b9a8697f2e34194c2a73a93105be23d}
```

```
contract.mintAirdropToken(1000000);
contract.totalAirdropSupply.call().then(console.log);
contract.totalSupply.call().then(console.log);
contract.setAirdrop(10);
contract.setAirdropLock(true);
contract.balanceOf.call(web3.eth.accounts[3]).then(console.log);

contract.currentTotalAirdrop.call().then(console.log);
```

操作演示

```
[ethereum@netkiller ~]$  truffle console
truffle(development)> var contract;
NetkillerAdvancedTokenAirDrop.deployed().then(function(instance)
{contract=instance;});undefined
truffle(development)>
NetkillerAdvancedTokenAirDrop.deployed().then(function(instance)
{contract=instance;});
```

```
undefined
truffle(development)> contract.mintAirdropToken(1000000);
{ tx:
'0xab251fafc30724273259442fc02bfa429235ff14535f4fb25e29bed7de09ea
11',
  receipt:
   { transactionHash:
'0xab251fafc30724273259442fc02bfa429235ff14535f4fb25e29bed7de09ea
11',
     transactionIndex: 0,
     blockHash:
'0x5498eb579a08e8da1ca2d5ad3d6dd5b0a5c21cc9a630809783d09b65e26929
b9',
     blockNumber: 28,
     gasUsed: 47701,
     cumulativeGasUsed: 47701,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.setAirdrop(10);
{ tx:
'0xd8a3a16328373858e3cc30c7947b3c3c00db447bb3306f2cf2af58fd0215ae
f8',
  receipt:
   { transactionHash:
'0xd8a3a16328373858e3cc30c7947b3c3c00db447bb3306f2cf2af58fd0215ae
f8',
     transactionIndex: 0,
     blockHash:
'0x5be96cfd95031a132fc643b06d01dba0b14f736fa9fa3f16dd5bbd9c7f0307
00',
     blockNumber: 29,
     gasUsed: 42209,
     cumulativeGasUsed: 42209,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> contract.setAirdropLock(true);
{ tx:
'0x7b8b5747a64111a500cce22371d78b2c3c4ce0b5a2b585b26c0278876044e2
31',
  receipt:
   { transactionHash:
'0x7b8b5747a64111a500cce22371d78b2c3c4ce0b5a2b585b26c0278876044e2
31',
     transactionIndex: 0,
     blockHash:
```

```
'0xd6960add791b36afac1126a00af16f2833043dcb3416d68bb4056db64313dc
93',
     blockNumber: 30,
     gasUsed: 42408,
     cumulativeGasUsed: 42408,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)>
contract.balanceOf.call(web3.eth.accounts[5]).then(console.log);
BigNumber { s: 1, e: 1, c: [ 10 ] }
undefined
```

4. Truffle 命令详解

```
neo@MacBook-Pro ~/ethereum/truffle % truffle help
Truffle v4.0.6 - a development framework for Ethereum

Usage: truffle <command> [options]

Commands:
  init      Initialize new and empty Ethereum project
  compile   Compile contract source files
  migrate   Run migrations to deploy contracts
  deploy    (alias for migrate)
  build     Execute build pipeline (if configuration present)
  test      Run JavaScript and Solidity tests
  debug     Interactively debug any transaction on the blockchain (experimen
tal)
  opcode    Print the compiled opcodes for a given contract
  console   Run a console with contract abstractions and commands availabl
e
  develop   Open a console with a local development blockchain
  create    Helper to create new contracts, migrations and tests
  install   Install a package from the Ethereum Package Registry
  publish   Publish a package to the Ethereum Package Registry
  networks  Show addresses for deployed contracts on each network
  watch     Watch filesystem for changes and rebuild the project automaticall
y
  serve     Serve the build directory on localhost and watch for changes
  exec      Execute a JS module within this Truffle environment
  unbox     Download a Truffle Box, a pre-built Truffle project
  version   Show version number and exit

See more at http://truffleframework.com/docs
```

4.1. version

输出版本号然后退出。

```
neo@MacBook-Pro ~/ethereum/truffle % truffle version
Truffle v4.0.6 (core: 4.0.6)
Solidity v0.4.19 (solc-js)
```

4.2. Truffle console 控制台

```
neo@MacBook-Pro ~/ethereum/truffle % truffle console
truffle(development)>
```

4.3. create

4.3.1. contract 创建合约

```
neo@MacBook-Pro ~/ethereum/truffle % truffle create contract
MyContract
neo@MacBook-Pro ~/ethereum/truffle % cat
contracts/MyContract.sol
pragma solidity ^0.4.4;

contract MyContract {
  function MyContract() {
    // constructor
  }
}
```

4.3.2. test 创建单元测试

```
neo@MacBook-Pro ~/ethereum/truffle % truffle create test MyTest
neo@MacBook-Pro ~/ethereum/truffle % cat test/my_test.js
contract('MyTest', function(accounts) {
  it("should assert true", function(done) {
    var my_test = MyTest.deployed();
    assert.isTrue(true);
    done();
  });
});
```

4.4. migrate

```
% truffle migrate --reset
```

4.5. compile

```
% truffle compile --all
```

4.6. test

运行测试

```
neo@MacBook-Pro ~/ethereum/truffle % truffle test
Using network 'development'.


  Contract: Migrations
    1) should assert true
```

```
    > No events were emitted


  0 passing (31ms)
  1 failing

  1) Contract: Migrations should assert true:
     ReferenceError: Migrations is not defined
      at Context.<anonymous> (test/migrations.js:3:22)
```

运行单个测试文件

```
neo@MacBook-Pro ~/ethereum/truffle % truffle test
test/migrations.js
```

4.7. watch

启动后监控文件系统的边龙并自动构建项目。

```
truffle watch
```

5. 合约开发

5.1. 构造方法

在 Truffer 中部署构造方法需要参数传递例子如下，MyContract 需要传递参数 _name：

```
pragma solidity ^0.4.19;

contract MyContract {

    string name;

    function MyContract(string _name) public{
        name = _name;
    }

    function getName() public view returns (string) {
        return name;
    }
}
```

migrations/3_initial_migration.js

```
var MyContract = artifacts.require("./MyContract.sol");

module.exports = function(deployer) {
  deployer.deploy(MyContract,"Netkiller");
};
```

给构造方法传递变量的方法是 deployer.deploy(MyContract,arg1, arg2, ...); arg1 是传递的参数。

多个合约传递方法是:

```
deployer.deploy([
  [ContractA, arg1, arg2, ...],
  ContractB,
  [ContractC, arg1]
]);
```

6. truffle console

6.1. 获取账号列表

```
truffle(development)> web3.eth.accounts
[ '0x8232ef29d29f46d3621350ab7097604247ed4830',
  '0x3c1ba8b80b9a8697f2e34194c2a73a93105be23d' ]

truffle(development)> web3.eth.getAccounts(function(err,res) {
accounts = res; });
undefined
truffle(development)> accounts[0]
'0x8232ef29d29f46d3621350ab7097604247ed4830'
```

6.2. 余额

```
truffle(development)> web3.eth.getBalance(web3.eth.accounts[0])
BigNumber { s: 1, e: 22, c: [ 206250000 ] }
```

6.3. 实例化合约

```
var contract;
Conference.deployed().then(function(instance)
{contract=instance;});
```

6.4. 访问 public 变量

```
truffle(development)> contract.quota.call().then(console.log);
BigNumber { s: 1, e: 1, c: [ 50 ] }
undefined
```

6.5. 调用 public 函数

```
var contract;
Conference.deployed().then(function(instance)
{contract=instance;});
contract.buyTicket();
```

函数参数 call(param1, param2 .....)

```
truffle(development)>
contract.balanceOf.call(web3.eth.accounts[0]).then(console.log)
;
BigNumber { s: 1, e: 27, c: [ 12000000000000 ] }
undefined
```

7. 测试

7.1. balanceOf

```
it("should return the balance of token owner", function() {
  var token;
  return Token.deployed().then(function(instance){
    token = instance;
    return token.balanceOf.call(accounts[0]);
  }).then(function(result){
    assert.equal(result.toNumber(), 1000000, 'balance is
wrong');
  })
});
```

7.2. transfer

```
it("should transfer right token", function() {
  var token;
  return Token.deployed().then(function(instance){
    token = instance;
    return token.transfer(accounts[1], 500000);
  }).then(function(){
    return token.balanceOf.call(accounts[0]);
  }).then(function(result){
    assert.equal(result.toNumber(), 500000, 'accounts[0]
balance is wrong');
    return token.balanceOf.call(accounts[1]);
  }).then(function(result){
    assert.equal(result.toNumber(), 500000, 'accounts[1]
balance is wrong');
  })
});
```

transferFrom

```
it("should give accounts[1] authority to spend account[0]'s
token", function() {
  var token;
  return Token.deployed().then(function(instance){
   token = instance;
   return token.approve(accounts[1], 200000);
  }).then(function(){
   return token.allowance.call(accounts[0], accounts[1]);
  }).then(function(result){
   assert.equal(result.toNumber(), 200000, 'allowance is
wrong');
   return token.transferFrom(accounts[0], accounts[2], 200000,
{from: accounts[1]});
  }).then(function(){
   return token.balanceOf.call(accounts[0]);
  }).then(function(result){
   assert.equal(result.toNumber(), 300000, 'accounts[0] balance
is wrong');
   return token.balanceOf.call(accounts[1]);
  }).then(function(result){
   assert.equal(result.toNumber(), 500000, 'accounts[1] balance
is wrong');
   return token.balanceOf.call(accounts[2]);
  }).then(function(result){
   assert.equal(result.toNumber(), 200000, 'accounts[2] balance
is wrong');
  })
});
```

Transfer Event

```
it("should show the transfer event", function() {
  var token;
  return Token.deployed().then(function(instance){
    token = instance;
    return token.transfer(accounts[1], 100000);
  }).then(function(result){
    console.log(result.logs[0].event)
  })
});
```

8. TRUFFLE BOXES

Truffle Boxes http://truffleframework.com/boxes/ 是一些有用的实例，学习 truffle 开发是供开发者参考学习。

使用下面命令下载例子

```
truffle unbox webpack
```

9. Zeppelin Solidity - OpenZeppelin is a library for writing secure Smart Contracts on Ethereum.

OpenZeppelin is an open framework of reusable and secure smart contracts in the Solidity language.

网站： https://openzeppelin.org

Github: https://github.com/OpenZeppelin/zeppelin-solidity

9.1. ERC20

```
pragma solidity ^0.4.19;
import "zeppelin-solidity/contracts/token/StandardToken.sol";

contract NeoCoin is StandardToken {
  string public name = "NeoCoin";
  string public symbol = "BLC";
  uint8 public decimals = 4;
  uint256 public INITIAL_SUPPLY = 666666;
  function NeoCoin() {
    totalSupply = INITIAL_SUPPLY;
    balances[msg.sender] = INITIAL_SUPPLY;
  }
}
```

9.2. ERC872

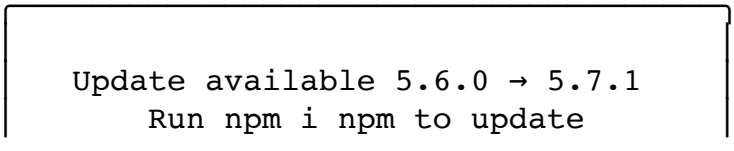
创建项目目录

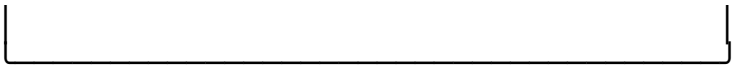
```
neo@MacBook-Pro ~/ethereum/truffle % mkdir TokenERC827
neo@MacBook-Pro ~/ethereum/truffle % cd TokenERC827
neo@MacBook-Pro ~/ethereum/truffle/TokenERC827 % truffle init
Downloading...
```

```
Unpacking...
Setting up...
Unbox successful. Sweet!

Commands:

  Compile:        truffle compile
  Migrate:        truffle migrate
  Test contracts: truffle test
```

安装 zeppelin-solidity

```
neo@MacBook-Pro ~/ethereum/truffle/TokenERC827 % npm init -y
Wrote to /Users/neo/ethereum/truffle/TokenERC827/package.json:

{
  "name": "TokenERC827",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

neo@MacBook-Pro ~/ethereum/truffle/TokenERC827 % npm install -E
zeppelin-solidity
npm notice created a lockfile as package-lock.json. You should
commit this file.
npm WARN TokenERC827@1.0.0 No description
npm WARN TokenERC827@1.0.0 No repository field.

+ zeppelin-solidity@1.7.0
added 8 packages in 2.591s
```





合约被安装在 node_modules/zeppelin-solidity/contracts 目录

创建合约和测试文件

```
neo@MacBook-Pro ~/ethereum/truffle/TokenERC827 % truffle create
contract TokenERC827
neo@MacBook-Pro ~/ethereum/truffle/TokenERC827 % truffle create
test TokenERC827
```

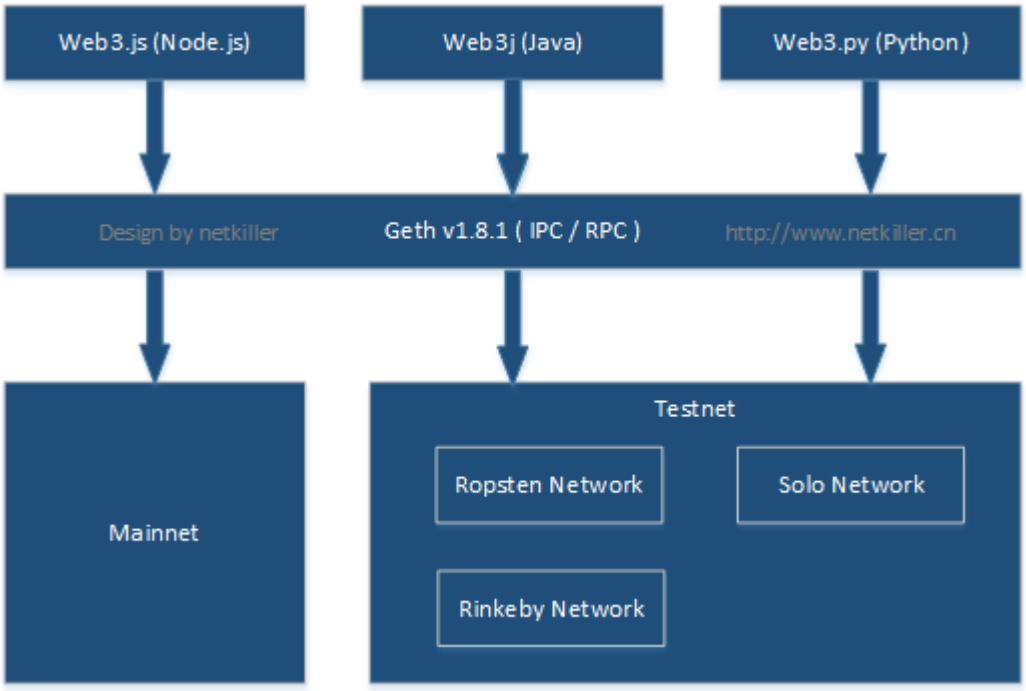
编辑合约文件

```
pragma solidity ^0.4.19;

import "zeppelin-
solidity/contracts/token/ERC827/ERC827Token.sol";

contract TokenERC827 is ERC827Token {

  string public name = "NetkillerCoin";
  string public symbol = "NKC";
  uint8 public decimals = 4;
  uint256 public INITIAL_SUPPLY = 1000000;

  function TokenERC827() public {
    // constructor
    totalSupply = INITIAL_SUPPLY;
    balances[msg.sender] = INITIAL_SUPPLY;
  }
}
```

第 16 章 web3.js - 1.0.0

注意本章采用 web3.js v1.0.0 版本，这个版本仍在beta阶段，还没有release。

你看到网上很多实例，安照例子的步骤操作，发现无法成功，很可能跟版本有关。v1.0.0 与之前的版本还是有很大差异，所以选择版本很重要。因为 v1.0.0 很快就会 release 所以本章不会在用早起版本举例。



1. 开发环境

```
npm init
npm install web3 --save
npm install solc
```

1.1. Ropsten 测试网

```
geth --testnet --syncmode light --cache 1024 --ipcpath
/Users/neo/Library/Ethereum/geth.ipc
```

Ropsten 测试网上转账的例子

```
fs = require('fs');
var net = require('net');
var Web3 = require('web3');
var web3 = new Web3('/Users/neo/Library/Ethereum/geth.ipc',
net);

console.log(web3.version)
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x70682386d0dE84B1e549DC3c4305CCB2D261b2a8";
const coinbase = "0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7";
const toAddress = "0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";

balanceWei = web3.eth.getBalance(coinbase);
console.log(balanceWei);

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});

web3.eth.personal.unlockAccount(coinbase,
"netkiller").then(function(result){
        console.log(result)

contract.methods.balanceOf(coinbase).call().then(console.log).c
atch(console.error);

contract.methods.balanceOf(toAddress).call().then(console.log).
catch(console.error);
});


contract.methods.transfer(toAddress,
```

```
10000).send().then(function(receipt){

        console.log(receipt);

}).catch(console.error);

contract.methods.balanceOf(coinbase).call().then(console.log).c
atch(console.error);
contract.methods.balanceOf(toAddress).call().then(console.log).
catch(console.error);
```

2. truffle-contract

```
neo@MacBook-Pro ~/ethereum/web3 % npm install truffle-contract
```

3. 连接到以太坊客户端

3.1. http 方式

```
var Web3 = require('web3');
var web3 = new Web3('http://localhost:8545');
```

查看连接状态

```
> web3.currentProvider
HttpProvider {
  host: 'http://localhost:8545',
  timeout: 0,
  connected: true,
  headers: undefined }
```

connected: true 表示连接成功。

3.2. WebSocket 方式

```
var Web3 = require('web3');
var web3 = new Web3(Web3.givenProvider ||
'ws://remotenode.com:8546');
```

3.3. IPC 方式

```
// Using the IPC provider in node.js
```

```
var net = require('net');
var Web3 = require('web3');
var web3 = new Web3('/Users/myuser/Library/Ethereum/geth.ipc',
net); // mac os path
```

4. web3

4.1. version 显示web3版本号

```
> web3.version
'1.0.0-beta.30'
```

5. web3.eth

5.1. 查看账号列表

```
var Web3 = require('web3');
var web3 = new Web3('http://localhost:8545');

web3.eth.getAccounts().then(console.log);
```

5.2. 查询矿工账号

```
var Web3 = require('web3');
var web3 = new Web3('http://localhost:8545');

web3.eth.getCoinbase().then(console.log);
```

Callback 方式

```
web3.eth.getCoinbase(
    function(error, result){
    if (error) {
        console.error(error);
    } else {
        console.log(result);
    }
 })
```

5.3. 获得余额

```
web3.eth.getBalance(req.query.address).then(function(balance){
      res.json({"status": true, "code":0, "data":
{"account":req.query.address, "balance":
web3.utils.fromWei(balance)}});
});
```

Callback 方式

```
web3.eth.getBalance(req.query.address, function (error, wei) {
    if (!error) {
        var balance = web3.utils.fromWei(wei, 'ether');
        res.json({"status": true, "code":0, "data":
{"account":req.query.address, "balance": balance}});
    }else{
        console.log(error);
        res.json({"status": false, "code":1, "data":
{"error":error.message}});
    }
});
```

捕捉错误

```
router.get('/balance.json', function(req, res) {

        try {
                web3.eth.getBalance(req.query.address, function
(error, wei) {
                    if (!error) {
                        var balance = web3.utils.fromWei(wei,
'ether');
                        res.json({"status": true, "code":0,
"data":{"account":req.query.address, "balance": balance}});
                    }else{
                        console.log(error);
                        res.json({"status": false, "code":1,
"data":{"error":error.message}})
```

```
                }
                 });
        }
        catch(error){
                 res.json({"status": false, "code":1, "data":
{"error":error.message}});
        };

});
```

5.4. web3.eth.sendTransaction()

```
web3.eth.sendTransaction({
    from: coinbase,
    to: '0x2C687bfF93677D69bd20808a36E4BC2999B4767C',
    value: web3.utils.toWei('2','ether')
},
function(error, result){
    if(!error) {
        console.log("#" + result + "#")
    } else {
        console.error(error);
    }
});

var code =
"0x603d80600c6000396000f3007c0100000000000000000000000000000000
00000000000000000000000006000350463c6888fa18114602d57005b600760043
502806000526020 6000f3";

web3.eth.sendTransaction({from: coinbase, data: code},
function(err, transactionHash) {
  if (!err)
    console.log(transactionHash); //
"0x7f9fade1c0d57a7af66ab4ead7c2eb7b11a91385"
});

web3.eth.sendTransaction({from: coinbase, data:
code}).then(function(receipt){
        console.log(receipt);
});
```

5.5. web3.eth.sendSignedTransaction() 私钥签名转账

5.5.1. 例子1

```
var account = web3.eth.accounts.privateKeyToAccount(privateKey);

web3.eth.accounts.signTransaction({
  from: account.address,
  to: "0x0013a861865d74b13ba94713d4e84d97c57e7081",
  gas: "3000000",
  value: '100000000000000000',
  gasPrice: '0x09184e72a000',
  data: "0x00"
}, account.privateKey)
.then(function(result) {
  console.log("Results: ", result)

  web3.eth.sendSignedTransaction(result.rawTransaction)
    .on('receipt', console.log);
})
```

5.5.2. 例子2

获取 pending 状态的区块

```
[ethereum@netkiller web3.example]$ vim test.js
[ethereum@netkiller web3.example]$ export
PRIVATE_KEY=585a219fd6a5583b325e96770a88e69660f404efc06e56be71d8
2beedb7a989e
[ethereum@netkiller web3.example]$ echo $PRIVATE_KEY
585a219fd6a5583b325e96770a88e69660f404efc06e56be71d82beedb7a989e
[ethereum@netkiller web3.example]$ node
> process.env["PRIVATE_KEY"]
'585a219fd6a5583b325e96770a88e69660f404efc06e56be71d82beedb7a989
e'
```

```
[ethereum@netkiller web3.example]$ cat transfer.js
fs = require('fs');
const Web3 = require('web3');
var Tx = require('ethereumjs-tx');
const web3 = new Web3('http://localhost:8545');
console.log(web3.version)

coinbase        = "0xaa96686a050e4916afbe9f6d8c5107062fa646dd";
address         = "0x372fda02e8a1eca513f2ee5901dc55b8b5dd7411"
contractAddress = "0x9ABcF16f6685fE1F79168534b1D30056c90B8A8A"

const main = async () => {
        var balance = await web3.eth.getBalance(coinbase);
        console.log(`Balance ETH: ${balance} \n`);

        const abi = fs.readFileSync('output/NetkillerToken.abi',
'utf-8');
        const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: address});

        var balance = await
contract.methods.balanceOf(address).call();
        console.log(`Balance before send: ${balance} \n`);

        var count = await
web3.eth.getTransactionCount(coinbase);
        const gasPrice = await web3.eth.getGasPrice();
        console.log(`gasPrice: ${gasPrice}\n`)
        var gasLimit = 1000000;
        var transferAmount = 1000;
    // Chain ID of Ropsten Test Net is 3, replace it to 1 for
Main Net
    var chainId = 1;

    var rawTransaction = {
        "from": coinbase,
        /* "nonce": "0x" + count.toString(16),*/
        "nonce":  web3.utils.toHex(count),
        "gasPrice": web3.utils.toHex(gasPrice),
        "gasLimit": web3.utils.toHex(gasLimit),
        "to": contractAddress,
        "value": "0x0",
        "data": contract.methods.transfer(address,
transferAmount).encodeABI(),
        "chainId": web3.utils.toHex(chainId)
    };
```

```
    console.log(`Raw of Transaction:
\n${JSON.stringify(rawTransaction, null, '\t')}\n`);

    // The private key for myAddress in .env
    var privateKey = new Buffer(process.env["PRIVATE_KEY"],
'hex');
    var tx = new Tx(rawTransaction);
    tx.sign(privateKey);
    var serializedTx = tx.serialize();

    // Comment out these four lines if you don't really want to
send the TX right now
    console.log(`Attempting to send signed tx:
${serializedTx.toString('hex')}\n`);

    var receipt = await web3.eth.sendSignedTransaction('0x' +
serializedTx.toString('hex'));

    // The receipt info of transaction, Uncomment for debug
    console.log(`Receipt info: \n${JSON.stringify(receipt, null,
'\t')}\n`);

    // The balance may not be updated yet, but let's check
        var balance = await
contract.methods.balanceOf(address).call();
        console.log(`Balance after send: ${balance}`);
}

main();
```

```
[ethereum@netkiller web3.example]$ node test.js
1.0.0-beta.34
Balance ETH: 8695480352861952

Balance before send: 100

gasPrice: 3000000000

Raw of Transaction:
{
        "from": "0xaa96686a050e4916afbe9f6d8c5107062fa646dd",
        "nonce": "0x20",
        "gasPrice": "0xb2d05e00",
```

```
        "gasLimit": "0xf4240",
        "to": "0x9ABcF16f6685fE1F79168534b1D30056c90B8A8A",
        "value": "0x0",
        "data":
"0xa9059cbb000000000000000000000000372fda02e8a1eca513f2ee5901dc5
5b8b5dd74110000000000000000000000000000000000000000000000000000
000000003e8",
        "chainId": "0x1"
}

Attempting to send signed tx:
f8a92084b2d05e00830f4240949abcf16f6685fe1f79168534b1d30056c90b8a
8a80b844a9059cbb000000000000000000000000372fda02e8a1eca513f2ee59
01dc55b8b5dd74110000000000000000000000000000000000000000000000000
00000000000003e825a05017058348d8f751cc40e71c13a63b8d8e21683841bd
002c9f6bf691d34b6a4ba07df3fa8792aa66cc272ff89373509f6335272a4298
e98177536a79c8ab7c947c

Receipt info:
{
        "blockHash":
"0x32f45f27040b1df616ff4efd25557416793782541b995a6d4ecbd66f84417
83f",
        "blockNumber": 5518045,
        "contractAddress": null,
        "cumulativeGasUsed": 5054117,
        "from": "0xaa96686a050e4916afbe9f6d8c5107062fa646dd",
        "gasUsed": 36184,
        "logs": [
                {
                        "address":
"0x9ABcF16f6685fE1F79168534b1D30056c90B8A8A",
                        "topics": [

"0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b
3ef",

"0x000000000000000000000000aa96686a050e4916afbe9f6d8c5107062fa64
6dd",

"0x000000000000000000000000372fda02e8a1eca513f2ee5901dc55b8b5dd7
411"
                        ],
                        "data":
"0x00000000000000000000000000000000000000000000000000000000000
3e8",
                        "blockNumber": 5518045,
                        "transactionHash":
```

```
"0x203ebcbe1cc8340c4b5a13f4e6c36a4f63142754437e0f43b0ff5e5c0bf51
2cc",
                        "transactionIndex": 104,
                        "blockHash":
"0x32f45f27040b1df616ff4efd25557416793782541b995a6d4ecbd66f84417
83f",
                        "logIndex": 94,
                        "removed": false,
                        "id": "log_7ef3f104"
                }
        ],
        "logsBloom":
"0x00000000000000000000000000000000000000000000000000000000000
00000000000400000000000000000000000000000000100000000400000000
00000000000000000000000800000000000000000000000000000000000000
00000000400000000000000000000000000000000000000001000000
00000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000
02000000020000000000000000000000000000120000000000000000000
02000000000000000000000000001000000000000000000000000000000
000",
        "status": true,
        "to": "0x9abcf16f6685fe1f79168534b1d30056c90b8a8a",
        "transactionHash":
"0x203ebcbe1cc8340c4b5a13f4e6c36a4f63142754437e0f43b0ff5e5c0bf51
2cc",
        "transactionIndex": 104
}

Balance after send: 1100
```

5.6. web3.eth.getBlock() 获取区块

获取 pending 状态的区块

```
web3.eth.getBlock(
    "pending",
function (error, block) {
    if (error) {
        console.error(error);
    } else {
        console.log(block.transactions.length);
```

```
    }
})
```

6. 账号管理

6.1. web3.eth.personal.unlockAccount()

```
var coinbase = "0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7";
//console.log(coinbase)
web3.eth.personal.unlockAccount(coinbase, "your
password").then(console.log);
```

7. 智能合约

7.1. 部署合约

解锁账号

```
> personal.unlockAccount(eth.accounts[0], "netkiller",5000)
true
> miner.stop()
true
> miner.start(2)
```

编译只能合约

```
solc --bin --abi --optimize -o ./output helloworld.sol
```

```
var Web3 = require("web3");
var fs = require ('fs');

var web3 = new Web3("http://localhost:8545");

var abi = JSON.parse(fs.readFileSync('output/Netkiller.abi'));
var bin =
'0x'+fs.readFileSync('output/Netkiller.bin').toString();

var myContract = new web3.eth.Contract(abi, null, {
    from: '0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7'
});

myContract.deploy({ data: bin }).send({
    from: '0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7',
    gas: 1500000,
    gasPrice: '30000000000000'
```

```
}).then(function(newContractInstance){
    console.log(newContractInstance.options.address)
});
```

7.2. 使用最佳手续费创建合约

```
var fs = require('fs');
var net = require('net');
var Web3 = require("web3");
var fs = require ('fs');

var web3 = new Web3("http://localhost:8545");

var abi =
JSON.parse(fs.readFileSync('NetkillerAdvancedToken.abi'));
var bin =
'0x'+fs.readFileSync('NetkillerAdvancedToken.bin').toString();
var address = '0x22c57F0537414FD95b9f0f08f1E51d8b96F14029';
var myContract = new web3.eth.Contract(abi, null, {from:
address});

var options = { data: bin, arguments: [100000000, 'Netkiller
Test Coin', 'NTC', 18] };

web3.eth.getGasPrice().then(function(gasPrice){
    myContract.deploy(options).estimateGas(function(err, gas){
        console.log(gas);
        web3.eth.personal.unlockAccount(req.body.from,
req.body.password).then(function(error){
            myContract.deploy(options).send({
                from: address,
                gas: gas,
                gasPrice: gasPrice
            }).then(function(newContractInstance){
                console.log(newContractInstance.options.address)
            });
        });
    });
});
```

7.3. 调用合约

部署智能合约，你可以使用钱包部署，也可以使用Truffer部署，不管你采用什么方式，最终我们需要合约地址。

```
pragma solidity ^0.4.18;

contract MetaCoin {
        mapping (address => uint) balances;

        event Transfer(address indexed _from, address indexed
_to, uint256 _value);

        function MetaCoin() public {
                balances[tx.origin] = 10000;
        }

        function sendCoin(address receiver, uint amount) public
returns(bool sufficient) {
                if (balances[msg.sender] < amount) return false;
                balances[msg.sender] -= amount;
                balances[receiver] += amount;
                Transfer(msg.sender, receiver, amount);
                return true;
        }

        function getBalanceInEth(address addr) public view
returns(uint){
                return convert(getBalance(addr),2);
        }

        function getBalance(address addr) public view
returns(uint) {
                return balances[addr];
        }

        function convert(uint amount,uint conversionRate) public
pure returns (uint convertedAmount)
        {
                return amount * conversionRate;
        }

}
```

这里使用 Truffle 部署

```
neo@MacBook-Pro ~/ethereum/truffle % truffle compile --all
Compiling ./contracts/MetaCoin.sol...
Writing artifacts to ./build/contracts

neo@MacBook-Pro ~/ethereum/truffle % truffle migrate --reset
Using network 'development'.

Running migration: 5_initial_migration.js
  Deploying MetaCoin...
  ...
0x9c006b398733a1d8679cbb00493ca75ff063f51c34521ae67a70523deeebf9
c4
  MetaCoin: 0xfb88de099e13c3ed21f80a7a1e49f8caecf10df6
Saving successful migration to network...
  ...
0x755a48ef99e488d7cf8460d718773a5afe73f760fb87697e51c40f3e6086f1
0b
Saving artifacts...
```

得到合约地址 MetaCoin: 0xfb88de099e13c3ed21f80a7a1e49f8caecf10df6

编译合约获得 abi 接口

```
neo@MacBook-Pro ~/ethereum/truffle % solc --bin --abi --optimize
-o ./output contracts/MetaCoin.sol
neo@MacBook-Pro ~/ethereum/truffle % cat output/MetaCoin.abi
[{"constant":true,"inputs":
[{"name":"addr","type":"address"}],"name":"getBalanceInEth","out
puts":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"receiver","type":"address"},
{"name":"amount","type":"uint256"}],"name":"sendCoin","outputs":
[{"name":"sufficient","type":"bool"}],"payable":false,"stateMuta
```

```
bility":"nonpayable","type":"function"},
{"constant":true,"inputs":[{"name":"amount","type":"uint256"},
{"name":"conversionRate","type":"uint256"}],"name":"convert","ou
tputs":
[{"name":"convertedAmount","type":"uint256"}],"payable":false,"s
tateMutability":"pure","type":"function"},
{"constant":true,"inputs":
[{"name":"addr","type":"address"}],"name":"getBalance","outputs"
:
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"inputs":
[],"payable":false,"stateMutability":"nonpayable","type":"constr
uctor"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"_from","type":"address"},
{"indexed":true,"name":"_to","type":"address"},
{"indexed":false,"name":"_value","type":"uint256"}],"name":"Tran
sfer","type":"event"}]
```


```
var Web3 = require("web3");
// 创建web3对象
var web3 = new Web3("http://localhost:9545");
// 合约ABI
```


```
var abi = [{"constant":true,"inputs":
[{"name":"addr","type":"address"}],"name":"getBalanceInEth","out
puts":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"receiver","type":"address"},
{"name":"amount","type":"uint256"}],"name":"sendCoin","outputs":
[{"name":"sufficient","type":"bool"}],"payable":false,"stateMuta
bility":"nonpayable","type":"function"},
{"constant":true,"inputs":[{"name":"amount","type":"uint256"},
{"name":"conversionRate","type":"uint256"}],"name":"convert","ou
tputs":
[{"name":"convertedAmount","type":"uint256"}],"payable":false,"s
tateMutability":"pure","type":"function"},
{"constant":true,"inputs":
[{"name":"addr","type":"address"}],"name":"getBalance","outputs"
:
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"inputs":
[],"payable":false,"stateMutability":"nonpayable","type":"constr
uctor"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"_from","type":"address"},
```

```
{"indexed":true,"name":"_to","type":"address"},
{"indexed":false,"name":"_value","type":"uint256"}],"name":"Tran
sfer","type":"event"}];
// 合约地址
var address = "0xfb88de099e13c3ed21f80a7a1e49f8caecf10df6";
var tokenContract = new web3.eth.Contract(abi, address);
// 调用函数
tokenContract.methods.getBalance("0x627306090abab3a6e1400e9345bc
60c78a8bef57").call(null,function(error,result){
        console.log("getBalance "+result);
});
tokenContract.methods.getBalanceInEth("0x627306090abab3a6e1400e9
345bc60c78a8bef57").call(null,function(error,result){
        console.log("getBalanceInEth "+result);
});
tokenContract.methods.getBalance("0xf17f52151ebef6c7334fad080c57
04d77216b732").call(null,function(error,result){
        console.log("getBalance 2 "+result);
});
tokenContract.methods.sendCoin("0xf17f52151ebef6c7334fad080c5704
d77216b732",387).send({from:
'0x627306090abab3a6e1400e9345bc60c78a8bef57'})
.on('transactionHash', function(hash){
}).on('confirmation', function(confirmationNumber, receipt){
}).on('receipt', function(receipt){
    // receipt example
    console.log(receipt); //查询这里可以得到结果
}).on('error', console.error); // If a out of gas error, the
second parameter is the receipt.
```

运行结果

```
getBalance 10000
getBalanceInEth 20000
getBalance 2 0
{ transactionHash:
'0x0f7514413865219d70873634c00d1b4746c5faa436283786f5414b483b6d6
333',
  transactionIndex: 0,
  blockHash:
'0x5a7662d14f78b6b4d64b05c0ec1e1e641bd39440467f0476409f4e49c21f2
87a',
  blockNumber: 11,
```

```
  gasUsed: 51024,
  cumulativeGasUsed: 51024,
  contractAddress: null,
  status: 1,
  events:
   { Transfer:
      { logIndex: 0,
        transactionIndex: 0,
        transactionHash:
'0x0f7514413865219d70873634c00d1b4746c5faa436283786f5414b483b6d6
333',
        blockHash:
'0x5a7662d14f78b6b4d64b05c0ec1e1e641bd39440467f0476409f4e49c21f2
87a',
        blockNumber: 11,
        address: '0xFB88dE099e13c3ED21F80a7a1E49f8CAEcF10df6',
        type: 'mined',
        id: 'log_8b567824',
        returnValues: [Result],
        event: 'Transfer',
        signature:
'0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b
3ef',
        raw: [Object] } } }
```

再次运行

```
getBalance 9613
getBalanceInEth 19226
getBalance 2 387
{ transactionHash:
'0x6d33487d9067494288190f7a64b3118c20560479e01afe8d53e37ca46a600
1e4',
  transactionIndex: 0,
  blockHash:
'0x28f6d416c68dbbd698d1a0b0fdc4ce842694d82318a38ea6395413fd15397
642',
  blockNumber: 12,
  gasUsed: 36024,
  cumulativeGasUsed: 36024,
  contractAddress: null,
  status: 1,
  events:
```

```
   { Transfer:
      { logIndex: 0,
        transactionIndex: 0,
        transactionHash:
'0x6d33487d9067494288190f7a64b3118c20560479e01afe8d53e37ca46a600
1e4',
        blockHash:
'0x28f6d416c68dbbd698d1a0b0fdc4ce842694d82318a38ea6395413fd15397
642',
        blockNumber: 12,
        address: '0xFB88dE099e13c3ED21F80a7a1E49f8CAEcF10df6',
        type: 'mined',
        id: 'log_e5cfee46',
        returnValues: [Result],
        event: 'Transfer',
        signature:
'0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b
3ef',
        raw: [Object] } } }
```

第二次运行会打印出账号2的余额

代码优化，由于开发阶段 abi 接口经常变化，从 solc 编译的abi 文件中直接读取最为方便。

```
const fs = require ('fs');
var Web3 = require("web3");
var web3 = new Web3("http://localhost:9545");
var abi = JSON.parse(fs.readFileSync('MetaCoin.abi'));
var address = "0xfb88de099e13c3ed21f80a7a1e49f8caecf10df6";
var tokenContract = new web3.eth.Contract(abi, address);
tokenContract.methods.getBalance("0x627306090abab3a6e1400e9345bc
60c78a8bef57").call(null,function(error,result){
        console.log("getBalance "+result);
});
```

7.4. event

下面以ERC20代币为例演示事件如何工作

```
neo@MacBook-Pro ~/ethereum/web3 % cat event.js

fs = require('fs');
var net = require('net');
var Web3 = require('web3');
var web3 = new Web3('/Users/neo/Library/Ethereum/geth.ipc',
net);

console.log(web3.version)
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x70682386d0dE84B1e549DC3c4305CCB2D261b2a8";
const coinbase = "0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7";
const toAddress = "0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";

balanceWei = web3.eth.getBalance(coinbase);
console.log(balanceWei);

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});

contract.events.Transfer({
    fromBlock: 0,
    toBlock:'latest'
}, function(error, event){ console.log(event); })
.on('data', function(event){
    console.log(event); // same results as the optional callback
above
})
.on('changed', function(event){
    // remove event from local database
})
.on('error', console.error);
```

运行后程序不会退出，会源源不断的打印出每笔交易

```
neo@MacBook-Pro ~/ethereum/web3 % node event.js
```

```
{ address: '0x70682386d0dE84B1e549DC3c4305CCB2D261b2a8',
  blockNumber: 0,
  transactionHash:
'0x0000000000000000000000000000000000000000000000000000000000000000',
  transactionIndex: 0,
  blockHash:
'0x0000000000000000000000000000000000000000000000000000000000000000',
  logIndex: 0,
  removed: false,
  id: 'log_247b1779',
  returnValues:
   Result {
     '0': '0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7',
     '1': '0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA',
     '2': '10000000',
     from: '0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7',
     to: '0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA',
     value: '10000000' },
  event: 'Transfer',
  signature:
'0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef',
  raw:
   { data:
'0x0000000000000000000000000000000000000000000000000000000000989680',
     topics:
      [
'0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef',

'0x000000000000000000000000b94054c174995ae2a9e7fcf6c7924635fba8ecf7',

'0x000000000000000000000000f56b81a2bcb964d2806071e9be4289a5559bb0fa' ] } }
```

8.

8.1.

```
const Web3 = require('web3');

const web3 = new Web3(new
Web3.providers.WebsocketProvider('wss://mainnet.infura.io/ws'))
;

var subscription = web3.eth.subscribe('pendingTransactions',
function(error, result){
    if (!error)
        console.log(result);
})
.on("data", function(transaction){
    console.log(transaction);
});

// unsubscribes the subscription
subscription.unsubscribe(function(error, success){
    if(success)
        console.log('Successfully unsubscribed!');
});
```

```
const Web3 = require('web3');

const web3wss = new Web3(new
Web3.providers.WebsocketProvider('wss://mainnet.infura.io/ws'))
;
const web3 = new
Web3("https://mainnet.infura.io/CsS9shwaAab0z7B4LP2d");


var subscription = web3wss.eth.subscribe('pendingTransactions',
function(error, result){
    if (!error){
```

```
        console.log("--------------------" + result +"---------
----------");
        var receipt =
web3.eth.getTransactionReceipt(result).then(console.log);
     }
})
.on("data", function(transaction){
    console.log(transaction);
});

// unsubscribes the subscription
subscription.unsubscribe(function(error, success){
    if(success)
        console.log('Successfully unsubscribed!');
});
```

8.2. 订阅 newBlockHeaders

```
#!/usr/bin/env node
const Web3 = require('web3');

const web3 = new Web3(new
Web3.providers.WebsocketProvider('wss://mainnet.infura.io/ws'))
;

const subscription = web3.eth.subscribe('newBlockHeaders',
(error, blockHeader) => {
  if (error) return console.error(error);

  console.log('Successfully subscribed!', blockHeader);
}).on('data', (blockHeader) => {
  console.log('data: ', blockHeader);
});

// unsubscribes the subscription
subscription.unsubscribe((error, success) => {
  if (error) return console.error(error);

  console.log('Successfully unsubscribed!');
});
```

8.3. 订阅 log

```
#!/usr/bin/env node
const Web3 = require('web3');

const web3 = new Web3(new
Web3.providers.WebsocketProvider('wss://mainnet.infura.io/ws'))
;

var subscription = web3.eth.subscribe('logs', {
        fromBlock: 5709426,
    address: '0x7fFdCccC3E7e33C6163393195A947A6d45f25814'
}, function(error, result){
    if (!error)
        console.log(result);
})
.on("data", function(log){
    console.log(log);
})
.on("changed", function(log){
});

// unsubscribes the subscription
subscription.unsubscribe(function(error, success){
    if(success)
        console.log('Successfully unsubscribed!');
});
```

8.4. 订阅同步状态

```
const Web3 = require('web3');

const web3 = new Web3(new
Web3.providers.WebsocketProvider('wss://mainnet.infura.io/ws'))
;

var subscription = web3.eth.subscribe('syncing',
```

```
function(error, sync){
    if (!error)
        console.log(sync);
})
.on("data", function(sync){
    // show some syncing stats
})
.on("changed", function(isSyncing){
    if(isSyncing) {
        // stop app operation
    } else {
        // regain app operation
    }
});

// unsubscribes the subscription
subscription.unsubscribe(function(error, success){
    if(success)
        console.log('Successfully unsubscribed!');
});
```

9. utils

9.1. web3.utils.toWei()

```
web3.utils.toWei('1', 'ether');
> "1000000000000000000"

web3.utils.toWei('1', 'finney');
> "1000000000000000"

web3.utils.toWei('1', 'szabo');
> "1000000000000"

web3.utils.toWei('1', 'shannon');
> "1000000000"
```

9.2. 将 Wei 转换到指定单位

```
web3.utils.fromWei('1', 'ether');
> "0.000000000000000001"

web3.utils.fromWei('1', 'finney');
> "0.000000000000001"

web3.utils.fromWei('1', 'szabo');
> "0.000000000001"

web3.utils.fromWei('1', 'shannon');
> "0.000000001"
```

10. web3 编译合约

10.1. solc.compile

```
pragma solidity ^0.4.0;
contract HelloWorldContract {
  function sayHi() constant returns (string){
    return 'Hello World';
  }
}
```

```
const fs = require ('fs');
const solc = require ('solc');
const input = fs.readFileSync('HelloWorldContract.sol');
const output = solc.compile(input.toString(), 1);
for (var contractName in output.contracts){
 console.log(contractName + ': ' +
output.contracts[contractName].bytecode)
}
```

```
> const fs = require ('fs');
undefined
> const solc = require ('solc');

undefined
> const input = fs.readFileSync('HelloWorldContract.sol');
undefined
> const output = solc.compile(input.toString(), 1);

undefined
> for (var contractName in output.contracts){
...  console.log(contractName + ': ' +
output.contracts[contractName].bytecode)
... }
:HelloWorldContract:
```

```
6060604052341561000f57600080fd5b61014e8061001e6000396000f300606
060405260043610610040576fffffffff7c010000000000000000000000000000
000000000000000000000000000000000000600350416630c49c36c8114610045575
b600080fd5b341561005057600080fd5b6100586100cf565b60405160208082
528190810183818151815260200191508051906020019080838360005b83811
015610094578082015183820152602001610078565b50505050905090810190
601f1680156100c1578082038051600183602003610100a031916815260200
191505b509250505060405180910390f35b6100d7610110565b604080519081
01604052600b81527f48656c6c6f20576f726c640000000000000000000000000000
0000000000000000000602082015290509056 5b6020604051908101604052600
08152905600a165627a7a723058206dc75b1be2caa91f056c3682d24390fcb
090a96bbf1b2b4c11cdd807518dc03c0029
undefined
```

11. web3admin

https://github.com/DecentricCorp/web3admin

```
npm install web3admin
```

```
const Web3 = require('web3')
var web3 = new Web3('http://localhost:8545')
const web3Admin = require('web3admin')
setTimeout(function(){
    web3Admin.extend(web3)
    console.log("turning on mining", web3.miner.start())
    console.log("isMining?", web3.eth.mining)
    console.log("isMining?", web3.eth.mining)
}, 1000)
```

12. ABI-encoded

ABI-encoded 实际上就是构造方法变量数据

```
var abi = require('ethereumjs-abi')

var parameterTypes = ["address", "uint256", "bool"];
var parameterValues = ["0x1234567812345678", "0x314159268",
true];

var encoded = abi.rawEncode(parameterTypes, parameterValues);

console.log(encoded.toString('hex'));
```

在线生成工具 https://abi.sonnguyen.ws/， https://abi.hashex.org/

还有一种方法可以找到 ABI 数据
https://etherscan.io/tx/0xc53f52b287ec1c71e9d203dd08cc68035ef40bf5155f8b0711da227c01d84d70 查看 Input Data: 0029之后所有的数据就是 ABI

13. 实用例子

13.1. 数据写入到区块链中

做一笔交易，并写入数据到区块链中

```
let Web3 = require("web3");
let fs = require("fs");
let web3 = new Web3("http://localhost:8545");

let log = {
    time:(new Date).getTime(),
    type:"info",
    msg:"Web3 Test!!!"
};
let str = JSON.stringify(log);
console.log(str);
let data = Buffer.from(str).toString('hex');
data = '0x'+data;
console.log(data);

//将数据写入到交易中
let coinbase = "0x5c18a33df2cc41a1beddc91133b8422e89f041b7"
console.log(coinbase)
let user1 = "0xc2b9e316f246d35052118e51b55c75bfe99d247e";
web3.eth.personal.unlockAccount(coinbase, "coinbase");
let address = web3.eth.sendTransaction({
    from:coinbase,
    to:user1,
    value:'0x00',
    data:data
},function(error, hash){
    console.log(hash);
});
```

运行上面程序，会产生一个交易，记下hash值，然后启动挖矿。

```
> miner.start();
null
# 过一段时间后停止
> miner.stop();
true
```

然后查看这比交易

```
let Web3 = require("web3");
let fs = require("fs");
let web3 = new Web3("http://localhost:8545");
let address
="0xb15681eb4bdb6b9670d305fb341ebbc95d45c2ede0ea5034ef432b74f30
b1b4f";
//从交易地址获取数据
web3.eth.getTransaction(address).then(console.log);

web3.eth.getTransaction(address,function(error, result){
    //console.log(result);
    inputData = result.input;
    res_str =
Buffer.from(inputData.replace('0x',''),'hex').toString();
    res_json = JSON.parse(res_str);
    console.log(res_json);
});
```

结果输出

```
{ blockHash:
'0x78dacc2af60900d2e4cae90b71e27446e6e883df36c53f21cbc9e071f7a5
86f4',
  blockNumber: 1258,
  from: '0x5c18a33DF2cc41a1bedDC91133b8422e89f041B7',
  gas: 90000,
  gasPrice: '18000000000',
  hash:
'0xb15681eb4bdb6b9670d305fb341ebbc95d45c2ede0ea5034ef432b74f30b
1b4f',
```

```
  input:
'0x7b2274696d65223a31353138393331343523537372c2274797065223a22
696e666f222c226d7367223a225765623320546573742121227d',
  nonce: 4,
  to: '0xc2b9e316F246d35052118E51B55C75BfE99d247e',
  transactionIndex: 0,
  value: '0',
  v: '0x41',
  r:
'0x7fcd86c7fd975a0e98bd0e61a99da950b0155cd6c4581fefa4defbdcd404
a930',
  s:
'0x16f14ce1fbfadb9d59f343f8ac235cdd73dcedec5db1025ef91206b8bb17
a827' }
{ time: 1518931452577, type: 'info', msg: 'Web3 Test!!!' }
```

{ time: 1518931452577, type: 'info', msg: 'Web3 Test!!!' } 就是保存在区块链中的数据。

13.2. 编译部署智能合约

```
console.log('Setting up...');
const fs = require ('fs');
const solc = require ('solc');
const Web3 = require ('web3');
const web3 = new Web3("http://localhost:8545");
console.log('Reading Contract...');
const input = fs.readFileSync('Netkiller.sol');
console.log('Compiling Contract...');
const output = solc.compile(input.toString(), 1);
//console.log(output);
const bytecode = output.contracts[':Netkiller'].bytecode;
//console.log(bytecode);
const abi = output.contracts[':Netkiller'].interface;
//console.log(abi);
//Contract Object
//const helloWorldContract =
web3.eth.contract(JSON.parse(abi));

var myContract = new web3.eth.Contract(JSON.parse(abi),
'0x5c18a33df2cc41a1beddc91133b8422e89f041b7', {
```

```
    from: '0x5c18a33df2cc41a1beddc91133b8422e89f041b7', //
default from address
    gasPrice: '20000000000' // default gas price in wei, 20
gwei in this case
});

console.log('unlocking Coinbase account');
const password = "chen1980";
web3.eth.personal.unlockAccount("0x5c18a33df2cc41a1beddc91133b8
422e89f041b7", password,100);

console.log("Deploying the contract");
```

13.3. 部署合约

```
var Web3 = require('web3');
var net = require('net');
var web3 = new Web3(new
Web3.providers.IpcProvider("~/netkiller/ethereum/geth.ipc",net)
);

var abi = [...];
var bin = "";

var tokenContract = new web3.eth.Contract(abi, null, {
    from: '0xFB88dE099e13c3ED21F80a7a1E49f8CAEcF10df6' // 目前
web3没有api来解锁账户,只能自己事先解锁
});

tokenContract.deploy({
    data: bin,
    arguments: ['netkiller'] // 这里是构造函数传值，如果构造函数没有
参数，请删除这行。
}).send({
    from: '0xFB88dE099e13c3ED21F80a7a1E49f8CAEcF10df6',
    gas: 1500000,
    gasPrice: '30000000000000'
}, function(error, transactionHash){
    console.log("deploy tx hash:"+transactionHash)
})
.on('error', function(error){ console.error(error) })
.on('transactionHash', function(transactionHash){
```

```
console.log("hash:",transactionHash)})
.on('receipt', function(receipt){
   console.log(receipt.contractAddress) // contains the new
contract address
})
.on('confirmation', function(confirmationNumber, receipt)
{console.log("receipt,",receipt)})
.then(function(newContractInstance){
    console.log(newContractInstance.options.address) //
instance with the new contract address
});
```

13.4. ERC20 Example

通过Web3操作代币转账

```
fs = require('fs');
const Web3 = require('web3');
const web3 = new Web3('http://localhost:8545');
web3.version
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x05A97632C197a0496bc939C4e666c2E03Cb95DD4";
const toAddress = "0x2C687bfF93677D69bd20808a36E4BC2999B4767C";

var coinbase;

web3.eth.getCoinbase().then(function (address){
  coinbase = address;
  console.log(address);
});

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});

contract.methods.balanceOf('0x5c18a33DF2cc41a1bedDC91133b8422e8
9f041B7').call().then(console.log).catch(console.error);
contract.methods.balanceOf('0x2C687bfF93677D69bd20808a36E4BC299
9B4767C').call().then(console.log).catch(console.error);

web3.eth.personal.unlockAccount(coinbase,
```

```
"netkiller").then(console.log);
contract.methods.transfer('0x2C687bfF93677D69bd20808a36E4BC2999
B4767C', 100).send().then(console.log).catch(console.error);

contract.methods.balanceOf('0x2C687bfF93677D69bd20808a36E4BC299
9B4767C').call().then(console.log).catch(console.error);
```

14. HD Wallet(Hierarchical Deterministic wallet)

BIP32 定义 Hierarchical Deterministic wallet (简称 "HD Wallet")，是一个系统可以从单个seed产生树状结构储存多组 keypairs（私钥和公钥）

BIP39 定义钱包助记词和seed生成规则，一般由 12 -24个单字组成，称为 mnemonic。助记词列表，https://github.com/bitcoin/bips/blob/master/bip-0039/english.txt

BIP44 基于 BIP32 的系统，赋予树状结构中的各层特殊的意义。让同一个 seed 可以支援多币种、多帐户等 （btc一般是 m/44'/0'/0'/0，eth一般是 m/44'/60'/0'/0）

```
npm install bip39 ethereumjs-wallet ethereumjs-util --save
```

14.1. 创建项目

导入开发包

```
var bip39 = require('bip39')
var hdkey = require('ethereumjs-wallet/hdkey')
var util = require('ethereumjs-util')
```

生成 mnemonic code

```
var mnemonic = bip39.generateMnemonic()
```

生成 HD Wallet 首先将 mnemonic code 转成 binary二进制的 seed

```
var seed = bip39.mnemonicToSeed(mnemonic)
```

生成 Master Key 地址 "m/44'/60'/0'/0/0" 使用 seed 生成 HD Wallet。

```
var hdwallet = hdkey.fromMasterSeed(seed)
```

从路径 m/44'/60'/0'/0/0 导入 Master Key 并生成 Wallet 中第一个帐户的第一组 keypair。

```
var key1 = hdwallet.derivePath("m/44'/60'/0'/0/0")
```

使用 keypair 中的公钥产生 address。

```
var address1 = util.pubToAddress(key1._hdkey._publicKey, true)
```

获得以太坊钱包地址

```
address1 = util.toChecksumAddress(address1.toString('hex'))
```

操作演示

```
[ethereum@netkiller web3.js]$ node
> var bip39 = require('bip39')
undefined
> var hdkey = require('ethereumjs-wallet/hdkey')
undefined
> var util = require('ethereumjs-util')
undefined
> var mnemonic = bip39.generateMnemonic()
undefined
> mnemonic
'client dune unfair assume level width bind control mad member old crystal'
> var seed = bip39.mnemonicToSeed(mnemonic)
undefined
> seed
<Buffer 51 12 a3 47 f3 bb b9 24 80 ac 05 6c ce 8c 9f dd b2 98 c8 d3 06 8f 4d 0b 6c 90 86 aa d4
b6 41 36 35 5f b4 42 89 b5 e4 6d 43 9b cf 8d 6a d7 9b 45 3e 5a ... >
> var hdwallet = hdkey.fromMasterSeed(seed)
undefined
> hdwallet
EthereumHDKey {
  _hdkey:
   HDKey {
     versions: { private: 76066276, public: 76067358 },
     depth: 0,
     index: 0,
     _privateKey: <Buffer 1c 37 00 1b f7 1d a5 de 3a 8a 4c e8 54 2d 69 78 81 f3 aa a9 d5 3e 64
74 bd ea 76 28 44 07 d3 04>,
     _publicKey: <Buffer 03 77 fc 6b c7 f3 e3 51 01 db 95 0a a9 0f c0 7f 31 40 af 75 f8 7a 4f 5a
c3 4c 93 ac cb 44 a3 20 5f>,
     chainCode: <Buffer 51 fe 32 23 a0 ab aa 10 5d ff 90 28 26 dc fc 86 fc 5f 8c dc 1b b7 39 31
7e 2d b8 a4 77 33 45 3a>,
     _fingerprint: 1056395940,
     parentFingerprint: 0,
     _identifier: <Buffer 3e f7 52 a4 ed 86 00 f7 ac 4d 1a b4 15 1c 0d 87 cd 7d fe de> } }
> var key1 = hdwallet.derivePath("m/44'/60'/0'/0/0")
undefined
> key1
EthereumHDKey {
  _hdkey:
   HDKey {
     versions: { private: 76066276, public: 76067358 },
     depth: 5,
     index: 0,
     _privateKey: <Buffer f5 92 b7 bf 06 ca 9f d7 69 6b a9 5d 6e d8 e3 57 de 6a 23 79 b6 d5 fe
1f fd 53 c6 b4 b0 63 cd 4a>,
     _publicKey: <Buffer 02 99 ff bb ea 3d 80 e1 8c d5 54 a1 6e 6a ca b2 4b 7e 69 3d 1d 2d 8a 68
f8 61 bf 18 dc 4a f8 d0 26>,
     chainCode: <Buffer 5a 9b b2 0e 7a 62 07 b0 82 db e5 5a 1f 17 4b 47 8a 64 cf 40 67 d5 49 09
89 da aa 33 66 00 d7 e6>,
```

```
    _fingerprint: 3510386860,
    parentFingerprint: 1205114865,
    _identifier: <Buffer d1 3c 40 ac 09 92 fc d7 a4 14 8e d8 91 d1 a7 21 55 7e b8 e3> } }
> var address1 = util.pubToAddress(key1._hdkey._publicKey, true)
undefined
> address1
<Buffer 37 2f da 02 e8 a1 ec a5 13 f2 ee 59 01 dc 55 b8 b5 dd 74 11>
> address1.toString('hex')
'372fda02e8a1eca513f2ee5901dc55b8b5dd7411'
```

14.2. 生成第二个钱包

只需递增最后一位数即可 "m/44'/60'/0'/0/1"， "m/44'/60'/0'/0/2"，"m/44'/60'/0'/0/3"

```
var key2 = hdwallet.derivePath("m/44'/60'/0'/0/1")
```

使用 keypair 中的公钥产生 address。

```
var address2 = util.pubToAddress(key2._hdkey._publicKey, true)
```

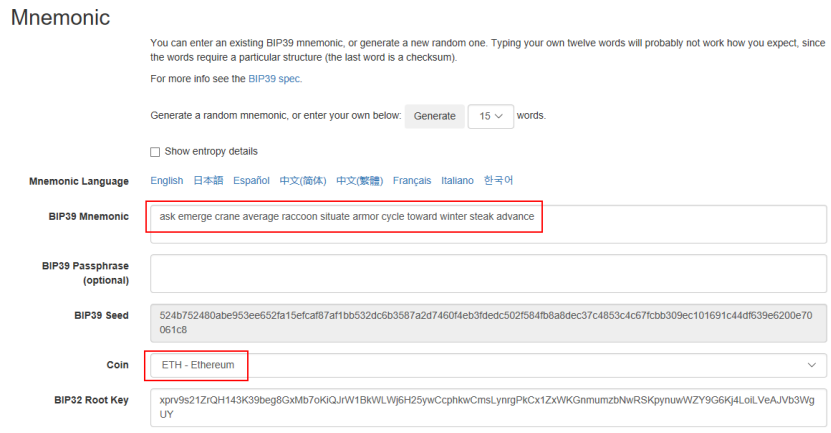
获得以太坊钱包地址

```
address2 = util.toChecksumAddress(address2.toString('hex'))
```

14.3. Mnemonic Code Converter

https://iancoleman.io/bip39/

输入 mnemonic，选择 ETH - Ethereum



系统将计算出钱包地址

Path Toggle	Address Toggle
m/44'/60'/0'/0/0	0xF78dFDe408d00991D01592A0f4d0dAC35Ea8072E
m/44'/60'/0'/0/1	0x0500111Fd8192419adaA0fe9056AA1E8C0286e41
m/44'/60'/0'/0/2	0xE1944A0F14b5E523327eb9f5f1caB60F88aF4582
m/44'/60'/0'/0/3	0x9d316F94E889777352cE2A86b65Ebbb5029feb12
m/44'/60'/0'/0/4	0x2eEAd68E98ca5F94761f1E03C6E14C2C5ee5af6e
m/44'/60'/0'/0/5	0xaE1c7c6b3CC6f483A6f6Ac80d49cD76c3945D8dc
m/44'/60'/0'/0/6	0x669E6Bbd25F74BAFa65b246658BF90d0dbCbF344
m/44'/60'/0'/0/7	0x206EFCc6e4D16c35D10867080227541578d33D75
m/44'/60'/0'/0/8	0x02dD8478DD4240A3100A18cE774F63EF2D7244ae
m/44'/60'/0'/0/9	0x03590bCB42e7f4c25E467fD1883040aC155654d1
m/44'/60'/0'/0/10	0x9033fEB4C45576c748e0645C5cC7DC798595D222

14.4. HD Wallet 例子

```
var bip39 = require('bip39');
var hdkey = require('ethereumjs-wallet/hdkey');
var util = require('ethereumjs-util');

mnemonic = 'client dune unfair assume level width bind control mad member old crystal';
var seed = bip39.mnemonicToSeed(mnemonic);
var hdwallet = hdkey.fromMasterSeed(seed);
var key1 = hdwallet.derivePath("m/44'/60'/0'/0/0");
var address1 = util.pubToAddress(key1._hdkey._publicKey, true);
var address = util.toChecksumAddress(address1.toString('hex'));

coinbase = "0xaa96686a050e4916afbe9f6d8c5107062fa646dd";
contractAddress = "0x9ABcF16f6685fE1F79168534b1D30056c90B8A8A"

console.log(address);

fs = require('fs');
const Web3 = require('web3');
const HDWalletProvider = require("truffle-hdwallet-provider");

const web3 = new Web3(new HDWalletProvider(mnemonic,'http://localhost:8545'));
console.log(web3.version)

web3.eth.getBalance("0xaa96686a050e4916afbe9f6d8c5107062fa646dd").then(function(balance){
      console.log( web3.utils.fromWei(balance) );
});

web3.eth.getBalance(address).then(function(balance){
      console.log( web3.utils.fromWei(balance) );
});

const abi = fs.readFileSync('output/NetkillerToken.abi', 'utf-8');
const contract = new web3.eth.Contract(JSON.parse(abi), contractAddress, { from: address, gas:
```

```
100000});

contract.methods.balanceOf(coinbase).call().then(console.log).catch(console.log);
contract.methods.balanceOf(address).call().then(console.log).catch(console.log);
```

14.5. 获得钱包地址和私钥

```
const bip39 = require('bip39');
const hdkey = require('ethereumjs-wallet/hdkey');

const mnemonic = 'client dune unfair assume level width bind control mad member old crystal';
const hdwallet = hdkey.fromMasterSeed(bip39.mnemonicToSeed(mnemonic));
const path = "m/44'/60'/0'/0/0";
const wallet = hdwallet.derivePath(path).getWallet();

const address = `0x${wallet.getAddress().toString('hex')}`;
const privateKey = wallet.getPrivateKey().toString('hex');
// wallet._privKey.toString('hex');
console.log(`Address: ${address}`);
console.log(`Private Key: ${privateKey}`);
```

14.6. truffle.js 例子

```
npm install truffle-hdwallet-provider
```

```
var HDWalletProvider = require("truffle-hdwallet-provider");

var mnemonic = "opinion destroy betray ...";

module.exports = {
  networks: {
    development: {
      host: "localhost",
      port: 8545,
      network_id: "*" // Match any network id
    },
    ropsten: {
      provider: new HDWalletProvider(mnemonic, "https://ropsten.infura.io/"),
      network_id: 3
    }
  }
};
```

14.7. Mnemonic To Seed 加密

没有加密 seed 是很不安全的，任何人都能通过 Mnemonic 还原出公钥和私钥。

```
var bip39 = require('bip39')
var hdkey = require('ethereumjs-wallet/hdkey')
var util = require('ethereumjs-util')
```

```
var mnemonic = bip39.generateMnemonic()

var password = "http://www.netkiller.cn"
var seed = bip39.mnemonicToSeed(mnemonic, password)
var hdwallet = hdkey.fromMasterSeed(seed)
var key1 = hdwallet.derivePath("m/44'/60'/0'/0/0")
var address1 = util.pubToAddress(key1._hdkey._publicKey, true)
address1 = util.toChecksumAddress(address1.toString('hex'))
```

14.8. 中文助记词

```
var bip39 = require('bip39')
var mnemonic = bip39.generateMnemonic(128, null, bip39.wordlists.chinese_simplified)
console.log(mnemonic)

var seed = bip39.mnemonicToSeed(mnemonic)
var hdwallet = hdkey.fromMasterSeed(seed)
var key1 = hdwallet.derivePath("m/44'/60'/0'/0/0")
var address1 = util.pubToAddress(key1._hdkey._publicKey, true)
address1 = util.toChecksumAddress(address1.toString('hex'))
```

演示

```
> var bip39 = require('bip39')
undefined

> var mnemonic = bip39.generateMnemonic(128, bip39.randomBytes,
bip39.wordlists.chinese_simplified)
undefined

> console.log(mnemonic)
俄 判 菌 诗 仪 偏 方 都 激 输 失 稀
undefined
```

14.9. 代币转账

15. 从 .ethereum/keystore 文件导入私钥

```
[ethereum@netkiller web3.example]$ npm install keythereum
```

代码

```
var keyth=require('keythereum')
// Mac
// keystore = './Appdata/roaming/ethereum';
// Ubuntu
// keystore = '~/.ethereum';
keystore = '../.ethereum';
var
keyobj=keyth.importFromFile('0x372fda02e8a1eca513f2ee5901dc55b8
b5dd7411',keystore)
var privateKey=keyth.recover('12345678',keyobj)
privateKey.toString('hex')
```

操作演示

```
> var keyth=require('keythereum')
undefined
> keystore = '../.ethereum';
'../.ethereum'
> var
keyobj=keyth.importFromFile('0x372fda02e8a1eca513f2ee5901dc55b8
b5dd7411',keystore)
undefined
> var privateKey=keyth.recover('12345678',keyobj)
undefined
> privateKey.toString('hex')
'f592b7bf06ca9fd7696ba95d6ed8e357de6a2379b6d5fe1ffd53c6b4b063cd
4a'
```

16. Express + web3.js 实现简单网页钱包

下面的例子，实现查询余额，创建账号，ETH转账，代币转账。

16.1. 创建项目

安装以太坊环境

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/gcc/gc
c.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/golang
/golang-1.10.2.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/
ethereum/centos/go-ethereum-1.8.8.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/
ethereum/systemd/private.sh | bash

curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/node.j
s/binrary/node-v10.1.0.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/node.j
s/binrary/profile.d.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/
ethereum/truffle/truffle.sh | bash
```

安装开发包

```
npm install express
npm install web3
npm install ejs
```

16.2. 主程序 main.js

```
var express = require('express');
var app = express();

app.use(express.static('public'));
app.set("view engine","ejs");
app.set('views', __dirname + '/views');

var async = require('async');

fs = require('fs');
var net = require('net');
var Web3 = require('web3');
var web3 = new Web3('/home/ethereum/.ethereum/geth.ipc', net);
const abi = fs.readFileSync( __dirname + '/abi/NKC.abi', 'utf-
8');
const coinbase = "0xaa96686a050e4916afbe9f6d62fa646dd8c51070"
const contractAddress =
"0x5F75DA091aBb25e055B91172C04371Ff4Dd563a0";

console.log(web3.version)


app.get('/', function (req, res) {
  //  res.send('Hello World');
   res.render("index",{});
})

app.get('/account.html', function (req, res) {
  var accounts;
  web3.eth.getAccounts(function(err, acc) {
    accounts = acc
    res.render("account",{"accounts":accounts});
  });
})

app.get('/new', function (req, res) {

web3.eth.personal.newAccount(req.query.password).then(function(
){
    res.redirect('/account.html');
```

```
  });
})

app.get('/balance.html', function (req, res) {

  web3.eth.getAccounts(function(err, accounts) {
    res.render("balance",{"accounts":accounts});
  });
})
app.post('/showbalance.html', function (req, res) {
  //
web3.eth.getBalance(req.query.account).then(function(balance){
  //   res.render("transfer",{"account":req.query.account,
"balance": balance});
  // });

  res.render("showbalance",{"account": "sss", "balance":
1000});
})

app.get('/getbalance.html', function (req, res) {
  var contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});
  web3.eth.getBalance(req.query.account).then(function(balance)
{

contract.methods.balanceOf(req.query.account).call().then(funct
ion(token){
      // console.log(contract.symbol.call());
      // contract.methods.symbol().call().then(console.log);
      contract.methods.symbol().call().then(function(name){
        res.render("showbalance",{"account":req.query.account,
"balance": web3.utils.fromWei(balance, 'ether'), "token":
token, "name": name});
      });

    });

  });
})

app.get('/transfer.html', function (req, res) {
  var contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});
  web3.eth.getAccounts(function(err, accounts) {
    contract.methods.symbol().call().then(function(symbol){
      res.render("transfer",{"accounts":accounts, "symbol":
symbol});
```

```
    });
  });
})

app.get('/send', function (req, res) {
  // console.log(req.query)
  web3.eth.personal.unlockAccount(req.query.from,
req.query.password).then(function(error){
    if(req.query.token == "ETH"){
      web3.eth.sendTransaction({
        from: req.query.from,
        to: req.query.to,
        value: web3.utils.toWei(req.query.amount ,'ether')
      },
      function(error, result){
          if(!error) {
              console.log("#" + result + "#")
              res.render("done",{"hash":result});
          } else {
              console.error(error);
          }
      });

    }else{
      var contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: req.query.from , gas: 1000000});
      contract.methods.transfer(req.query.to,
req.query.amount).send().then(function(hash){
        console.log(hash)
        res.render("done",{"hash":hash.transactionHash});
      });
    }
  });
})

var server = app.listen(8080, function () {

  var host = server.address().address
  var port = server.address().port

  console.log("应用实例，访问地址为 http://%s:%s", host, port)

})
```

16.3. ABI 文件 abi/NKC.abi

[{"constant":true,"inputs":[],"name":"name","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"_spender","type":"address"},
{"name":"_value","type":"uint256"}],"name":"approve","outputs":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabi
lity":"nonpayable","type":"function"},
{"constant":true,"inputs":[],"name":"totalSupply","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability
":"view","type":"function"},{"constant":false,"inputs":
[{"name":"_from","type":"address"},
{"name":"_to","type":"address"},
{"name":"_value","type":"uint256"}],"name":"transferFrom","outp
uts":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabi
lity":"nonpayable","type":"function"},
{"constant":true,"inputs":[],"name":"decimals","outputs":
[{"name":"","type":"uint8"}],"payable":false,"stateMutability":
"view","type":"function"},{"constant":false,"inputs":
[{"name":"_value","type":"uint256"}],"name":"burn","outputs":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabi
lity":"nonpayable","type":"function"},
{"constant":true,"inputs":
[{"name":"","type":"address"}],"name":"balanceOf","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability
":"view","type":"function"},{"constant":false,"inputs":
[{"name":"_from","type":"address"},
{"name":"_value","type":"uint256"}],"name":"burnFrom","outputs"
:
[{"name":"success","type":"bool"}],"payable":false,"stateMutabi
lity":"nonpayable","type":"function"},
{"constant":true,"inputs":[],"name":"symbol","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"_to","type":"address"},
{"name":"_value","type":"uint256"}],"name":"transfer","outputs"
:
[],"payable":false,"stateMutability":"nonpayable","type":"funct
ion"},{"constant":false,"inputs":
[{"name":"_spender","type":"address"},
{"name":"_value","type":"uint256"},
{"name":"_extraData","type":"bytes"}],"name":"approveAndCall","
outputs":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabi
lity":"nonpayable","type":"function"},
{"constant":true,"inputs":[{"name":"","type":"address"},

```
{"name":"","type":"address"}],"name":"allowance","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability
":"view","type":"function"},{"inputs":
[{"name":"initialSupply","type":"uint256"},
{"name":"tokenName","type":"string"},
{"name":"tokenSymbol","type":"string"}],"payable":false,"stateM
utability":"nonpayable","type":"constructor"},
{"anonymous":false,"inputs":
[{"indexed":true,"name":"from","type":"address"},
{"indexed":true,"name":"to","type":"address"},
{"indexed":false,"name":"value","type":"uint256"}],"name":"Tran
sfer","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"from","type":"address"},
{"indexed":false,"name":"value","type":"uint256"}],"name":"Burn
","type":"event"}]
```

16.4. 页面视图

16.4.1. views/account.ejs

```
<%- include header.ejs %>

<h1>Users</h1>
<ul id="accounts">
    <% accounts.forEach(function(account, index){ %>
    <li><%= index %>, <%= account %></li>
    <% }) %>
</ul>

<p>
新建账号
<form method="get" action="/new">
    密码：<input type="password" name="password" />
    <input type="submit" value="新建账号" />
</form>
</p>
```

16.4.2. views/balance.ejs

```
<%- include header.ejs %>

<h1>Account</h1>
<form method="get" action="/getbalance.html">

    <select name="account">
        <% accounts.forEach(function(account, index){ %>
        <option value ="<%= account %>"><%= account %></option>
        <% }) %>
    </select>
    <input type="submit" value="Submit" />
</form>
```

16.4.3. views/done.ejs

```
<%- include header.ejs %>

<p>转账完成</p>
<p>查看交易
<a href="https://etherscan.io/tx/<%= hash %>"
target="etherscan">主网<%= hash %></a> <br />
</p>
```

16.4.4. views/header.ejs

```
<a href="/account.html">账号</a> | <a href="/balance.html">余额
</a> | <a href="/transfer.html">转账</a>
<br />
<hr />
```

16.4.5. views/index.ejs

```
<%- include header.ejs %>

Welcome !!!
```

16.4.6. views/showbalance.ejs

```
<%- include header.ejs %>
<p>
<h1>Account: <%= account %>, Balance: <%= balance %></h1>
</p>

<p>
    Token: <%= token%> <%= name%>
</p>
```

16.4.7. views/transfer.ejs

```
<%- include header.ejs %>

<h1>Account</h1>
<form method="get" action="/send">
    From:
    <select name="from">
        <% accounts.forEach(function(account, index){ %>
        <option value ="<%= account %>"><%= account %></option>
        <% }) %>
    </select>
    <br />
    To:
    <select name="to">
        <% accounts.forEach(function(account, index){ %>
        <option value ="<%= account %>"><%= account %></option>
        <% }) %>
    </select>
```

```
    <select name="token">
        <option value ="ETH">ETH</option>
        <option value ="<%= symbol %>"><%= symbol %></option>
    </select>

    <br />
    金额: <input type="text" name="amount" /> ETH
    <br />
    密码: <input type="password" name="password" />
    <br />
    <input type="submit" value="Submit" />
</form>
```

16.5. 启动 Node 服务

```
neo@MacBook-Pro ~/example % node main.js
```

浏览器访问 http://localhost:8080/ 可以进入钱包

第 17 章 web3j v3.4.0 - Jave Client

官方网站 https://web3j.io

文档 https://github.com/web3j/web3j/tree/master/docs/source

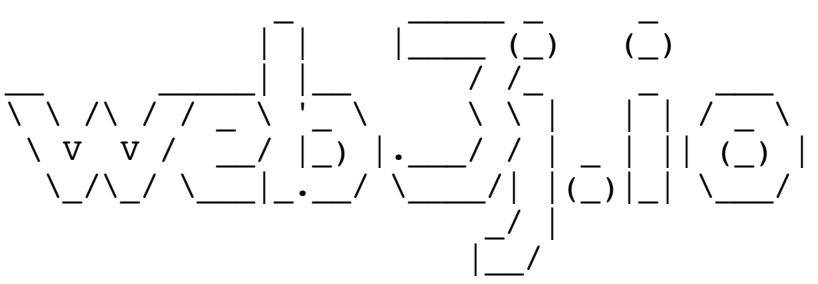
Java 客户端与 Server 之间采用json-rpc协议连接。

1. 安装命令行工具

web3j 命令用于将 sol 合约文件转换为 java 文件。

1.1. Mac OS

```
brew tap web3j/web3j
brew install web3j

neo@MacBook-Pro ~ % web3j
```

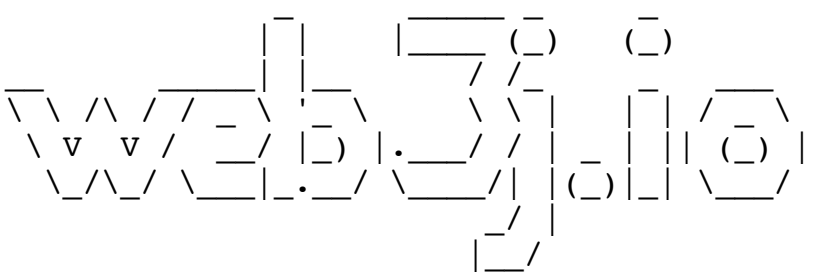


```
Usage: web3j version|wallet|solidity ...
```

1.2. 二进制包安装

下载二进制文件 https://github.com/web3j/web3j/releases

```
wget
https://github.com/web3j/web3j/releases/download/v3.2.0/web3j-
3.2.0.zip
unzip web3j-3.2.0.zip

$ ./web3j-3.2.0/bin/web3j
```



```
Usage: web3j version|wallet|solidity ...
```

2. 启动以太坊

首先启动服务

```
neo@netkiller ~ % geth --networkid 123456 --rpc --
rpcaddr="0.0.0.0" --rpccorsdomain "*" --nodiscover
INFO [02-01|23:35:12] Starting peer-to-peer node
instance=Geth/v1.8.8-stable-4bb3c89d/linux-amd64/go1.10.2
INFO [02-01|23:35:12] Allocated cache and file handles
database=/home/neo/.ethereum/geth/chaindata cache=128
handles=1024
INFO [02-01|23:35:12] Initialised chain configuration
config="{ChainID: 15 Homestead: 0 DAO: <nil> DAOSupport: false
EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil> Engine:
unknown}"
INFO [02-01|23:35:12] Disk storage enabled for ethash caches
dir=/home/neo/.ethereum/geth/ethash count=3
INFO [02-01|23:35:12] Disk storage enabled for ethash DAGs
dir=/home/neo/.ethash                count=2
INFO [02-01|23:35:12] Initialising Ethereum protocol
versions="[63 62]" network=123456
INFO [02-01|23:35:12] Loaded most recent local header
number=719 hash=61330b…82786e td=108754979
INFO [02-01|23:35:12] Loaded most recent local full block
number=719 hash=61330b…82786e td=108754979
INFO [02-01|23:35:12] Loaded most recent local fast block
number=719 hash=61330b…82786e td=108754979
INFO [02-01|23:35:12] Loaded local transaction journal
transactions=0 dropped=0
INFO [02-01|23:35:12] Regenerated local transaction journal
transactions=0 accounts=0
WARN [02-01|23:35:12] Blockchain not empty, fast sync disabled
INFO [02-01|23:35:12] Starting P2P networking
INFO [02-01|23:35:12] RLPx listener up
self="enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029
cf4a6c4693d2f470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5b
6b17a8a4b40527@[::]:30303?discport=0"
INFO [02-01|23:35:12] IPC endpoint opened:
/home/neo/.ethereum/geth.ipc
INFO [02-01|23:35:12] HTTP endpoint opened: http://0.0.0.0:8545
INFO [02-01|23:35:15] Mapped network port
proto=tcp extport=30303 intport=30303 interface="UPNP IGDv1-IP1"
```

Web3j 将使用这个地址连接 HTTP endpoint opened:
http://your_ip_address:8545

3. Maven pom.xml 文件

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>

        <groupId>cn.netkiller</groupId>
        <artifactId>ethereum</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>jar</packaging>

        <name>ethereum</name>
        <url>http://maven.apache.org</url>

        <properties>
                <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
        </properties>

        <dependencies>
                <dependency>
                        <groupId>org.web3j</groupId>
                        <artifactId>core</artifactId>
                        <version>3.4.0</version>
                </dependency>
        </dependencies>
</project>
```

4. Java 与 Solidity 数据类型映射关系

```
boolean -> bool
BigInteger -> uint/int
byte[] -> bytes
String -> string and address types
List<> -> dynamic/static array
```

5. 常量

DefaultBlockParameterName.LATEST	当前块地址
DefaultBlockParameterName.PENDING	处理中块地址

5.1. 默认 Gas

获取默认GAS Transaction.DEFAULT_GAS

]

```
package cn.netkiller.wallet.ethereum;

import org.web3j.protocol.core.methods.request.Transaction;

public class Test {

        public Test() {
                // TODO Auto-generated constructor stub
        }

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                System.out.println(Transaction.DEFAULT_GAS);

        }

}
```

5.2. 默认 gaslimit gasprice

已经废弃

```
BigInteger gasLimit = Contract.GAS_LIMIT;
BigInteger gasPrice = Contract.GAS_PRICE;
```

推荐使用

```
                        BigInteger gasPrice = BigInteger.ZERO;
                        BigInteger gasLimit = BigInteger.ZERO;

                        gasPrice = DefaultGasProvider.GAS_PRICE;
                        gasLimit = DefaultGasProvider.GAS_LIMIT;

                        System.out.println("gasPrice: " +
gasPrice.toString());
                        System.out.println("gasLimit: " +
gasLimit.toString());
```

6. 连接到服务器获取版本号

```
package cn.netkiller.ethereum;

import java.io.IOException;

import org.web3j.protocol.Web3j;
import org.web3j.protocol.core.methods.response.Web3ClientVersion;
import org.web3j.protocol.http.HttpService;

public class Web3JClient {
        // TODO Auto-generated method stub

        public static void main(String[] args) {
                String url = "http://172.16.0.1:8545/";
                Web3j web3j = Web3j.build(new HttpService(url)); // defaults to
http://localhost:8545/

                try {
                        Web3ClientVersion web3ClientVersion =
web3j.web3ClientVersion().send();
                        String clientVersion =
web3ClientVersion.getWeb3ClientVersion();
                        System.out.println(clientVersion);
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }

}
```

运行结果

```
Geth/v1.8.8-stable-4bb3c89d/linux-amd64/go1.10.2
```

除了 TCP 方式连接，还支持 IPC 方式。这种方式比较少用，可以使用 localhost 替代。

```
// OS X/Linux/Unix:
Web3j web3 = Web3j.build(new UnixIpcService("/path/to/socketfile"));
...

// Windows
Web3j web3 = Web3j.build(new WindowsIpcService("/path/to/namedpipefile"));
...
```

7. 获得以太坊状态信息

7.1. 获取客户端版本

```
                        Web3ClientVersion web3ClientVersion =
web3j.web3ClientVersion().send();
                        String clientVersion =
web3ClientVersion.getWeb3ClientVersion();
                        System.out.println("客户端版本: " +
clientVersion);
```

7.2. 协议版本

```
                        EthProtocolVersion ethProtocolVersion =
web3j.ethProtocolVersion().send();
                        String protocolVersion =
ethProtocolVersion.getProtocolVersion();
                        System.out.println("协议版本" +
protocolVersion);
```

7.3. 查看当前区块

```
                        EthBlockNumber ethBlockNumber =
web3j.ethBlockNumber().send();
                        BigInteger blockNumber =
ethBlockNumber.getBlockNumber();
                        System.out.println("当前区块: " +
blockNumber);
```

7.4. 同步状态

```
                        EthSyncing ethSyncing =
web3j.ethSyncing().send();
                        boolean isSyncing =
ethSyncing.isSyncing();
                        System.out.println("同步状态: " +
isSyncing);
```

7.5. 挖矿状态

```
                        EthMining ethMining =
web3j.ethMining().send();
                        boolean isMining = ethMining.isMining();
                        System.out.println("挖矿状态: " +
isMining);
```

7.6. 矿工账号

```
                        EthCoinbase ethCoinbase =
web3j.ethCoinbase().send();
                        String coinbase =
ethCoinbase.getAddress();
                        System.out.println("矿工账号: " +
coinbase);
```

7.7. 挖矿速度

```
                        EthHashrate ethHashrate =
web3j.ethHashrate().send();
```

```
                        BigInteger hashRate =
ethHashrate.getHashrate();
                        System.out.println("挖矿速度: " +
hashRate);
```

7.8. Gas 价格

```
                        EthGasPrice ethGasPrice =
web3j.ethGasPrice().send();
                        BigInteger gasPrice =
ethGasPrice.getGasPrice();
                        System.out.println("Gas 价格: " +
gasPrice);
```

7.9. 评估GAS

```
        EthEstimateGas ethEstimateGas =
web3.ethEstimateGas(Transaction.createEthCallTransaction(credent
ials.getAddress(), null, encodedFunction)).sendAsync().get();
        BigInteger estimateGas = ethEstimateGas.getAmountUsed();
        System.out.println(estimateGas);


ethEstimateGas.getAmountUsed().divide(BigInteger.valueOf(100));
```

7.10. 节点数量

```
                        NetPeerCount netPeerCount =
web3j.netPeerCount().send();
                        BigInteger peerCount =
netPeerCount.getQuantity();
                        System.out.println("节点数量: " +
```

```
peerCount);
```

8. 单位转换

8.1. GWEI to WEI

GWEI 转化为 WEI

```
BigInteger wei = Convert.toWei(BigDecimal.valueOf(0.001),
Convert.Unit.GWEI).toBigInteger();

BigInteger value = Convert.toWei("1.0",
Convert.Unit.ETHER).toBigInteger();
```

9. 账号管理

9.1. 获得账号列表

```
        public List<String> getAccountlist() {

                try {
                        return
web3j.ethAccounts().send().getAccounts();
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return null;
        }
```

9.2. 获得账号信息

```
        public String getAccount(int index) {
                String account = null;

                try {
                        account =
web3j.ethAccounts().send().getAccounts().get(index);

                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                return account;
        }
```

9.3. 创建账号

```
package cn.netkiller.example.ethereum.account;

import java.io.IOException;
import java.math.BigInteger;
import java.util.List;

import org.web3j.protocol.admin.Admin;
import
org.web3j.protocol.admin.methods.response.NewAccountIdentifier;
import
org.web3j.protocol.admin.methods.response.PersonalListAccounts;
import
org.web3j.protocol.admin.methods.response.PersonalUnlockAccount
;
//import org.web3j.protocol.http.HttpService;
import org.web3j.protocol.ipc.UnixIpcService;

public class AccountTest {
        private static Admin admin;

        public AccountTest() {
                // TODO Auto-generated constructor stub
                // admin = Admin.build(new
HttpService("http://127.0.0.1:8545"));
                admin = Admin.build(new
UnixIpcService("/Users/neo/Library/Ethereum/geth.ipc"));

        }

        private void createAccount() throws IOException {
                String password = "12345678";
                NewAccountIdentifier newAccountIdentifier =
admin.personalNewAccount(password).send();
                String address =
newAccountIdentifier.getAccountId();
                System.out.println("New account address: " +
address);
        }

        public static void main(String[] args) throws
IOException {
                // TODO Auto-generated method stub
                AccountTest account = new AccountTest();
                account.createAccount();
        }
```

```
}
```

9.4. 解锁账号

```
Admin web3j = Admin.build(new HttpService());  // defaults to
http://localhost:8545/
PersonalUnlockAccount personalUnlockAccount =
web3j.personalUnlockAccount("0x000...", "a
password").sendAsync().get();
if (personalUnlockAccount.accountUnlocked()) {
    // send a transaction
}
```

```
package cn.netkiller.example.ethereum.account;

import java.io.IOException;
import java.math.BigInteger;
import java.util.List;

import org.web3j.protocol.admin.Admin;
import
org.web3j.protocol.admin.methods.response.NewAccountIdentifier;
import
org.web3j.protocol.admin.methods.response.PersonalListAccounts;
import
org.web3j.protocol.admin.methods.response.PersonalUnlockAccount
;
//import org.web3j.protocol.http.HttpService;
import org.web3j.protocol.ipc.UnixIpcService;

public class AccountTest {
        private static Admin admin;

        public AccountTest() {
                // TODO Auto-generated constructor stub
                // admin = Admin.build(new
HttpService("http://127.0.0.1:8545"));
                admin = Admin.build(new
UnixIpcService("/Users/neo/Library/Ethereum/geth.ipc"));
```

```
        }

        private void unlockAccount() {
                String address =
"0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";
                String password = "12345678";
                // 账号解锁持续时间 单位秒 缺省值300秒
                BigInteger unlockDuration =
BigInteger.valueOf(60L);
                try {
                        PersonalUnlockAccount
personalUnlockAccount = admin.personalUnlockAccount(address,
password, unlockDuration).send();
                        Boolean isUnlocked =
personalUnlockAccount.accountUnlocked();
                        System.out.println("Account unlock " +
isUnlocked);
                } catch (IOException e) {
                        e.printStackTrace();
                }
        }

        public static void main(String[] args) throws
IOException {
                // TODO Auto-generated method stub
                AccountTest account = new AccountTest();
                account.unlockAccount();
        }

}
```

10. Credentials

```
package cn.netkiller.ethereum.credentials;

import java.io.IOException;
import java.math.BigInteger;

import org.web3j.crypto.Credentials;
import org.web3j.crypto.ECKeyPair;
import org.web3j.protocol.Web3j;
import org.web3j.protocol.http.HttpService;

public class CredentialsTest {

        public static void main(String[] args) {
                // TODO Auto-generated method stub

                String url = "http://172.16.0.1:8545/";
                Web3j web3j = Web3j.build(new
HttpService(url)); // defaults to http://localhost:8545/

                try {
                        String account =
web3j.ethAccounts().send().getAccounts().get(0);
                        Credentials credentials =
Credentials.create(account);
                        ECKeyPair keyPair =
credentials.getEcKeyPair();
                        BigInteger privateKey =
keyPair.getPrivateKey();
                        BigInteger publicKey =
keyPair.getPublicKey();

                        System.out.println(privateKey);
                        System.out.println("---");
                        System.out.println(publicKey);
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }
```

```
}
```

11. 交易

11.1. 获取余额

```
        public BigInteger getBalance(String account) throws IOException
{

                EthGetBalance ethGetBalance =
web3j.ethGetBalance(account, DefaultBlockParameterName.LATEST).send();
                BigInteger balance = ethGetBalance.getBalance();
                return balance;

        }
```

11.2. 通过 Keystore 转账

```
        public void transfer(String account, float coin)
                        throws InterruptedException, IOException,
TransactionException, Exception {
                String password = "";
                String walletfile = "/Users/neo/netkiller/UTC--2018-01-
20T04-04-06.786586541Z--83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6";
                Credentials credentials =
WalletUtils.loadCredentials(password, walletfile);
                TransactionReceipt transactionReceipt =
Transfer.sendFunds(web3j, credentials, account,
BigDecimal.valueOf(coin), Unit.ETHER).send();
                System.out.println(transactionReceipt.getStatus());
        }
```

11.3. 通过私钥转账

```
package cn.netkiller.ethereum.transaction;

import org.web3j.crypto.Credentials;
import org.web3j.protocol.Web3j;
import org.web3j.protocol.core.methods.response.TransactionReceipt;
import org.web3j.protocol.http.HttpService;
import org.web3j.tx.Transfer;
```

```
import org.web3j.utils.Convert;

import java.math.BigDecimal;

public class TransactionTest {
        public static void main(String[] args) throws Exception {
                Web3j web3j = Web3j.build(new
HttpService("https://ropsten.infura.io/CsS9shwaAab0z7B4LP2d"));
                String toAddress =
"0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";
                Credentials credentials =
Credentials.create("16690967F2BADABE13A067066558537228D8AF63ECCB022FFBDC
970EC717BC3A");

                TransactionReceipt transactionReceipt =
Transfer.sendFunds(web3j, credentials, toAddress,
BigDecimal.valueOf(0.002), Convert.Unit.ETHER).send();


System.out.println(transactionReceipt.getTransactionHash());
        }
}
```

11.4. 指定 gas 费用

```
package cn.netkiller.ethereum.transaction;

import org.web3j.crypto.Credentials;
import org.web3j.crypto.TransactionEncoder;
import org.web3j.crypto.RawTransaction;
import org.web3j.protocol.Web3j;
import org.web3j.protocol.core.DefaultBlockParameterName;
import org.web3j.protocol.core.methods.response.EthGetTransactionCount;
import org.web3j.protocol.core.methods.response.EthSendTransaction;
import org.web3j.protocol.http.HttpService;
import org.web3j.utils.Convert;
import org.web3j.utils.Numeric;

import java.math.BigInteger;
import java.util.concurrent.ExecutionException;

public class RawTransactionTest {

        public static void main(String[] args) throws
InterruptedException, ExecutionException {
                // TODO Auto-generated method stub
                // 设置需要的矿工费
                BigInteger gasPrice =
BigInteger.valueOf(18_000_000_000L);
```

```
                BigInteger gasLimit = BigInteger.valueOf(4_300_000);
                // System.out.println(gasPrice);

                // 连接 ropsten测试环境，这里使用的是infura这个客户端
                Web3j web3j = Web3j.build(new
HttpService("https://ropsten.infura.io/CsS9shwaAab0z7B4LP2d"));
                // 转出账户地址
                String fromAddress =
"0x22c57F0537414FD95b9f0f08f1E51d8b96F14029";
                // 接收账户地址
                String toAddress =
"0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";
                // 转账人的私钥
                Credentials credentials =
Credentials.create("16697AC066558537CADABF68BDE13A06790967F2BC3A228DB022
FF0EC717B3EC");

                // Nonce 就相当于数据中PK主键，每次Nonce会做 +1 操作
                EthGetTransactionCount ethGetTransactionCount =
web3j.ethGetTransactionCount(fromAddress,
DefaultBlockParameterName.LATEST).sendAsync().get();
                BigInteger nonce =
ethGetTransactionCount.getTransactionCount();

                // 创建交易并转0.05个以太币
                BigInteger value = Convert.toWei("0.05",
Convert.Unit.ETHER).toBigInteger();
                RawTransaction rawTransaction =
RawTransaction.createEtherTransaction(nonce, gasPrice, gasLimit,
toAddress, value);

                // 对交易做签名
                byte[] signedMessage =
TransactionEncoder.signMessage(rawTransaction, credentials);
                String hexValue = Numeric.toHexString(signedMessage);

                // 发送交易
                EthSendTransaction ethSendTransaction =
web3j.ethSendRawTransaction(hexValue).sendAsync().get();
                String transactionHash =
ethSendTransaction.getTransactionHash();

                // 获得到transactionHash后就可以到以太坊的网站上查询这笔交易的状
态了
                System.out.println("https://ropsten.etherscan.io/tx/" +
transactionHash);

        }

}
```

11.5. 查询 Transaction Information

```
        public TransactionReceipt getTransactionReceipt(String
transactionHash) throws InterruptedException, ExecutionException {
                EthGetTransactionReceipt ethGetTransactionReceipt =
web3.ethGetTransactionReceipt(transactionHash).sendAsync().get();
                return ethGetTransactionReceipt.getResult();
        }
```

执行返回结果

```
TransactionReceipt{

transactionHash='0xece52bdbc6d4fa0c8eba7578a7c6e537883265199fa07ef8e5b10
38e4bcdefb9',
        transactionIndex='0x13',
blockHash='0x2642b35670872a0e024d30ab2393b6bd4f7dab449bf4fc3eac067e2677c
bc085',
        blockNumber='0x344a79', cumulativeGasUsed='0x806f54',
gasUsed='0x8fee', contractAddress='null', root='null',
        status='0x1', from='0x22c57f0537414fd95b9f0f08f1e51d8b96f14029',
to='0xb3cedc76e75fcd278c988b22963c2f35c99c10b7', l
        ogs=[Log{removed=false, logIndex='0xa', transactionIndex='0x13',
transactionHash='0xece52bdbc6d4fa0c8eba7578a7c6e537883265199fa07ef8e5b10
38e4bcdefb9',

blockHash='0x2642b35670872a0e024d30ab2393b6bd4f7dab449bf4fc3eac067e2677c
bc085', blockNumber='0x344a79',
        address='0xb3cedc76e75fcd278c988b22963c2f35c99c10b7',
data='0x000000000000000000000000000000000000000000000000000000000000000a
',
        type='null', topics=
[0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef,
0x00000000000000000000000022c57f0537414fd95b9f0f08f1e51d8b96f14029,
0x000000000000000000000000cdf0253d8362d6c3334c8f28a6bfd74c90d03d92]}],

logsBloom='0x00000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000001000000000000000000000000000000000000000000000
00000000000000000000802000001000000000000000000000000000800000000000000000
00000000000000000000000000000000000000000000100000000000000000000004000000
00000000000000000000800000000000000000000000000000000000000000000000040000
00000000000000000000000000000000000000000000200000000000000040000000000000
00002000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000'
}
```

注意：该函数只能返回 TxReceipt Status: Success 状态数据，无法返回 Pending 状态的数据。

11.6. 交易结果查询

```
EthTransaction transaction =
web3.ethGetTransactionByHash("TRANSACTION_HASH").sendAsync().get();
System.out.println(transaction.getResult());
```

11.7. RawTransaction 编码与解码

```
                String hexTransaction =
Numeric.toHexString(TransactionEncoder.encode(rawTransaction));
                RawTransaction tx1 =
TransactionDecoder.decode(hexTransaction);
```

12. 钱包

12.1. 创建钱包

```
package cn.netkiller.ethereum.wallet;

import java.io.File;
import java.io.IOException;
import java.security.InvalidAlgorithmParameterException;
import java.security.NoSuchAlgorithmException;
import java.security.NoSuchProviderException;

import org.web3j.crypto.CipherException;
import org.web3j.crypto.WalletUtils;

public class WalletMain {

        public void createWallet() throws NoSuchAlgorithmException,
NoSuchProviderException,
                        InvalidAlgorithmParameterException, CipherException,
IOException {

                File file = new File("/tmp/ethereum/keystore");
                String password = "passw0rd";
                String fileName =
WalletUtils.generateFullNewWalletFile(password, file);
                System.out.println(fileName);

        }

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                WalletMain wallet = new WalletMain();
                try {
                        wallet.createWallet();
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }
}
```

运行结果

```
neo@MacBook-Pro ~ % mkdir -p /tmp/ethereum/keystore
neo@MacBook-Pro ~ % ll /tmp/ethereum/keystore
total 8
-rw-r--r--  1 neo  wheel   491B Feb  4 18:30 UTC--2018-02-04T10-30-
```

```
58.476000000Z--75d01e920d6e018445dae504058ce4d968fd2a58.json

neo@MacBook-Pro ~ % cat /tmp/ethereum/keystore/UTC--2018-02-04T10-30-
58.476000000Z--75d01e920d6e018445dae504058ce4d968fd2a58.json
{"address":"75d01e920d6e018445dae504058ce4d968fd2a58","id":"80700448-69bc-475a-
aaf9-f2b836f17b13","version":3,"crypto":{"cipher":"aes-128-
ctr","ciphertext":"fe86f5dbd61d15d092f9d6870e70bff7ed99a7925703ea71eef23669c8e3e
c62","cipherparams":
{"iv":"d058819ab660cd062080b405591ba143"},"kdf":"scrypt","kdfparams":
{"dklen":32,"n":262144,"p":1,"r":8,"salt":"f69c535137b08667dbac53b8001313f5b43f8
1fce67a5d0e94b518c97d212d14"},"mac":"c247e34760bc838c3a4c8b2da286ccc6acec244bbc1
3fc6cc9ce28e88a7319d5"}}
```

12.2. 从钱包取出账号

```
package cn.netkiller.ethereum.wallet;

import java.io.File;
import java.io.IOException;
import java.security.InvalidAlgorithmParameterException;
import java.security.NoSuchAlgorithmException;
import java.security.NoSuchProviderException;

import org.web3j.crypto.CipherException;
import org.web3j.crypto.Credentials;
import org.web3j.crypto.WalletUtils;

public class WalletMain {

        public void walletAddress() throws IOException, CipherException {

                File file = new File(
                                "/tmp/ethereum/keystore/UTC--2018-02-04T10-43-
27.339000000Z--7cab470df532710d13078c5cdc0812a27f70cf51.json");
                String password = "passw0rd";
                Credentials credentials = WalletUtils.loadCredentials(password,
file);
                System.out.println(credentials.getAddress());

        }

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                WalletMain wallet = new WalletMain();
                try {
                        wallet.walletAddress();
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }
}
```

运行结果

```
0x7cab470df532710d13078c5cdc0812a27f70cf51
```

12.3. 生成助记词钱包

```
                String keystore = WalletUtils.getDefaultKeyDirectory();
                System.out.println("生成keystore文件的默认目录：" + keystore);
                // 通过密码及keystore目录生成钱包
                Bip39Wallet wallet =
WalletUtils.generateBip39Wallet("yourpassword", new File(keystore));
                // keystore文件名
                System.out.println(wallet.getFilename());
                // 12个单词的助记词
                System.out.println(wallet.getMnemonic());
```

12.4. 随机产生助记词

```
package cn.netkiller.example.ethereum.mnemonic;

import java.security.SecureRandom;
import org.web3j.crypto.MnemonicUtils;

public class MnemonicUtilsTest {

        public MnemonicUtilsTest() {
                // TODO Auto-generated constructor stub
        }

        public static void main(String[] args) {

                // TODO Auto-generated method stub
                byte[] initialEntropy = new byte[16];
                SecureRandom secureRandom = new SecureRandom();
                secureRandom.nextBytes(initialEntropy);

                String mnemonic =
MnemonicUtils.generateMnemonic(initialEntropy);
                System.out.println(mnemonic);

        }

}
```

12.5. 导入 BIP39 钱包

```
                Credentials credentials =
WalletUtils.loadBip39Credentials("password", "spoon crisp length scrub train
scrap initial inherit airport that answer tornado");
                // 钱包地址
                System.out.println(credentials.getAddress());
                // 公钥16进制字符串表示

System.out.println(credentials.getEcKeyPair().getPublicKey().toString(16));
                // 私钥16进制字符串表示

System.out.println(credentials.getEcKeyPair().getPrivateKey().toString(16));
```

13. 智能合约

```
neo@netkiller ~/ethereum/solidity % cat netkiller.sol
pragma solidity ^0.4.18;

contract Netkiller {
    string name;
    int num;
    function Netkiller() public{
        name = "default";
        num = 1;
    }
    function setName(string _name) public{
        name = _name;
    }
    function getName() public view returns(string){
        return name;
    }
    function setNum(int n) public{
        num = n;
    }
    function addNum(int m) public view returns(int res){
        res = m + num;
    }
}
```

编译智能合约

```
$ solc /path/to/<smart-contract>.sol --bin --abi --optimize -o output/
$ web3j solidity generate /path/to/<smart-contract>.bin /path/to/<smart-
contract>.abi -o /path/to/src/main/java -p com.your.organisation.name
```

```
$ solc netkiller.sol --bin --abi --optimize -o output/
$ web3j solidity generate output/Netkiller.bin output/Netkiller.abi -p
cn.netkiller.ethereum.contract -o java
```

```
                _     _____ _     _
               | |    |____ (_)   (_)
__      _____| |__      / /_      _    ___
\ \ /\ / / _ \ '_ \     \ \ |    | |  / _ \
 \ V  V /  __/ |_) |.___/ / | _  | || (_) |
  \_/\_/ \___|_.__/ \____/| |(_)|_| \___/
                        _/ |
                       |__/

Generating cn.netkiller.ethereum.contract.Netkiller ... File written to java
```

```
neo@netkiller ~/ethereum/solidity % ll
java/cn/netkiller/ethereum/contract/Netkiller.java
-rw-rw-r-- 1 neo neo 5.9K Feb  3 23:02
java/cn/netkiller/ethereum/contract/Netkiller.java
```

启动以太坊，并开始挖矿。注意参数 --mine --minerthreads 1 ，你也可以启动后在 JavaScript 控制台钟启动挖矿。

```
neo@netkiller ~ % geth --networkid 123456 --rpc --rpcaddr="0.0.0.0" --
rpccorsdomain "*" --mine --minerthreads 1
```

```
package cn.netkiller.ethereum;

import java.math.BigInteger;

import org.web3j.crypto.Credentials;
import org.web3j.crypto.WalletUtils;
import org.web3j.protocol.Web3j;
import org.web3j.protocol.http.HttpService;
import org.web3j.tx.Contract;
import org.web3j.tx.ManagedTransaction;

import cn.netkiller.ethereum.contract.Netkiller;

public class ContractTest {

        public static void main(String[] args) throws Exception {
                // TODO Auto-generated method stub

                String walletfile = "/Users/neo/Downloads/UTC--2018-01-20T04-04-
06.786586541Z--83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6";

                Web3j web3j = Web3j.build(new
HttpService("http://172.16.0.1:8545"));
                Credentials credentials = WalletUtils.loadCredentials("",
walletfile);

                Netkiller contract = Netkiller.deploy(web3j, credentials,
ManagedTransaction.GAS_PRICE, Contract.GAS_LIMIT).send();

                System.out.println(contract.isValid());
                if (contract.isValid()) {
                        System.out.println("---");
                        String contractAddress = contract.getContractAddress();
                        System.out.println(contractAddress);
                        System.out.println("---");
                        String result = contract.getName().send();
                        System.out.println(result);

                        contract.setName("Netkiller").send();
```

```
                        System.out.println(contract.getName().send());
                        System.out.println("---");
                        contract.setNum(BigInteger.valueOf(8)).send();

System.out.println(contract.addNum(BigInteger.valueOf(8)).send());
                        System.out.println("---");
                } else {
                        System.out.println("Deploy ERROR !!!");
                }
        }

}
```

运行结果

```
true
---
0xef872f1b344a4b7c765c7d765a3cc82b741777a9
---
default
Netkiller
---
16
---
```

在程序运行是，去看 geth 打印的日志，有如下记录打印

```
INFO [02-04|00:04:43] Submitted transaction
fullhash=0x9f70ccb600294d2dd6dda08d090362131b107d42a692f27dd4a3b7548dbaf22c
recipient=0xEF872F1b344a4B7C765c7D765a3cC82b741777a9
```

13.1. 载入合约

```
HelloWorld contract = HelloWorld.load(contractAddress,web3j,credentials,
ManagedTransaction.GAS_PRICE, Contract.GAS_LIMIT);
```

14. ERC20合约

14.1. balanceOf

```
        @SuppressWarnings("rawtypes")
        public BigInteger getTokenBalance(String account,
String contractAddress) throws InterruptedException,
ExecutionException {
                Function function = new Function("balanceOf",
Arrays.<Type>asList(new Address(account)), Arrays.
<TypeReference<?>>asList(new TypeReference<Uint256>() {
                }));

                String encodedFunction =
FunctionEncoder.encode(function);

                EthCall response =
web3.ethCall(Transaction.createEthCallTransaction(account,
contractAddress, encodedFunction),
DefaultBlockParameterName.LATEST).sendAsync().get();

                List<Type> result =
FunctionReturnDecoder.decode(response.getValue(),
function.getOutputParameters());

                BigInteger balance = BigInteger.ZERO;
                if (result.size() == 1) {
                        balance = (BigInteger)
result.get(0).getValue();
                }
                return balance;
        }
```

合约 balance 是不含小数点的，因为不同合约采用的小数点位数不同，无法使用以太坊单位直接换算。可以使用下面方法添加小数：

```
        public BigDecimal formatBalance(BigInteger balance, int
```

```
decimal) {
                BigDecimal value = new BigDecimal(balance);
                value =
value.divide(BigDecimal.TEN.pow(decimal));
                return value;
        }
```

14.2. name

```
                String methodName = "name";
                String fromAddr = emptyAddress;
                List<Type> inputParameters = new ArrayList<>
();
                List<TypeReference<?>> outputParameters = new
ArrayList<>();
                TypeReference<Utf8String> typeReference = new
TypeReference<Utf8String>() {};
outputParameters.add(typeReference);

                Function function = new Function(methodName,
inputParameters,outputParameters);
```

```
@SuppressWarnings("rawtypes")
        public String getName(String contractAddress) {
                String name = null;

                Function function = new Function("name",
Arrays.<Type>asList(), Arrays.<TypeReference<?>>asList(new
TypeReference<Utf8String>() {
                }));

                String data = FunctionEncoder.encode(function);
                Transaction transaction =
Transaction.createEthCallTransaction(null, contractAddress,
data);

                EthCall ethCall;
                try {
                        ethCall = web3.ethCall(transaction,
```

```
DefaultBlockParameterName.LATEST).sendAsync().get();
                        List<Type> results =
FunctionReturnDecoder.decode(ethCall.getValue(),
function.getOutputParameters());
                        name =
results.get(0).getValue().toString();
                } catch (InterruptedException |
ExecutionException e) {
                        e.printStackTrace();
                }
                return name;
        }
```

14.3. 合约转账

```
        @SuppressWarnings("rawtypes")
        public String sendTokenTransaction(String fromAddress,
String password, String toAddress, BigInteger amount) {
                String txHash = null;

                try {

                        PersonalUnlockAccount
personalUnlockAccount =
admin.personalUnlockAccount(fromAddress, password,
BigInteger.valueOf(10)).send();
                        if
(personalUnlockAccount.accountUnlocked()) {
                                String methodName = "transfer";
                                List<Type> inputParameters =
new ArrayList<>();
                                List<TypeReference<?>>
outputParameters = new ArrayList<>();

                                Address tAddress = new
Address(toAddress);

                                Uint256 value = new
Uint256(amount);
                                inputParameters.add(tAddress);
                                inputParameters.add(value);

                                TypeReference<Bool>
```

```
typeReference = new TypeReference<Bool>() {
                                };

outputParameters.add(typeReference);

                                Function function = new
Function(methodName, inputParameters, outputParameters);

                                String data =
FunctionEncoder.encode(function);

                                EthGetTransactionCount
ethGetTransactionCount =
web3.ethGetTransactionCount(fromAddress,
DefaultBlockParameterName.PENDING).sendAsync().get();
                                BigInteger nonce =
ethGetTransactionCount.getTransactionCount();
                                BigInteger gasPrice =
Convert.toWei(BigDecimal.valueOf(5),
Convert.Unit.GWEI).toBigInteger();

                                Transaction transaction =
Transaction.createFunctionCallTransaction(fromAddress, nonce,
gasPrice, BigInteger.valueOf(60000), this.contractAddress,
data);

                                EthSendTransaction
ethSendTransaction =
web3.ethSendTransaction(transaction).sendAsync().get();
                                txHash =
ethSendTransaction.getTransactionHash();
                        }
                } catch (Exception e) {
                        e.printStackTrace();
                }

                return txHash;
        }
```

14.4. 完整的 ERC20 代币开发库

这是一个万能的代币接口，只要知道合约地址，即可操作该合约。传统做法是使用web3j 命令将 .sol 编译成 Java Class 但这种类只能操作自

己的合约。

```
package cn.netkiller.wallet.ethereum;

import java.io.IOException;
import java.math.BigDecimal;
import java.math.BigInteger;
import java.util.concurrent.ExecutionException;

import org.web3j.protocol.core.methods.request.Transaction;
import
org.web3j.protocol.core.methods.response.TransactionReceipt;
import org.web3j.tx.Contract;

public class TestToken {

        public TestToken() {
                // TODO Auto-generated constructor stub
        }

        @SuppressWarnings("deprecation")
        public static void main(String[] args) {
                // TODO Auto-generated method stub

                try {
                        Token token = new
Token("0xb3cedc76e75fcd278c988b22963c2f35c99c10b7",
"166970EDB022C717B3ECCADAB6558537228FFBDE1F68AC063A06790967F2BC
3A");
                        String owner = token.getOwner();
                        System.out.println("代币创建者: " +
owner);

                        String name = token.getName();
                        System.out.println("代币名称: " + name);

                        String symbol = token.getSymbol();
                        System.out.println("代币符号: " +
symbol);

                        int decimal = token.getDecimals();
                        System.out.println("小数位数: " +
decimal);
```

```
                        BigInteger totalSupply =
token.getTotalSupply();
                        System.out.println("发行总量: " +
totalSupply);

                        BigInteger tokenBalance =
token.getTokenBalance("0x22c57F0537414FD95b9f0f08f1E51d8b96F140
29");
                        System.out.println("代币余额:" +
tokenBalance);

                        BigDecimal val =
token.formatBalance(tokenBalance, decimal);
                        System.out.println("格式化后: " + val);

                        String transactionHash =
token.sendTransaction("0xCdF0253d8362d6c3334c8F28A6BFd74c90d03d
92", BigInteger.valueOf(10));
                        System.out.println("代币转账: " +
transactionHash);
                        TransactionReceipt transactionReceipt =
token.getTransactionReceipt("0xece52bdbc6d4fa0c8eba7578a7c6e537
883265199fa07ef8e5b1038e4bcdefb9");
                        System.out.println("转账状态: " +
transactionReceipt.toString());

                        String hash =
token.setApprove("0xCdF0253d8362d6c3334c8F28A6BFd74c90d03d92",
BigInteger.valueOf(100));
                        System.out.println("设置授信: " + hash);

                        BigInteger value =
token.getAllowance("0x22c57F0537414FD95b9f0f08f1E51d8b96F14029"
, "0xCdF0253d8362d6c3334c8F28A6BFd74c90d03d92");
                        System.out.println("查询授信: " + value);

                        Token token1 = new
Token("0xb3cedc76e75fcd278c988b22963c2f35c99c10b7",
"8D160B668E63CC04CEE44C398C184121D63C3F5D189671D985A6FB3719FB1B
5E");
                        System.out.println("授信转出: " +
token1.sendTransactionFrom("0x22c57F0537414FD95b9f0f08f1E51d8b9
6F14029", "0xCdF0253d8362d6c3334c8F28A6BFd74c90d03d92",
BigInteger.valueOf(20)));

                        //
```

```
System.out.println(token1.getAllowance("0x22c57F0537414FD95b9f0
f08f1E51d8b96F14029",
"0xCdF0253d8362d6c3334c8F28A6BFd74c90d03d92"));
                } catch (InterruptedException |
ExecutionException | IOException e) {
                        e.printStackTrace();

                }

        }

}
```

运行结果

```
代币创建者：0x22c57f0537414fd95b9f0f08f1e51d8b96f14029
代币名称：Netkiller Test Coin
代币符号：NTC
小数位数：4
发行总量：1000000000000
代币余额:999999999430
格式化后：99999999.943
代币转账：
0xe851f682457672f2ca5ddbc3ad276dd9fa56ea81e838cf9a4b1eb8c97d0d9
8fd
转账状态：
TransactionReceipt{transactionHash='0xece52bdbc6d4fa0c8eba7578a
7c6e537883265199fa07ef8e5b1038e4bcdefb9',
transactionIndex='0x13',
blockHash='0x2642b35670872a0e024d30ab2393b6bd4f7dab449bf4fc3eac
067e2677cbc085', blockNumber='0x344a79',
cumulativeGasUsed='0x806f54', gasUsed='0x8fee',
contractAddress='null', root='null', status='0x1',
from='0x22c57f0537414fd95b9f0f08f1e51d8b96f14029',
to='0xb3cedc76e75fcd278c988b22963c2f35c99c10b7', logs=
[Log{removed=false, logIndex='0xa', transactionIndex='0x13',
transactionHash='0xece52bdbc6d4fa0c8eba7578a7c6e537883265199fa0
7ef8e5b1038e4bcdefb9',
blockHash='0x2642b35670872a0e024d30ab2393b6bd4f7dab449bf4fc3eac
067e2677cbc085', blockNumber='0x344a79',
```

address='0xb3cedc76e75fcd278c988b22963c2f35c99c10b7',
data='0x0000000000000000000000000000000000000000000000000000000
00000000a', type='null', topics=
[0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523
b3ef,
0x00000000000000000000000022c57f0537414fd95b9f0f08f1e51d8b96f14
029,
0x000000000000000000000000cdf0253d8362d6c3334c8f28a6bfd74c90d03
d92]}],
logsBloom='0x00000000000000000000000000000000000000000000000000
000000000000000000000000000000000000010000000000000000000000000
000000000000000000000000000000000000008020000001000000000000000
000000000000800000000000000000000000000000000000000000000000000
000000010000000000000000000004000000000000000000000000000800000
000000000000000000000000000000000000000000400000000000000000000
000000000000000000000000002000000000000000400000000000000002000
000000000000000000000000000000000000000000000000000000000000000
000000000000000000000'}
设置授信:
0x2a6c32b5939de91d9f06f7b73913433ee0b28642a3afc095454a7dcd59da3
234
查询授信：100
授信转出:
0x29f4824b35c47a0fc18adf2d8c72c902a3c518f249bc4235aace804e5a9f1
7df

如需代码，有偿提供，请联系作者。

15. Infura

```
package cn.netkiller.ethereum.infura;

import org.web3j.protocol.Web3j;
import
org.web3j.protocol.core.methods.response.Web3ClientVersion;
import org.web3j.protocol.http.HttpService;

public class Infura {
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                try {
                        Web3j web3 = Web3j.build(new
HttpService("https://rinkeby.infura.io/CsS9shwaAab0z7B4LP2d"));
                        Web3ClientVersion web3ClientVersion =
web3.web3ClientVersion().send();

System.out.println(web3ClientVersion.getWeb3ClientVersion());
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        }
}
```

16. 助记词

16.1. 获取随机助记词

```
package cn.netkiller.ethereum.mnemonic;

import java.io.IOException;
import java.security.SecureRandom;
import java.util.List;

import org.bitcoinj.wallet.DeterministicSeed;
import org.bitcoinj.wallet.UnreadableWalletException;

public class MnemonicTest {
        public static void main(String[] args) throws
UnreadableWalletException, IOException {
                // TODO Auto-generated method stub

                String passphrase = "";
                SecureRandom secureRandom = new SecureRandom();
                long creationTimeSeconds =
System.currentTimeMillis() / 1000;
                DeterministicSeed deterministicSeed = new
DeterministicSeed(secureRandom, 128, passphrase,
creationTimeSeconds);
                List<String> mnemonicCode =
deterministicSeed.getMnemonicCode();
                System.out.println(String.join(" ",
mnemonicCode));

        }
}
```

输出

```
romance rhythm session oyster upgrade include hammer chimney
float bridge autumn accident
```

16.2. 助记词导出公钥和私钥

```
package cn.netkiller.ethereum.mnemonic;

import java.math.BigInteger;
import java.util.List;

import org.bitcoinj.crypto.ChildNumber;
import org.bitcoinj.crypto.DeterministicKey;
import org.bitcoinj.crypto.HDUtils;
import org.bitcoinj.wallet.DeterministicKeyChain;
import org.bitcoinj.wallet.DeterministicSeed;
import org.bitcoinj.wallet.UnreadableWalletException;
import org.web3j.crypto.Credentials;

public class Test {

        public static void main(String[] args) throws
UnreadableWalletException {
                // TODO Auto-generated method stub
                String seedCode = "client dune unfair assume
level width bind control mad member old crystal";

                // BitcoinJ
                DeterministicSeed seed = new
DeterministicSeed(seedCode, null, "", 1409478661L);
                DeterministicKeyChain chain =
DeterministicKeyChain.builder().seed(seed).build();
                List<ChildNumber> keyPath =
HDUtils.parsePath("M/44H/60H/0H/0/0");
                DeterministicKey key =
chain.getKeyByPath(keyPath, true);
                BigInteger privKey = key.getPrivKey();

                // Web3j

                Credentials credentials =
Credentials.create(privKey.toString(16));
                String address = credentials.getAddress();
                String privateKey = privKey.toString(16);
                System.out.println(address);
                System.out.println(privateKey);
```

```
        }
}
```

输出

```
0x372fda02e8a1eca513f2ee5901dc55b8b5dd7411
f592b7bf06ca9fd7696ba95d6ed8e357de6a2379b6d5fe1ffd53c6b4b063cd4
a
```

17. 过滤器 (Filter)

18. Subscription

在区块链上发生的事件时，能不通知到订阅事件者。

提示

主意 https://docs.web3j.io/filters.html 有一段话：

Note: filters are not supported on Infura.

目前 Infura 不支持。

18.1. 接收所有添加到区块链的新区块

```
Subscription subscription = web3j.blockObservable(false).subscribe(block -> {
    ...
});
```

18.2. 接收所有添加到区块链的新交易

```
Subscription subscription = web3j.transactionObservable().subscribe(tx -> {
    ...
});
```

18.3. 接收所有待处理的事务

```
Subscription subscription = web3j.pendingTransactionObservable().subscribe(tx ->
{
    ...
});
```

测试环境 Mac, 首先启动 Ethereum Wallet，然后启动下面程序，回到 Ethereum Wallet 中做一笔转账。

```
package cn.netkiller.example.ethereum.subscription;
```

```
import org.web3j.protocol.Web3j;
import org.web3j.protocol.ipc.UnixIpcService;

import rx.Subscription;

public class PendingTest {

        public PendingTest() {
                // TODO Auto-generated constructor stub
        }

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                System.out.println("Subscription Starting...");
                Web3j web3 = Web3j.build(new
UnixIpcService("/Users/neo/Library/Ethereum/geth.ipc"));
                Subscription subscription =
web3.pendingTransactionObservable().subscribe(block -> {
                        // System.out.println(block.toString());
                        System.out.println("block number: " +
block.getBlockHash());
                });
                // subscription.unsubscribe();
        }

}
```

输出

```
block number: 0x74f7dd053dadcf01599dc85d4abf60662695e78ce7531335f44dc03f49dee326
```

18.4. 将区块块重放到当前的当前位置

```
Subscription subscription = catchUpToLatestAndSubscribeToNewBlocksObservable(
        <startBlockNumber>, <fullTxObjects>)
        .subscribe(block -> {
            ...
});
```


18.5. 过滤主题

```
EthFilter filter = new EthFilter(DefaultBlockParameterName.EARLIEST,
DefaultBlockParameterName.LATEST, <contract-
address>).addSingleTopic(...)|.addOptionalTopics(..., ...)|...;

web3j.ethLogObservable(filter).subscribe(log -> {
    ...
});
```

18.6. 停止订阅 Subscriptions

```
subscription.unsubscribe();
```

18.7.

```
Web3j web3 = Web3j.build(new HttpService("http://127.0.0.1:8080"));
        System.out.println("Connected to Ethereum client version: "
                + web3.toString());
        subscription = web3.blockObservable(false)
                .subscribe(tx -> {
                    System.out.println("observation tx:" + tx.getRawResponse());
                });
        txSubscription = web3.transactionObservable().subscribe((tx) -> {
            System.out.println("txSubscription hash:" + tx.getHash() +
":::::address:" + tx.getTo());
            if (!ValidationUtil.isEmpty(tx.getTo())) {
                System.out.println("getBlockNumber:" +
tx.getBlockNumber().longValue());
                System.out.println("getValue:" + tx.getValue());
                System.out.println("getTo:" + tx.getTo());
                System.out.println("getFrom:" + tx.getFrom());
                System.out.println("getHash:" + tx.getHash());
            }
        });
        pendingSubscription = web3.pendingTransactionObservable().subscribe(tx -
> {
            System.out.println("pending hash:" + tx.getHash() + ":::::address:"
+ tx.getTo());
        });
        latestSubscription = web3.catchUpToLatestTransactionObservable(new
DefaultBlockParameterNumber(2576860)).subscribe(block ->
                System.out.println("*+observer+getblock" +
block.getBlockNumber())
        );
```

19. 解锁账号

```
Admin admin = Admin.build(new HttpService());  // defaults to
http://localhost:8545/
PersonalUnlockAccount personalUnlockAccount =
admin.personalUnlockAccount("0x000...", "a
password").sendAsync().get();
if (personalUnlockAccount.accountUnlocked()) {
    // send a transaction
}
```

20. IBAN (International Bank Account Number)

https://github.com/ethereum/wiki/wiki/ICAP:-Inter-exchange-Client-Address-Protocol

```
iban:XE4214YF25M7C0Q6QFUF989GYBCR29987SX?amount=100&token=ETH
```

https://github.com/arturmkrtchyan/iban4j

21. Springboot with Ethereum (web3j)

21.1. Maven

```
                <dependency>
                        <groupId>org.web3j</groupId>
                        <artifactId>web3j-spring-boot-
starter</artifactId>
                        <version>1.6.0</version>
                </dependency>
```

21.2. application.properties

```
web3j.client-
address=https://ropsten.infura.io/CsS9shwaAab0z7B4LP2d
web3j.admin-client=true
```

21.3. TestRestController

```
package cn.netkiller.wallet.restful;

import java.io.IOException;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.web3j.protocol.Web3j;
import
org.web3j.protocol.core.methods.response.Web3ClientVersion;
```

```
@RestController
public class TestRestController {
	private static final Logger logger =
LoggerFactory.getLogger(TestRestController.class);

	@Autowired
	private Web3j web3j;

	public TestRestController() {
		// TODO Auto-generated constructor stub
	}

	@GetMapping("/version")
	public String version() throws IOException {
		Web3ClientVersion web3ClientVersion =
web3j.web3ClientVersion().send();
		String clientVersion =
web3ClientVersion.getWeb3ClientVersion();
		logger.info(clientVersion);
		return clientVersion;
	}

}
```

21.4. 测试

```
neo@MacBook-Pro ~ % curl http://localhost:8080/version
Geth/v1.8.3-stable/linux-amd64/go1.10
```

第 18 章 web3.py - A python interface for interacting with the Ethereum blockchain and ecosystem.

文档地址 http://web3py.readthedocs.io/

1. 安装 web3.py 开发环境

1.1. CentOS

```
[root@iZj6c39y62jl5b1wmfv6u8Z ~]# yum install python34-pip

[root@iZj6c39y62jl5b1wmfv6u8Z ~]# pip3 install --upgrade pip
Collecting pip
  Downloading
http://mirrors.aliyun.com/pypi/packages/0f/74/ecd13431bcc456ed3
90b44c8a6e917c1820365cbebcb6a8974d1cd045ab4/pip-10.0.1-py2.py3-
none-any.whl (1.3MB)
    100% |████████████████████████████████| 1.3MB 68.9MB/s
Installing collected packages: pip
  Found existing installation: pip 8.1.2
    Uninstalling pip-8.1.2:
      Successfully uninstalled pip-8.1.2
Successfully installed pip-10.0.1
```

```
python3.6 -m venv env --without-pip
source env/bin/activate
curl https://bootstrap.pypa.io/get-pip.py | python3
python3.6 -m pip install web3
```

1.2. MAC OS

首先去 python.org 官网下载 Python 3.6.5 dmg格式，安装后进入终端。

```
You should consider upgrading via the 'pip install --upgrade
pip' command.
neo@MacBook-Pro ~/ethereum/web3.py % pip3.6 install --upgrade
pip
Cache entry deserialization failed, entry ignored
Collecting pip
  Using cached
https://files.pythonhosted.org/packages/0f/74/ecd13431bcc456ed3
90b44c8a6e917c1820365cbebcb6a8974d1cd045ab4/pip-10.0.1-py2.py3-
none-any.whl
Installing collected packages: pip
  Found existing installation: pip 9.0.3
    Uninstalling pip-9.0.3:
      Successfully uninstalled pip-9.0.3
Successfully installed pip-10.0.1

neo@MacBook-Pro ~/ethereum/web3.py % pip3.6 install web3
```

2. 连接到以太坊节点

2.1. HTTP

```
>>> from web3 import Web3
>>> web3 = Web3(Web3.HTTPProvider("http://127.0.0.1:8545")
```

2.2. IPC

```
>>> from web3 import Web3
>>> web3 = Web3(Web3.IPCProvider("~/Library/Ethereum/geth.ipc"))
```

2.3. Websocket

```
>>> from web3 import Web3
>>> web3 = Web3(Web3.WebsocketProvider("ws://127.0.0.1:8546")
```

3. 交易

3.1. 发送 ETH

```
from web3 import Web3
web3 = Web3(Web3.IPCProvider("~/Library/Ethereum/geth.ipc"))

fromAddress = '0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA'
toAddress = '0x997e5CA600E19447D0B82aFBf9c7F00De2B39B16'
value = 0.0001

web3.personal.unlockAccount(fromAddress, '12345678')
web3.eth.sendTransaction({'to': toAddress, 'from': fromAddress,
'value': value})
```

3.2. 签名发送 ETH

```
from web3 import Web3
web3 =
Web3(Web3.HTTPProvider("https://ropsten.infura.io/CsS9shwaAab0z7
B4LP2d"))

privateKey =
'8D16063C3F665A6FB37195D18968E63CC04CEE4BFB1B98C184121D4C31D9875
E'
fromAddress = "0xb3cedc76e75fcd278c988b22963c2f35c99c10b7"
toAddress = '0x997e5CA600E19447D0B82aFBf9c7F00De2B39B16'
amount = 0.0001

signed_txn = web3.eth.account.signTransaction(dict(
    nonce=web3.eth.getTransactionCount(fromAddress),
    gasPrice=web3.eth.gasPrice,
    gas=21000,
    to=toAddress,
    value=amount,
    data=b'',
  ),
  private_key,
```

```
)
web3.eth.sendRawTransaction(signed_txn.rawTransaction)
```

4. ERC20 代币合约

```
from web3 import Web3
web3 =
Web3(Web3.HTTPProvider("https://ropsten.infura.io/CsS9shwaAab0z7
B4LP2d"))

contract =
web3.eth.contract(address='0x00000000000000000000000000000000000
0dead', abi=...)

contract.functions.balanceOf(address).call()

contract.functions.transfer(to, value).call()
```

4.1. 签名发送ERC20代币

```
from web3 import Web3
from web3 import Web3, HTTPProvider, IPCProvider
# w3 = Web3(Web3.IPCProvider("~/Library/Ethereum/geth.ipc"))

w3 =
Web3(HTTPProvider("https://ropsten.infura.io/CsS9shwaAab0z7B4LP2
d"))

fromAddress = '0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7'
toAddress = '0xCdF0253d8362d6c3334c8F28A6BFd74c90d03d92'
contractAddress='0xfB6916095ca1df60bB79Ce92cE3Ea74c37c5d359'
private_key = b''


# '~/.ethereum/testnet/keystore/UTC--2018-03-02T12-12-
51.966823000Z--b94054c174995ae2a9e7fcf6c7924635fba8ecf7'
with open('/Users/neo/Library/Ethereum/testnet/keystore/UTC-
-2018-03-02T12-12-51.966823000Z--
b94054c174995ae2a9e7fcf6c7924635fba8ecf7') as keyfile:
    encrypted_key = keyfile.read()
    private_key = w3.eth.account.decrypt(encrypted_key, '')
```

```
print(private_key)

interface='[ { "constant": true, "inputs": [], "name": "name",
"outputs": [ { "name": "", "type": "string" } ], "payable":
false, "stateMutability": "view", "type": "function" }, {
"constant": false, "inputs": [ { "name": "_spender", "type":
"address" }, { "name": "_value", "type": "uint256" } ], "name":
"approve", "outputs": [ { "name": "success", "type": "bool" } ],
"payable": false, "stateMutability": "nonpayable", "type":
"function" }, { "constant": true, "inputs": [], "name":
"totalSupply", "outputs": [ { "name": "", "type": "uint256" } ],
"payable": false, "stateMutability": "view", "type": "function"
}, { "constant": false, "inputs": [ { "name": "_from", "type":
"address" }, { "name": "_to", "type": "address" }, { "name":
"_value", "type": "uint256" } ], "name": "transferFrom",
"outputs": [ { "name": "success", "type": "bool" } ], "payable":
false, "stateMutability": "nonpayable", "type": "function" }, {
"constant": true, "inputs": [], "name": "decimals", "outputs": [
{ "name": "", "type": "uint8" } ], "payable": false,
"stateMutability": "view", "type": "function" }, { "constant":
false, "inputs": [ { "name": "_value", "type": "uint256" } ],
"name": "burn", "outputs": [ { "name": "success", "type": "bool"
} ], "payable": false, "stateMutability": "nonpayable", "type":
"function" }, { "constant": true, "inputs": [ { "name": "",
"type": "address" } ], "name": "balanceOf", "outputs": [ {
"name": "", "type": "uint256" } ], "payable": false,
"stateMutability": "view", "type": "function" }, { "constant":
false, "inputs": [ { "name": "_from", "type": "address" }, {
"name": "_value", "type": "uint256" } ], "name": "burnFrom",
"outputs": [ { "name": "success", "type": "bool" } ], "payable":
false, "stateMutability": "nonpayable", "type": "function" }, {
"constant": true, "inputs": [], "name": "symbol", "outputs": [ {
"name": "", "type": "string" } ], "payable": false,
"stateMutability": "view", "type": "function" }, { "constant":
false, "inputs": [ { "name": "_to", "type": "address" }, {
"name": "_value", "type": "uint256" } ], "name": "transfer",
"outputs": [], "payable": false, "stateMutability":
"nonpayable", "type": "function" }, { "constant": false,
"inputs": [ { "name": "_spender", "type": "address" }, { "name":
"_value", "type": "uint256" }, { "name": "_extraData", "type":
"bytes" } ], "name": "approveAndCall", "outputs": [ { "name":
"success", "type": "bool" } ], "payable": false,
"stateMutability": "nonpayable", "type": "function" }, {
"constant": true, "inputs": [ { "name": "", "type": "address" },
{ "name": "", "type": "address" } ], "name": "allowance",
"outputs": [ { "name": "", "type": "uint256" } ], "payable":
false, "stateMutability": "view", "type": "function" }, {
"inputs": [ { "name": "initialSupply", "type": "uint256",
```

```
"index": 0, "typeShort": "uint", "bits": "256", "displayName":
"initial Supply", "template": "elements_input_uint", "value":
"100000" }, { "name": "tokenName", "type": "string", "index": 1,
"typeShort": "string", "bits": "", "displayName": "token Name",
"template": "elements_input_string", "value": "NEO" }, { "name":
"tokenSymbol", "type": "string", "index": 2, "typeShort":
"string", "bits": "", "displayName": "token Symbol", "template":
"elements_input_string", "value": "#" } ], "payable": false,
"stateMutability": "nonpayable", "type": "constructor" }, {
"anonymous": false, "inputs": [ { "indexed": true, "name":
"from", "type": "address" }, { "indexed": true, "name": "to",
"type": "address" }, { "indexed": false, "name": "value",
"type": "uint256" } ], "name": "Transfer", "type": "event" }, {
"anonymous": false, "inputs": [ { "indexed": true, "name":
"from", "type": "address" }, { "indexed": false, "name":
"value", "type": "uint256" } ], "name": "Burn", "type": "event"
} ]'
contract = w3.eth.contract(address=contractAddress,
abi=interface)

nonce = w3.eth.getTransactionCount(fromAddress)
print(nonce)

txn =
contract.functions.transfer(toAddress,5,).buildTransaction({
    'chainId': 3,
    'gas': 30000,
    'gasPrice': w3.toWei('1', 'gwei'),
    'nonce': nonce,
    })

print(txn)

signed_txn = w3.eth.account.signTransaction(txn,
private_key=private_key)
print(signed_txn.hash)
print(signed_txn.rawTransaction)

tmp = w3.eth.sendRawTransaction(signed_txn.rawTransaction)

txhash = w3.toHex(w3.sha3(signed_txn.rawTransaction))

print("https://ropsten.etherscan.io/tx/"+txhash)
```

第 19 章 Ethereum iOS

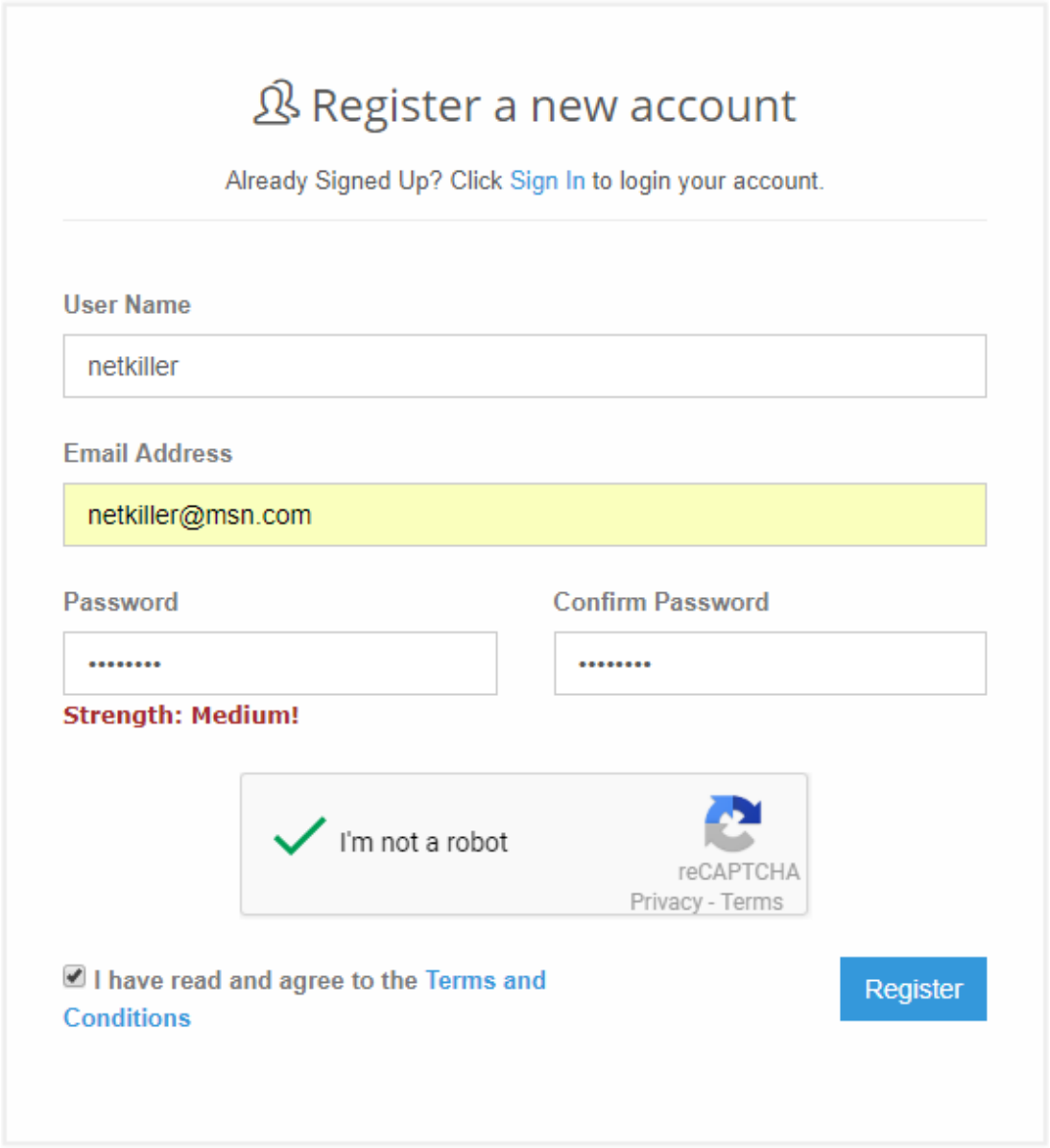
https://github.com/MercuryProtocol/web3.swift

https://medium.com/mercuryprotocol/introducing-web3-swift-for-ethereum-ios-development-1e02212b662b

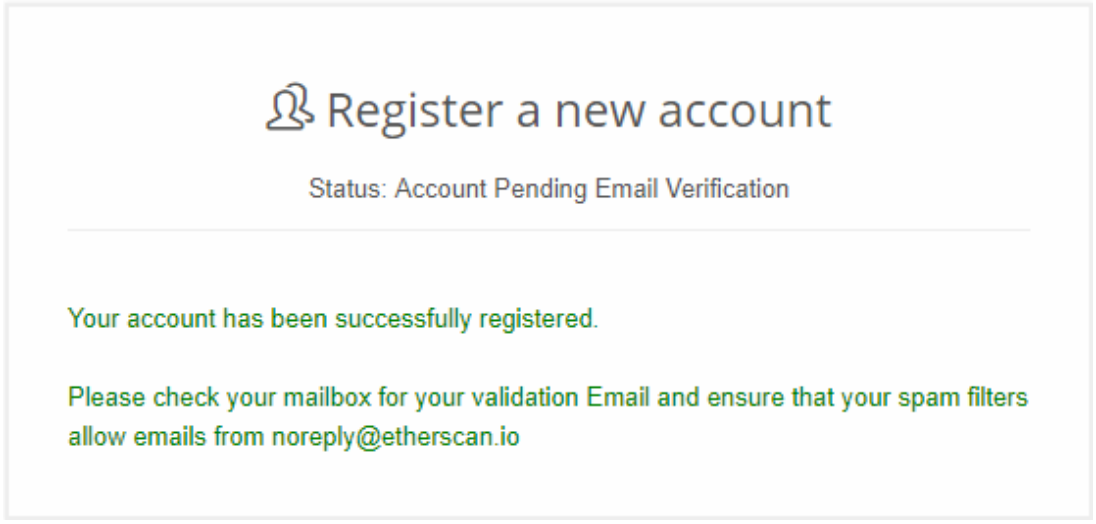
第 20 章 Ethereum Developer APIs

1. API Keys

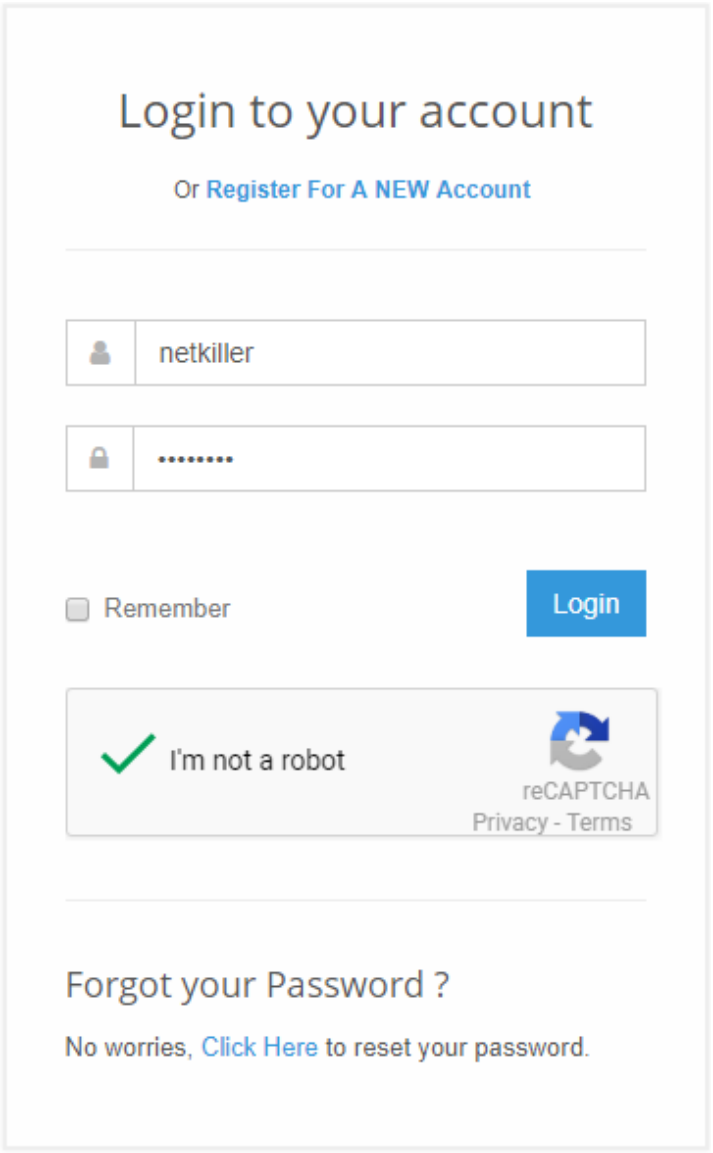
https://etherscan.io/register



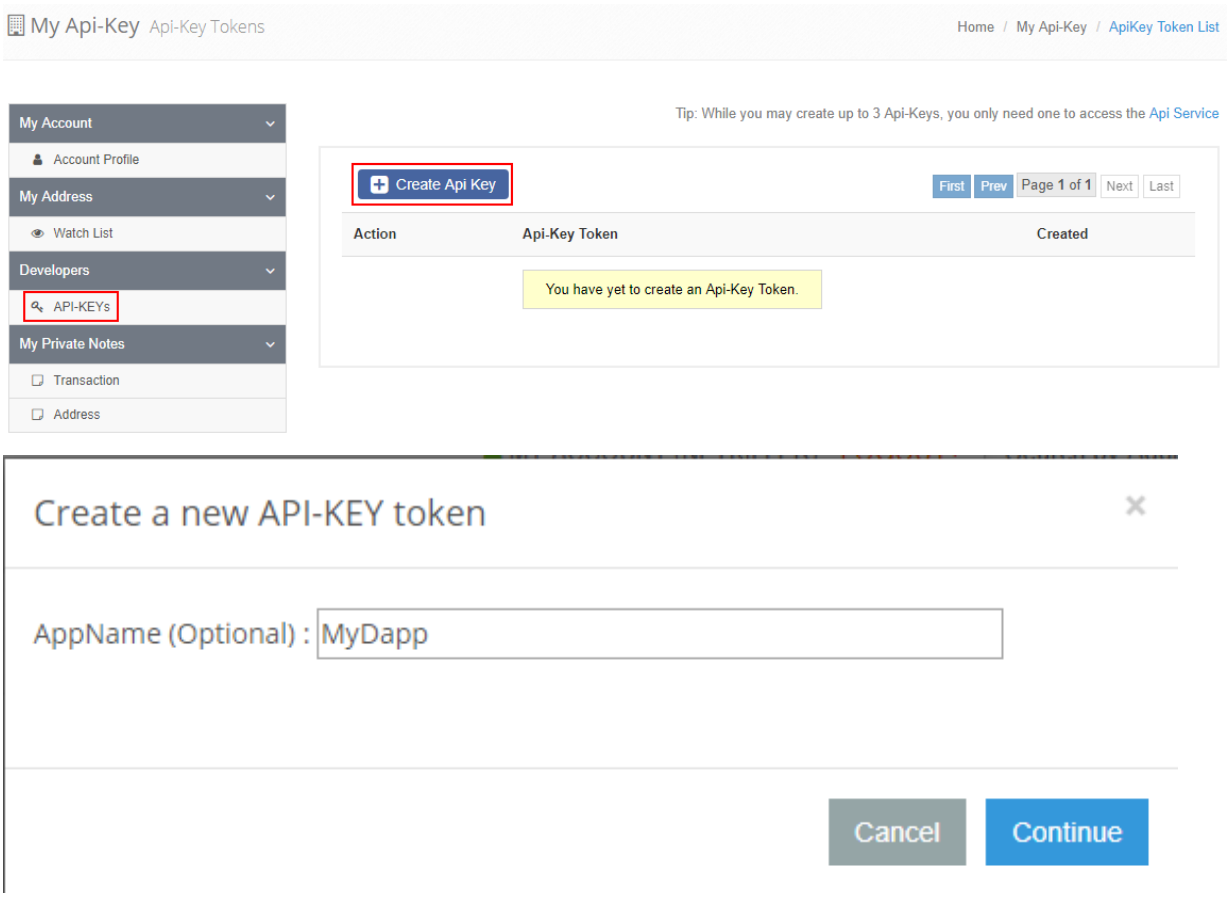
前往邮箱查看以太坊发出的确认邮件，并点击里面链接地址。


Register a new account
Status: Account Pending Email Verification
Your account has been successfully registered.
Please check your mailbox for your validation Email and ensure that your spam filters allow emails from noreply@etherscan.io

登陆

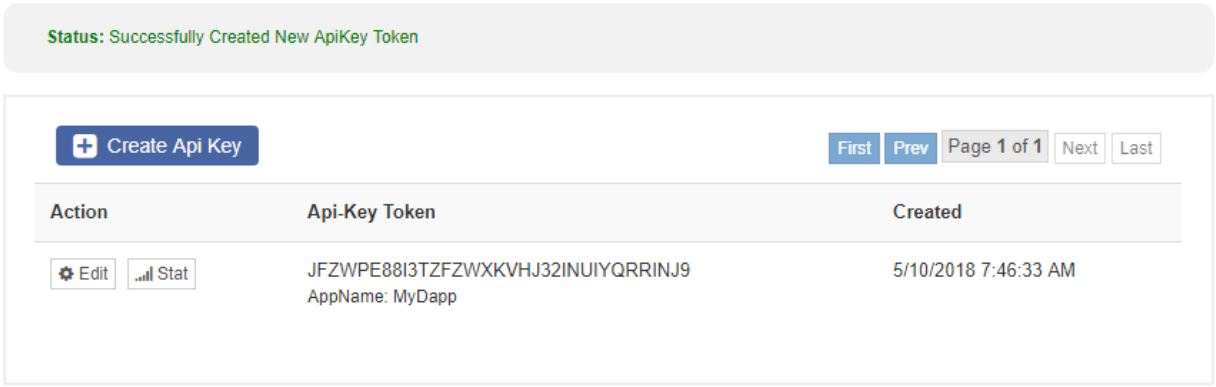

Login to your account
Or Register For A NEW Account
netkiller
Remember
Login
I'm not a robot
reCAPTCHA
Privacy - Terms
Forgot your Password ?
No worries, Click Here to reset your password.

创建 API Key


My Api-Key
Api-Key Tokens
Home / My Api-Key / ApiKey Token List
My Account
Account Profile
My Address
Watch List
Developers
API-KEYs
My Private Notes
Transaction
Address
Tip: While you may create up to 3 Api-Keys, you only need one to access the Api Service
Create Api Key
First Prev Page 1 of 1 Next Last
Action
Api-Key Token
Created
You have yet to create an Api-Key Token.

Create a new API-KEY token
AppName (Optional) :
MyDapp
Cancel
Continue

创建完成


Status: Successfully Created New ApiKey Token
Create Api Key
First Prev Page 1 of 1 Next Last
Action
Api-Key Token
Created
Edit
Stat
JFZWPE88I3TZFZWXKVHJ32INUIYQRRINJ9
AppName: MyDapp
5/10/2018 7:46:33 AM

2. 账号

2.1. 余额

获得账号的余额

```
https://api.etherscan.io/api?
module=account&action=balance&address=0xddbd2b932c763ba5b1b7ae3b
362eac3e8d40121a&tag=latest&apikey=JFZWPE88I3TZFZWXKVHJ32INUIYQR
RINJ9
```

2.2. 查询区块

```
http://api.etherscan.io/api?
module=account&action=txlist&address=0x3e827461Cc53ed7c75A29187C
fF39629FCAE3661&startblock=0&endblock=99999999&sort=asc&apikey=R
T5JW37AKEZVSW3C91Z86IGI2FF7JDPF1N
```

Spring boot 2.0.2 实现交易记录查询

```
package cn.netkiller.api.restful;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
```

```
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.client.RestTemplate;

@RestController
public class TransactionsRestController {
	private static final Logger logger =
LoggerFactory.getLogger(TransactionsRestController.class);

	private static final String url =
"http://api.etherscan.io/api?module={module}&action=
{action}&address={address}&startblock={startblock}&endblock=
{endblock}&sort={sort}&apikey={apikey}";

	@Autowired
	public TestRestController() {
		// TODO Auto-generated constructor stub
	}

	@GetMapping("/block/{startblock}/{endblock}/{address}")
	public TransactionsResponse block(@PathVariable String
startblock, @PathVariable String endblock, @PathVariable String
address) {
		return this.getTransactionsByAddress(startblock,
endblock, address);
	}

	private TransactionsResponse
getTransactionsByAddress(String startblock, String endblock,
String address) {
		Map<String, String> params = new HashMap<String,
String>();
		params.put("module", "account");
		params.put("action", "txlist");
		params.put("address", address);
		params.put("startblock", startblock);
		params.put("endblock", endblock);
		params.put("sort", "asc");
		params.put("apikey",
"RT5JW37AKEZVSW3C91Z86IGI2FF7JDPF1N");
		RestTemplate restTemplate = new RestTemplate();
		TransactionsResponse result =
restTemplate.getForObject(url, TransactionsResponse.class,
params);
		logger.info(params.toString());
		logger.info(result.toString());
		return result;
	}
}
```

```
class Transactions {

        private String blockNumber;
        private String timeStamp;
        private String hash;
        private String nonce;
        private String blockHash;
        private String transactionIndex;
        private String from;
        private String to;
        private String value;
        private String gas;
        private String gasPrice;
        private String isError;
        private String txreceipt_status;
        private String input;
        private String contractAddress;
        private String cumulativeGasUsed;
        private String gasUsed;
        private String confirmations;

        public Transactions() {
        }

        public String getBlockNumber() {
                return blockNumber;
        }

        public void setBlockNumber(String blockNumber) {
                this.blockNumber = blockNumber;
        }

        public String getTimeStamp() {
                return timeStamp;
        }

        public void setTimeStamp(String timeStamp) {
                this.timeStamp = timeStamp;
        }

        public String getHash() {
                return hash;
        }

        public void setHash(String hash) {
                this.hash = hash;
        }
```

```
	public String getNonce() {
		return nonce;
	}

	public void setNonce(String nonce) {
		this.nonce = nonce;
	}

	public String getBlockHash() {
		return blockHash;
	}

	public void setBlockHash(String blockHash) {
		this.blockHash = blockHash;
	}

	public String getTransactionIndex() {
		return transactionIndex;
	}

	public void setTransactionIndex(String transactionIndex)
{
		this.transactionIndex = transactionIndex;
	}

	public String getFrom() {
		return from;
	}

	public void setFrom(String from) {
		this.from = from;
	}

	public String getTo() {
		return to;
	}

	public void setTo(String to) {
		this.to = to;
	}

	public String getValue() {
		return value;
	}

	public void setValue(String value) {
		this.value = value;
```

```
	}

	public String getGas() {
		return gas;
	}

	public void setGas(String gas) {
		this.gas = gas;
	}

	public String getGasPrice() {
		return gasPrice;
	}

	public void setGasPrice(String gasPrice) {
		this.gasPrice = gasPrice;
	}

	public String getIsError() {
		return isError;
	}

	public void setIsError(String isError) {
		this.isError = isError;
	}

	public String getTxreceipt_status() {
		return txreceipt_status;
	}

	public void setTxreceipt_status(String txreceipt_status)
{
		this.txreceipt_status = txreceipt_status;
	}

	public String getInput() {
		return input;
	}

	public void setInput(String input) {
		this.input = input;
	}

	public String getContractAddress() {
		return contractAddress;
	}

	public void setContractAddress(String contractAddress) {
```

```
                this.contractAddress = contractAddress;
        }

        public String getCumulativeGasUsed() {
                return cumulativeGasUsed;
        }

        public void setCumulativeGasUsed(String
cumulativeGasUsed) {
                this.cumulativeGasUsed = cumulativeGasUsed;
        }

        public String getGasUsed() {
                return gasUsed;
        }

        public void setGasUsed(String gasUsed) {
                this.gasUsed = gasUsed;
        }

        public String getConfirmations() {
                return confirmations;
        }

        public void setConfirmations(String confirmations) {
                this.confirmations = confirmations;
        }

        @Override
        public String toString() {
                return "Transactions [blockNumber=" +
blockNumber + ", timeStamp=" + timeStamp + ", hash=" + hash + ",
nonce=" + nonce + ", blockHash=" + blockHash + ",
transactionIndex=" + transactionIndex + ", from=" + from + ",
to=" + to + ", value=" + value + ", gas=" + gas + ", gasPrice="
+ gasPrice + ", isError=" + isError + ", txreceipt_status=" +
txreceipt_status + ", input=" + input + ", contractAddress=" +
contractAddress + ", cumulativeGasUsed=" + cumulativeGasUsed +
", gasUsed=" + gasUsed + ", confirmations=" + confirmations +
"]";
        }

}

class TransactionsResponse {
        private String status;
        private String message;
        private List<Transactions> result = new
```

```
ArrayList<Transactions>();

        public TransactionsResponse() {
        }

        public String getStatus() {
                return status;
        }

        public void setStatus(String status) {
                this.status = status;
        }

        public String getMessage() {
                return message;
        }

        public void setMessage(String message) {
                this.message = message;
        }

        public List<Transactions> getResult() {
                return result;
        }

        public void setResult(List<Transactions> result) {
                this.result = result;
        }

        @Override
        public String toString() {
                return "TransactionsResponse [status=" + status
+ ", message=" + message + ", result=" + result + "]";
        }

}
```

2.3. 查询区块

```
http://api.etherscan.io/api?
module=account&action=txlistinternal&address=0x3e827461Cc53ed7c7
5A29187CfF39629FCAE3661&startblock=0&endblock=99999999&sort=asc&
apikey=RT5JW37AKEZVSW3C91Z86IGI2FF7JDPF1N
```

3. 查询交易

3.1. 检查合约执行状态

```
https://api.etherscan.io/api?
module=transaction&action=getstatus&txhash=0x6cec54f03e6d9271f2
c51e0696de03df4bba82171f2aff464c63554614915500&apikey=RT5JW37AK
EZVSW3C91Z86IGI2FF7JDPF1N
```

4. Geth/Parity Proxy APIs

4.1.

```
https://api.etherscan.io/api?
module=proxy&action=eth_blockNumber&apikey=RT5JW37AKEZVSW3C91Z8
6IGI2FF7JDPF1N
```

```
        private String getBlockNumber() {
                final String url =
"https://api.etherscan.io/api?
module=proxy&action=eth_blockNumber&apikey=RT5JW37AKEZVSW3C91Z8
6IGI2FF7JDPF1N";
                Map<String, String> params = new
HashMap<String, String>();
                params.put("module", "proxy");
                params.put("action", "eth_blockNumber");
                params.put("apikey",
"RT5JW37AKEZVSW3C91Z86IGI2FF7JDPF1N");
                RestTemplate restTemplate = new RestTemplate();
                JsonRpc result = restTemplate.getForObject(url,
JsonRpc.class, params);

                return
Integer.valueOf(result.getResult().substring(2),
16).toString();
        }
```

5. JSON RPC 原生交口调用

JSON RPC 文档 https://github.com/ethereum/wiki/wiki/JSON-RPC#eth_getlogs

```
neo@MacBook-Pro ~ % curl -XPOST -H "Accept: application/json" -H
"Content-Type: application/json" -d
'{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0x5c18a33df2cc41a1beddc91133b8422e89f041b7", "latest"],"id":1}'
http://localhost:8545
{"jsonrpc":"2.0","id":1,"result":"0x153e5f3aeb667075800"}
```

```
curl -X POST --data '{"jsonrpc":"2.0","method":"eth_accounts","params":
[],"id":1}'

curl -X POST --data
'{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0x407d73d8a49eeb85d32cf465507dd71d507100c1", "latest"],"id":1}'
```

第 21 章 infura

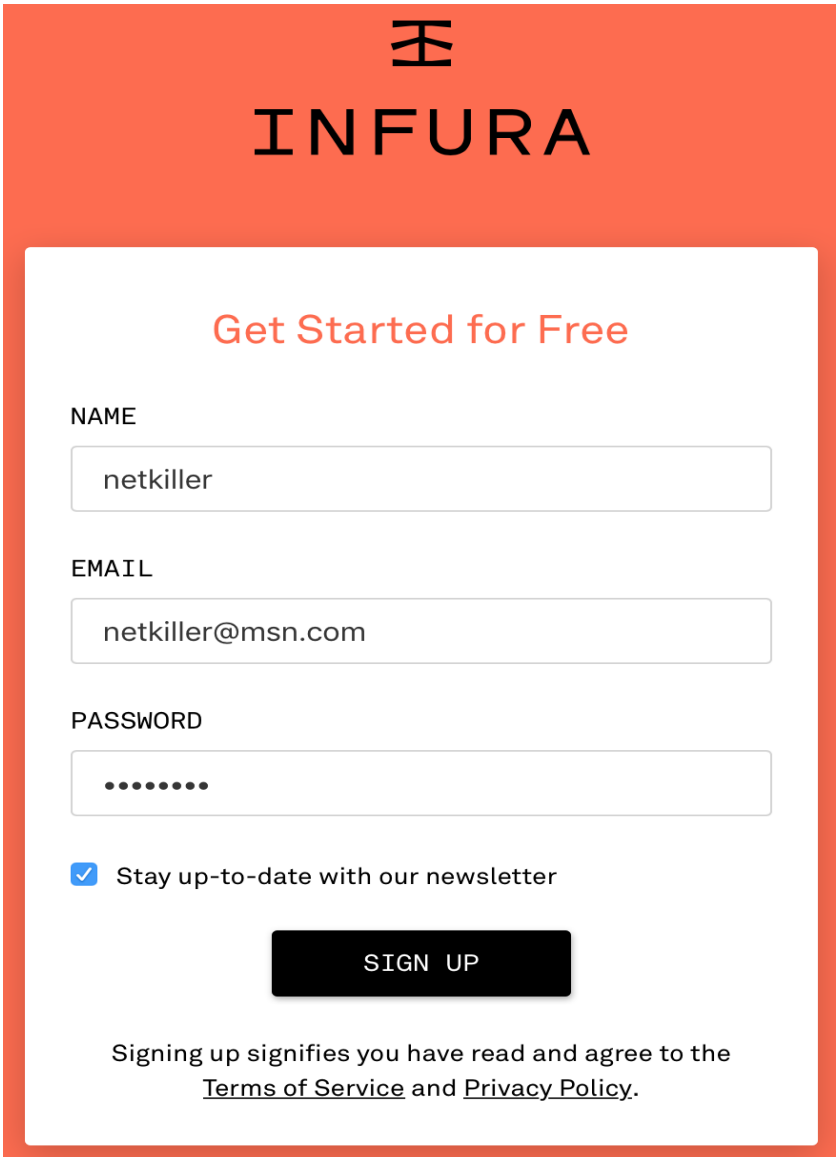
https://infura.io/

SCALABLE BLOCKCHAIN INFRASTRUCTURE

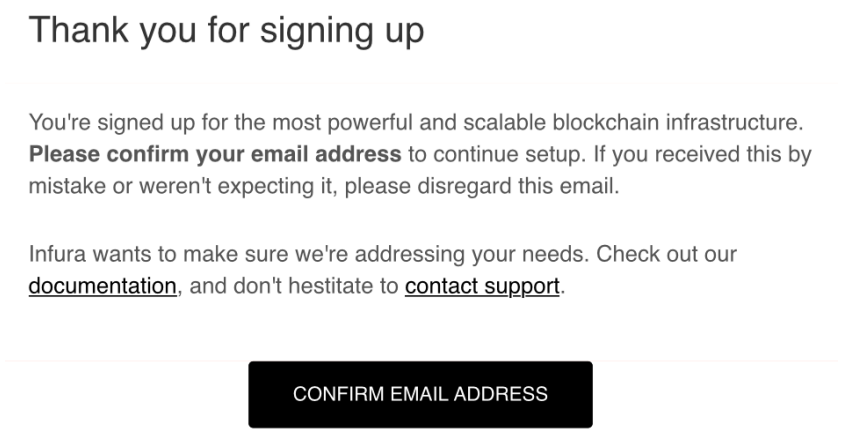
We provide secure, reliable, and scalable access to Ethereum and IPFS.

1. Infura 3.0

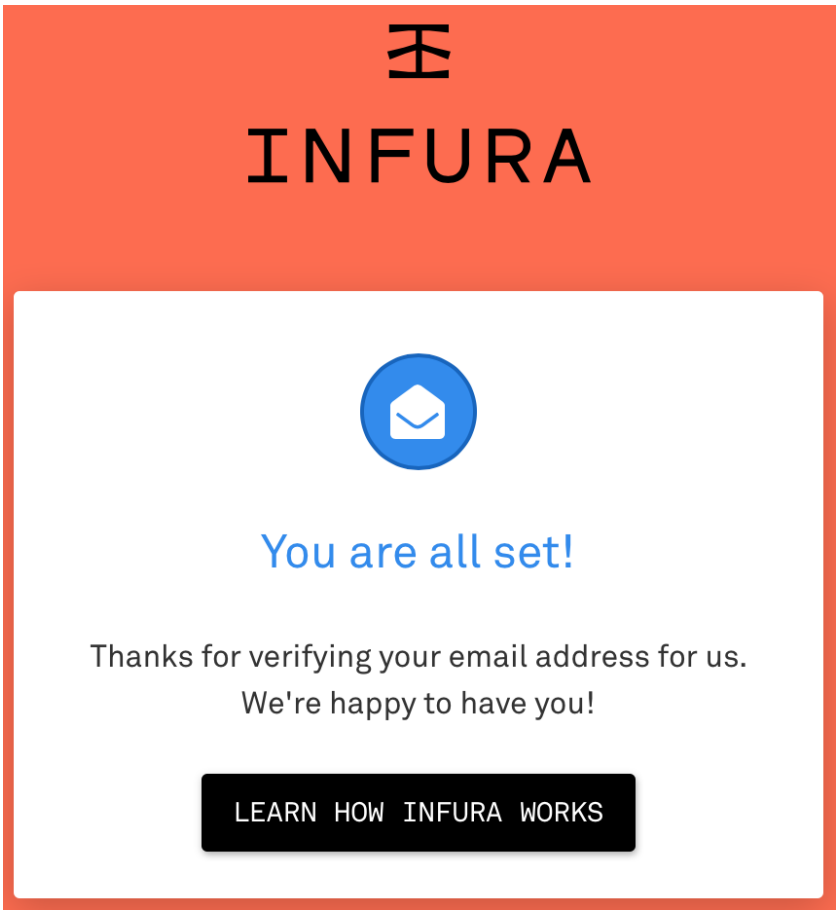
注册 Infura 3.0



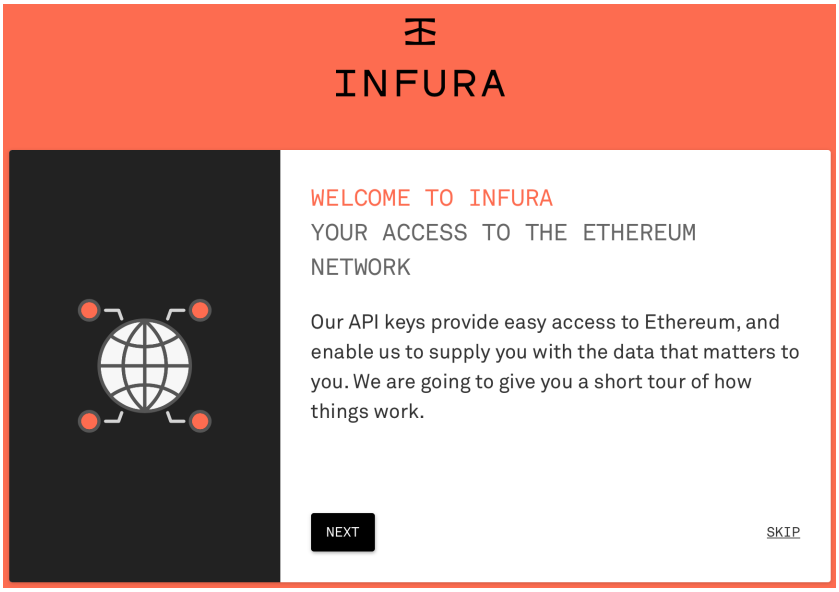
邮件确认



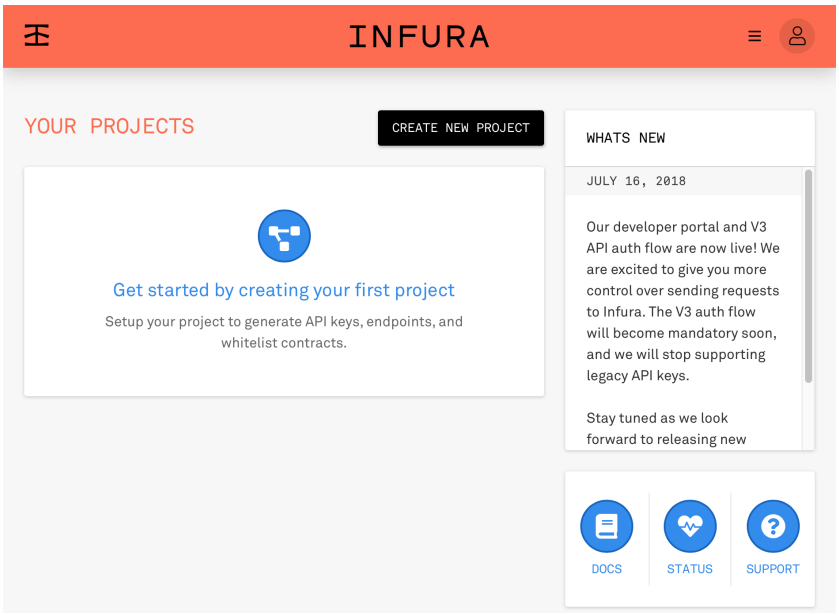
登录



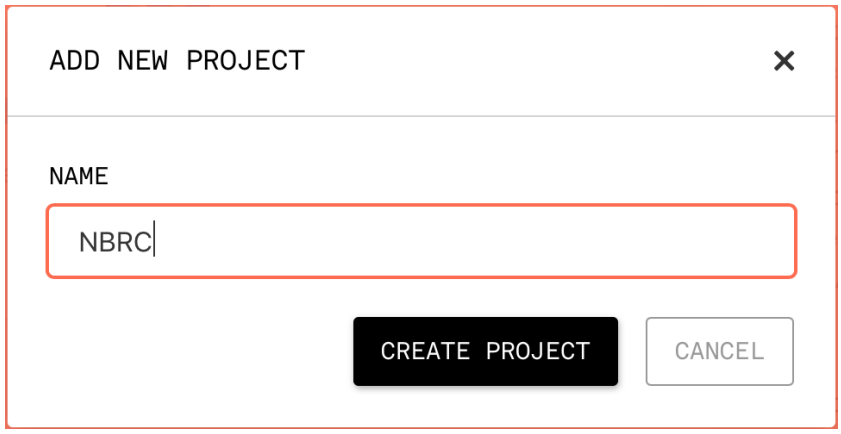
点击 SKIP 按钮，跳过教学模式


INFURA
WELCOME TO INFURA
YOUR ACCESS TO THE ETHEREUM NETWORK
Our API keys provide easy access to Ethereum, and enable us to supply you with the data that matters to you. We are going to give you a short tour of how things work.
NEXT
SKIP

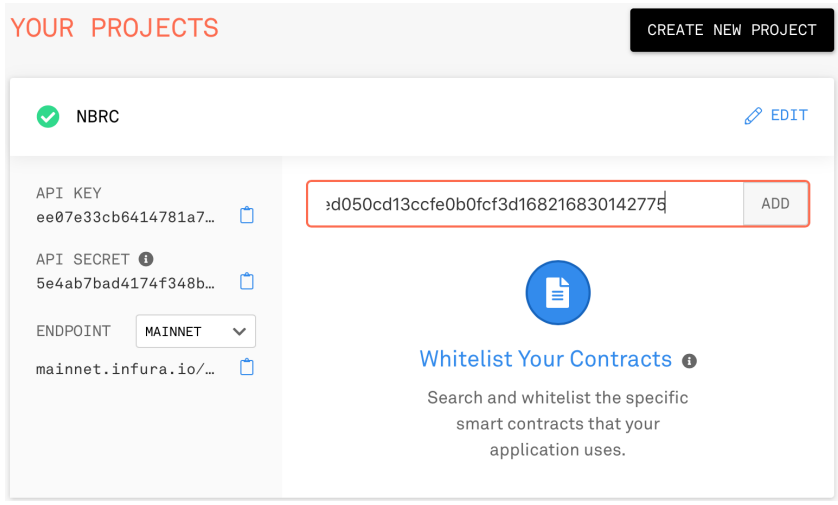
创建项目


INFURA
YOUR PROJECTS
CREATE NEW PROJECT
Get started by creating your first project
Setup your project to generate API keys, endpoints, and whitelist contracts.
WHATS NEW
JULY 16, 2018
Our developer portal and V3 API auth flow are now live! We are excited to give you more control over sending requests to Infura. The V3 auth flow will become mandatory soon, and we will stop supporting legacy API keys.
Stay tuned as we look forward to releasing new
DOCS
STATUS
SUPPORT

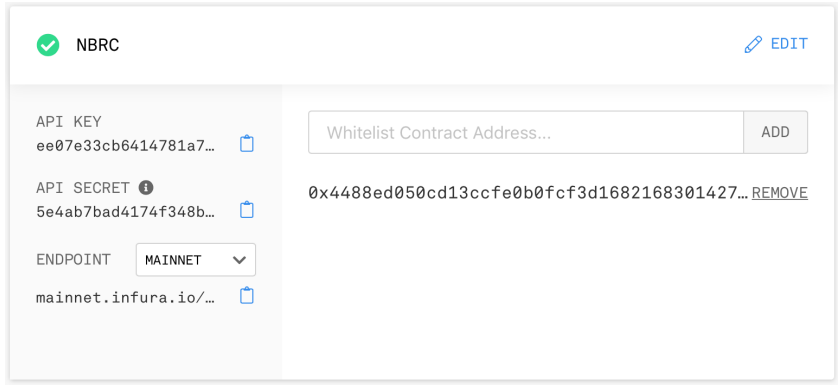
输入项目名称



添加合约到白名单



添加完成



复制 Infura 地址:

```
https://mainnet.infura.io/v3/ee07e33cb6414781a72deaf3b303ca3b
https://ropsten.infura.io/v3/ee07e33cb6414781a72deaf3b303ca3b
```

2. websocket

```
wss://mainnet.infura.io/ws
wss://ropsten.infura.io/ws
wss://rinkeby.infura.io/ws
```

2.1. 订阅 newBlockHeaders

```
#!/usr/bin/env node
const Web3 = require('web3');

const web3 = new Web3(new
Web3.providers.WebsocketProvider('wss://mainnet.infura.io/ws'))
;

const subscription = web3.eth.subscribe('newBlockHeaders',
(error, blockHeader) => {
  if (error) return console.error(error);

  console.log('Successfully subscribed!', blockHeader);
}).on('data', (blockHeader) => {
  console.log('data: ', blockHeader);
});

// unsubscribes the subscription
subscription.unsubscribe((error, success) => {
  if (error) return console.error(error);

  console.log('Successfully unsubscribed!');
});
```

3. 配置 Truffle

安装 truffle-hdwallet-provider

```
$ npm install truffle-hdwallet-provider
```

修改 truffle.js 文件

```
var HDWalletProvider = require("truffle-hdwallet-provider");

// infura 为你提供的 apikey 请与你申请到的 key 保持一致
var infura_apikey = "CsS9shwaAab0z7B4LP2d";

// 你以太坊钱包地址 进入 MetaMask -> Settings -> reveal seed words
复制到此处
var mnemonic = "drill hunt food team moment mistake bird
attitude tunnel ecology sister resist";

module.exports = {
  networks: {
    development: {
      host: "127.0.0.1",
      port: 7545,
      network_id: "*"
    },
    private: {
      host: "localhost",
      port: 8545,
      network_id: "*" // Match any network id
    },
    ropsten: {
      provider: new HDWalletProvider(mnemonic,
"https://ropsten.infura.io/"+infura_apikey),
      network_id: 3,
      gas: 3012388,
      gasPrice: 30000000000
    },
```

```
    main: {
      provider: new HDWalletProvider(mnemonic,
"https://mainnet.infura.io/"+infura_apikey),
      network_id: 3,
      gas: 3012388,
      gasPrice: 1000000000
    }
  }
};
```

部署合约到 ropsten 测试网， 在命令行输入如下命令，通过 --network 设置发布的目标网络：

```
truffle migrate --network ropsten
```

主网络发布

```
truffle migrate --network main
```

4. infura.io web3.js 开发

4.1. Web3 通过 infura 连接到 Ropsten 测试网络

```
fs = require('fs');
const Web3 = require('web3');
const web3 = new
Web3('https://ropsten.infura.io/CsS9shwaAab0z7B4LP2d');
console.log(web3.version)
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x70682386d0dE84B1e549DC3c4305CCB2D261b2a8";
const coinbase = "0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7";
const toAddress = "0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";

balanceWei = web3.eth.getBalance(coinbase);
console.log(balanceWei);

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});

contract.methods.balanceOf(coinbase).call().then(console.log).ca
tch(console.error);
contract.methods.balanceOf(toAddress).call().then(console.log).c
atch(console.error);
```

4.2. 使用 truffle-hdwallet-provider 连接到 https://ropsten.infura.io

```
fs = require('fs');
const Web3 = require('web3');
const HDWalletProvider = require("truffle-hdwallet-provider");
const mnemonic = "drill hunt food team moment mistake bird
attitude tunnel ecology sister resist";
const web3 = new Web3(new
HDWalletProvider(mnemonic,'https://ropsten.infura.io/CsS8shwaCab
```

```
0a7B4LP2d'));
console.log(web3.version)
const abi = fs.readFileSync('output/TokenERC20.abi', 'utf-8');

const contractAddress =
"0x70682386d0dE84B1e549DC3c4305CCB2D261b2a8";
const coinbase = "0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7";
const toAddress = "0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";

balanceWei = web3.eth.getBalance(coinbase);
console.log(balanceWei);

const contract = new web3.eth.Contract(JSON.parse(abi),
contractAddress, { from: coinbase , gas: 100000});

contract.methods.balanceOf(coinbase).call().then(console.log).ca
tch(console.log);
contract.methods.balanceOf(toAddress).call().then(console.log).c
atch(console.log);
```

4.3. 转账

```
        const coinbase =
"0xB94054c174995AE2A9E7fcf6c7924635FBa8ECF7";
        const toAddress =
"0xf56b81a2bcb964D2806071e9Be4289A5559BB0fA";
        const privateKey =
"e33ea581d88e0bd2270c0fd109604039a3de59671b6d69882b4cb4688d3dcff
d"

        var nonce = await
web3.eth.getTransactionCount(coinbase);
        var gasPrice = await web3.eth.getGasPrice();
        console.log(`gasPrice: ${gasPrice}\n`)
    var gasLimit = 1000000;
        var transferAmount = 1000;
    var chainId = 1;

    var rawTransaction = {
        "from": coinbase,
        "nonce":  web3.utils.toHex(count),
        "gasPrice": web3.utils.toHex(gasPrice),
        "gasLimit": web3.utils.toHex(gasLimit),
```

```
        "to": toAddress,
        "value": "100",
        "data": "0x0",
        "chainId": web3.utils.toHex(chainId)
    };

    var privateKey = new Buffer(privateKey, 'hex');
    var tx = new Tx(rawTransaction);
    tx.sign(privateKey);
    var serializedTx = tx.serialize();

    web3.eth.sendSignedTransaction('0x' +
serializedTx.toString('hex')).on('receipt', console.log);
```

```
'use strict';
const Web3 = require('web3');

const wsAddress = 'wss://rinkeby.infura.io/ws';
const contractJson = '(taken from solc or remix online
compiler)';
const privateKey = '0xOOOX';
const contractAddress = '0xOOOX';
const walletAddress = '0xOOOX';

const webSocketProvider = new
Web3.providers.WebsocketProvider(wsAddress);
const web3 = new Web3(new
Web3.providers.WebsocketProvider(webSocketProvider));
const contract = new web3.eth.Contract(
  JSON.parse(contractJson),
  contractAddress
);
// change this to whatever contract method you are trying to
call, E.G. SimpleStore("Hello World")
const query = contract.methods.SimpleStore('Hello World');
const encodedABI = query.encodeABI();
const tx = {
  from: walletAddress,
  to: contractAddress,
  gas: 2000000,
  data: encodedABI,
};

const account =
```

```
web3.eth.accounts.privateKeyToAccount(privateKey);
console.log(account);
web3.eth.getBalance(walletAddress).then(console.log);

web3.eth.accounts.signTransaction(tx, privateKey).then(signed =>
{
  const tran = web3.eth
    .sendSignedTransaction(signed.rawTransaction)
    .on('confirmation', (confirmationNumber, receipt) => {
      console.log('=> confirmation: ' + confirmationNumber);
    })
    .on('transactionHash', hash => {
      console.log('=> hash');
      console.log(hash);
    })
    .on('receipt', receipt => {
      console.log('=> reciept');
      console.log(receipt);
    })
    .on('error', console.error);
});
```

4.4. 执行合约

```
const fs = require('fs');
const Web3 = require('web3');
const web3 = new
Web3("https://mainnet.infura.io/CsS9shwaAab0z7B4LP2d");
const Tx = require('ethereumjs-tx');

const abi = [{"constant":true,"inputs":
[],"name":"name","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability":
"view","type":"function"},{"constant":false,"inputs":
[{"name":"_status","type":"bool"}],"name":"setAirdropStatus","ou
tputs":
[{"name":"status","type":"bool"}],"payable":false,"stateMutabili
ty":"nonpayable","type":"function"},{"constant":false,"inputs":
[{"name":"_spender","type":"address"},
{"name":"_value","type":"uint256"}],"name":"approve","outputs":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabil
ity":"nonpayable","type":"function"},{"constant":true,"inputs":
[],"name":"totalSupply","outputs":
```

```
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"_from","type":"address"},
{"name":"_to","type":"address"},
{"name":"_value","type":"uint256"}],"name":"transferFrom","outpu
ts":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabil
ity":"nonpayable","type":"function"},{"constant":true,"inputs":
[],"name":"airdropStatus","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"v
iew","type":"function"},{"constant":true,"inputs":
[],"name":"decimals","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"_mintedAmount","type":"uint256"}],"name":"mintAirdropT
oken","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"functi
on"},{"constant":false,"inputs":
[{"name":"_value","type":"uint256"}],"name":"burn","outputs":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabil
ity":"nonpayable","type":"function"},{"constant":false,"inputs":
[{"name":"_lock","type":"bool"}],"name":"setLock","outputs":
[{"name":"status","type":"bool"}],"payable":false,"stateMutabili
ty":"nonpayable","type":"function"},{"constant":false,"inputs":
[{"name":"_address","type":"address"}],"name":"balanceOf","outpu
ts":
[{"name":"balance","type":"uint256"}],"payable":false,"stateMuta
bility":"nonpayable","type":"function"},
{"constant":true,"inputs":
[],"name":"airdropCurrentTotal","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":false,"inputs":
[{"name":"target","type":"address"},
{"name":"mintedAmount","type":"uint256"}],"name":"mintToken","ou
tputs":
[],"payable":false,"stateMutability":"nonpayable","type":"functi
on"},{"constant":false,"inputs":
[{"name":"_from","type":"address"},
{"name":"_value","type":"uint256"}],"name":"burnFrom","outputs":
[{"name":"success","type":"bool"}],"payable":false,"stateMutabil
ity":"nonpayable","type":"function"},{"constant":true,"inputs":
[],"name":"owner","outputs":
[{"name":"","type":"address"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":true,"inputs":
[],"name":"symbol","outputs":
[{"name":"","type":"string"}],"payable":false,"stateMutability":
"view","type":"function"},{"constant":false,"inputs":
[{"name":"_amount","type":"uint256"}],"name":"setAirdropAmount",
```

```
"outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"functi
on"},{"constant":false,"inputs":
[{"name":"_to","type":"address"},
{"name":"_value","type":"uint256"}],"name":"transfer","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"functi
on"},{"constant":true,"inputs":
[{"name":"","type":"address"}],"name":"frozenAccount","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"v
iew","type":"function"},{"constant":true,"inputs":
[],"name":"airdropTotalSupply","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"constant":true,"inputs":
[{"name":"_owner","type":"address"},
{"name":"_spender","type":"address"}],"name":"allowance","output
s":
[{"name":"remaining","type":"uint256"}],"payable":false,"stateMu
tability":"view","type":"function"},{"constant":true,"inputs":
[{"name":"","type":"address"}],"name":"touched","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"v
iew","type":"function"},{"constant":false,"inputs":
[{"name":"target","type":"address"},
{"name":"freeze","type":"bool"}],"name":"freezeAccount","outputs
":
[],"payable":false,"stateMutability":"nonpayable","type":"functi
on"},{"constant":false,"inputs":
[{"name":"newOwner","type":"address"}],"name":"transferOwnership
","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"functi
on"},{"constant":true,"inputs":[],"name":"lock","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"v
iew","type":"function"},{"constant":true,"inputs":
[],"name":"airdropAmount","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability"
:"view","type":"function"},{"inputs":
[{"name":"initialSupply","type":"uint256"},
{"name":"tokenName","type":"string"},
{"name":"tokenSymbol","type":"string"},
{"name":"decimalUnits","type":"uint256"}],"payable":false,"state
Mutability":"nonpayable","type":"constructor"},
{"anonymous":false,"inputs":
[{"indexed":true,"name":"from","type":"address"},
{"indexed":true,"name":"to","type":"address"},
{"indexed":false,"name":"value","type":"uint256"}],"name":"Trans
fer","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"from","type":"address"},
{"indexed":false,"name":"value","type":"uint256"}],"name":"Burn"
,"type":"event"},{"anonymous":false,"inputs":
```

```
[{"indexed":true,"name":"owner","type":"address"},
{"indexed":true,"name":"spender","type":"address"},
{"indexed":false,"name":"value","type":"uint256"}],"name":"Appro
val","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"target","type":"address"},
{"indexed":false,"name":"frozen","type":"bool"}],"name":"FrozenF
unds","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"target","type":"address"},
{"indexed":false,"name":"value","type":"uint256"}],"name":"AirDr
op","type":"event"}];
const address = "0x3e827461Cc53CAE366175A291ed7c629F87CfF39";
const key =
"19A57E4F6274AF1E0B9C3F8F7E3503876A850AFEE1912B8B9C5D9358EDEA036
2"

const contractAddress =
"0x44cCf3d1601427Fe0B0f7588eD058216830cd13C";
const contract = new web3.eth.Contract(abi, contractAddress, {
"from": address});

contract.methods.balanceOf(address).call().then(function(balance
){
    console.log(balance)
});

contract.methods.decimals().call().then(function(decimals){
    console.log(decimals)
});

web3.eth.getGasPrice().then(function(gasPrice){
    var price = Number(gasPrice);

    web3.eth.getTransactionCount(address).then(function(nonce){
        var amount = "1000000";

contract.methods.mintAirdropToken(amount).estimateGas().then(fun
ction(gas){
            var rawTransaction = {
                "nonce": web3.utils.toHex(nonce),
                "from": address,
                "to": contractAddress,
                "gas": web3.utils.toHex(gas),
                "gasPrice": web3.utils.toHex(price),
                // "gasLimit":
this.web3.utils.toHex(gasLimit.gasLimit),
                "value": "0x0",
                "data":
contract.methods.mintAirdropToken(amount).encodeABI()
```

```
            };

            console.log(rawTransaction);

            var privateKey = new Buffer.from(key, 'hex');
            var tx = new Tx(rawTransaction);
            tx.sign(privateKey);
            var serializedTx = tx.serialize();

            web3.eth.sendSignedTransaction('0x' +
serializedTx.toString('hex')).on('receipt', function(txhash){
                console.log(txhash);
            });
        });
    });
});

web3.eth.getGasPrice().then(function(gasPrice){
    var price = Number(gasPrice);

    web3.eth.getTransactionCount(address).then(function(nonce){
        var amount = "10";

contract.methods.setAirdropAmount(amount).estimateGas().then(fun
ction(gas){
            var rawTransaction = {
                "nonce": web3.utils.toHex(nonce),
                "from": address,
                "to": contractAddress,
                "gas": web3.utils.toHex(gas),
                "gasPrice": web3.utils.toHex(price),
                "value": "0x0",
                "data":
contract.methods.setAirdropAmount(amount).encodeABI()
            };

            console.log(rawTransaction);

            var privateKey = new Buffer.from(key, 'hex');
            var tx = new Tx(rawTransaction);
            tx.sign(privateKey);
            var serializedTx = tx.serialize();

            web3.eth.sendSignedTransaction('0x' +
serializedTx.toString('hex')).on('receipt', function(txhash){
                console.log(txhash);
            });
        });
```

```
    });
});

web3.eth.getGasPrice().then(function(gasPrice){
    var price = Number(gasPrice);

    web3.eth.getTransactionCount(address).then(function(nonce){
        var status = true;

contract.methods.setAirdropStatus(status).estimateGas().then(fun
ction(gas){
            var rawTransaction = {
                "nonce": web3.utils.toHex(nonce),
                "from": address,
                "to": contractAddress,
                "gas": web3.utils.toHex(gas),
                "gasPrice": web3.utils.toHex(price),
                "value": "0x0",
                "data":
contract.methods.setAirdropStatus(status).encodeABI()
            };
            console.log(rawTransaction);

            var privateKey = new Buffer.from(key, 'hex');
            var tx = new Tx(rawTransaction);
            tx.sign(privateKey);
            var serializedTx = tx.serialize();

            web3.eth.sendSignedTransaction('0x' +
serializedTx.toString('hex')).on('receipt', function(txhash){
                console.log(txhash);
            });
        });
    });
});
```

5. Infura IPFS

5.1. 上传文件

```
neo@MacBook-Pro /tmp % cat hello.txt
http://www.netkiller.cn

neo@MacBook-Pro /tmp % curl "https://ipfs.infura.io:5001/api/v0/add?
pin=false" \
    -X POST \
    -H "Content-Type: multipart/form-data" \
    -F file=@"hello.txt"
{"Name":"hello.txt","Hash":"QmToi4pgQH4LQX8wGMt5H8EV2dA7hD8S2EccRpd8YGUg
ac","Size":"32"}
```

5.2. 查看文件

```
neo@MacBook-Pro /tmp % curl "https://ipfs.infura.io:5001/api/v0/cat?
arg=QmToi4pgQH4LQX8wGMt5H8EV2dA7hD8S2EccRpd8YGUgac"
http://www.netkiller.cn
```

5.3. 下载文件

```
neo@MacBook-Pro /tmp % curl -s "https://ipfs.infura.io:5001/api/v0/get?
arg=QmToi4pgQH4LQX8wGMt5H8EV2dA7hD8S2EccRpd8YGUgac&archive=true" -o
hello.tgz

neo@MacBook-Pro /tmp % tar zxvf hello.tgz
x QmToi4pgQH4LQX8wGMt5H8EV2dA7hD8S2EccRpd8YGUgac

neo@MacBook-Pro /tmp % cat
QmToi4pgQH4LQX8wGMt5H8EV2dA7hD8S2EccRpd8YGUgac
http://www.netkiller.cn
```

5.4. 创建目录

```
neo@MacBook-Pro /tmp % curl
"https://ipfs.infura.io:5001/api/v0/files/mkdir?arg=/netkiller"

neo@MacBook-Pro /tmp % curl
"https://ipfs.infura.io:5001/api/v0/files/stat?arg=/netkiller"
{"Hash":"QmUNLLsPACCz1vLxQVkXqqLX5R1X345qqfHbsf67hvA3Nn","Size":0,"Cumul
ativeSize":4,"Blocks":0,"Type":"directory"}
```

5.5. 查看文件状态

```
neo@MacBook-Pro /tmp % curl
"https://ipfs.infura.io:5001/api/v0/files/stat?arg=/netkiller"
{"Hash":"QmUNLLsPACCz1vLxQVkXqqLX5R1X345qqfHbsf67hvA3Nn","Size":0,"Cumul
ativeSize":4,"Blocks":0,"Type":"directory"}
```

5.6. 查看IPFS版本号

```
neo@MacBook-Pro ~/workspace % curl
"https://ipfs.infura.io:5001/api/v0/version"
{"Version":"0.4.14","Commit":"","Repo":"6","System":"amd64/linux","Golan
g":"go1.10"}
```

6. Infura 2.0 (已经废弃)

6.1. 注册账号

https://infura.io/signup

输入用户名和邮箱即可注册

这些地址你需要牢记

NETWORK	DESCRIPTION	URL
Mainnet	production network	https://mainnet.infura.io/CsS9shwaAab0z7B4LP2d
Ropsten	test network	https://ropsten.infura.io/CsS9shwaAab0z7B4LP2d
INFURAnet	test network	https://infuranet.infura.io/CsS9shwaAab0z7B4LP2d
Kovan	test network	https://kovan.infura.io/CsS9shwaAab0z7B4LP2d
Rinkeby	test network	https://rinkeby.infura.io/CsS9shwaAab0z7B4LP2d
IPFS	gateway	https://ipfs.infura.io

NETWORK DESCRIPTION URL
Mainnet production network https://mainnet.infura.io/CsS9shwaAab0z7B4LP2d
Ropsten test network https://ropsten.infura.io/CsS9shwaAab0z7B4LP2d

INFURAnet test network https://infuranet.infura.io/CsS9shwaAab0z7B4LP2d
Kovan test network https://kovan.infura.io/CsS9shwaAab0z7B4LP2d
Rinkeby test network https://rinkeby.infura.io/CsS9shwaAab0z7B4LP2d
IPFS gateway https://ipfs.infura.io

这是你的 api key CsS9shwaAab0z7B4LP2d 请保存好

Mainnet 是以太坊主网

IPFS 是一个分布式区中心化的文件系统。

6.2. infura 接口

6.2.1. jsonrpc

JSONRPC 文档 https://github.com/INFURA/infura/blob/master/docs/source/index.html.md#choosing-a-client-to-handle-your-request

```
$ curl -X POST -H "Content-Type: application/json" \
  --data '{"jsonrpc": "2.0", "id": 1, "method": "eth_blockNumber", "params": []}' \
  "https://mainnet.infura.io/CsS9shwaAab0z7B4LP2d"
```

6.2.2. INFURA API

```
https://api.infura.io/v1/jsonrpc/mainnet/eth_blockNumber?token=CsS9shwaAab0z7B4LP2d
```

第 22 章 以太坊案例

1. EtherDelta

https://github.com/etherdelta/smart_contract

2. 以太猫（CryptoKitties）

首先你要拥有以太币，可以去交易所购买，然后使用谷歌浏览器进入下面网站，点击“Start meow” 按钮，再点击 “I Installed MetaMask” 安装浏览器插件，

以太猫官网 https://www.cryptokitties.co

合约源码地址

https://etherscan.io/address/0x06012c8cf97bead5deae237070f9587f8e7a266d#code

下面是以太猫智能合约：

```
pragma solidity ^0.4.11;


/**
 * @title Ownable
 * @dev The Ownable contract has an owner address, and provides
basic authorization control
 * functions, this simplifies the implementation of "user
permissions".
 */
contract Ownable {
  address public owner;


  /**
   * @dev The Ownable constructor sets the original `owner` of
the contract to the sender
   * account.
   */
  function Ownable() {
    owner = msg.sender;
  }
```

```
  /**
   * @dev Throws if called by any account other than the owner.
   */
  modifier onlyOwner() {
    require(msg.sender == owner);
    _;
  }


  /**
   * @dev Allows the current owner to transfer control of the
contract to a newOwner.
   * @param newOwner The address to transfer ownership to.
   */
  function transferOwnership(address newOwner) onlyOwner {
    if (newOwner != address(0)) {
      owner = newOwner;
    }
  }

}



/// @title Interface for contracts conforming to ERC-721: Non-
Fungible Tokens
/// @author Dieter Shirley <dete@axiomzen.co>
(https://github.com/dete)
contract ERC721 {
    // Required methods
    function totalSupply() public view returns (uint256 total);
    function balanceOf(address _owner) public view returns
(uint256 balance);
    function ownerOf(uint256 _tokenId) external view returns
(address owner);
    function approve(address _to, uint256 _tokenId) external;
    function transfer(address _to, uint256 _tokenId) external;
    function transferFrom(address _from, address _to, uint256
_tokenId) external;

    // Events
    event Transfer(address from, address to, uint256 tokenId);
    event Approval(address owner, address approved, uint256
tokenId);

    // Optional
    // function name() public view returns (string name);
    // function symbol() public view returns (string symbol);
```

```
    // function tokensOfOwner(address _owner) external view
returns (uint256[] tokenIds);
    // function tokenMetadata(uint256 _tokenId, string
_preferredTransport) public view returns (string infoUrl);

    // ERC-165 Compatibility
(https://github.com/ethereum/EIPs/issues/165)
    function supportsInterface(bytes4 _interfaceID) external
view returns (bool);
}


// // Auction wrapper functions


// Auction wrapper functions




/// @title SEKRETOOOO
contract GeneScienceInterface {
    /// @dev simply a boolean to indicate this is the contract
we expect to be
    function isGeneScience() public pure returns (bool);

    /// @dev given genes of kitten 1 & 2, return a genetic
combination - may have a random factor
    /// @param genes1 genes of mom
    /// @param genes2 genes of sire
    /// @return the genes that are supposed to be passed down
the child
    function mixGenes(uint256 genes1, uint256 genes2, uint256
targetBlock) public returns (uint256);
}




/// @title A facet of KittyCore that manages special access
privileges.
/// @author Axiom Zen (https://www.axiomzen.co)
```

```
/// @dev See the KittyCore contract documentation to understand
how the various contract facets are arranged.
contract KittyAccessControl {
    // This facet controls access control for CryptoKitties.
There are four roles managed here:
    //
    //     - The CEO: The CEO can reassign other roles and
change the addresses of our dependent smart
    //         contracts. It is also the only role that can
unpause the smart contract. It is initially
    //         set to the address that created the smart
contract in the KittyCore constructor.
    //
    //     - The CFO: The CFO can withdraw funds from KittyCore
and its auction contracts.
    //
    //     - The COO: The COO can release gen0 kitties to
auction, and mint promo cats.
    //
    // It should be noted that these roles are distinct without
overlap in their access abilities, the
    // abilities listed for each role above are exhaustive. In
particular, while the CEO can assign any
    // address to any role, the CEO address itself doesn't have
the ability to act in those roles. This
    // restriction is intentional so that we aren't tempted to
use the CEO address frequently out of
    // convenience. The less we use an address, the less likely
it is that we somehow compromise the
    // account.

    /// @dev Emited when contract is upgraded - See README.md
for updgrade plan
    event ContractUpgrade(address newContract);

    // The addresses of the accounts (or contracts) that can
execute actions within each roles.
    address public ceoAddress;
    address public cfoAddress;
    address public cooAddress;

    // @dev Keeps track whether the contract is paused. When
that is true, most actions are blocked
    bool public paused = false;

    /// @dev Access modifier for CEO-only functionality
    modifier onlyCEO() {
        require(msg.sender == ceoAddress);
```

```
        _;
    }

    /// @dev Access modifier for CFO-only functionality
    modifier onlyCFO() {
        require(msg.sender == cfoAddress);
        _;
    }

    /// @dev Access modifier for COO-only functionality
    modifier onlyCOO() {
        require(msg.sender == cooAddress);
        _;
    }

    modifier onlyCLevel() {
        require(
            msg.sender == cooAddress ||
            msg.sender == ceoAddress ||
            msg.sender == cfoAddress
        );
        _;
    }

    /// @dev Assigns a new address to act as the CEO. Only
available to the current CEO.
    /// @param _newCEO The address of the new CEO
    function setCEO(address _newCEO) external onlyCEO {
        require(_newCEO != address(0));

        ceoAddress = _newCEO;
    }

    /// @dev Assigns a new address to act as the CFO. Only
available to the current CEO.
    /// @param _newCFO The address of the new CFO
    function setCFO(address _newCFO) external onlyCEO {
        require(_newCFO != address(0));

        cfoAddress = _newCFO;
    }

    /// @dev Assigns a new address to act as the COO. Only
available to the current CEO.
    /// @param _newCOO The address of the new COO
    function setCOO(address _newCOO) external onlyCEO {
        require(_newCOO != address(0));
```

```
        cooAddress = _newCOO;
    }

    /*** Pausable functionality adapted from OpenZeppelin ***/

    /// @dev Modifier to allow actions only when the contract
IS NOT paused
    modifier whenNotPaused() {
        require(!paused);
        _;
    }

    /// @dev Modifier to allow actions only when the contract
IS paused
    modifier whenPaused {
        require(paused);
        _;
    }

    /// @dev Called by any "C-level" role to pause the
contract. Used only when
    ///  a bug or exploit is detected and we need to limit
damage.
    function pause() external onlyCLevel whenNotPaused {
        paused = true;
    }

    /// @dev Unpauses the smart contract. Can only be called by
the CEO, since
    ///  one reason we may pause the contract is when CFO or
COO accounts are
    ///  compromised.
    /// @notice This is public rather than external so it can
be called by
    ///  derived contracts.
    function unpause() public onlyCEO whenPaused {
        // can't unpause if contract was upgraded
        paused = false;
    }
}




/// @title Base contract for CryptoKitties. Holds all common
structs, events and base variables.
/// @author Axiom Zen (https://www.axiomzen.co)
/// @dev See the KittyCore contract documentation to understand
```

```
how the various contract facets are arranged.
contract KittyBase is KittyAccessControl {
    /*** EVENTS ***/

    /// @dev The Birth event is fired whenever a new kitten
comes into existence. This obviously
    ///  includes any time a cat is created through the
giveBirth method, but it is also called
    ///  when a new gen0 cat is created.
    event Birth(address owner, uint256 kittyId, uint256
matronId, uint256 sireId, uint256 genes);

    /// @dev Transfer event as defined in current draft of
ERC721. Emitted every time a kitten
    ///  ownership is assigned, including births.
    event Transfer(address from, address to, uint256 tokenId);

    /*** DATA TYPES ***/

    /// @dev The main Kitty struct. Every cat in CryptoKitties
is represented by a copy
    ///  of this structure, so great care was taken to ensure
that it fits neatly into
    ///  exactly two 256-bit words. Note that the order of the
members in this structure
    ///  is important because of the byte-packing rules used by
Ethereum.
    ///  Ref:
http://solidity.readthedocs.io/en/develop/miscellaneous.html
    struct Kitty {
        // The Kitty's genetic code is packed into these 256-
bits, the format is
        // sooper-sekret! A cat's genes never change.
        uint256 genes;

        // The timestamp from the block when this cat came into
existence.
        uint64 birthTime;

        // The minimum timestamp after which this cat can
engage in breeding
        // activities again. This same timestamp is used for
the pregnancy
        // timer (for matrons) as well as the siring cooldown.
        uint64 cooldownEndBlock;

        // The ID of the parents of this kitty, set to 0 for
gen0 cats.
```

```
        // Note that using 32-bit unsigned integers limits us
to a "mere"
        // 4 billion cats. This number might seem small until
you realize
        // that Ethereum currently has a limit of about 500
million
        // transactions per year! So, this definitely won't be
a problem
        // for several years (even as Ethereum learns to
scale).
        uint32 matronId;
        uint32 sireId;

        // Set to the ID of the sire cat for matrons that are
pregnant,
        // zero otherwise. A non-zero value here is how we know
a cat
        // is pregnant. Used to retrieve the genetic material
for the new
        // kitten when the birth transpires.
        uint32 siringWithId;

        // Set to the index in the cooldown array (see below)
that represents
        // the current cooldown duration for this Kitty. This
starts at zero
        // for gen0 cats, and is initialized to
floor(generation/2) for others.
        // Incremented by one for each successful breeding
action, regardless
        // of whether this cat is acting as matron or sire.
        uint16 cooldownIndex;

        // The "generation number" of this cat. Cats minted by
the CK contract
        // for sale are called "gen0" and have a generation
number of 0. The
        // generation number of all other cats is the larger of
the two generation
        // numbers of their parents, plus one.
        // (i.e. max(matron.generation, sire.generation) + 1)
        uint16 generation;
    }

    /*** CONSTANTS ***/

    /// @dev A lookup table indicating the cooldown duration
after any successful
```

```
    ///  breeding action, called "pregnancy time" for matrons
and "siring cooldown"
    ///  for sires. Designed such that the cooldown roughly
doubles each time a cat
    ///  is bred, encouraging owners not to just keep breeding
the same cat over
    ///  and over again. Caps out at one week (a cat can breed
an unbounded number
    ///  of times, and the maximum cooldown is always seven
days).
    uint32[14] public cooldowns = [
        uint32(1 minutes),
        uint32(2 minutes),
        uint32(5 minutes),
        uint32(10 minutes),
        uint32(30 minutes),
        uint32(1 hours),
        uint32(2 hours),
        uint32(4 hours),
        uint32(8 hours),
        uint32(16 hours),
        uint32(1 days),
        uint32(2 days),
        uint32(4 days),
        uint32(7 days)
    ];

    // An approximation of currently how many seconds are in
between blocks.
    uint256 public secondsPerBlock = 15;

    /*** STORAGE ***/

    /// @dev An array containing the Kitty struct for all
Kitties in existence. The ID
    ///  of each cat is actually an index into this array. Note
that ID 0 is a negacat,
    ///  the unKitty, the mythical beast that is the parent of
all gen0 cats. A bizarre
    ///  creature that is both matron and sire... to itself!
Has an invalid genetic code.
    ///  In other words, cat ID 0 is invalid... ;-)
    Kitty[] kitties;

    /// @dev A mapping from cat IDs to the address that owns
them. All cats have
    ///  some valid owner address, even gen0 cats are created
with a non-zero owner.
```

```
    mapping (uint256 => address) public kittyIndexToOwner;

    // @dev A mapping from owner address to count of tokens
that address owns.
    //  Used internally inside balanceOf() to resolve ownership
count.
    mapping (address => uint256) ownershipTokenCount;

    /// @dev A mapping from KittyIDs to an address that has
been approved to call
    ///  transferFrom(). Each Kitty can only have one approved
address for transfer
    ///  at any time. A zero value means no approval is
outstanding.
    mapping (uint256 => address) public kittyIndexToApproved;

    /// @dev A mapping from KittyIDs to an address that has
been approved to use
    ///  this Kitty for siring via breedWith(). Each Kitty can
only have one approved
    ///  address for siring at any time. A zero value means no
approval is outstanding.
    mapping (uint256 => address) public sireAllowedToAddress;

    /// @dev The address of the ClockAuction contract that
handles sales of Kitties. This
    ///  same contract handles both peer-to-peer sales as well
as the gen0 sales which are
    ///  initiated every 15 minutes.
    SaleClockAuction public saleAuction;

    /// @dev The address of a custom ClockAuction subclassed
contract that handles siring
    ///  auctions. Needs to be separate from saleAuction
because the actions taken on success
    ///  after a sales and siring auction are quite different.
    SiringClockAuction public siringAuction;

    /// @dev Assigns ownership of a specific Kitty to an
address.
    function _transfer(address _from, address _to, uint256
_tokenId) internal {
        // Since the number of kittens is capped to 2^32 we
can't overflow this
        ownershipTokenCount[_to]++;
        // transfer ownership
        kittyIndexToOwner[_tokenId] = _to;
        // When creating new kittens _from is 0x0, but we can't
```

```
account that address.
        if (_from != address(0)) {
            ownershipTokenCount[_from]--;
            // once the kitten is transferred also clear sire
allowances
            delete sireAllowedToAddress[_tokenId];
            // clear any previously approved ownership exchange
            delete kittyIndexToApproved[_tokenId];
        }
        // Emit the transfer event.
        Transfer(_from, _to, _tokenId);
    }

    /// @dev An internal method that creates a new kitty and
stores it. This
    ///  method doesn't do any checking and should only be
called when the
    ///  input data is known to be valid. Will generate both a
Birth event
    ///  and a Transfer event.
    /// @param _matronId The kitty ID of the matron of this cat
(zero for gen0)
    /// @param _sireId The kitty ID of the sire of this cat
(zero for gen0)
    /// @param _generation The generation number of this cat,
must be computed by caller.
    /// @param _genes The kitty's genetic code.
    /// @param _owner The inital owner of this cat, must be
non-zero (except for the unKitty, ID 0)
    function _createKitty(
        uint256 _matronId,
        uint256 _sireId,
        uint256 _generation,
        uint256 _genes,
        address _owner
    )
        internal
        returns (uint)
    {
        // These requires are not strictly necessary, our
calling code should make
        // sure that these conditions are never broken.
However! _createKitty() is already
        // an expensive call (for storage), and it doesn't hurt
to be especially careful
        // to ensure our data structures are always valid.
        require(_matronId == uint256(uint32(_matronId)));
        require(_sireId == uint256(uint32(_sireId)));
```

```
        require(_generation == uint256(uint16(_generation)));

        // New kitty starts with the same cooldown as parent
gen/2
        uint16 cooldownIndex = uint16(_generation / 2);
        if (cooldownIndex > 13) {
            cooldownIndex = 13;
        }

        Kitty memory _kitty = Kitty({
            genes: _genes,
            birthTime: uint64(now),
            cooldownEndBlock: 0,
            matronId: uint32(_matronId),
            sireId: uint32(_sireId),
            siringWithId: 0,
            cooldownIndex: cooldownIndex,
            generation: uint16(_generation)
        });
        uint256 newKittenId = kitties.push(_kitty) - 1;

        // It's probably never going to happen, 4 billion cats
is A LOT, but
        // let's just be 100% sure we never let this happen.
        require(newKittenId == uint256(uint32(newKittenId)));

        // emit the birth event
        Birth(
            _owner,
            newKittenId,
            uint256(_kitty.matronId),
            uint256(_kitty.sireId),
            _kitty.genes
        );

        // This will assign ownership, and also emit the
Transfer event as
        // per ERC721 draft
        _transfer(0, _owner, newKittenId);

        return newKittenId;
    }

    // Any C-level can fix how many seconds per blocks are
currently observed.
    function setSecondsPerBlock(uint256 secs) external
onlyCLevel {
        require(secs < cooldowns[0]);
```

```
        secondsPerBlock = secs;
    }
}
```

```
/// @title The external contract that is responsible for
generating metadata for the kitties,
///  it has one function that will return the data as bytes.
contract ERC721Metadata {
    /// @dev Given a token Id, returns a byte array that is
supposed to be converted into string.
    function getMetadata(uint256 _tokenId, string) public view
returns (bytes32[4] buffer, uint256 count) {
        if (_tokenId == 1) {
            buffer[0] = "Hello World! :D";
            count = 15;
        } else if (_tokenId == 2) {
            buffer[0] = "I would definitely choose a medi";
            buffer[1] = "um length string.";
            count = 49;
        } else if (_tokenId == 3) {
            buffer[0] = "Lorem ipsum dolor sit amet, mi e";
            buffer[1] = "st accumsan dapibus augue lorem,";
            buffer[2] = " tristique vestibulum id, libero";
            buffer[3] = " suscipit varius sapien aliquam.";
            count = 128;
        }
    }
}
```

```
/// @title The facet of the CryptoKitties core contract that
manages ownership, ERC-721 (draft) compliant.
/// @author Axiom Zen (https://www.axiomzen.co)
/// @dev Ref: https://github.com/ethereum/EIPs/issues/721
///  See the KittyCore contract documentation to understand how
the various contract facets are arranged.
contract KittyOwnership is KittyBase, ERC721 {

    /// @notice Name and symbol of the non fungible token, as
defined in ERC721.
    string public constant name = "CryptoKitties";
    string public constant symbol = "CK";

    // The contract that will return kitty metadata
```

```
    ERC721Metadata public erc721Metadata;

    bytes4 constant InterfaceSignature_ERC165 =
        bytes4(keccak256('supportsInterface(bytes4)'));

    bytes4 constant InterfaceSignature_ERC721 =
        bytes4(keccak256('name()')) ^
        bytes4(keccak256('symbol()')) ^
        bytes4(keccak256('totalSupply()')) ^
        bytes4(keccak256('balanceOf(address)')) ^
        bytes4(keccak256('ownerOf(uint256)')) ^
        bytes4(keccak256('approve(address,uint256)')) ^
        bytes4(keccak256('transfer(address,uint256)')) ^

bytes4(keccak256('transferFrom(address,address,uint256)')) ^
        bytes4(keccak256('tokensOfOwner(address)')) ^
        bytes4(keccak256('tokenMetadata(uint256,string)'));

    /// @notice Introspection interface as per ERC-165
(https://github.com/ethereum/EIPs/issues/165).
    ///  Returns true for any standardized interfaces
implemented by this contract. We implement
    ///  ERC-165 (obviously!) and ERC-721.
    function supportsInterface(bytes4 _interfaceID) external
view returns (bool)
    {
        // DEBUG ONLY
        //require((InterfaceSignature_ERC165 == 0x01ffc9a7) &&
(InterfaceSignature_ERC721 == 0x9a20483d));

        return ((_interfaceID == InterfaceSignature_ERC165) ||
(_interfaceID == InterfaceSignature_ERC721));
    }

    /// @dev Set the address of the sibling contract that
tracks metadata.
    ///  CEO only.
    function setMetadataAddress(address _contractAddress)
public onlyCEO {
        erc721Metadata = ERC721Metadata(_contractAddress);
    }

    // Internal utility functions: These functions all assume
that their input arguments
    // are valid. We leave it to public methods to sanitize
their inputs and follow
    // the required logic.
```

```
    /// @dev Checks if a given address is the current owner of
a particular Kitty.
    /// @param _claimant the address we are validating against.
    /// @param _tokenId kitten id, only valid when > 0
    function _owns(address _claimant, uint256 _tokenId)
internal view returns (bool) {
        return kittyIndexToOwner[_tokenId] == _claimant;
    }

    /// @dev Checks if a given address currently has
transferApproval for a particular Kitty.
    /// @param _claimant the address we are confirming kitten
is approved for.
    /// @param _tokenId kitten id, only valid when > 0
    function _approvedFor(address _claimant, uint256 _tokenId)
internal view returns (bool) {
        return kittyIndexToApproved[_tokenId] == _claimant;
    }

    /// @dev Marks an address as being approved for
transferFrom(), overwriting any previous
    ///  approval. Setting _approved to address(0) clears all
transfer approval.
    ///  NOTE: _approve() does NOT send the Approval event.
This is intentional because
    ///  _approve() and transferFrom() are used together for
putting Kitties on auction, and
    ///  there is no value in spamming the log with Approval
events in that case.
    function _approve(uint256 _tokenId, address _approved)
internal {
        kittyIndexToApproved[_tokenId] = _approved;
    }

    /// @notice Returns the number of Kitties owned by a
specific address.
    /// @param _owner The owner address to check.
    /// @dev Required for ERC-721 compliance
    function balanceOf(address _owner) public view returns
(uint256 count) {
        return ownershipTokenCount[_owner];
    }

    /// @notice Transfers a Kitty to another address. If
transferring to a smart
    ///  contract be VERY CAREFUL to ensure that it is aware of
ERC-721 (or
    ///  CryptoKitties specifically) or your Kitty may be lost
```

```
forever. Seriously.
    /// @param _to The address of the recipient, can be a user
or contract.
    /// @param _tokenId The ID of the Kitty to transfer.
    /// @dev Required for ERC-721 compliance.
    function transfer(
        address _to,
        uint256 _tokenId
    )
        external
        whenNotPaused
    {
        // Safety check to prevent against an unexpected 0x0
default.
        require(_to != address(0));
        // Disallow transfers to this contract to prevent
accidental misuse.
        // The contract should never own any kitties (except
very briefly
        // after a gen0 cat is created and before it goes on
auction).
        require(_to != address(this));
        // Disallow transfers to the auction contracts to
prevent accidental
        // misuse. Auction contracts should only take ownership
of kitties
        // through the allow + transferFrom flow.
        require(_to != address(saleAuction));
        require(_to != address(siringAuction));

        // You can only send your own cat.
        require(_owns(msg.sender, _tokenId));

        // Reassign ownership, clear pending approvals, emit
Transfer event.
        _transfer(msg.sender, _to, _tokenId);
    }

    /// @notice Grant another address the right to transfer a
specific Kitty via
    ///  transferFrom(). This is the preferred flow for
transfering NFTs to contracts.
    /// @param _to The address to be granted transfer approval.
Pass address(0) to
    ///  clear all approvals.
    /// @param _tokenId The ID of the Kitty that can be
transferred if this call succeeds.
    /// @dev Required for ERC-721 compliance.
```

```
    function approve(
        address _to,
        uint256 _tokenId
    )
        external
        whenNotPaused
    {
        // Only an owner can grant transfer approval.
        require(_owns(msg.sender, _tokenId));

        // Register the approval (replacing any previous
approval).
        _approve(_tokenId, _to);

        // Emit approval event.
        Approval(msg.sender, _to, _tokenId);
    }

    /// @notice Transfer a Kitty owned by another address, for
which the calling address
    ///  has previously been granted transfer approval by the
owner.
    /// @param _from The address that owns the Kitty to be
transfered.
    /// @param _to The address that should take ownership of
the Kitty. Can be any address,
    ///  including the caller.
    /// @param _tokenId The ID of the Kitty to be transferred.
    /// @dev Required for ERC-721 compliance.
    function transferFrom(
        address _from,
        address _to,
        uint256 _tokenId
    )
        external
        whenNotPaused
    {
        // Safety check to prevent against an unexpected 0x0
default.
        require(_to != address(0));
        // Disallow transfers to this contract to prevent
accidental misuse.
        // The contract should never own any kitties (except
very briefly
        // after a gen0 cat is created and before it goes on
auction).
        require(_to != address(this));
        // Check for approval and valid ownership
```

```
        require(_approvedFor(msg.sender, _tokenId));
        require(_owns(_from, _tokenId));

        // Reassign ownership (also clears pending approvals
and emits Transfer event).
        _transfer(_from, _to, _tokenId);
    }

    /// @notice Returns the total number of Kitties currently
in existence.
    /// @dev Required for ERC-721 compliance.
    function totalSupply() public view returns (uint) {
        return kitties.length - 1;
    }

    /// @notice Returns the address currently assigned
ownership of a given Kitty.
    /// @dev Required for ERC-721 compliance.
    function ownerOf(uint256 _tokenId)
        external
        view
        returns (address owner)
    {
        owner = kittyIndexToOwner[_tokenId];

        require(owner != address(0));
    }

    /// @notice Returns a list of all Kitty IDs assigned to an
address.
    /// @param _owner The owner whose Kitties we are interested
in.
    /// @dev This method MUST NEVER be called by smart contract
code. First, it's fairly
    ///  expensive (it walks the entire Kitty array looking for
cats belonging to owner),
    ///  but it also returns a dynamic array, which is only
supported for web3 calls, and
    ///  not contract-to-contract calls.
    function tokensOfOwner(address _owner) external view
returns(uint256[] ownerTokens) {
        uint256 tokenCount = balanceOf(_owner);

        if (tokenCount == 0) {
            // Return an empty array
            return new uint256[](0);
        } else {
            uint256[] memory result = new uint256[]
```

```
(tokenCount);
            uint256 totalCats = totalSupply();
            uint256 resultIndex = 0;

            // We count on the fact that all cats have IDs
starting at 1 and increasing
            // sequentially up to the totalCat count.
            uint256 catId;

            for (catId = 1; catId <= totalCats; catId++) {
                if (kittyIndexToOwner[catId] == _owner) {
                    result[resultIndex] = catId;
                    resultIndex++;
                }
            }

            return result;
        }
    }

    /// @dev Adapted from memcpy() by @arachnid (Nick Johnson
<arachnid@notdot.net>)
    ///  This method is licenced under the Apache License.
    ///  Ref: https://github.com/Arachnid/solidity-
stringutils/blob/2f6ca9accb48ae14c66f1437ec50ed19a0616f78/strin
gs.sol
    function _memcpy(uint _dest, uint _src, uint _len) private
view {
        // Copy word-length chunks while possible
        for(; _len >= 32; _len -= 32) {
            assembly {
                mstore(_dest, mload(_src))
            }
            _dest += 32;
            _src += 32;
        }

        // Copy remaining bytes
        uint256 mask = 256 ** (32 - _len) - 1;
        assembly {
            let srcpart := and(mload(_src), not(mask))
            let destpart := and(mload(_dest), mask)
            mstore(_dest, or(destpart, srcpart))
        }
    }

    /// @dev Adapted from toString(slice) by @arachnid (Nick
Johnson <arachnid@notdot.net>)
```

```
    ///  This method is licenced under the Apache License.
    ///  Ref: https://github.com/Arachnid/solidity-
stringutils/blob/2f6ca9accb48ae14c66f1437ec50ed19a0616f78/strin
gs.sol
    function _toString(bytes32[4] _rawBytes, uint256
_stringLength) private view returns (string) {
        var outputString = new string(_stringLength);
        uint256 outputPtr;
        uint256 bytesPtr;

        assembly {
            outputPtr := add(outputString, 32)
            bytesPtr := _rawBytes
        }

        _memcpy(outputPtr, bytesPtr, _stringLength);

        return outputString;
    }

    /// @notice Returns a URI pointing to a metadata package
for this token conforming to
    ///  ERC-721 (https://github.com/ethereum/EIPs/issues/721)
    /// @param _tokenId The ID number of the Kitty whose
metadata should be returned.
    function tokenMetadata(uint256 _tokenId, string
_preferredTransport) external view returns (string infoUrl) {
        require(erc721Metadata != address(0));
        bytes32[4] memory buffer;
        uint256 count;
        (buffer, count) = erc721Metadata.getMetadata(_tokenId,
_preferredTransport);

        return _toString(buffer, count);
    }
}



/// @title A facet of KittyCore that manages Kitty siring,
gestation, and birth.
/// @author Axiom Zen (https://www.axiomzen.co)
/// @dev See the KittyCore contract documentation to understand
how the various contract facets are arranged.
contract KittyBreeding is KittyOwnership {

    /// @dev The Pregnant event is fired when two cats
successfully breed and the pregnancy
```

```
    ///  timer begins for the matron.
    event Pregnant(address owner, uint256 matronId, uint256
sireId, uint256 cooldownEndBlock);

    /// @notice The minimum payment required to use
breedWithAuto(). This fee goes towards
    ///  the gas cost paid by whatever calls giveBirth(), and
can be dynamically updated by
    ///  the COO role as the gas price changes.
    uint256 public autoBirthFee = 2 finney;

    // Keeps track of number of pregnant kitties.
    uint256 public pregnantKitties;

    /// @dev The address of the sibling contract that is used
to implement the sooper-sekret
    ///  genetic combination algorithm.
    GeneScienceInterface public geneScience;

    /// @dev Update the address of the genetic contract, can
only be called by the CEO.
    /// @param _address An address of a GeneScience contract
instance to be used from this point forward.
    function setGeneScienceAddress(address _address) external
onlyCEO {
        GeneScienceInterface candidateContract =
GeneScienceInterface(_address);

        // NOTE: verify that a contract is what we expect -
https://github.com/Lunyr/crowdsale-
contracts/blob/cfadd15986c30521d8ba7d5b6f57b4fefcc7ac38/contrac
ts/LunyrToken.sol#L117
        require(candidateContract.isGeneScience());

        // Set the new contract address
        geneScience = candidateContract;
    }

    /// @dev Checks that a given kitten is able to breed.
Requires that the
    ///  current cooldown is finished (for sires) and also
checks that there is
    ///  no pending pregnancy.
    function _isReadyToBreed(Kitty _kit) internal view returns
(bool) {
        // In addition to checking the cooldownEndBlock, we
also need to check to see if
        // the cat has a pending birth; there can be some
```

```
period of time between the end
        // of the pregnacy timer and the birth event.
        return (_kit.siringWithId == 0) &&
(_kit.cooldownEndBlock <= uint64(block.number));
    }

    /// @dev Check if a sire has authorized breeding with this
matron. True if both sire
    ///  and matron have the same owner, or if the sire has
given siring permission to
    ///  the matron's owner (via approveSiring()).
    function _isSiringPermitted(uint256 _sireId, uint256
_matronId) internal view returns (bool) {
        address matronOwner = kittyIndexToOwner[_matronId];
        address sireOwner = kittyIndexToOwner[_sireId];

        // Siring is okay if they have same owner, or if the
matron's owner was given
        // permission to breed with this sire.
        return (matronOwner == sireOwner ||
sireAllowedToAddress[_sireId] == matronOwner);
    }

    /// @dev Set the cooldownEndTime for the given Kitty, based
on its current cooldownIndex.
    ///  Also increments the cooldownIndex (unless it has hit
the cap).
    /// @param _kitten A reference to the Kitty in storage
which needs its timer started.
    function _triggerCooldown(Kitty storage _kitten) internal {
        // Compute an estimation of the cooldown time in blocks
(based on current cooldownIndex).
        _kitten.cooldownEndBlock =
uint64((cooldowns[_kitten.cooldownIndex]/secondsPerBlock) +
block.number);

        // Increment the breeding count, clamping it at 13,
which is the length of the
        // cooldowns array. We could check the array size
dynamically, but hard-coding
        // this as a constant saves gas. Yay, Solidity!
        if (_kitten.cooldownIndex < 13) {
            _kitten.cooldownIndex += 1;
        }
    }

    /// @notice Grants approval to another user to sire with
one of your Kitties.
```

```
    /// @param _addr The address that will be able to sire with
your Kitty. Set to
    ///  address(0) to clear all siring approvals for this
Kitty.
    /// @param _sireId A Kitty that you own that _addr will now
be able to sire with.
    function approveSiring(address _addr, uint256 _sireId)
        external
        whenNotPaused
    {
        require(_owns(msg.sender, _sireId));
        sireAllowedToAddress[_sireId] = _addr;
    }

    /// @dev Updates the minimum payment required for calling
giveBirthAuto(). Can only
    ///  be called by the COO address. (This fee is used to
offset the gas cost incurred
    ///  by the autobirth daemon).
    function setAutoBirthFee(uint256 val) external onlyCOO {
        autoBirthFee = val;
    }

    /// @dev Checks to see if a given Kitty is pregnant and (if
so) if the gestation
    ///  period has passed.
    function _isReadyToGiveBirth(Kitty _matron) private view
returns (bool) {
        return (_matron.siringWithId != 0) &&
(_matron.cooldownEndBlock <= uint64(block.number));
    }

    /// @notice Checks that a given kitten is able to breed
(i.e. it is not pregnant or
    ///  in the middle of a siring cooldown).
    /// @param _kittyId reference the id of the kitten, any
user can inquire about it
    function isReadyToBreed(uint256 _kittyId)
        public
        view
        returns (bool)
    {
        require(_kittyId > 0);
        Kitty storage kit = kitties[_kittyId];
        return _isReadyToBreed(kit);
    }

    /// @dev Checks whether a kitty is currently pregnant.
```

```
    /// @param _kittyId reference the id of the kitten, any
user can inquire about it
    function isPregnant(uint256 _kittyId)
        public
        view
        returns (bool)
    {
        require(_kittyId > 0);
        // A kitty is pregnant if and only if this field is set
        return kitties[_kittyId].siringWithId != 0;
    }

    /// @dev Internal check to see if a given sire and matron
are a valid mating pair. DOES NOT
    ///  check ownership permissions (that is up to the
caller).
    /// @param _matron A reference to the Kitty struct of the
potential matron.
    /// @param _matronId The matron's ID.
    /// @param _sire A reference to the Kitty struct of the
potential sire.
    /// @param _sireId The sire's ID
    function _isValidMatingPair(
        Kitty storage _matron,
        uint256 _matronId,
        Kitty storage _sire,
        uint256 _sireId
    )
        private
        view
        returns(bool)
    {
        // A Kitty can't breed with itself!
        if (_matronId == _sireId) {
            return false;
        }

        // Kitties can't breed with their parents.
        if (_matron.matronId == _sireId || _matron.sireId ==
_sireId) {
            return false;
        }
        if (_sire.matronId == _matronId || _sire.sireId ==
_matronId) {
            return false;
        }

        // We can short circuit the sibling check (below) if
```

```
either cat is
        // gen zero (has a matron ID of zero).
        if (_sire.matronId == 0 || _matron.matronId == 0) {
            return true;
        }

        // Kitties can't breed with full or half siblings.
        if (_sire.matronId == _matron.matronId ||
_sire.matronId == _matron.sireId) {
            return false;
        }
        if (_sire.sireId == _matron.matronId || _sire.sireId ==
_matron.sireId) {
            return false;
        }

        // Everything seems cool! Let's get DTF.
        return true;
    }

    /// @dev Internal check to see if a given sire and matron
are a valid mating pair for
    ///  breeding via auction (i.e. skips ownership and siring
approval checks).
    function _canBreedWithViaAuction(uint256 _matronId, uint256
_sireId)
        internal
        view
        returns (bool)
    {
        Kitty storage matron = kitties[_matronId];
        Kitty storage sire = kitties[_sireId];
        return _isValidMatingPair(matron, _matronId, sire,
_sireId);
    }

    /// @notice Checks to see if two cats can breed together,
including checks for
    ///  ownership and siring approvals. Does NOT check that
both cats are ready for
    ///  breeding (i.e. breedWith could still fail until the
cooldowns are finished).
    ///  TODO: Shouldn't this check pregnancy and cooldowns?!?
    /// @param _matronId The ID of the proposed matron.
    /// @param _sireId The ID of the proposed sire.
    function canBreedWith(uint256 _matronId, uint256 _sireId)
        external
        view
```

```
        returns(bool)
    {
        require(_matronId > 0);
        require(_sireId > 0);
        Kitty storage matron = kitties[_matronId];
        Kitty storage sire = kitties[_sireId];
        return _isValidMatingPair(matron, _matronId, sire,
_sireId) &&
            _isSiringPermitted(_sireId, _matronId);
    }

    /// @dev Internal utility function to initiate breeding,
assumes that all breeding
    ///  requirements have been checked.
    function _breedWith(uint256 _matronId, uint256 _sireId)
internal {
        // Grab a reference to the Kitties from storage.
        Kitty storage sire = kitties[_sireId];
        Kitty storage matron = kitties[_matronId];

        // Mark the matron as pregnant, keeping track of who
the sire is.
        matron.siringWithId = uint32(_sireId);

        // Trigger the cooldown for both parents.
        _triggerCooldown(sire);
        _triggerCooldown(matron);

        // Clear siring permission for both parents. This may
not be strictly necessary
        // but it's likely to avoid confusion!
        delete sireAllowedToAddress[_matronId];
        delete sireAllowedToAddress[_sireId];

        // Every time a kitty gets pregnant, counter is
incremented.
        pregnantKitties++;

        // Emit the pregnancy event.
        Pregnant(kittyIndexToOwner[_matronId], _matronId,
_sireId, matron.cooldownEndBlock);
    }

    /// @notice Breed a Kitty you own (as matron) with a sire
that you own, or for which you
    ///  have previously been given Siring approval. Will
either make your cat pregnant, or will
    ///  fail entirely. Requires a pre-payment of the fee given
```

```
out to the first caller of giveBirth()
    /// @param _matronId The ID of the Kitty acting as matron
(will end up pregnant if successful)
    /// @param _sireId The ID of the Kitty acting as sire (will
begin its siring cooldown if successful)
    function breedWithAuto(uint256 _matronId, uint256 _sireId)
        external
        payable
        whenNotPaused
    {
        // Checks for payment.
        require(msg.value >= autoBirthFee);

        // Caller must own the matron.
        require(_owns(msg.sender, _matronId));

        // Neither sire nor matron are allowed to be on auction
during a normal
        // breeding operation, but we don't need to check that
explicitly.
        // For matron: The caller of this function can't be the
owner of the matron
        //   because the owner of a Kitty on auction is the
auction house, and the
        //   auction house will never call breedWith().
        // For sire: Similarly, a sire on auction will be owned
by the auction house
        //   and the act of transferring ownership will have
cleared any oustanding
        //   siring approval.
        // Thus we don't need to spend gas explicitly checking
to see if either cat
        // is on auction.

        // Check that matron and sire are both owned by caller,
or that the sire
        // has given siring permission to caller (i.e. matron's
owner).
        // Will fail for _sireId = 0
        require(_isSiringPermitted(_sireId, _matronId));

        // Grab a reference to the potential matron
        Kitty storage matron = kitties[_matronId];

        // Make sure matron isn't pregnant, or in the middle of
a siring cooldown
        require(_isReadyToBreed(matron));
```

```
        // Grab a reference to the potential sire
        Kitty storage sire = kitties[_sireId];

        // Make sure sire isn't pregnant, or in the middle of a
siring cooldown
        require(_isReadyToBreed(sire));

        // Test that these cats are a valid mating pair.
        require(_isValidMatingPair(
            matron,
            _matronId,
            sire,
            _sireId
        ));

        // All checks passed, kitty gets pregnant!
        _breedWith(_matronId, _sireId);
    }

    /// @notice Have a pregnant Kitty give birth!
    /// @param _matronId A Kitty ready to give birth.
    /// @return The Kitty ID of the new kitten.
    /// @dev Looks at a given Kitty and, if pregnant and if the
gestation period has passed,
    ///  combines the genes of the two parents to create a new
kitten. The new Kitty is assigned
    ///  to the current owner of the matron. Upon successful
completion, both the matron and the
    ///  new kitten will be ready to breed again. Note that
anyone can call this function (if they
    ///  are willing to pay the gas!), but the new kitten
always goes to the mother's owner.
    function giveBirth(uint256 _matronId)
        external
        whenNotPaused
        returns(uint256)
    {
        // Grab a reference to the matron in storage.
        Kitty storage matron = kitties[_matronId];

        // Check that the matron is a valid cat.
        require(matron.birthTime != 0);

        // Check that the matron is pregnant, and that its time
has come!
        require(_isReadyToGiveBirth(matron));

        // Grab a reference to the sire in storage.
```

```
        uint256 sireId = matron.siringWithId;
        Kitty storage sire = kitties[sireId];

        // Determine the higher generation number of the two
parents
        uint16 parentGen = matron.generation;
        if (sire.generation > matron.generation) {
            parentGen = sire.generation;
        }

        // Call the sooper-sekret gene mixing operation.
        uint256 childGenes = geneScience.mixGenes(matron.genes,
sire.genes, matron.cooldownEndBlock - 1);

        // Make the new kitten!
        address owner = kittyIndexToOwner[_matronId];
        uint256 kittenId = _createKitty(_matronId,
matron.siringWithId, parentGen + 1, childGenes, owner);

        // Clear the reference to sire from the matron
(REQUIRED! Having siringWithId
        // set is what marks a matron as being pregnant.)
        delete matron.siringWithId;

        // Every time a kitty gives birth counter is
decremented.
        pregnantKitties--;

        // Send the balance fee to the person who made birth
happen.
        msg.sender.send(autoBirthFee);

        // return the new kitten's ID
        return kittenId;
    }
}
```

```
/// @title Auction Core
/// @dev Contains models, variables, and internal methods for
```

```
the auction.
/// @notice We omit a fallback function to prevent accidental
sends to this contract.
contract ClockAuctionBase {

    // Represents an auction on an NFT
    struct Auction {
        // Current owner of NFT
        address seller;
        // Price (in wei) at beginning of auction
        uint128 startingPrice;
        // Price (in wei) at end of auction
        uint128 endingPrice;
        // Duration (in seconds) of auction
        uint64 duration;
        // Time when auction started
        // NOTE: 0 if this auction has been concluded
        uint64 startedAt;
    }

    // Reference to contract tracking NFT ownership
    ERC721 public nonFungibleContract;

    // Cut owner takes on each auction, measured in basis
points (1/100 of a percent).
    // Values 0-10,000 map to 0%-100%
    uint256 public ownerCut;

    // Map from token ID to their corresponding auction.
    mapping (uint256 => Auction) tokenIdToAuction;

    event AuctionCreated(uint256 tokenId, uint256
startingPrice, uint256 endingPrice, uint256 duration);
    event AuctionSuccessful(uint256 tokenId, uint256
totalPrice, address winner);
    event AuctionCancelled(uint256 tokenId);

    /// @dev Returns true if the claimant owns the token.
    /// @param _claimant - Address claiming to own the token.
    /// @param _tokenId - ID of token whose ownership to
verify.
    function _owns(address _claimant, uint256 _tokenId)
internal view returns (bool) {
        return (nonFungibleContract.ownerOf(_tokenId) ==
_claimant);
    }

    /// @dev Escrows the NFT, assigning ownership to this
```

```
contract.
    /// Throws if the escrow fails.
    /// @param _owner - Current owner address of token to
escrow.
    /// @param _tokenId - ID of token whose approval to verify.
    function _escrow(address _owner, uint256 _tokenId) internal
{
        // it will throw if transfer fails
        nonFungibleContract.transferFrom(_owner, this,
_tokenId);
    }

    /// @dev Transfers an NFT owned by this contract to another
address.
    /// Returns true if the transfer succeeds.
    /// @param _receiver - Address to transfer NFT to.
    /// @param _tokenId - ID of token to transfer.
    function _transfer(address _receiver, uint256 _tokenId)
internal {
        // it will throw if transfer fails
        nonFungibleContract.transfer(_receiver, _tokenId);
    }

    /// @dev Adds an auction to the list of open auctions. Also
fires the
    ///  AuctionCreated event.
    /// @param _tokenId The ID of the token to be put on
auction.
    /// @param _auction Auction to add.
    function _addAuction(uint256 _tokenId, Auction _auction)
internal {
        // Require that all auctions have a duration of
        // at least one minute. (Keeps our math from getting
hairy!)
        require(_auction.duration >= 1 minutes);

        tokenIdToAuction[_tokenId] = _auction;

        AuctionCreated(
            uint256(_tokenId),
            uint256(_auction.startingPrice),
            uint256(_auction.endingPrice),
            uint256(_auction.duration)
        );
    }

    /// @dev Cancels an auction unconditionally.
    function _cancelAuction(uint256 _tokenId, address _seller)
```

```
internal {
        _removeAuction(_tokenId);
        _transfer(_seller, _tokenId);
        AuctionCancelled(_tokenId);
    }

    /// @dev Computes the price and transfers winnings.
    /// Does NOT transfer ownership of token.
    function _bid(uint256 _tokenId, uint256 _bidAmount)
        internal
        returns (uint256)
    {
        // Get a reference to the auction struct
        Auction storage auction = tokenIdToAuction[_tokenId];

        // Explicitly check that this auction is currently
live.
        // (Because of how Ethereum mappings work, we can't
just count
        // on the lookup above failing. An invalid _tokenId
will just
        // return an auction object that is all zeros.)
        require(_isOnAuction(auction));

        // Check that the bid is greater than or equal to the
current price
        uint256 price = _currentPrice(auction);
        require(_bidAmount >= price);

        // Grab a reference to the seller before the auction
struct
        // gets deleted.
        address seller = auction.seller;

        // The bid is good! Remove the auction before sending
the fees
        // to the sender so we can't have a reentrancy attack.
        _removeAuction(_tokenId);

        // Transfer proceeds to seller (if there are any!)
        if (price > 0) {
            // Calculate the auctioneer's cut.
            // (NOTE: _computeCut() is guaranteed to return a
            // value <= price, so this subtraction can't go
negative.)
            uint256 auctioneerCut = _computeCut(price);
            uint256 sellerProceeds = price - auctioneerCut;
```

```
            // NOTE: Doing a transfer() in the middle of a
complex
            // method like this is generally discouraged
because of
            // reentrancy attacks and DoS attacks if the seller
is
            // a contract with an invalid fallback function. We
explicitly
            // guard against reentrancy attacks by removing the
auction
            // before calling transfer(), and the only thing
the seller
            // can DoS is the sale of their own asset! (And if
it's an
            // accident, they can call cancelAuction(). )
            seller.transfer(sellerProceeds);
        }

        // Calculate any excess funds included with the bid. If
the excess
        // is anything worth worrying about, transfer it back
to bidder.
        // NOTE: We checked above that the bid amount is
greater than or
        // equal to the price so this cannot underflow.
        uint256 bidExcess = _bidAmount - price;

        // Return the funds. Similar to the previous transfer,
this is
        // not susceptible to a re-entry attack because the
auction is
        // removed before any transfers occur.
        msg.sender.transfer(bidExcess);

        // Tell the world!
        AuctionSuccessful(_tokenId, price, msg.sender);

        return price;
    }

    /// @dev Removes an auction from the list of open auctions.
    /// @param _tokenId - ID of NFT on auction.
    function _removeAuction(uint256 _tokenId) internal {
        delete tokenIdToAuction[_tokenId];
    }

    /// @dev Returns true if the NFT is on auction.
    /// @param _auction - Auction to check.
```

```
    function _isOnAuction(Auction storage _auction) internal
view returns (bool) {
        return (_auction.startedAt > 0);
    }

    /// @dev Returns current price of an NFT on auction. Broken
into two
    ///  functions (this one, that computes the duration from
the auction
    ///  structure, and the other that does the price
computation) so we
    ///  can easily test that the price computation works
correctly.
    function _currentPrice(Auction storage _auction)
        internal
        view
        returns (uint256)
    {
        uint256 secondsPassed = 0;

        // A bit of insurance against negative values (or
wraparound).
        // Probably not necessary (since Ethereum guarnatees
that the
        // now variable doesn't ever go backwards).
        if (now > _auction.startedAt) {
            secondsPassed = now - _auction.startedAt;
        }

        return _computeCurrentPrice(
            _auction.startingPrice,
            _auction.endingPrice,
            _auction.duration,
            secondsPassed
        );
    }

    /// @dev Computes the current price of an auction. Factored
out
    ///  from _currentPrice so we can run extensive unit tests.
    ///  When testing, make this function public and turn on
    ///  `Current price computation` test suite.
    function _computeCurrentPrice(
        uint256 _startingPrice,
        uint256 _endingPrice,
        uint256 _duration,
        uint256 _secondsPassed
    )
```

```
        internal
        pure
        returns (uint256)
    {
        // NOTE: We don't use SafeMath (or similar) in this
function because
        //  all of our public functions carefully cap the
maximum values for
        //  time (at 64-bits) and currency (at 128-bits).
_duration is
        //  also known to be non-zero (see the require()
statement in
        //  _addAuction())
        if (_secondsPassed >= _duration) {
            // We've reached the end of the dynamic pricing
portion
            // of the auction, just return the end price.
            return _endingPrice;
        } else {
            // Starting price can be higher than ending price
(and often is!), so
            // this delta can be negative.
            int256 totalPriceChange = int256(_endingPrice) -
int256(_startingPrice);

            // This multiplication can't overflow,
_secondsPassed will easily fit within
            // 64-bits, and totalPriceChange will easily fit
within 128-bits, their product
            // will always fit within 256-bits.
            int256 currentPriceChange = totalPriceChange *
int256(_secondsPassed) / int256(_duration);

            // currentPriceChange can be negative, but if so,
will have a magnitude
            // less that _startingPrice. Thus, this result will
always end up positive.
            int256 currentPrice = int256(_startingPrice) +
currentPriceChange;

            return uint256(currentPrice);
        }
    }

    /// @dev Computes owner's cut of a sale.
    /// @param _price - Sale price of NFT.
    function _computeCut(uint256 _price) internal view returns
(uint256) {
```

```
        // NOTE: We don't use SafeMath (or similar) in this
function because
        //  all of our entry functions carefully cap the
maximum values for
        //  currency (at 128-bits), and ownerCut <= 10000 (see
the require()
        //  statement in the ClockAuction constructor). The
result of this
        //  function is always guaranteed to be <= _price.
        return _price * ownerCut / 10000;
    }

}
```

```
/**
 * @title Pausable
 * @dev Base contract which allows children to implement an
emergency stop mechanism.
 */
contract Pausable is Ownable {
  event Pause();
  event Unpause();

  bool public paused = false;


  /**
   * @dev modifier to allow actions only when the contract IS
paused
   */
  modifier whenNotPaused() {
    require(!paused);
    _;
  }

  /**
   * @dev modifier to allow actions only when the contract IS
NOT paused
   */
  modifier whenPaused {
    require(paused);
    _;
```

```
  }

  /**
   * @dev called by the owner to pause, triggers stopped state
   */
  function pause() onlyOwner whenNotPaused returns (bool) {
    paused = true;
    Pause();
    return true;
  }

  /**
   * @dev called by the owner to unpause, returns to normal
state
   */
  function unpause() onlyOwner whenPaused returns (bool) {
    paused = false;
    Unpause();
    return true;
  }
}


/// @title Clock auction for non-fungible tokens.
/// @notice We omit a fallback function to prevent accidental
sends to this contract.
contract ClockAuction is Pausable, ClockAuctionBase {

    /// @dev The ERC-165 interface signature for ERC-721.
    ///  Ref: https://github.com/ethereum/EIPs/issues/165
    ///  Ref: https://github.com/ethereum/EIPs/issues/721
    bytes4 constant InterfaceSignature_ERC721 =
bytes4(0x9a20483d);

    /// @dev Constructor creates a reference to the NFT
ownership contract
    ///  and verifies the owner cut is in the valid range.
    /// @param _nftAddress - address of a deployed contract
implementing
    ///  the Nonfungible Interface.
    /// @param _cut - percent cut the owner takes on each
auction, must be
    ///  between 0-10,000.
    function ClockAuction(address _nftAddress, uint256 _cut)
public {
        require(_cut <= 10000);
        ownerCut = _cut;
```

```
        ERC721 candidateContract = ERC721(_nftAddress);

require(candidateContract.supportsInterface(InterfaceSignature_
ERC721));
        nonFungibleContract = candidateContract;
    }

    /// @dev Remove all Ether from the contract, which is the
owner's cuts
    ///  as well as any Ether sent directly to the contract
address.
    ///  Always transfers to the NFT contract, but can be
called either by
    ///  the owner or the NFT contract.
    function withdrawBalance() external {
        address nftAddress = address(nonFungibleContract);

        require(
            msg.sender == owner ||
            msg.sender == nftAddress
        );
        // We are using this boolean method to make sure that
even if one fails it will still work
        bool res = nftAddress.send(this.balance);
    }

    /// @dev Creates and begins a new auction.
    /// @param _tokenId - ID of token to auction, sender must
be owner.
    /// @param _startingPrice - Price of item (in wei) at
beginning of auction.
    /// @param _endingPrice - Price of item (in wei) at end of
auction.
    /// @param _duration - Length of time to move between
starting
    ///  price and ending price (in seconds).
    /// @param _seller - Seller, if not the message sender
    function createAuction(
        uint256 _tokenId,
        uint256 _startingPrice,
        uint256 _endingPrice,
        uint256 _duration,
        address _seller
    )
        external
        whenNotPaused
    {
        // Sanity check that no inputs overflow how many bits
```

```
we've allocated
        // to store them in the auction struct.
        require(_startingPrice ==
uint256(uint128(_startingPrice)));
        require(_endingPrice ==
uint256(uint128(_endingPrice)));
        require(_duration == uint256(uint64(_duration)));

        require(_owns(msg.sender, _tokenId));
        _escrow(msg.sender, _tokenId);
        Auction memory auction = Auction(
            _seller,
            uint128(_startingPrice),
            uint128(_endingPrice),
            uint64(_duration),
            uint64(now)
        );
        _addAuction(_tokenId, auction);
    }

    /// @dev Bids on an open auction, completing the auction
and transferring
    ///  ownership of the NFT if enough Ether is supplied.
    /// @param _tokenId - ID of token to bid on.
    function bid(uint256 _tokenId)
        external
        payable
        whenNotPaused
    {
        // _bid will throw if the bid or funds transfer fails
        _bid(_tokenId, msg.value);
        _transfer(msg.sender, _tokenId);
    }

    /// @dev Cancels an auction that hasn't been won yet.
    ///  Returns the NFT to original owner.
    /// @notice This is a state-modifying function that can
    ///  be called while the contract is paused.
    /// @param _tokenId - ID of token on auction
    function cancelAuction(uint256 _tokenId)
        external
    {
        Auction storage auction = tokenIdToAuction[_tokenId];
        require(_isOnAuction(auction));
        address seller = auction.seller;
        require(msg.sender == seller);
        _cancelAuction(_tokenId, seller);
    }
```

```
    /// @dev Cancels an auction when the contract is paused.
    ///  Only the owner may do this, and NFTs are returned to
    ///  the seller. This should only be used in emergencies.
    /// @param _tokenId - ID of the NFT on auction to cancel.
    function cancelAuctionWhenPaused(uint256 _tokenId)
        whenPaused
        onlyOwner
        external
    {
        Auction storage auction = tokenIdToAuction[_tokenId];
        require(_isOnAuction(auction));
        _cancelAuction(_tokenId, auction.seller);
    }

    /// @dev Returns auction info for an NFT on auction.
    /// @param _tokenId - ID of NFT on auction.
    function getAuction(uint256 _tokenId)
        external
        view
        returns
    (
        address seller,
        uint256 startingPrice,
        uint256 endingPrice,
        uint256 duration,
        uint256 startedAt
    ) {
        Auction storage auction = tokenIdToAuction[_tokenId];
        require(_isOnAuction(auction));
        return (
            auction.seller,
            auction.startingPrice,
            auction.endingPrice,
            auction.duration,
            auction.startedAt
        );
    }

    /// @dev Returns the current price of an auction.
    /// @param _tokenId - ID of the token price we are
checking.
    function getCurrentPrice(uint256 _tokenId)
        external
        view
        returns (uint256)
    {
        Auction storage auction = tokenIdToAuction[_tokenId];
```

```
        require(_isOnAuction(auction));
        return _currentPrice(auction);
    }

}


/// @title Reverse auction modified for siring
/// @notice We omit a fallback function to prevent accidental
sends to this contract.
contract SiringClockAuction is ClockAuction {

    // @dev Sanity check that allows us to ensure that we are
pointing to the
    //  right auction in our setSiringAuctionAddress() call.
    bool public isSiringClockAuction = true;

    // Delegate constructor
    function SiringClockAuction(address _nftAddr, uint256 _cut)
public
        ClockAuction(_nftAddr, _cut) {}

    /// @dev Creates and begins a new auction. Since this
function is wrapped,
    /// require sender to be KittyCore contract.
    /// @param _tokenId - ID of token to auction, sender must
be owner.
    /// @param _startingPrice - Price of item (in wei) at
beginning of auction.
    /// @param _endingPrice - Price of item (in wei) at end of
auction.
    /// @param _duration - Length of auction (in seconds).
    /// @param _seller - Seller, if not the message sender
    function createAuction(
        uint256 _tokenId,
        uint256 _startingPrice,
        uint256 _endingPrice,
        uint256 _duration,
        address _seller
    )
        external
    {
        // Sanity check that no inputs overflow how many bits
we've allocated
        // to store them in the auction struct.
        require(_startingPrice ==
uint256(uint128(_startingPrice)));
        require(_endingPrice ==
```

```
uint256(uint128(_endingPrice)));
        require(_duration == uint256(uint64(_duration)));

        require(msg.sender == address(nonFungibleContract));
        _escrow(_seller, _tokenId);
        Auction memory auction = Auction(
            _seller,
            uint128(_startingPrice),
            uint128(_endingPrice),
            uint64(_duration),
            uint64(now)
        );
        _addAuction(_tokenId, auction);
    }

    /// @dev Places a bid for siring. Requires the sender
    /// is the KittyCore contract because all bid methods
    /// should be wrapped. Also returns the kitty to the
    /// seller rather than the winner.
    function bid(uint256 _tokenId)
        external
        payable
    {
        require(msg.sender == address(nonFungibleContract));
        address seller = tokenIdToAuction[_tokenId].seller;
        // _bid checks that token ID is valid and will throw if
bid fails
        _bid(_tokenId, msg.value);
        // We transfer the kitty back to the seller, the winner
will get
        // the offspring
        _transfer(seller, _tokenId);
    }

}




/// @title Clock auction modified for sale of kitties
/// @notice We omit a fallback function to prevent accidental
sends to this contract.
contract SaleClockAuction is ClockAuction {

    // @dev Sanity check that allows us to ensure that we are
pointing to the
    //  right auction in our setSaleAuctionAddress() call.
```

```
    bool public isSaleClockAuction = true;

    // Tracks last 5 sale price of gen0 kitty sales
    uint256 public gen0SaleCount;
    uint256[5] public lastGen0SalePrices;

    // Delegate constructor
    function SaleClockAuction(address _nftAddr, uint256 _cut)
public
        ClockAuction(_nftAddr, _cut) {}

    /// @dev Creates and begins a new auction.
    /// @param _tokenId - ID of token to auction, sender must
be owner.
    /// @param _startingPrice - Price of item (in wei) at
beginning of auction.
    /// @param _endingPrice - Price of item (in wei) at end of
auction.
    /// @param _duration - Length of auction (in seconds).
    /// @param _seller - Seller, if not the message sender
    function createAuction(
        uint256 _tokenId,
        uint256 _startingPrice,
        uint256 _endingPrice,
        uint256 _duration,
        address _seller
    )
        external
    {
        // Sanity check that no inputs overflow how many bits
we've allocated
        // to store them in the auction struct.
        require(_startingPrice ==
uint256(uint128(_startingPrice)));
        require(_endingPrice ==
uint256(uint128(_endingPrice)));
        require(_duration == uint256(uint64(_duration)));

        require(msg.sender == address(nonFungibleContract));
        _escrow(_seller, _tokenId);
        Auction memory auction = Auction(
            _seller,
            uint128(_startingPrice),
            uint128(_endingPrice),
            uint64(_duration),
            uint64(now)
        );
        _addAuction(_tokenId, auction);
```

```
    }

    /// @dev Updates lastSalePrice if seller is the nft
contract
    /// Otherwise, works the same as default bid method.
    function bid(uint256 _tokenId)
        external
        payable
    {
        // _bid verifies token ID size
        address seller = tokenIdToAuction[_tokenId].seller;
        uint256 price = _bid(_tokenId, msg.value);
        _transfer(msg.sender, _tokenId);

        // If not a gen0 auction, exit
        if (seller == address(nonFungibleContract)) {
            // Track gen0 sale prices
            lastGen0SalePrices[gen0SaleCount % 5] = price;
            gen0SaleCount++;
        }
    }

    function averageGen0SalePrice() external view returns
(uint256) {
        uint256 sum = 0;
        for (uint256 i = 0; i < 5; i++) {
            sum += lastGen0SalePrices[i];
        }
        return sum / 5;
    }

}


/// @title Handles creating auctions for sale and siring of
kitties.
///  This wrapper of ReverseAuction exists only so that users
can create
///  auctions with only one transaction.
contract KittyAuction is KittyBreeding {

    // @notice The auction contract variables are defined in
KittyBase to allow
    //  us to refer to them in KittyOwnership to prevent
accidental transfers.
    // `saleAuction` refers to the auction for gen0 and p2p
sale of kitties.
    // `siringAuction` refers to the auction for siring rights
```

```
of kitties.

    /// @dev Sets the reference to the sale auction.
    /// @param _address - Address of sale contract.
    function setSaleAuctionAddress(address _address) external
onlyCEO {
        SaleClockAuction candidateContract =
SaleClockAuction(_address);

        // NOTE: verify that a contract is what we expect -
https://github.com/Lunyr/crowdsale-
contracts/blob/cfadd15986c30521d8ba7d5b6f57b4fefcc7ac38/contrac
ts/LunyrToken.sol#L117
        require(candidateContract.isSaleClockAuction());

        // Set the new contract address
        saleAuction = candidateContract;
    }

    /// @dev Sets the reference to the siring auction.
    /// @param _address - Address of siring contract.
    function setSiringAuctionAddress(address _address) external
onlyCEO {
        SiringClockAuction candidateContract =
SiringClockAuction(_address);

        // NOTE: verify that a contract is what we expect -
https://github.com/Lunyr/crowdsale-
contracts/blob/cfadd15986c30521d8ba7d5b6f57b4fefcc7ac38/contrac
ts/LunyrToken.sol#L117
        require(candidateContract.isSiringClockAuction());

        // Set the new contract address
        siringAuction = candidateContract;
    }

    /// @dev Put a kitty up for auction.
    ///  Does some ownership trickery to create auctions in one
tx.
    function createSaleAuction(
        uint256 _kittyId,
        uint256 _startingPrice,
        uint256 _endingPrice,
        uint256 _duration
    )
        external
        whenNotPaused
    {
```

```
        // Auction contract checks input sizes
        // If kitty is already on any auction, this will throw
        // because it will be owned by the auction contract.
        require(_owns(msg.sender, _kittyId));
        // Ensure the kitty is not pregnant to prevent the
auction
        // contract accidentally receiving ownership of the
child.
        // NOTE: the kitty IS allowed to be in a cooldown.
        require(!isPregnant(_kittyId));
        _approve(_kittyId, saleAuction);
        // Sale auction throws if inputs are invalid and clears
        // transfer and sire approval after escrowing the
kitty.
        saleAuction.createAuction(
            _kittyId,
            _startingPrice,
            _endingPrice,
            _duration,
            msg.sender
        );
    }

    /// @dev Put a kitty up for auction to be sire.
    ///  Performs checks to ensure the kitty can be sired, then
    ///  delegates to reverse auction.
    function createSiringAuction(
        uint256 _kittyId,
        uint256 _startingPrice,
        uint256 _endingPrice,
        uint256 _duration
    )
        external
        whenNotPaused
    {
        // Auction contract checks input sizes
        // If kitty is already on any auction, this will throw
        // because it will be owned by the auction contract.
        require(_owns(msg.sender, _kittyId));
        require(isReadyToBreed(_kittyId));
        _approve(_kittyId, siringAuction);
        // Siring auction throws if inputs are invalid and
clears
        // transfer and sire approval after escrowing the
kitty.
        siringAuction.createAuction(
            _kittyId,
            _startingPrice,
```

```
            _endingPrice,
            _duration,
            msg.sender
        );
    }

    /// @dev Completes a siring auction by bidding.
    ///  Immediately breeds the winning matron with the sire on
auction.
    /// @param _sireId - ID of the sire on auction.
    /// @param _matronId - ID of the matron owned by the
bidder.
    function bidOnSiringAuction(
        uint256 _sireId,
        uint256 _matronId
    )
        external
        payable
        whenNotPaused
    {
        // Auction contract checks input sizes
        require(_owns(msg.sender, _matronId));
        require(isReadyToBreed(_matronId));
        require(_canBreedWithViaAuction(_matronId, _sireId));

        // Define the current price of the auction.
        uint256 currentPrice =
siringAuction.getCurrentPrice(_sireId);
        require(msg.value >= currentPrice + autoBirthFee);

        // Siring auction will throw if the bid fails.
        siringAuction.bid.value(msg.value - autoBirthFee)
(_sireId);
        _breedWith(uint32(_matronId), uint32(_sireId));
    }

    /// @dev Transfers the balance of the sale auction contract
    /// to the KittyCore contract. We use two-step withdrawal
to
    /// prevent two transfer calls in the auction bid function.
    function withdrawAuctionBalances() external onlyCLevel {
        saleAuction.withdrawBalance();
        siringAuction.withdrawBalance();
    }
}


/// @title all functions related to creating kittens
```

```
contract KittyMinting is KittyAuction {

    // Limits the number of cats the contract owner can ever
create.
    uint256 public constant PROMO_CREATION_LIMIT = 5000;
    uint256 public constant GEN0_CREATION_LIMIT = 45000;

    // Constants for gen0 auctions.
    uint256 public constant GEN0_STARTING_PRICE = 10 finney;
    uint256 public constant GEN0_AUCTION_DURATION = 1 days;

    // Counts the number of cats the contract owner has
created.
    uint256 public promoCreatedCount;
    uint256 public gen0CreatedCount;

    /// @dev we can create promo kittens, up to a limit. Only
callable by COO
    /// @param _genes the encoded genes of the kitten to be
created, any value is accepted
    /// @param _owner the future owner of the created kittens.
Default to contract COO
    function createPromoKitty(uint256 _genes, address _owner)
external onlyCOO {
        address kittyOwner = _owner;
        if (kittyOwner == address(0)) {
             kittyOwner = cooAddress;
        }
        require(promoCreatedCount < PROMO_CREATION_LIMIT);

        promoCreatedCount++;
        _createKitty(0, 0, 0, _genes, kittyOwner);
    }

    /// @dev Creates a new gen0 kitty with the given genes and
    ///  creates an auction for it.
    function createGen0Auction(uint256 _genes) external onlyCOO
{
        require(gen0CreatedCount < GEN0_CREATION_LIMIT);

        uint256 kittyId = _createKitty(0, 0, 0, _genes,
address(this));
        _approve(kittyId, saleAuction);

        saleAuction.createAuction(
            kittyId,
            _computeNextGen0Price(),
            0,
```

```
            GEN0_AUCTION_DURATION,
            address(this)
        );

        gen0CreatedCount++;
    }

    /// @dev Computes the next gen0 auction starting price,
given
    ///  the average of the past 5 prices + 50%.
    function _computeNextGen0Price() internal view returns
(uint256) {
        uint256 avePrice = saleAuction.averageGen0SalePrice();

        // Sanity check to ensure we don't overflow arithmetic
        require(avePrice == uint256(uint128(avePrice)));

        uint256 nextPrice = avePrice + (avePrice / 2);

        // We never auction for less than starting price
        if (nextPrice < GEN0_STARTING_PRICE) {
            nextPrice = GEN0_STARTING_PRICE;
        }

        return nextPrice;
    }
}


/// @title CryptoKitties: Collectible, breedable, and oh-so-
adorable cats on the Ethereum blockchain.
/// @author Axiom Zen (https://www.axiomzen.co)
/// @dev The main CryptoKitties contract, keeps track of
kittens so they don't wander around and get lost.
contract KittyCore is KittyMinting {

    // This is the main CryptoKitties contract. In order to
keep our code seperated into logical sections,
    // we've broken it up in two ways. First, we have several
seperately-instantiated sibling contracts
    // that handle auctions and our super-top-secret genetic
combination algorithm. The auctions are
    // seperate since their logic is somewhat complex and
there's always a risk of subtle bugs. By keeping
    // them in their own contracts, we can upgrade them without
disrupting the main contract that tracks
    // kitty ownership. The genetic combination algorithm is
kept seperate so we can open-source all of
```

```
    // the rest of our code without making it _too_ easy for
folks to figure out how the genetics work.
    // Don't worry, I'm sure someone will reverse engineer it
soon enough!
    //
    // Secondly, we break the core contract into multiple files
using inheritence, one for each major
    // facet of functionality of CK. This allows us to keep
related code bundled together while still
    // avoiding a single giant file with everything in it. The
breakdown is as follows:
    //
    //      - KittyBase: This is where we define the most
fundamental code shared throughout the core
    //             functionality. This includes our main data
storage, constants and data types, plus
    //             internal functions for managing these items.
    //
    //      - KittyAccessControl: This contract manages the
various addresses and constraints for operations
    //             that can be executed only by specific roles.
Namely CEO, CFO and COO.
    //
    //      - KittyOwnership: This provides the methods
required for basic non-fungible token
    //             transactions, following the draft ERC-721
spec (https://github.com/ethereum/EIPs/issues/721).
    //
    //      - KittyBreeding: This file contains the methods
necessary to breed cats together, including
    //             keeping track of siring offers, and relies
on an external genetic combination contract.
    //
    //      - KittyAuctions: Here we have the public methods
for auctioning or bidding on cats or siring
    //             services. The actual auction functionality
is handled in two sibling contracts (one
    //             for sales and one for siring), while auction
creation and bidding is mostly mediated
    //             through this facet of the core contract.
    //
    //      - KittyMinting: This final facet contains the
functionality we use for creating new gen0 cats.
    //             We can make up to 5000 "promo" cats that can
be given away (especially important when
    //             the community is new), and all others can
only be created and then immediately put up
    //             for auction via an algorithmically
```

```
determined starting price. Regardless of how they
    //             are created, there is a hard limit of 50k
gen0 cats. After that, it's all up to the
    //             community to breed, breed, breed!

    // Set in case the core contract is broken and an upgrade
is required
    address public newContractAddress;

    /// @notice Creates the main CryptoKitties smart contract
instance.
    function KittyCore() public {
        // Starts paused.
        paused = true;

        // the creator of the contract is the initial CEO
        ceoAddress = msg.sender;

        // the creator of the contract is also the initial COO
        cooAddress = msg.sender;

        // start with the mythical kitten 0 - so we don't have
generation-0 parent issues
        _createKitty(0, 0, 0, uint256(-1), address(0));
    }

    /// @dev Used to mark the smart contract as upgraded, in
case there is a serious
    ///  breaking bug. This method does nothing but keep track
of the new contract and
    ///  emit a message indicating that the new address is set.
It's up to clients of this
    ///  contract to update to the new contract address in that
case. (This contract will
    ///  be paused indefinitely if such an upgrade takes
place.)
    /// @param _v2Address new address
    function setNewAddress(address _v2Address) external onlyCEO
whenPaused {
        // See README.md for updgrade plan
        newContractAddress = _v2Address;
        ContractUpgrade(_v2Address);
    }

    /// @notice No tipping!
    /// @dev Reject all Ether from being sent here, unless it's
from one of the
    ///  two auction contracts. (Hopefully, we can prevent user
```

```
accidents.)
    function() external payable {
        require(
            msg.sender == address(saleAuction) ||
            msg.sender == address(siringAuction)
        );
    }

    /// @notice Returns all the relevant information about a
specific kitty.
    /// @param _id The ID of the kitty of interest.
    function getKitty(uint256 _id)
        external
        view
        returns (
        bool isGestating,
        bool isReady,
        uint256 cooldownIndex,
        uint256 nextActionAt,
        uint256 siringWithId,
        uint256 birthTime,
        uint256 matronId,
        uint256 sireId,
        uint256 generation,
        uint256 genes
    ) {
        Kitty storage kit = kitties[_id];

        // if this variable is 0 then it's not gestating
        isGestating = (kit.siringWithId != 0);
        isReady = (kit.cooldownEndBlock <= block.number);
        cooldownIndex = uint256(kit.cooldownIndex);
        nextActionAt = uint256(kit.cooldownEndBlock);
        siringWithId = uint256(kit.siringWithId);
        birthTime = uint256(kit.birthTime);
        matronId = uint256(kit.matronId);
        sireId = uint256(kit.sireId);
        generation = uint256(kit.generation);
        genes = kit.genes;
    }

    /// @dev Override unpause so it requires all external
contract addresses
    ///  to be set before contract can be unpaused. Also, we
can't have
    ///  newContractAddress set either, because then the
contract was upgraded.
    /// @notice This is public rather than external so we can
```

```
call super.unpause
    ///  without using an expensive CALL.
    function unpause() public onlyCEO whenPaused {
        require(saleAuction != address(0));
        require(siringAuction != address(0));
        require(geneScience != address(0));
        require(newContractAddress == address(0));

        // Actually unpause the contract.
        super.unpause();
    }

    // @dev Allows the CFO to capture the balance available to
the contract.
    function withdrawBalance() external onlyCFO {
        uint256 balance = this.balance;
        // Subtract all the currently pregnant kittens we have,
plus 1 of margin.
        uint256 subtractFees = (pregnantKitties + 1) *
autoBirthFee;

        if (balance > subtractFees) {
            cfoAddress.send(balance - subtractFees);
        }
    }
}
```

3. CryptoZombies

http://www.cryptozombies.io

一款游戏

4. Augur Project

Augur 是建立在以太坊平台上的去中心化预测市场平台，也是以太坊平台上第一个众筹的应用，利用 Augur任何人都可以为任何自己感兴趣的主题创建一个预测市场，并提供初始流动性。Augur 项目于2015年8月开启代币众筹，一共众筹到500万美元，代币名称为REP。

https://www.augur.net

合约代码地址 https://github.com/AugurProject/augur-core

Token sale contract and tools: https://github.com/AugurProject/token-sale

5. Golem

Golem 是一个去中心化的全球算力市场。Golem于2016年11月11日开启众筹，约半个小时完成众筹上限82万ETH，以当日ETH71元人民币的收盘价计算，共筹得约5822万元人民币，每枚代币成本约为0.071元人民币，代币名称为GNT。

Golem ICO众筹合约代码由Token.sol和GNTAllocation.sol两个文件组成，其中GNTAllocation.sol中定义了分配给Golem公司和开发者的Token数量，主要部分的合约写在Token.sol中。

合约源码地址： https://github.com/golemfactory/golem-crowdfunding

6. FirstBlood

第一滴血是一个可以让玩家随时随地参与竞赛并且获得赏金的去中心化电竞平台。第一滴血ICO于2016年9月26日开始，达成众筹额度仅耗费两分钟，共筹得约465，313ETH，发放代币1SF约7910万枚，每枚代币成本约为0.5元人民币。

合约源码地址：https://github.com/Firstbloodio/token

7. Bancor

Bancor协议是一种基于以太坊的底层货币协议，通过区块链技术和智能合约实现小型加密数字货币之间的连续流动性和异步价格发现。Bancor于2017年6月13日开启众筹，数据显示，在ICO于18:00结束时已经筹集了超过39万个以太，发行了将近8000万的BNT。

合约源码地址： https://github.com/bancorprotocol/contracts

第 23 章 FAQ

1. Error: etherbase missing: etherbase address must be explicitly specified

```
ERROR[02-10|16:12:45] Cannot start mining without etherbase
err="etherbase address must be explicitly specified"
Error: etherbase missing: etherbase address must be explicitly
specified
    at web3.js:3143:20
    at web3.js:6347:15
    at web3.js:5081:36
    at <anonymous>:1:1
```

原因是当前环境没有账户，需要建立一个账户

```
> personal.newAccount("chen")
"0x1b94732fca6f62a4f74fb2f7c80bfc89d567fdfb"
```

现在启动挖矿就不会出现问题了

3. Error: authentication needed: password or unlock

```
> eth.sendTransaction({from:
'0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6', to:
'0xe8abf98484325fd6afc59b804ac15804b978e607', value:
web3.toWei(1, "ether")})
Error: authentication needed: password or unlock
    at web3.js:3143:20
    at web3.js:6347:15
    at web3.js:5081:36
    at <anonymous>:1:1
```

解锁转出账号

```
>
personal.unlockAccount("0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb
5a6", "your_password", 300)
true
> eth.sendTransaction({from:
'0x83fda0ba7e6cfa8d7319d78fa0e6b753a2bcb5a6', to:
'0xe8abf98484325fd6afc59b804ac15804b978e607', value:
web3.toWei(1, "ether")})
"0xd9e8c8fdc71e24ee8052048de4ff0acd7157b872393f37344c8ec2083f3f
e48f"
```

4. 新增节点后不生效

新增节点显示 true ,但是使用 admin.peers 查看不到。

```
>
admin.addPeer("enode://c4586276391b3c88ec23889d1bc825d0c7d69bd5765d45456
86f835608068b8dc48799d2686a04ea0f9e17aed099bf9b56935679fa6493e9b17151624
a320714@172.16.0.17:30303")
true
> admin.peers
[]
```

查看节点，如果发现 ip: "::" 同时 discovery: 0

```
> admin.nodeInfo
{
  enode:
"enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f
470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5b6b17a8a4b40527@[::]:3
0303?discport=0",
  id:
"9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f470b642f
d2cc3507f7e851df60aaeb730a1270b7a477f91ec5b6b17a8a4b40527",
  ip: "::",
  listenAddr: "[::]:30303",
  name: "Geth/v1.7.3-stable-4bb3c89d/linux-amd64/go1.9.1",
  ports: {
    discovery: 0,
    listener: 30303
  },
  protocols: {
    eth: {
      difficulty: 131072,
      genesis:
"0x611596e7979cd4e7ca1531260fa706093a5492ecbdf58f20a39545397e424d04",
      head:
"0x611596e7979cd4e7ca1531260fa706093a5492ecbdf58f20a39545397e424d04",
      network: 123456
    }
  }
}
>
```

解决方案启动时可能增加了 --nodiscover 参数，去掉参数后可以解决。

有三种方法新增节点

第一种，启动指定

```
geth --bootnodes
enode://pubkey1@ip1:port1,enode://pubkey2@ip2:port2,enode://pubkey3@ip3:
port3
```

第二种，在控制台中添加

```
>
admin.addPeer('enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d6702
9cf4a6c4693d2f470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5b6b17a8a
4b40527@172.16.0.1:30303')
```

第三种，在文件 ~/.ethereum/geth/static-nodes.json 中添加节点数据

```
[

"enode://9f6490ffb5236f2ddc5710ae73d47c740e0a3644bbd2d67029cf4a6c4693d2f
470b642fd2cc3507f7e851df60aaeb730a1270b7a477f91ec5b6b17a8a4b40527@172.16
.0.17:30303",
  "enode://pubkey@ip:port"
]
```

这个文件内容是一个数组，类似 ["","",""] 。

5. Unhandled rejection Error: Returned error: The method personal_unlockAccount does not exist/is not available

```
> Unhandled rejection Error: Returned error: The method
personal_unlockAccount does not exist/is not available
    at Object.ErrorResponse
(/Users/neo/ethereum/web3/node_modules/web3-core-
helpers/src/errors.js:29:16)
    at /Users/neo/ethereum/web3/node_modules/web3-core-
requestmanager/src/index.js:137:36
    at XMLHttpRequest.request.onreadystatechange
(/Users/neo/ethereum/web3/node_modules/web3-providers-
http/src/index.js:77:13)
    at XMLHttpRequestEventTarget.dispatchEvent
(/Users/neo/ethereum/web3/node_modules/xhr2/lib/xhr2.js:64:18)
    at XMLHttpRequest._setReadyState
(/Users/neo/ethereum/web3/node_modules/xhr2/lib/xhr2.js:354:12)
    at XMLHttpRequest._onHttpResponseEnd
(/Users/neo/ethereum/web3/node_modules/xhr2/lib/xhr2.js:509:12)
    at IncomingMessage.<anonymous>
(/Users/neo/ethereum/web3/node_modules/xhr2/lib/xhr2.js:469:24)
    at IncomingMessage.emit (events.js:165:20)
    at endReadableNT (_stream_readable.js:1101:12)
    at process._tickCallback
(internal/process/next_tick.js:152:19)
```

解决方法

```
$ geth --rpc --rpcapi personal,db,eth,net,web3
```

在 --rpcapi 选项中增加 personal

6. Error: exceeds block gas limit

```
neo@MacBook-Pro ~/ethereum/truffle/Conference % truffle migrate --
reset
\Using network 'development'.

Running migration: 1_initial_conference.js
  Deploying Conference...
  ... undefined
Error encountered, bailing. Network state unknown. Review successful
transactions manually.
Error: exceeds block gas limit
    at Object.InvalidResponse
(/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/er
rors.js:38:1)
    at
/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/req
uestmanager.js:86:1
    at /usr/local/lib/node_modules/truffle/build/webpack:/~/truffle-
migrate/index.js:225:1
    at /usr/local/lib/node_modules/truffle/build/webpack:/~/truffle-
provider/wrapper.js:134:1
    at XMLHttpRequest.request.onreadystatechange
(/usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/ht
tpprovider.js:128:1)
    at XMLHttpRequestEventTarget.dispatchEvent
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js
:64:1)
    at XMLHttpRequest._setReadyState
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js
:354:1)
    at XMLHttpRequest._onHttpResponseEnd
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js
:509:1)
    at IncomingMessage.<anonymous>
(/usr/local/lib/node_modules/truffle/build/webpack:/~/xhr2/lib/xhr2.js
:469:1)
    at IncomingMessage.emit (events.js:165:20)
```

```
neo@MacBook-Pro ~/ethereum/truffle/contracts % geth attach
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.1-stable/darwin-amd64/go1.10
coinbase: 0x5c18a33df2cc41a1beddc91133b8422e89f041b7
```

```
at block: 5381 (Wed, 28 Feb 2018 23:20:05 CST)
 datadir: /Users/neo/Library/Ethereum
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0
rpc:1.0 txpool:1.0 web3:1.0

> web3.eth.getBlock("pending").gasLimit
4712388
```

或者

```
$ truffle console
truffle(development)> web3.eth.getBlock("pending").gasLimit
6712390
```

修改 truffle.js 文件，加入 gas 值为上面所查询的值:

```
module.exports = {
  networks: {
    development: {
      host: "localhost",
      port: 8545,
      gas: 6712390,
      network_id: "*" // Match any network id
    }
  }
};
```

7. Migrations.sol:11:3: Warning: Defining constructors as functions with the same name as the contract is deprecated. Use "constructor(...) { ... }" instead.

Solidity 0.4.23 合约构造方法与之前的版本不同

```
[ethereum@netkiller truffle]$ cat contracts/Migrations.sol
pragma solidity ^0.4.17;

contract Migrations {
  address public owner;
  uint public last_completed_migration;

  modifier restricted() {
    if (msg.sender == owner) _;
  }

  function Migrations() public {
    owner = msg.sender;
  }

  function setCompleted(uint completed) public restricted {
    last_completed_migration = completed;
  }

  function upgrade(address new_address) public restricted {
    Migrations upgraded = Migrations(new_address);
    upgraded.setCompleted(last_completed_migration);
  }
}
```

```
[ethereum@netkiller truffle]$ cat contracts/Migrations.sol
pragma solidity ^0.4.17;

contract Migrations {
```

```
  address public owner;
  uint public last_completed_migration;

  modifier restricted() {
    if (msg.sender == owner) _;
  }

  constructor() public {
    owner = msg.sender;
  }

  function setCompleted(uint completed) public restricted {
    last_completed_migration = completed;
  }

  function upgrade(address new_address) public restricted {
    Migrations upgraded = Migrations(new_address);
    upgraded.setCompleted(last_completed_migration);
  }
}
```

8. Exception in thread "main" rx.exceptions.OnErrorNotImplementedException: Invalid response received: okhttp3.internal.http.RealResponseBody@6c25e6c4

异常出现在 Subscription 订阅的时候，原因是 web3j Subscription 操作只能使用 ipc, rpc， infura.io 不支持该操作。

```
Exception in thread "main"
rx.exceptions.OnErrorNotImplementedException: Invalid response
received: okhttp3.internal.http.RealResponseBody@6c25e6c4
        at
rx.internal.util.InternalObservableUtils$ErrorNotImplementedAct
ion.call(InternalObservableUtils.java:386)
        at
rx.internal.util.InternalObservableUtils$ErrorNotImplementedAct
ion.call(InternalObservableUtils.java:383)
        at
rx.internal.util.ActionSubscriber.onError(ActionSubscriber.java
:44)
        at
rx.observers.SafeSubscriber._onError(SafeSubscriber.java:153)
        at
rx.observers.SafeSubscriber.onError(SafeSubscriber.java:115)
        at
rx.internal.operators.OnSubscribeMap$MapSubscriber.onError(OnSu
bscribeMap.java:88)
        at
rx.internal.operators.OnSubscribeFilter$FilterSubscriber.onErro
r(OnSubscribeFilter.java:90)
        at
rx.internal.operators.OperatorMerge$MergeSubscriber.reportError
(OperatorMerge.java:266)
        at
rx.internal.operators.OperatorMerge$MergeSubscriber.checkTermin
ate(OperatorMerge.java:818)
        at
```

```
rx.internal.operators.OperatorMerge$MergeSubscriber.emitLoop(Op
eratorMerge.java:579)
        at
rx.internal.operators.OperatorMerge$MergeSubscriber.emit(Operat
orMerge.java:568)
        at
rx.internal.operators.OperatorMerge$MergeSubscriber.onError(Ope
ratorMerge.java:276)
        at
rx.internal.operators.OnSubscribeMap$MapSubscriber.onError(OnSu
bscribeMap.java:88)
        at rx.Observable.unsafeSubscribe(Observable.java:10334)
        at
rx.internal.operators.OnSubscribeMap.call(OnSubscribeMap.java:4
8)
        at
rx.internal.operators.OnSubscribeMap.call(OnSubscribeMap.java:3
3)
        at
rx.internal.operators.OnSubscribeLift.call(OnSubscribeLift.java
:48)
        at
rx.internal.operators.OnSubscribeLift.call(OnSubscribeLift.java
:30)
        at rx.Observable.unsafeSubscribe(Observable.java:10327)
        at
rx.internal.operators.OnSubscribeFilter.call(OnSubscribeFilter.
java:45)
        at
rx.internal.operators.OnSubscribeFilter.call(OnSubscribeFilter.
java:30)
        at rx.Observable.unsafeSubscribe(Observable.java:10327)
        at
rx.internal.operators.OnSubscribeMap.call(OnSubscribeMap.java:4
8)
        at
rx.internal.operators.OnSubscribeMap.call(OnSubscribeMap.java:3
3)
        at rx.Observable.subscribe(Observable.java:10423)
        at rx.Observable.subscribe(Observable.java:10390)
        at rx.Observable.subscribe(Observable.java:10195)
        at
cn.netkiller.example.ethereum.subscription.PendingTest.main(Pen
dingTest.java:20)
Caused by:
org.web3j.protocol.exceptions.ClientConnectionException:
Invalid response received:
okhttp3.internal.http.RealResponseBody@6c25e6c4
```

```
        at
org.web3j.protocol.http.HttpService.performIO(HttpService.java:
114)
        at org.web3j.protocol.Service.send(Service.java:31)
        at
org.web3j.protocol.core.Request.send(Request.java:71)
        at
org.web3j.protocol.core.filters.PendingTransactionFilter.sendRe
quest(PendingTransactionFilter.java:24)
        at
org.web3j.protocol.core.filters.Filter.run(Filter.java:45)
        at
org.web3j.protocol.rx.JsonRpc2_0Rx.run(JsonRpc2_0Rx.java:73)
        at
org.web3j.protocol.rx.JsonRpc2_0Rx.lambda$ethPendingTransaction
HashObservable$1(JsonRpc2_0Rx.java:55)
        at rx.Observable.unsafeSubscribe(Observable.java:10327)
        ... 14 more
```

部分 III. Hyperledger

第 24 章 Hyperledger Fabric v2.0.0

区块链技术发展至今，形成了公有链、联盟链和邦链三种种主流技术平台。

公有链：面向大众，用户可以匿名参与，非常方便，账本数据也公开，加上强大的智能合约，因此公有链极大地促进了区块链概念和技术的普及，比如比特币、Ethereum平台等。

联盟链：考虑到商业应用对安全、隐私、监管、审计、性能的需求，提高准入门槛，增加了安全、隐私、可监管审计等商业特性，是区块链技术在商业领域的应用探索。

邦链：暂时资料比较少。

概念

通道（ Channel ）：通道是构建在 Hyperledger Fabric 区块链网络上的私有区块链，实现了数据的隔离和保密。通道中的 Chaincode 和交易只有加入该通道的节点（ Peer ）可见。同一个节点可以加入多个通道，并为每个通道内容维护一个账本。每一个通道即为一条逻辑上的区块链。可以按照业务来划分通道，也可以按照行政职能和隐私策略来划分通道。

节点（ Peer ）： 维护账本的网络节点，通常区块链网络架构中存在多种角色，如 endorser 和 committer 。

排序服务或共识服务（ Order Services ）： 提供排序服务或共识服务的网络节点，完成交易的排序和区块打包等工作，支持可插拔的共识组件，当前生产环境下使用 Kafka 进行交易排序。

分布式账本（ Distribute Ledger ）： 由网络中若干去中心化节点共同维护的数据账本。

组织（ Org ）：联盟链中按照访问和使用账本的网络节点，一个联盟（或者一个区块链网络）有多个组织（成员），一个组织内可以有多个节点（ Peer ），每个节点参与账本和世界状态维护。

智能合约（ Smart Contract ）：根据特定条件自动执行的合约程序。智能合约是区块链的重要特征，是用户与区块链进行交互，利用区块链实现业务逻辑的重要途径。

链码（ Chaincode ）：链码是 Hyperledger Fabric 对智能合约的一种实现方式，是运行于 Hyperledger Fabric 网络之上一段应用程序代码，也是用户与 Hyperledger Fabric 交互的唯一途径。

链（ Chain ）：一个链即是一个由若干区块通过特定指向链接、摘要算法或加密算法锚定组成的数据集合。

1. 安装 Hyperledger Fabric v1.1.x

1.1. 依赖工具

```
yum -y install epel-release
yum install -y git
yum install -y golang
```

1.2. 安装Docker

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualization/docker/docker.centos7.ce.sh
| bash
curl -s https://raw.githubusercontent.com/oscm/shell/master/virtualization/docker/registry-
mirror.sh        | bash
or
curl -fsSL https://get.docker.com/ | sh

创建2个docker网络,如下:
docker network create fabric_noops
docker network create fabric_pbft
```

安装 docker-compose

```
curl -s https://raw.githubusercontent.com/oscm/shell/master/virtualization/docker/docker-
compose/docker-compose.sh        | bash
```

1.3. 安装 Node.js 环境

```
yum install -y nodejs
npm install
```

1.4. 安装 hyperledger 1.1.0

运行后 hyperledger 相关镜像被安装到 Docker 中

```
cd /usr/local/src
curl -s https://raw.githubusercontent.com/hyperledger/fabric/release/scripts/bootstrap-1.1.0.sh
| bash
```

由于上面脚本会安装所有节点，速度较慢，作者建议你参考下一章节，手工安装所需最低配置节点。

1.5. 手工安装 hyperledger v 1.1.0 开发环境

对于开发环境，最小化的环境，包括 1 个 peer 节点、1 个 Orderer 节点、1 个 CA 节点。

准备一个服务器或者虚拟机，安装 CentOS 操作系统。

如果你是在已有的Docker上安装，建议你删除所有容器后在安装。以免出现冲突等情况。

```
docker stop $(docker ps -q) && docker rm $(docker ps -aq)
```

1.5.1. 登录 docker

```
[root@localhost ~]# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker
ID, head over to https://hub.docker.com to create one.
Username: netkiller
Password:
Login Succeeded
```

1.5.2. Docker 安装

```
docker pull hyperledger/fabric-baseimage:latest \
  && docker pull hyperledger/fabric-membersrvc:latest \
  && docker pull hyperledger/fabric-peer:latest \
  && docker pull hyperledger/fabric-orderer:latest \
  && docker pull hyperledger/fabric-ca:latest \
  && docker pull hyperledger/blockchain-explorer:latest
```

安装会出现下面问题

```
[root@localhost ~]# docker search fabric-peer | grep hyperledger/fabric-peer
hyperledger/fabric-peer            Fabric Peer docker image for Hyperledger Pro…   69

[root@localhost ~]# docker pull hyperledger/fabric-peer:latest
Error response from daemon: manifest for hyperledger/fabric-peer:latest not found
```

可以 search 到的镜像 pull 不了，原因是 fabric-peer:latest，latest 不存在，你需要指定版本号。

```
docker pull hyperledger/fabric-ca:x86_64-1.1.0 \
&& docker pull hyperledger/fabric-peer:x86_64-1.1.0 \
&& docker pull hyperledger/fabric-orderer:x86_64-1.1.0 \
&& docker pull hyperledger/fabric-couchdb:x86_64-1.1.0 \
&& docker pull hyperledger/fabric-tools:x86_64-1.1.0

docker tag hyperledger/fabric-ca:x86_64-1.1.0 hyperledger/fabric-ca \
&& docker tag hyperledger/fabric-peer:x86_64-1.1.0 hyperledger/fabric-peer \
&& docker tag hyperledger/fabric-orderer:x86_64-1.1.0 hyperledger/fabric-orderer \
&& docker tag hyperledger/fabric-couchdb:x86_64-1.1.0 hyperledger/fabric-couchdb \
&& docker tag hyperledger/fabric-tools:x86_64-1.1.0 hyperledger/fabric-tools
```

查看镜像

```
[root@localhost src]# docker images
REPOSITORY                    TAG                 IMAGE ID            CREATED             SIZE
hyperledger/fabric-orderer    latest              368c78b6f03b        2 months ago
151MB
hyperledger/fabric-orderer    x86_64-1.1.0        368c78b6f03b        2 months ago
151MB
hyperledger/fabric-peer       latest              c2ab022f0bdb        2 months ago
154MB
hyperledger/fabric-peer       x86_64-1.1.0        c2ab022f0bdb        2 months ago
154MB
```

```
hyperledger/fabric-membersrvc   latest              b3654d32e4f9        15 months ago
1.42GB
```

1.5.3. 编译安装

```
git config --global core.autocrlf false
$ git clone https://github.com/hyperledger/fabric.git
$ make docker
$ git clone https://github.com/hyperledger/fabric-ca.git
$ make docker

cd fabric/devenv
vagrant box add hyperledger/fabric-baseimage centos7.box
vagrant up

yum -y install epel-release
yum -y install git
yum -y install golang
yum -y install python-pip
pip install --upgrade backports.ssl_match_hostname
pip install docker-compose

docker-compose -version
```

1.6. 启动 docker 虚拟机

体验 Hyperledger Fabric 在 https://github.com/hyperledger/fabric/tree/release/examples 下面有一些例子供用户学习。这里我选择的是 fabric-samples

这里我们最小化启动，需要四个节点，分别是 ca, peer, order, couchdb。

创建文件 docker-compose.yml

```
[root@localhost ~]# mkdir -p docker
[root@localhost ~]# cd docker
[root@localhost docker]# vim docker-compose.yml
#
# Copyright IBM Corp All Rights Reserved
#
# SPDX-License-Identifier: Apache-2.0
#
version: '2'

networks:
  basic:

services:
  ca.example.com:
    image: hyperledger/fabric-ca
    environment:
      - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
      - FABRIC_CA_SERVER_CA_NAME=ca.example.com
      - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/fabric-ca-server-
config/ca.org1.example.com-cert.pem
      - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/fabric-ca-server-
config/4239aa0dcd76daeeb8ba0cda701851d14504d31aad1b2ddddbac6a57365e497c_sk
    ports:
      - "7054:7054"
```

```
    command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
    volumes:
      - ./crypto-config/peerOrganizations/org1.example.com/ca/:/etc/hyperledger/fabric-ca-
server-config
    container_name: ca.example.com
    networks:
      - basic

  orderer.example.com:
    container_name: orderer.example.com
    image: hyperledger/fabric-orderer
    environment:
      - ORDERER_GENERAL_LOGLEVEL=debug
      - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
      - ORDERER_GENERAL_GENESISMETHOD=file
      - ORDERER_GENERAL_GENESISFILE=/etc/hyperledger/configtx/genesis.block
      - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
      - ORDERER_GENERAL_LOCALMSPDIR=/etc/hyperledger/msp/orderer/msp
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric/orderer
    command: orderer
    ports:
      - 7050:7050
    volumes:
        - ./config/:/etc/hyperledger/configtx
        - ./crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/:/etc/hyperledger/msp/order
er
        - ./crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/:/etc/hyperledger/msp/pee
rOrg1
    networks:
      - basic

  peer0.org1.example.com:
    container_name: peer0.org1.example.com
    image: hyperledger/fabric-peer
    environment:
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      - CORE_PEER_ID=peer0.org1.example.com
      - CORE_LOGGING_PEER=debug
      - CORE_CHAINCODE_LOGGING_LEVEL=DEBUG
      - CORE_PEER_LOCALMSPID=Org1MSP
      - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/peer/
      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
      # # the following setting starts chaincode containers on the same
      # # bridge network as the peers
      # # https://docs.docker.com/compose/networking/
      - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}_basic
      - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
      - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb:5984
      # The CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME and
CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD
      # provide the credentials for ledger to connect to CouchDB.  The username and password
must
      # match the username and password set for the associated CouchDB.
      - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=
      - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric
    command: peer node start
    # command: peer node start --peer-chaincodedev=true
    ports:
      - 7051:7051
      - 7053:7053
    volumes:
        - /var/run/:/host/var/run/
        - ./crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/msp:/etc/hyperledger/msp/
peer
        - ./crypto-config/peerOrganizations/org1.example.com/users:/etc/hyperledger/msp/users
        - ./config:/etc/hyperledger/configtx
    depends_on:
      - orderer.example.com
```

```
      - couchdb
    networks:
      - basic

  couchdb:
    container_name: couchdb
    image: hyperledger/fabric-couchdb
    # Populate the COUCHDB_USER and COUCHDB_PASSWORD to set an admin user and password
    # for CouchDB.  This will prevent CouchDB from operating in an "Admin Party" mode.
    environment:
      - COUCHDB_USER=
      - COUCHDB_PASSWORD=
    ports:
      - 5984:5984
    networks:
      - basic

  cli:
    container_name: cli
    image: hyperledger/fabric-tools
    tty: true
    environment:
      - GOPATH=/opt/gopath
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      - CORE_LOGGING_LEVEL=DEBUG
      - CORE_PEER_ID=cli
      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
      - CORE_PEER_LOCALMSPID=Org1MSP
      -
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizati
ons/org1.example.com/users/Admin@org1.example.com/msp
      - CORE_CHAINCODE_KEEPALIVE=10
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
    command: /bin/bash
    volumes:
        - /var/run/:/host/var/run/
        - ./../chaincode/:/opt/gopath/src/github.com/
        - ./crypto-config:/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
    networks:
        - basic
    #depends_on:
    #  - orderer.example.com
    #  - peer0.org1.example.com
    #  - couchdb
```

启动 Docker

```
[root@localhost docker]# docker-compose -f docker-compose.yml up -d ca.example.com
orderer.example.com peer0.org1.example.com couchdb
```

查看进程

```
[root@localhost docker]# docker-compose ps
         Name                       Command               State                         Ports
-------------------------------------------------------------------------------------------------------
----------------
ca.example.com           sh -c fabric-ca-server sta ...   Up      0.0.0.0:7054->7054/tcp
cli                      /bin/bash                        Up
couchdb                  tini -- /docker-entrypoint ...   Up      4369/tcp, 0.0.0.0:5984-
>5984/tcp, 9100/tcp
orderer.example.com      orderer                          Up      0.0.0.0:7050->7050/tcp
```

```
peer0.org1.example.com   peer node start                 Up      0.0.0.0:7051->7051/tcp,
0.0.0.0:7053->7053/tcp
```

1.7. 管理 hyperledger

1.7.1. CouchDB 管理界面

```
[root@localhost fabcar]# curl http://localhost:5984
{"couchdb":"Welcome","version":"2.0.0","vendor":{"name":"The Apache Software Foundation"}}
```

http://localhost:5984/_utils/

1.8. 部署 chaincode

1.8.1. channel 管理

Hyperledger Fabric Channel 可以理解为 vlan （交换机术语） 用来实现区块隔离。

```
[root@localhost docker]# docker-compose exec peer0.org1.example.com bash
root@dcb09db1cbc8:/go/src/github.com/hyperledger/fabric#
```

1.8.1.1. 列出 channel

```
root@595ec455c0ff:/opt/gopath/src/github.com/hyperledger/fabric# peer channel list
2018-02-07 03:24:41.151 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-07 03:24:41.152 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default
signing identity
2018-02-07 03:24:41.154 UTC [channelCmd] InitCmdFactory -> INFO 003 Endorser and orderer
connections initialized
2018-02-07 03:24:41.155 UTC [msp/identity] Sign -> DEBU 004 Sign: plaintext:
0A85070A5B08031A0B08F9E2E9D30510...631A0D0A0B4765744368616E6E656C73
2018-02-07 03:24:41.156 UTC [msp/identity] Sign -> DEBU 005 Sign: digest:
238CBAB61A0524954DC3C511588EB8FC1F886E636A8800131EBE16FB95FB0C9A
2018-02-07 03:24:41.167 UTC [channelCmd] list -> INFO 006 Channels peers has joined to:
2018-02-07 03:24:41.167 UTC [channelCmd] list -> INFO 007 mychannel
2018-02-07 03:24:41.167 UTC [main] main -> INFO 008 Exiting.....
```

1.8.1.2. 创建 Channel

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp
```

```
peer channel create -o orderer.example.com:7050 -c mychannel -f
/etc/hyperledger/configtx/channel.tx
```

1.8.1.3. 加入 Channel

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp
peer channel join -b mychannel.block
```

1.8.2. 部署连

```
[root@localhost basic-network]# docker-compose exec cli bash
root@b1ded848f967:/opt/gopath/src/github.com/hyperledger/fabric/peer#
```

安装合约

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizati
ons/org1.example.com/users/Admin@org1.example.com/msp
peer chaincode install -n fabcar -v 1.0 -p github.com/fabcar
```

实例化合约

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizati
ons/org1.example.com/users/Admin@org1.example.com/msp
peer chaincode instantiate -o orderer.example.com:7050 -C mychannel -n fabcar -v 1.0 -c
'{"Args":[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
```

初始化合约

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizati
ons/org1.example.com/users/Admin@org1.example.com/msp
peer chaincode invoke -o orderer.example.com:7050 -C mychannel -n fabcar -c
'{"function":"initLedger","Args":[""]}'
```

1.8.3. 查询合约

```
root@b1ded848f967:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode query -o
```

```
orderer.example.com:7050 -C mychannel -n fabcar -c '{"function":"queryCar","Args":["CAR9"]}'
2018-02-07 05:11:59.737 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-07 05:11:59.737 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default
signing identity
2018-02-07 05:11:59.738 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using default
escc
2018-02-07 05:11:59.738 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using default
vscc
2018-02-07 05:11:59.738 UTC [msp/identity] Sign -> DEBU 005 Sign: plaintext:
0A93070A6908031A0C089F95EAD30510...0A0871756572794361720A0443415239
2018-02-07 05:11:59.739 UTC [msp/identity] Sign -> DEBU 006 Sign: digest:
84E6F021BEA8C5F2D97B1C8BFEA4BB07B91DBC167E0BBD188260B234DC4620E8
Query Result: {"colour":"brown","make":"Holden","model":"Barina","owner":"Shotaro"}
2018-02-07 05:11:59.771 UTC [main] main -> INFO 007 Exiting.....
```

1.8.4. 调用合约

调用合约，新增一条记录，然后做查询操作。

```
peer chaincode invoke -o orderer.example.com:7050 -C mychannel -n fabcar -c
'{"function":"createCar","Args":["CAR15", "Toyota", "Rezi", "White", "Neo"]}'
peer chaincode query -o orderer.example.com:7050 -C mychannel -n fabcar -c
'{"function":"queryCar","Args":["CAR15"]}'
```

演示结果

```
root@b1ded848f967:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode invoke -o
orderer.example.com:7050 -C mychannel -n fabcar -c '{"function":"createCar","Args":["CAR15",
"Toyota", "Rezi", "White", "Neo"]}'
2018-02-07 05:16:57.415 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-07 05:16:57.415 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default
signing identity
2018-02-07 05:16:57.417 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using default
escc
2018-02-07 05:16:57.417 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using default
vscc
2018-02-07 05:16:57.418 UTC [msp/identity] Sign -> DEBU 005 Sign: plaintext:
0A93070A6908031A0C08C997EAD30510...52657A690A0557686974650A034E656F
2018-02-07 05:16:57.418 UTC [msp/identity] Sign -> DEBU 006 Sign: digest:
D9BDF565353FFAD37CDC64DBFBD06DA8B5A049AA32305E1668A695E6522C3043
2018-02-07 05:16:57.443 UTC [msp/identity] Sign -> DEBU 007 Sign: plaintext:
0A93070A6908031A0C08C997EAD30510...4B5F674D96D8DE9BE699613B72ED46B1
2018-02-07 05:16:57.443 UTC [msp/identity] Sign -> DEBU 008 Sign: digest:
2951E3522A902D2644E08508E52A1D833AF9E11A420EE933851CC5673B005375
2018-02-07 05:16:57.451 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> DEBU 009 ESCC invoke
result: version:1 response:<status:200 message:"OK" > payload:"\n
\327\324\006\316\004\230\3262{\300\\|\275\035\336\243\316L\232\247x\020n`2\346\334\031\n(<\262\0
22\204\001\nn\022T\n\006fabcar\022J\032H\n\005CAR15\032?
{\"make\":\"Toyota\",\"model\":\"Rezi\",\"colour\":\"White\",\"owner\":\"Neo\"}\022\026\n\004lsc
c\022\016\n\014\n\006fabcar\022\002\010\001\032\003\010\310\001\"\r\022\006fabcar\032\0031.0"
endorsement:<endorser:"\n\007Org1MSP\022\200\006-----BEGIN -----
\nMIICGjCCAcCgAwIBAgIRAPlwF/rUZUP9mqN4wSml4iswCgYIKoZIzj0EAwIwczEL\nMAkGA1UEBhMCVVMxEzARBgNVBAgT
CkNhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiBG\ncmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBA
MTE2Nh\nLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcwODMxMDkxNDMyWhcNMjcwODI5MDkxNDMy\nWjBbMQswCQYDVQQGEwJVUz
ETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMN\nU2FuIEZyYW5jaXNjbzEfMB0GA1UEAxMWcGVlcjAub3JnMS5leGFt
cGxlLmNvbTBZ\nMBMGByqGSM49AgEGCCqGSM49AwEHA0IABHihxW6ks3B2+5XdbAVq3CBgxRRRZ22x\nzzpqnD86nKkz7fBE
lBuhlXl2K6rTxyY2OBOB0ts8keqZ93xueRGymrajTTBLMA4G\nA1UdDwEB/wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIw
QkMCKAIEI5qg3Ndtru\nuLoM2nAYUdFFBNMarRst3dusalc2Xkl8MAoGCCqGSM49BAMCA0gAMEUCIQD4j0Rn\ne1rrd0FSCz
sR6u+IuuPK5dI/kR/bh7+VLf0TNgIgCfUtkJvfvzVEwZLFoFyjoHtr\ntvwzNUS1U0hEqIaDeo4=\n-----END -----\n"
signature:"0E\002!\000\364q\r\026\267\205\357\245\006\364\354v\333r92\022l\2267[Yb@F\263N\230\32
```

```
4\351.\025\002 \022K\275P\234\262A\3338\244\337\216\340q%-
K_gM\226\330\336\233\346\231a;r\355F\261" >
2018-02-07 05:16:57.451 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 00a Chaincode invoke
successful. result: status:200
2018-02-07 05:16:57.451 UTC [main] main -> INFO 00b Exiting.....


root@b1ded848f967:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode query -o
orderer.example.com:7050 -C mychannel -n fabcar -c '{"function":"queryCar","Args":["CAR15"]}'
2018-02-07 05:17:07.383 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-07 05:17:07.383 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining default
signing identity
2018-02-07 05:17:07.383 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using default
escc
2018-02-07 05:17:07.383 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using default
vscc
2018-02-07 05:17:07.384 UTC [msp/identity] Sign -> DEBU 005 Sign: plaintext:
0A93070A6908031A0C08D397EAD30510...087175657279436172 0A054341523135
2018-02-07 05:17:07.384 UTC [msp/identity] Sign -> DEBU 006 Sign: digest:
FBFD595C716FB185BABBBD3709040F6D1538964931BA47A8B653E878C2084C4B
Query Result: {"colour":"White","make":"Toyota","model":"Rezi","owner":"Neo"}
2018-02-07 05:17:07.411 UTC [main] main -> INFO 007 Exiting.....
```

2. Ubuntu 环境安装 Hyperledger Fabric v1.1.0

2.1. 安装 Docker

Ubuntu apt 库中携带的 docker.io 版本过低，我们从官网安装Docker CE（社区版）取代他。

确保环境是干净的，卸载旧版本的 Docker

```
sudo apt-get remove docker docker-engine docker.io
```

从官网安装新版本的 Docker

```
sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"

sudo apt-get update
sudo apt-get install docker-ce

sudo apt-get install python-pip
pip install docker-compose
```

启动 docker

```
neo@netkiller ~ % sudo systemctl start docker
neo@netkiller ~ % sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor
preset: enabled)
   Active: active (running) since Sat 2018-03-10 04:56:01 HST; 4s ago
     Docs: https://docs.docker.com
 Main PID: 27572 (dockerd)
    Tasks: 18
   Memory: 21.9M
      CPU: 578ms
```

```
   CGroup: /system.slice/docker.service
           ├─27572 /usr/bin/dockerd -H fd://
           └─27587 docker-containerd --config
/var/run/docker/containerd/containerd.toml

Mar 10 04:56:00 netkiller dockerd[27572]: time="2018-03-
10T04:56:00.020801698-10:00" level=warning msg="Your kernel does not
support swap memory limit"
Mar 10 04:56:00 netkiller dockerd[27572]: time="2018-03-
10T04:56:00.020948715-10:00" level=warning msg="Your kernel does not
support cgroup rt period"
Mar 10 04:56:00 netkiller dockerd[27572]: time="2018-03-
10T04:56:00.020991877-10:00" level=warning msg="Your kernel does not
support cgroup rt runtime"
Mar 10 04:56:00 netkiller dockerd[27572]: time="2018-03-
10T04:56:00.024334084-10:00" level=info msg="Loading containers: start."
Mar 10 04:56:00 netkiller dockerd[27572]: time="2018-03-
10T04:56:00.786564515-10:00" level=info msg="Default bridge (docker0) is
assigned with an IP address 172.17.0.0/16. Daemon option --bip can be
use
Mar 10 04:56:01 netkiller dockerd[27572]: time="2018-03-
10T04:56:01.243512581-10:00" level=info msg="Loading containers: done."
Mar 10 04:56:01 netkiller dockerd[27572]: time="2018-03-
10T04:56:01.262158514-10:00" level=info msg="Docker daemon"
commit=7390fc6 graphdriver(s)=btrfs version=17.12.1-ce
Mar 10 04:56:01 netkiller dockerd[27572]: time="2018-03-
10T04:56:01.262329696-10:00" level=info msg="Daemon has completed
initialization"
Mar 10 04:56:01 netkiller dockerd[27572]: time="2018-03-
10T04:56:01.280874987-10:00" level=info msg="API listen on
/var/run/docker.sock"
Mar 10 04:56:01 netkiller systemd[1]: Started Docker Application
Container Engine.
```

2.2. 安装 Hyperledger Fabric v1.1.0 Docker 镜像

```
sudo docker pull hyperledger/fabric-ca:x86_64-1.1.0
sudo docker pull hyperledger/fabric-peer:x86_64-1.1.0
sudo docker pull hyperledger/fabric-orderer:x86_64-1.1.0
sudo docker pull hyperledger/fabric-couchdb:x86_64-1.1.0
sudo docker pull hyperledger/fabric-tools:x86_64-1.1.0

sudo docker tag hyperledger/fabric-ca:x86_64-1.1.0 hyperledger/fabric-ca
sudo docker tag hyperledger/fabric-peer:x86_64-1.1.0 hyperledger/fabric-
peer
sudo docker tag hyperledger/fabric-orderer:x86_64-1.1.0
hyperledger/fabric-orderer
sudo docker tag hyperledger/fabric-couchdb:x86_64-1.1.0
hyperledger/fabric-couchdb
```

```
sudo docker tag hyperledger/fabric-tools:x86_64-1.1.0
hyperledger/fabric-tools
```

```
neo@netkiller ~ % sudo docker images
REPOSITORY                    TAG                 IMAGE ID
CREATED             SIZE
hyperledger/fabric-couchdb    latest              380446aa57b6        2
weeks ago         1.5GB
hyperledger/fabric-couchdb    x86_64-1.1.0        380446aa57b6        2
weeks ago         1.5GB
hyperledger/fabric-tools      latest              322eaa2b8786        3
weeks ago         1.33GB
hyperledger/fabric-tools      x86_64-1.1.0        322eaa2b8786        3
weeks ago         1.33GB
hyperledger/fabric-orderer    latest              659d92c1be85        3
weeks ago         151MB
hyperledger/fabric-orderer    x86_64-1.1.0        659d92c1be85        3
weeks ago         151MB
hyperledger/fabric-peer       latest              28c7c07db540        3
weeks ago         154MB
hyperledger/fabric-peer       x86_64-1.1.0        28c7c07db540        3
weeks ago         154MB
hyperledger/fabric-ca         latest              fe3c9b6542cf        3
weeks ago         238MB
hyperledger/fabric-ca         x86_64-1.1.0        fe3c9b6542cf        3
weeks ago         238MB
```

2.3. docker-compose

```
neo@netkiller ~ % sudo apt install python3-pip
neo@netkiller ~ % pip3 install docker-compose
```

或者

```
neo@netkiller ~ % sudo curl -sL
https://github.com/docker/compose/releases/download/1.19.0/docker-
compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose
neo@netkiller ~ % sudo chmod +x /usr/local/bin/docker-compose
neo@netkiller ~ % docker-compose --version
```

```
docker-compose version 1.19.0, build 9e633ef
```

3. Netkiller OSCM 一键安装

3.1. 安装Docker

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualization/docke
r/docker.centos7.ce.sh   | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualization/docke
r/registry-mirror.sh     | bash
```

安装 docker-compose

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualization/docke
r/docker-compose/docker-compose.sh | bash
```

3.2. 清理 Docker 容器和镜像

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/hyperledg
er/fabric/uninstall.sh | bash
```

3.3. Hyperledger Fabric 1.0.6

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/hyperledg
er/fabric/1.0.6/all-in-one.sh | bash
```

3.4. Hyperledger Fabric 1.1.0

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/hyperledg
er/fabric/1.1.0/all-in-one.sh | bash
```

3.5. Hyperledger Fabric 1.2.0

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/hyperledg
er/fabric/uninstall.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/hyperledg
er/fabric/1.2.0/all-in-one.sh | bash
```

4. CentOS 8.0 安装 Fabric 2.0.0

4.1. CentOS 8 初始化

```
[root@localhost ~]# dnf update -y
```

禁用防火墙

```
[root@localhost ~]# systemctl disable firewalld
Removed /etc/systemd/system/multi-
user.target.wants/firewalld.service.
Removed /etc/systemd/system/dbus-
org.fedoraproject.FirewallD1.service.
```

4.2. 安装依赖命令和语言

```
if [ ! -f /usr/bin/bunzip2 ];then
    dnf install -y bzip2
fi

if [ ! -f /usr/bin/git ];then
    dnf install -y git
fi

if [ ! -f /usr/bin/go ];then
    dnf install -y golang
fi
```

```
curl -L --retry 5 --retry-delay 3
https://github.com/hyperledger/fabric/releases/download/v1.4.6/
hyperledger-fabric-linux-amd64-1.4.6.tar.gz | tar xz
curl -L --retry 5 --retry-delay 3
https://github.com/hyperledger/fabric-
ca/releases/download/v1.4.6/hyperledger-fabric-ca-linux-amd64-
1.4.6.tar.gz | tar xz

mkdir -p /srv/hyperledger/fabric
mv bin config /srv/hyperledger/fabric
PATH=$PATH:/srv/hyperledger/fabric/bin
```

4.3. 安装 Docker

```
dnf config-manager --add-
repo=https://download.docker.com/linux/centos/docker-ce.repo

dnf install -y docker-ce

systemctl enable docker
systemctl start docker
```

安装 docker-compose

```
sudo curl -L
"https://github.com/docker/compose/releases/download/1.25.4/doc
ker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-
compose
chmod +x /usr/local/bin/docker-compose
```

4.4. 安装 Fabric 2.0.0

```
IMAGES=(ca peer orderer couchdb ccenv kafka zookeeper tools
javaenv)
FABRIC_TAG=latest
for IMAGE in ${IMAGES}; do
    echo "==> FABRIC IMAGE: $IMAGE"
    echo
    docker pull hyperledger/fabric-$IMAGE:$FABRIC_TAG
    echo
done
```

按照完成检查镜像

```
[root@localhost src]# docker images
REPOSITORY                     TAG                 IMAGE ID
CREATED             SIZE
hyperledger/fabric-zookeeper   latest              21e55e9a2862
12 days ago         276MB
hyperledger/fabric-kafka       latest              28a93b376dbe
12 days ago         270MB
hyperledger/fabric-couchdb     latest              03ac1654afc5
12 days ago         261MB
hyperledger/fabric-ca          latest              3b96a893c1e4
2 weeks ago         150MB
hyperledger/fabric-tools       latest              0f9743ac0662
2 weeks ago         1.49GB
hyperledger/fabric-ccenv       latest              191911f4454f
2 weeks ago         1.36GB
hyperledger/fabric-orderer     latest              84eaba5388e7
2 weeks ago         120MB
hyperledger/fabric-peer        latest              5a52faa5d8c2
2 weeks ago         128MB
```

5. fabric-samples

安装 fabric-samples

```
git clone https://github.com/hyperledger/fabric-samples.git
cd fabric-samples
```

fabric-samples 包含了很多例子，我们只运行几个常用的例子。

5.1. test-network

进入fabric-samples/test-network目录启动 Hyperledger Fabric 2.0，测试网络是否可用

```
[root@localhost ~]# cd fabric-samples/test-network/
[root@localhost ~]# ./network.sh up
```

如果现实如下，恭喜！你的Hyperledger Fabric 2.0已经安装成功！

```
Creating network "net_test" with the default driver
Creating volume "net_orderer.example.com" with default driver
Creating volume "net_peer0.org1.example.com" with default driver
Creating volume "net_peer0.org2.example.com" with default driver
Creating orderer.example.com    ... done
Creating peer0.org2.example.com ... done
Creating peer0.org1.example.com ... done
CONTAINER ID        IMAGE                                  COMMAND             CREATED
STATUS                  PORTS                                NAMES
8d0c74b9d6af        hyperledger/fabric-orderer:latest   "orderer"           4 seconds
ago       Up Less than a second   0.0.0.0:7050->7050/tcp             orderer.example.com
ea1cf82b5b99        hyperledger/fabric-peer:latest      "peer node start"   4 seconds
ago       Up Less than a second   0.0.0.0:7051->7051/tcp
peer0.org1.example.com
cd8d9b23cb56        hyperledger/fabric-peer:latest      "peer node start"   4 seconds
ago       Up 1 second             7051/tcp, 0.0.0.0:9051->9051/tcp
peer0.org2.example.com
```

5.2. fabcar

```
cd fabric-samples/fabcar
```

启动

```
[root@localhost fabcar]# ./startFabric.sh

# don't rewrite paths for Windows Git Bash users
export MSYS_NO_PATHCONV=1

docker-compose -f docker-compose.yml down
Stopping peer0.org1.example.com ... done
Stopping couchdb ... done
Stopping orderer.example.com ... done
Stopping ca.example.com ... done
Removing peer0.org1.example.com ... done
Removing couchdb ... done
Removing orderer.example.com ... done
Removing ca.example.com ... done
Removing network net_basic

docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com
peer0.org1.example.com couchdb
Creating network "net_basic" with the default driver
Creating couchdb
Creating orderer.example.com
Creating ca.example.com
Creating peer0.org1.example.com

# wait for Hyperledger Fabric to start
# incase of errors when running later commands, issue export FABRIC_START_TIMEOUT=
<larger number>
export FABRIC_START_TIMEOUT=10
#echo ${FABRIC_START_TIMEOUT}
sleep ${FABRIC_START_TIMEOUT}

# Create the channel
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e
"CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp"
peer0.org1.example.com peer channel create -o orderer.example.com:7050 -c mychannel -f
/etc/hyperledger/configtx/channel.tx
2018-02-06 04:27:38.822 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-06 04:27:38.822 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining
default signing identity
2018-02-06 04:27:38.825 UTC [channelCmd] InitCmdFactory -> INFO 003 Endorser and orderer
connections initialized
2018-02-06 04:27:38.825 UTC [msp] GetLocalMSP -> DEBU 004 Returning existing local MSP
2018-02-06 04:27:38.825 UTC [msp] GetDefaultSigningIdentity -> DEBU 005 Obtaining
default signing identity
2018-02-06 04:27:38.826 UTC [msp] GetLocalMSP -> DEBU 006 Returning existing local MSP
2018-02-06 04:27:38.826 UTC [msp] GetDefaultSigningIdentity -> DEBU 007 Obtaining
default signing identity
2018-02-06 04:27:38.826 UTC [msp/identity] Sign -> DEBU 008 Sign: plaintext:
0A88060A074F7267314D535012FC052D...53616D706C65436F6E736F727469756D
2018-02-06 04:27:38.826 UTC [msp/identity] Sign -> DEBU 009 Sign: digest:
D2F2DB3135CE892465FB7B6C2C89A6C566EC6E0081034AF00AF62014ED098E10
2018-02-06 04:27:38.827 UTC [msp] GetLocalMSP -> DEBU 00a Returning existing local MSP
2018-02-06 04:27:38.827 UTC [msp] GetDefaultSigningIdentity -> DEBU 00b Obtaining
default signing identity
2018-02-06 04:27:38.827 UTC [msp] GetLocalMSP -> DEBU 00c Returning existing local MSP
2018-02-06 04:27:38.827 UTC [msp] GetDefaultSigningIdentity -> DEBU 00d Obtaining
default signing identity
2018-02-06 04:27:38.827 UTC [msp/identity] Sign -> DEBU 00e Sign: plaintext:
0ABF060A1508021A0608BADDE4D30522...4447D0A2A449E39255165BEE40760F3E
2018-02-06 04:27:38.827 UTC [msp/identity] Sign -> DEBU 00f Sign: digest:
BE32EAB1C6CE6A2C7F79FED4D1C646702EE07A3D64014B5EE2B77F8B35E9CF5F
2018-02-06 04:27:38.935 UTC [msp] GetLocalMSP -> DEBU 010 Returning existing local MSP
2018-02-06 04:27:38.935 UTC [msp] GetDefaultSigningIdentity -> DEBU 011 Obtaining
default signing identity
```

```
2018-02-06 04:27:38.935 UTC [msp] GetLocalMSP -> DEBU 012 Returning existing local MSP
2018-02-06 04:27:38.935 UTC [msp] GetDefaultSigningIdentity -> DEBU 013 Obtaining
default signing identity
2018-02-06 04:27:38.935 UTC [msp/identity] Sign -> DEBU 014 Sign: plaintext:
0ABF060A1508021A0608BADDE4D30522...96444BF5E3AA12080A021A0012021A00
2018-02-06 04:27:38.936 UTC [msp/identity] Sign -> DEBU 015 Sign: digest:
11D42E8978C507DC0C33304B81E089DB5DAB72967A153BD52932C5A4054C3D4B
2018-02-06 04:27:38.937 UTC [channelCmd] readBlock -> DEBU 016 Got status: &{NOT_FOUND}
2018-02-06 04:27:38.937 UTC [msp] GetLocalMSP -> DEBU 017 Returning existing local MSP
2018-02-06 04:27:38.937 UTC [msp] GetDefaultSigningIdentity -> DEBU 018 Obtaining
default signing identity
2018-02-06 04:27:38.939 UTC [channelCmd] InitCmdFactory -> INFO 019 Endorser and orderer
connections initialized
2018-02-06 04:27:39.139 UTC [msp] GetLocalMSP -> DEBU 01a Returning existing local MSP
2018-02-06 04:27:39.139 UTC [msp] GetDefaultSigningIdentity -> DEBU 01b Obtaining
default signing identity
2018-02-06 04:27:39.140 UTC [msp] GetLocalMSP -> DEBU 01c Returning existing local MSP
2018-02-06 04:27:39.140 UTC [msp] GetDefaultSigningIdentity -> DEBU 01d Obtaining
default signing identity
2018-02-06 04:27:39.140 UTC [msp/identity] Sign -> DEBU 01e Sign: plaintext:
0ABF060A1508021A0608BBDDE4D30522...75CF3E958E8612080A021A0012021A00
2018-02-06 04:27:39.140 UTC [msp/identity] Sign -> DEBU 01f Sign: digest:
26B8D88C16CF0DC0573F9BB9A69EC1BF1FEA8C12D052F832B3DB7B2A10D2CA6D
2018-02-06 04:27:39.145 UTC [channelCmd] readBlock -> DEBU 020 Received block: 0
2018-02-06 04:27:39.146 UTC [main] main -> INFO 021 Exiting.....
# Join peer0.org1.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e
"CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp"
peer0.org1.example.com peer channel join -b mychannel.block
2018-02-06 04:27:39.465 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-06 04:27:39.465 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining
default signing identity
2018-02-06 04:27:39.467 UTC [channelCmd] InitCmdFactory -> INFO 003 Endorser and orderer
connections initialized
2018-02-06 04:27:39.468 UTC [msp/identity] Sign -> DEBU 004 Sign: plaintext:
0A86070A5C08011A0C08BBDDE4D30510...2A0C0B6B1D4B1A080A000A000A000A00
2018-02-06 04:27:39.468 UTC [msp/identity] Sign -> DEBU 005 Sign: digest:
455E4221B3AE6DD1238BF5C8893970131846C096F1465D4CA89385AF3C7C0B2F
2018-02-06 04:27:39.991 UTC [channelCmd] executeJoin -> INFO 006 Peer joined the
channel!
2018-02-06 04:27:39.991 UTC [main] main -> INFO 007 Exiting.....
Creating cli
2018-02-06 04:27:46.123 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-06 04:27:46.123 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining
default signing identity
2018-02-06 04:27:46.123 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using
default escc
2018-02-06 04:27:46.123 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using
default vscc
2018-02-06 04:27:47.513 UTC [golang-platform] getCodeFromFS -> DEBU 005 getCodeFromFS
github.com/fabcar
2018-02-06 04:27:48.448 UTC [golang-platform] func1 -> DEBU 006 Discarding GOROOT
package bytes
2018-02-06 04:27:48.449 UTC [golang-platform] func1 -> DEBU 007 Discarding GOROOT
package encoding/json
2018-02-06 04:27:48.449 UTC [golang-platform] func1 -> DEBU 008 Discarding GOROOT
package fmt
2018-02-06 04:27:48.449 UTC [golang-platform] func1 -> DEBU 009 Discarding provided
package github.com/hyperledger/fabric/core/chaincode/shim
2018-02-06 04:27:48.449 UTC [golang-platform] func1 -> DEBU 00a Discarding provided
package github.com/hyperledger/fabric/protos/peer
2018-02-06 04:27:48.449 UTC [golang-platform] func1 -> DEBU 00b Discarding GOROOT
package strconv
2018-02-06 04:27:48.449 UTC [golang-platform] GetDeploymentPayload -> DEBU 00c done
2018-02-06 04:27:48.458 UTC [msp/identity] Sign -> DEBU 00d Sign: plaintext:
0A86070A5C08031A0C08C4DDE4D30510...939FFF060000FFFF9C08DC0700200000
```

```
2018-02-06 04:27:48.459 UTC [msp/identity] Sign -> DEBU 00e Sign: digest:
881EE9535C8EC4371B72CE38BC52BD950864D1CA50F1F64B707EC7492B38AC75
2018-02-06 04:27:48.512 UTC [chaincodeCmd] install -> DEBU 00f Installed remotely
response:<status:200 payload:"OK" >
2018-02-06 04:27:48.512 UTC [main] main -> INFO 010 Exiting.....
2018-02-06 04:27:48.845 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-06 04:27:48.845 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining
default signing identity
2018-02-06 04:27:48.847 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using
default escc
2018-02-06 04:27:48.848 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using
default vscc
2018-02-06 04:27:48.849 UTC [msp/identity] Sign -> DEBU 005 Sign: plaintext:
0A91070A6708031A0C08C4DDE4D30510...324D53500A04657363630A0476736363
2018-02-06 04:27:48.849 UTC [msp/identity] Sign -> DEBU 006 Sign: digest:
7279142B6DCADB92F178A42EB4812C9103108F2AEE8F40F8AD0125D3E13AEC25
2018-02-06 04:28:51.790 UTC [msp/identity] Sign -> DEBU 007 Sign: plaintext:
0A91070A6708031A0C08C4DDE4D30510...33975E7D10605FC261528FBDEEC0A9A8
2018-02-06 04:28:51.790 UTC [msp/identity] Sign -> DEBU 008 Sign: digest:
6A3D493ACD5E447FAD3D91C518CD60C45D0DB8E20D5DCFB70B45DA7572C666C0
2018-02-06 04:28:51.885 UTC [main] main -> INFO 009 Exiting.....
2018-02-06 04:29:02.974 UTC [msp] GetLocalMSP -> DEBU 001 Returning existing local MSP
2018-02-06 04:29:02.974 UTC [msp] GetDefaultSigningIdentity -> DEBU 002 Obtaining
default signing identity
2018-02-06 04:29:02.976 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using
default escc
2018-02-06 04:29:02.976 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 004 Using
default vscc
2018-02-06 04:29:02.977 UTC [msp/identity] Sign -> DEBU 005 Sign: plaintext:
0A93070A6908031A0C088EDEE4D30510...1A0E0A0A696E69744C65646765720A00
2018-02-06 04:29:02.977 UTC [msp/identity] Sign -> DEBU 006 Sign: digest:
F5979DA908E3058B28762469FEECB498F506FB2A7D62F01ED56C5CCE3354D5CE
2018-02-06 04:29:03.103 UTC [msp/identity] Sign -> DEBU 007 Sign: plaintext:
0A93070A6908031A0C088EDEE4D30510...EEB1365266BC4EF15DDAB42DD4202559
2018-02-06 04:29:03.103 UTC [msp/identity] Sign -> DEBU 008 Sign: digest:
027F185369407246B34B7247E6B83E08078761FFA13C43F508CD7257BBF5BC7B
2018-02-06 04:29:03.113 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> DEBU 009 ESCC
invoke result: version:1 response:<status:200 message:"OK" > payload:"\n
e\350\256\026\2161o\374\224\311\210g\322\001\234\332Vt\026+\204\034\013\310;\371\234\353
\030`:\360\022\267\006\n\240\006\022\205\006\n\006fabcar\022\372\005\032J\n\004CAR0\032B
{\"make\":\"Toyota\",\"model\":\"Prius\",\"colour\":\"blue\",\"owner\":\"Tomoko\"}\032G\
n\004CAR1\032?
{\"make\":\"Ford\",\"model\":\"Mustang\",\"colour\":\"red\",\"owner\":\"Brad\"}\032N\n\0
04CAR2\032F{\"make\":\"Hyundai\",\"model\":\"Tucson\",\"colour\":\"green\",\"owner\":\"J
in
Soo\"}\032N\n\004CAR3\032F{\"make\":\"Volkswagen\",\"model\":\"Passat\",\"colour\":\"yel
low\",\"owner\":\"Max\"}\032G\n\004CAR4\032?
{\"make\":\"Tesla\",\"model\":\"S\",\"colour\":\"black\",\"owner\":\"Adriana\"}\032K\n\0
04CAR5\032C{\"make\":\"Peugeot\",\"model\":\"205\",\"colour\":\"purple\",\"owner\":\"Mic
hel\"}\032H\n\004CAR6\032@{\"make\":\"Chery\",\"model\":\"S22L\",\"colour\":\"white\",\"
owner\":\"Aarav\"}\032H\n\004CAR7\032@{\"make\":\"Fiat\",\"model\":\"Punto\",\"colour\":
\"violet\",\"owner\":\"Pari\"}\032J\n\004CAR8\032B{\"make\":\"Tata\",\"model\":\"Nano\",
\"colour\":\"indigo\",\"owner\":\"Valeria\"}\032M\n\004CAR9\032E{\"make\":\"Holden\",\"m
odel\":\"Barina\",\"colour\":\"brown\",\"owner\":\"Shotaro\"}\022\026\n\004lscc\022\016\
n\014\n\006fabcar\022\002\010\001\032\003\010\310\001\"\r\022\006fabcar\032\0031.0"
endorsement:<endorser:"\n\007Org1MSP\022\200\006-----BEGIN -----
\nMIICGjCCAcCgAwIBAgIRAPlwF/rUZUP9mqN4wSml4iswCgYIKoZIzj0EAwIwczEL\nMAkGA1UEBhMCVVMxEzAR
BgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiBG\ncmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS
5jb20xHDAaBgNVBAMTE2Nh\nLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcwODMxMDkxNDMyWhcNMjcwODI5MDkxNDMy
\nWjBbMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMN\nU2FuIEZyYW5jaXNjbzEf
MB0GA1UEAxMWcGVlcjAub3JnMS5leGFtcGxlLmNvbTBZ\nMBMGByqGSM49AgEGCCqGSM49AwEHA0IABHihxW6ks3
B2+5XdbAVq3CBgxRRRZ22x\nzzpqnD86nKkz7fBElBuhlXl2K6rTxyY2OBOB0ts8keqZ93xueRGymrajTTBLMA4G
\nA1UdDwEB/wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIwQkMCKAIEI5qg3Ndtru\nuLoM2nAYUdFFBNMarRst
3dusalc2Xkl8MAoGCCqGSM49BAMCA0gAMEUCIQD4j0Rn\ne1rrd0FSCzsR6u+IuuPK5dI/kR/bh7+VLf0TNgIgCf
UtkJvfvzVEwZLFoFyjoHtr\ntvwzNUS1U0hEqIaDeo4=\n-----END -----\n"
signature:"0E\002!\000\271u\265\3621\223\255|\353h\003\364g\363\3474\310>\032\351\225\32
```

```
6\306z\356\324\013\366\235\334J\237\002
^>\200AhmDWh\261\301\204Ye\346\345\356\2616Rf\274N\361]\332\264-\324 %Y" >
2018-02-06 04:29:03.113 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 00a Chaincode
invoke successful. result: status:200
2018-02-06 04:29:03.113 UTC [main] main -> INFO 00b Exiting.....

Total setup execution time : 136 secs ...


Start by installing required packages run 'npm install'
Then run 'node enrollAdmin.js', then 'node registerUser'

The 'node invoke.js' will fail until it has been updated with valid arguments
The 'node query.js' may be run at anytime once the user has been registered
```

至此 hyperledger 开发环境已经启动完毕

```
node enrollAdmin.js
node registerUser.js
node invoke.js
node query.js
```

```
[root@localhost fabcar]# node enrollAdmin.js
 Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded admin from persistence
Assigned the admin user to the fabric client ::
{"name":"admin","mspid":"Org1MSP","roles":null,"affiliation":"","enrollmentSecret":"","e
nrollment":
{"signingIdentity":"9995fba0ac327e43983b07d09a50423de8cb510176484566abfce0cb86e5f594","i
dentity":{"certificate":"-----BEGIN CERTIFICATE-----
\nMIIB8TCCAZegAwIBAgIUNH9h0PUYxF1vpUqzEXKRfVc2a7YwCgYIKoZIzj0EAwIw\nczELMAkGA1UEBhMCVVMx
EzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh\nbiBGcmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbX
BsZS5jb20xHDAaBgNVBAMT\nE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTgwMjA2MDQ0NTAwWhcNMTkwMjA2MDQ0
\nNTAwWjAQMQ4wDAYDVQQDEwVhZG1pbjBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IA\nBEhyVSI9Dl7S2fIYQdiJ
AB2zeXR9aHIQSVSG7auK3y3yVvABOQPA/Kyn2iMAl4rr\nky/0FYY5B+lxYLLSype/2zKjbDBqMA4GA1UdDwEB/w
QEAwIHgDAMBgNVHRMBAf8E\nAjAAMB0GA1UdDgQWBBRotEXOHR/h1ZLwHV/eAUT80bIQFTArBgNVHSMEJDAigCBC
\nOaoNzXba7ri6DNpwGFHRRQTTGq0bLd3brGpXNl5JfDAKBggqhkjOPQQDAgNIADBF\nAiEAnBH8WJvb24o2eC5V
mMvtQMoB8NDBTpdq5RNyVJx97HcCIBhWMM6R7crkL8M7\n3wmZ1lNNkAJmpRoes7cNxO/ak5Xy\n-----END
CERTIFICATE-----\n"}}}

[root@localhost fabcar]# node registerUser.js
 Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded admin from persistence
Successfully registered user1 - secret:ZjzLwIaVjEAV
Successfully enrolled member user "user1"
User1 was successfully registered and enrolled and is ready to intreact with the fabric
network

[root@localhost fabcar]# node query.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Query has completed, checking results
Response is  [{"Key":"CAR0", "Record":
{"colour":"blue","make":"Toyota","model":"Prius","owner":"Tomoko"}},{"Key":"CAR1",
"Record":{"colour":"red","make":"Ford","model":"Mustang","owner":"Brad"}},{"Key":"CAR2",
"Record":{"colour":"green","make":"Hyundai","model":"Tucson","owner":"Jin Soo"}},
{"Key":"CAR3", "Record":
```

```
{"colour":"yellow","make":"Volkswagen","model":"Passat","owner":"Max"}},{"Key":"CAR4",
"Record":{"colour":"black","make":"Tesla","model":"S","owner":"Adriana"}},{"Key":"CAR5",
"Record":{"colour":"purple","make":"Peugeot","model":"205","owner":"Michel"}},
{"Key":"CAR6", "Record":
{"colour":"white","make":"Chery","model":"S22L","owner":"Aarav"}},{"Key":"CAR7",
"Record":{"colour":"violet","make":"Fiat","model":"Punto","owner":"Pari"}},
{"Key":"CAR8", "Record":
{"colour":"indigo","make":"Tata","model":"Nano","owner":"Valeria"}},{"Key":"CAR9",
"Record":{"colour":"brown","make":"Holden","model":"Barina","owner":"Shotaro"}}]
```

invoke.js 脚本会出现下面错误

```
[root@localhost fabcar]# node invoke.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Assigning transaction_id:
424c946e96d43005e4b815f56fa43bd58d1370404f6713b3ce267928d4499b78
error: [client-utils.js]: sendPeersProposal - Promise is rejected: Error: 2 UNKNOWN:
chaincode error (status: 500, message: Invalid Smart Contract function name.)
    at new createStatusError (/root/fabric-
samples/fabcar/node_modules/grpc/src/client.js:65:15)
    at /root/fabric-samples/fabcar/node_modules/grpc/src/client.js:568:15
Transaction proposal was bad
Failed to send Proposal or receive valid response. Response null or status is not 200.
exiting...
Failed to invoke successfully :: Error: Failed to send Proposal or receive valid
response. Response null or status is not 200. exiting...
```

这里出错，是因为没有提供相关的调用方法和参数，暂时不去纠结，继续做实验。

5.2.1. 智能合约

```
[root@localhost fabric-samples]# cat chaincode/fabcar/fabcar.go
/*
 * Licensed to the Apache Software Foundation (ASF) under one
 * or more contributor license agreements.  See the NOTICE file
 * distributed with this work for additional information
 * regarding copyright ownership.  The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License.  You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing,
 * software distributed under the License is distributed on an
 * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 * KIND, either express or implied.  See the License for the
 * specific language governing permissions and limitations
 * under the License.
 */

/*
 * The sample smart contract for documentation topic:
 * Writing Your First Blockchain Application
```

```
 */

package main

/* Imports
 * 4 utility libraries for formatting, handling bytes, reading and writing JSON, and
string manipulation
 * 2 specific Hyperledger Fabric specific libraries for Smart Contracts
 */
import (
        "bytes"
        "encoding/json"
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        sc "github.com/hyperledger/fabric/protos/peer"
)

// Define the Smart Contract structure
type SmartContract struct {
}

// Define the car structure, with 4 properties.  Structure tags are used by
encoding/json library
type Car struct {
        Make   string `json:"make"`
        Model  string `json:"model"`
        Colour string `json:"colour"`
        Owner  string `json:"owner"`
}

/*
 * The Init method is called when the Smart Contract "fabcar" is instantiated by the
blockchain network
 * Best practice is to have any Ledger initialization in separate function -- see
initLedger()
 */
func (s *SmartContract) Init(APIstub shim.ChaincodeStubInterface) sc.Response {
        return shim.Success(nil)
}

/*
 * The Invoke method is called as a result of an application request to run the Smart
Contract "fabcar"
 * The calling application program has also specified the particular smart contract
function to be called, with arguments
 */
func (s *SmartContract) Invoke(APIstub shim.ChaincodeStubInterface) sc.Response {

        // Retrieve the requested Smart Contract function and arguments
        function, args := APIstub.GetFunctionAndParameters()
        // Route to the appropriate handler function to interact with the ledger
appropriately
        if function == "queryCar" {
                return s.queryCar(APIstub, args)
        } else if function == "initLedger" {
                return s.initLedger(APIstub)
        } else if function == "createCar" {
                return s.createCar(APIstub, args)
        } else if function == "queryAllCars" {
                return s.queryAllCars(APIstub)
        } else if function == "changeCarOwner" {
                return s.changeCarOwner(APIstub, args)
        }
```

```
        return shim.Error("Invalid Smart Contract function name.")
}

func (s *SmartContract) queryCar(APIstub shim.ChaincodeStubInterface, args []string)
sc.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of arguments. Expecting 1")
        }

        carAsBytes, _ := APIstub.GetState(args[0])
        return shim.Success(carAsBytes)
}

func (s *SmartContract) initLedger(APIstub shim.ChaincodeStubInterface) sc.Response {
        cars := []Car{
                Car{Make: "Toyota", Model: "Prius", Colour: "blue", Owner: "Tomoko"},
                Car{Make: "Ford", Model: "Mustang", Colour: "red", Owner: "Brad"},
                Car{Make: "Hyundai", Model: "Tucson", Colour: "green", Owner: "Jin
Soo"},
                Car{Make: "Volkswagen", Model: "Passat", Colour: "yellow", Owner:
"Max"},
                Car{Make: "Tesla", Model: "S", Colour: "black", Owner: "Adriana"},
                Car{Make: "Peugeot", Model: "205", Colour: "purple", Owner: "Michel"},
                Car{Make: "Chery", Model: "S22L", Colour: "white", Owner: "Aarav"},
                Car{Make: "Fiat", Model: "Punto", Colour: "violet", Owner: "Pari"},
                Car{Make: "Tata", Model: "Nano", Colour: "indigo", Owner: "Valeria"},
                Car{Make: "Holden", Model: "Barina", Colour: "brown", Owner: "Shotaro"},
        }

        i := 0
        for i < len(cars) {
                fmt.Println("i is ", i)
                carAsBytes, _ := json.Marshal(cars[i])
                APIstub.PutState("CAR"+strconv.Itoa(i), carAsBytes)
                fmt.Println("Added", cars[i])
                i = i + 1
        }

        return shim.Success(nil)
}

func (s *SmartContract) createCar(APIstub shim.ChaincodeStubInterface, args []string)
sc.Response {

        if len(args) != 5 {
                return shim.Error("Incorrect number of arguments. Expecting 5")
        }

        var car = Car{Make: args[1], Model: args[2], Colour: args[3], Owner: args[4]}

        carAsBytes, _ := json.Marshal(car)
        APIstub.PutState(args[0], carAsBytes)

        return shim.Success(nil)
}

func (s *SmartContract) queryAllCars(APIstub shim.ChaincodeStubInterface) sc.Response {

        startKey := "CAR0"
        endKey := "CAR999"

        resultsIterator, err := APIstub.GetStateByRange(startKey, endKey)
        if err != nil {
                return shim.Error(err.Error())
        }
```

```
        defer resultsIterator.Close()

        // buffer is a JSON array containing QueryResults
        var buffer bytes.Buffer
        buffer.WriteString("[")

        bArrayMemberAlreadyWritten := false
        for resultsIterator.HasNext() {
                queryResponse, err := resultsIterator.Next()
                if err != nil {
                        return shim.Error(err.Error())
                }
                // Add a comma before array members, suppress it for the first array
member
                if bArrayMemberAlreadyWritten == true {
                        buffer.WriteString(",")
                }
                buffer.WriteString("{\"Key\":")
                buffer.WriteString("\"")
                buffer.WriteString(queryResponse.Key)
                buffer.WriteString("\"")

                buffer.WriteString(", \"Record\":")
                // Record is a JSON object, so we write as-is
                buffer.WriteString(string(queryResponse.Value))
                buffer.WriteString("}")
                bArrayMemberAlreadyWritten = true
        }
        buffer.WriteString("]")

        fmt.Printf("- queryAllCars:\n%s\n", buffer.String())

        return shim.Success(buffer.Bytes())
}

func (s *SmartContract) changeCarOwner(APIstub shim.ChaincodeStubInterface, args
[]string) sc.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of arguments. Expecting 2")
        }

        carAsBytes, _ := APIstub.GetState(args[0])
        car := Car{}

        json.Unmarshal(carAsBytes, &car)
        car.Owner = args[1]

        carAsBytes, _ = json.Marshal(car)
        APIstub.PutState(args[0], carAsBytes)

        return shim.Success(nil)
}

// The main function is only relevant in unit test mode. Only included here for
completeness.
func main() {

        // Create a new Smart Contract
        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract: %s", err)
        }
}
```

5.2.2. 创建记录

```
[root@localhost fabcar]# cp invoke.js createCar.js
[root@localhost fabcar]# vim createCar.js
```

```
var request = {
                //targets: let default to the peer assigned to the client
                chaincodeId: 'fabcar',
                fcn: '',
                args: [''],
                chainId: 'mychannel',
                txId: tx_id
        };
```

改为

```
        var request = {
                //targets: let default to the peer assigned to the client
                chaincodeId: 'fabcar',
                fcn: 'createCar',
                args: ['CAR10', 'Chevy', 'Volt', 'Red', 'Nick'],
                chainId: 'mychannel',
                txId: tx_id
        };
```

运行结果

```
[root@localhost fabcar]# node createCar.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Assigning transaction_id:
907d35d7c5debf952f28135b2c341797acdb7ef1f1389607fb04891a27e8ba19
Transaction proposal was good
Successfully sent Proposal and received ProposalResponse: Status - 200, message - "OK"
info: [EventHub.js]: _connect - options {}
The transaction has been committed on peer localhost:7053
Send transaction promise and event listener promise have completed
Successfully sent transaction to the orderer.
Successfully committed the change to the ledger by the peer
```

5.2.3. 查询单条记录

查找 CAR5 这条记录的数据。

```
[root@localhost fabcar]# node queryOne.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Query has completed, checking results
Response is  {"colour":"purple","make":"Peugeot","model":"205","owner":"Michel"}
[root@localhost fabcar]# cat queryOne.js
'use strict';
/*
* Copyright IBM Corp All Rights Reserved
*
* SPDX-License-Identifier: Apache-2.0
*/
/*
 * Chaincode query
 */

var Fabric_Client = require('fabric-client');
var path = require('path');
var util = require('util');
var os = require('os');

//
var fabric_client = new Fabric_Client();

// setup the fabric network
var channel = fabric_client.newChannel('mychannel');
var peer = fabric_client.newPeer('grpc://localhost:7051');
channel.addPeer(peer);

//
var member_user = null;
var store_path = path.join(__dirname, 'hfc-key-store');
console.log('Store path:'+store_path);
var tx_id = null;

// create the key value store as defined in the fabric-client/config/default.json 'key-
value-store' setting
Fabric_Client.newDefaultKeyValueStore({ path: store_path
}).then((state_store) => {
        // assign the store to the fabric client
        fabric_client.setStateStore(state_store);
        var crypto_suite = Fabric_Client.newCryptoSuite();
        // use the same location for the state store (where the users' certificate are
kept)
        // and the crypto store (where the users' keys are kept)
        var crypto_store = Fabric_Client.newCryptoKeyStore({path: store_path});
        crypto_suite.setCryptoKeyStore(crypto_store);
        fabric_client.setCryptoSuite(crypto_suite);

        // get the enrolled user from persistence, this user will sign all requests
        return fabric_client.getUserContext('user1', true);
}).then((user_from_store) => {
        if (user_from_store && user_from_store.isEnrolled()) {
                console.log('Successfully loaded user1 from persistence');
                member_user = user_from_store;
        } else {
                throw new Error('Failed to get user1.... run registerUser.js');
        }

        // queryCar chaincode function - requires 1 argument, ex: args: ['CAR4'],
        // queryAllCars chaincode function - requires no arguments , ex: args: [''],
        const request = {
                //targets : --- letting this default to the peers assigned to the
channel
```

```
                chaincodeId: 'fabcar',
                fcn: 'queryCar',
                args: ['CAR10']
        };

        // send the query proposal to the peer
        return channel.queryByChaincode(request);
}).then((query_responses) => {
        console.log("Query has completed, checking results");
        // query_responses could have more than one  results if there multiple peers
were used as targets
        if (query_responses && query_responses.length == 1) {
                if (query_responses[0] instanceof Error) {
                        console.error("error from query = ", query_responses[0]);
                } else {
                        console.log("Response is ", query_responses[0].toString());
                }
        } else {
                console.log("No payloads were returned from query");
        }
}).catch((err) => {
        console.error('Failed to query successfully :: ' + err);
});
```

运行结果

```
[root@localhost fabcar]# node queryOne.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Query has completed, checking results
Response is  {"colour":"Red","make":"Chevy","model":"Volt","owner":"Nick"}
```

5.2.4. 修改汽车所有者

```
[root@localhost fabcar]# cp invoke.js changeCarOwner.js
[root@localhost fabcar]# vim changeCarOwner.js
```

```
        var request = {
                //targets: let default to the peer assigned to the client
                chaincodeId: 'fabcar',
                fcn: '',
                args: [''],
                chainId: 'mychannel',
                txId: tx_id
        };
```

修改为

```
        var request = {
                //targets: let default to the peer assigned to the client
                chaincodeId: 'fabcar',
                fcn: 'changeCarOwner',
                args: ['CAR10', 'Neo'],
                chainId: 'mychannel',
                txId: tx_id
        };
```

运行结果

```
[root@localhost fabcar]# node changeCarOwner.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Assigning transaction_id:
b183a7bac6161ff7c7e9974e20bb1a72b2ac5f11ab698a2ae4f63cfdbc33a735
Transaction proposal was good
Successfully sent Proposal and received ProposalResponse: Status - 200, message - "OK"
info: [EventHub.js]: _connect - options {}
The transaction has been committed on peer localhost:7053
Send transaction promise and event listener promise have completed
Successfully sent transaction to the orderer.
Successfully committed the change to the ledger by the peer


[root@localhost fabcar]# node queryOne.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Query has completed, checking results
Response is  {"colour":"Red","make":"Chevy","model":"Volt","owner":"Neo"}
```

现在我们可以看到 所有者已经改为 Neo

5.3. balance-transfer

```
[root@localhost fabric-samples]# cd balance-transfer/
[root@localhost balance-transfer]# ./runApp.sh

Stopping peer0.org1.example.com ... done
Stopping peer0.org2.example.com ... done
Stopping peer1.org2.example.com ... done
Stopping peer1.org1.example.com ... done
Stopping ca_peerOrg1 ... done
Stopping orderer.example.com ... done
Stopping ca_peerOrg2 ... done
Removing peer0.org1.example.com ... done
Removing peer0.org2.example.com ... done
Removing peer1.org2.example.com ... done
Removing peer1.org1.example.com ... done
Removing ca_peerOrg1 ... done
Removing orderer.example.com ... done
Removing ca_peerOrg2 ... done
Removing network artifacts_default
```

```
========== No containers available for deletion ==========


========== No images available for deletion ===========

Creating network "artifacts_default" with the default driver
Creating ca_peerOrg2
Creating orderer.example.com
Creating ca_peerOrg1
Creating peer1.org2.example.com
Creating peer0.org2.example.com
Creating peer0.org1.example.com
Creating peer1.org1.example.com
/root/fabric-samples/balance-transfer


============== node modules installed already =============

[2018-02-06 09:14:54.916] [DEBUG] Helper - [crypto_ecdsa_aes]: constructor, keySize: 256
[2018-02-06 09:14:54.921] [DEBUG] Helper - [crypto_ecdsa_aes]: Hash algorithm: SHA2,
hash output size: 256
[2018-02-06 09:14:55.046] [DEBUG] Helper - [utils.CryptoKeyStore]: CryptoKeyStore,
constructor - start
[2018-02-06 09:14:55.048] [DEBUG] Helper - [utils.CryptoKeyStore]: constructor, no super
class specified, using config: fabric-client/lib/impl/FileKeyValueStore.js
[2018-02-06 09:14:55.056] [DEBUG] Helper - [crypto_ecdsa_aes]: constructor, keySize: 256
[2018-02-06 09:14:55.056] [DEBUG] Helper - [crypto_ecdsa_aes]: Hash algorithm: SHA2,
hash output size: 256
[2018-02-06 09:14:55.057] [DEBUG] Helper - [utils.CryptoKeyStore]: CryptoKeyStore,
constructor - start
[2018-02-06 09:14:55.057] [DEBUG] Helper - [utils.CryptoKeyStore]: constructor, no super
class specified, using config: fabric-client/lib/impl/FileKeyValueStore.js
[2018-02-06 09:14:55.082] [INFO] SampleWebApp - ****************** SERVER STARTED
************************
[2018-02-06 09:14:55.082] [INFO] SampleWebApp - **************  http://localhost:4000
******************
```

安装 jq 工具

```
yum install -y jq
```

API 测试结果

```
[root@localhost balance-transfer]# ./testAPIs.sh
POST request Enroll on Org1  ...

{"success":true,"secret":"uXlJIPvBgLbh","message":"Jim enrolled
Successfully","token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTc5NjMwMDUsInVz
ZXJuYW1lIjoiSmltIiwib3JnTmFtZSI6Im9yZzEiLCJpYXQiOjE1MTc5MjcwMDV9.pBqwAVac32NIA_x5S163h9_
lRfIHMx4shX05D0geO5k"}

ORG1 token is
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTc5NjMwMDUsInVzZXJuYW1lIjoiSmltIiwib3J
nTmFtZSI6Im9yZzEiLCJpYXQiOjE1MTc5MjcwMDV9.pBqwAVac32NIA_x5S163h9_lRfIHMx4shX05D0geO5k
```

```
POST request Enroll on Org2 ...

{"success":true,"secret":"kXcltqsAoiDf","message":"Barry enrolled
Successfully","token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTc5NjMwMDcsInVz
ZXJuYW1lIjoiQmFycnkiLCJvcmdOYW1lIjoib3JnMiIsImlhdCI6MTUxNzkyNzAwN30.ePexFJ2r9WoYNma5iKom
Ju3anEvg3PTgBEqI6K-s6F0"}

ORG2 token is
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTc5NjMwMDcsInVzZXJuYW1lIjoiQmFycnkiLCJ
vcmdOYW1lIjoib3JnMiIsImlhdCI6MTUxNzkyNzAwN30.ePexFJ2r9WoYNma5iKomJu3anEvg3PTgBEqI6K-s6F0


POST request Create channel  ...

{"success":true,"message":"Channel 'mychannel' created Successfully"}

POST request Join channel on Org1

{"success":true,"message":"Successfully joined peers in organization org1 to the channel
'mychannel'"}

POST request Join channel on Org2

{"success":true,"message":"Successfully joined peers in organization org2 to the channel
'mychannel'"}

POST Install chaincode on Org1

Successfully Installed chaincode on organization org1

POST Install chaincode on Org2

Successfully Installed chaincode on organization org2

POST instantiate chaincode on peer1 of Org1

Failed to order the transaction. Error code: undefined

POST invoke chaincode on peers of Org1 and Org2

Transacton ID is Failed to order the transaction. Error code: undefined


GET query chaincode on peer1 of Org1

a now has Error: 2 UNKNOWN: could not find chaincode with name 'mycc' - make sure the
chaincode mycc has been successfully instantiated and try again after the move

GET query Block by blockNumber

Error: 2 UNKNOWN: chaincode error (status: 500, message: Failed to get block number 1,
error Entry not found in index)
    at new createStatusError (/root/fabric-samples/balance-
transfer/node_modules/grpc/src/client.js:65:15)
    at /root/fabric-samples/balance-transfer/node_modules/grpc/src/client.js:568:15

GET query Transaction by TransactionID

Error: 2 UNKNOWN: chaincode error (status: 500, message: Failed to get transaction with
id Failed, error Entry not found in index)
    at new createStatusError (/root/fabric-samples/balance-
transfer/node_modules/grpc/src/client.js:65:15)
    at /root/fabric-samples/balance-transfer/node_modules/grpc/src/client.js:568:15
```

```
GET query ChainInfo

{"height":{"low":1,"high":0,"unsigned":true},"currentBlockHash":{"buffer":
{"type":"Buffer","data":
[8,1,18,32,180,19,234,224,76,239,188,107,147,219,120,185,108,73,201,120,81,0,242,221,32,
3,94,175,16,200,181,113,100,97,129,57]},"offset":4,"markedOffset":-1,"limit":36,"littleE
ndian":true,"noAssert":false},"previousBlockHash":{"buffer":{"type":"Buffer","data":
[]},"offset":0,"markedOffset":-1,"limit":0,"littleEndian":false,"noAssert":false}}

GET query Installed chaincodes

["name: mycc, version: v0, path: github.com/example_cc"]

GET query Instantiated chaincodes

[]

GET query Channels

{"channels":[{"channel_id":"mychannel"}]}

Total execution time : 364 secs ...
```

上面 ERROR 可能是因为 hyperledger v1.1.0 版本的缘故。此时的 fabric-samples 仅仅支持 v1.0.0。

5.4. first-network

```
cd fabric-samples/first-network
```

```
[root@localhost first-network]# ./byfn.sh -m generate
Generating certs and genesis block for with channel 'mychannel' and CLI timeout of '10'
Continue (y/n)? y
proceeding ...
which: no cryptogen in (/root/fabric-samples/first-network/../bin:/root/fabric-
samples/first-
network:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/srv/php/bin::/root/
bin:/usr/local/src/phalcon-devtools)
cryptogen tool not found. exiting
```

```
[root@localhost first-network]# ./byfn.sh -m up
```

6. e2e_cli

运行这个命令请慎重，这里是为了不被先前的实验产生的容易所影响，清理 Docker 虚拟机，

```
[root@localhost ~]# docker rmi -f $(docker images -q)
```

克隆源码，因为我们需要 example 目录中的文件，所以将 https://github.com/hyperledger/fabric.git 源码克隆到本地。

```
[root@localhost ~]# go get github.com/hyperledger/fabric

[root@localhost ~]# git clone --depth=1
https://github.com/hyperledger/fabric.git
[root@localhost ~]# cd fabric
[root@localhost fabric]# make release
[root@localhost ~]# cd ~/fabric/examples/e2e_cli/
[root@localhost e2e_cli]#
```

7. Hyperledger Composer

Build Blockchain applications and business networks your way

https://hyperledger.github.io/composer/

8. 创世区块

创建创世区块需要两个配置文件，一个是 crypto-config.yaml 文件， 另一个是 configtx.yaml 文件

8.1. crypto-config.yaml

crypto-config.yaml 证书配置文件

```
OrdererOrgs:
  - Name: Orderer
    Domain: example.com
    Specs:
      - Hostname: orderer
PeerOrgs:
  - Name: Org1
    Domain: org1.example.com
    Template:
      Count: 1
    Users:
      Count: 1
```

Name：定义名称

Domain与Hostname：组合成为节点的名称，也是生成后的文件夹的名称。

Template: 模板数量

Users：用来指定添加进节点的默认用户数

Count：用来指定每个org下边所拥有的节点数。

如果有多个Peer节点参考下面配置。

```
[root@localhost netkiller]# vim crypto-config.yaml

OrdererOrgs:
  - Name: Orderer
    Domain: example.com
    Specs:
      - Hostname: orderer
PeerOrgs:
  - Name: Org1
    Domain: org1.example.com
    Template:
      Count: 2
    Users:
      Count: 1
  - Name: Org2
    Domain: org2.example.com
    Template:
      Count: 2
    Users:
      Count: 1
```

8.2. configtx.yaml

9. hyperledger/fabric-ca

go 安装方式

```
go get github.com/hyperledger/fabric-ca
```

10. Restful 接口

10.1. 注册

```
curl -i -H "Accept: application/json" -H "Content-Type:
application/json" -X POST -d '
{

    "enrollId": "jim",
    "enrollSecret": "6avZQLwcUe9b"

}' http://localhost:7050/registrar
```

10.2.

```
curl -i -H "Accept: application/json" -H "Content-Type:
application/json" -X POST -d '
{

   "jsonrpc": "2.0",
   "method": "deploy",
   "params": {
    "type": 1,
    "chaincodeID":{
        "name": "mycc"
    },
    "ctorMsg": {
         "function":"init",
         "args":[]
     },
    "secureContext": "jim"
  },
   "id": 1

}' http://localhost:7050/chaincode
```

第 25 章 Hyperledger Fabric 运维

1. 背景

由于区块链是区中心化，与传统运维不同，所以之前你积累的经验，不一定适用于区块链。要想运维好区块链项目，就必须理解去中心化这个概念。

首先谈谈传统运维，总结为三个字“中心化”，当然有人反对并抛出“分布式”感念，传统运维的分布式仍然建立在中心化的基础之上。

我们来看看传统应用模式，决多数应用都可以概括为：

用户 -> WEB -> Application -> Cache -> Database

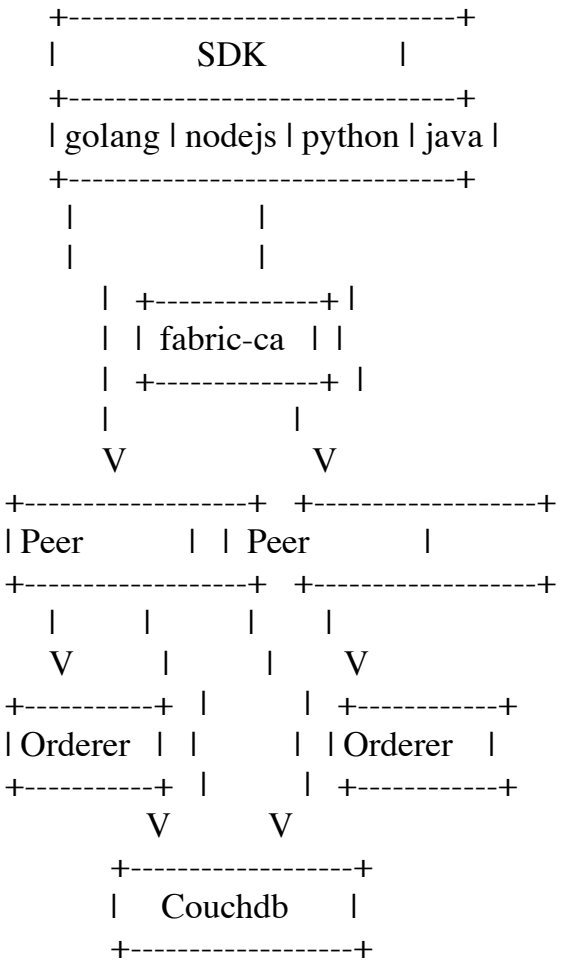
可以在这个体系下面做灵活变化，例如加入所有引擎、分布式文件系统，大数据等等应用，但都离不开这个模式。

区块链完全不同，如果举一个最接近的例子，我想可能与多数据中心远程异地灾备比较接近。

2. 部署拓扑

什么是区块链呢？ 区块链实际上就是数据库，一个只能插入和查询的数据库，数据不能被修改和删除，并且这个数据库没有DBA管理员角色。这么一说你应该明白了把，实际上运维区块链就是在维护一个分布式数据库。

网上的绝大多数安装例子中，均采用 docker 部署方案，但无一例外的是，全部安装在一个物理机上。如果是生产环境，我们必须分开不是，首先要做的工作是化整为零，拆解应用，搞明白每个容器的功能和作用。然后我们将应用拆分，独立部署到物理节点上去。



接下来我们要做的工作是将上面拓扑图种的技术点分分击破。

由于 Hyperledger Fabric 是建立在 Docker 基础之上的。所以不建议你去除 Docker 转而使用传统的本地编译安装方式。我们仍然保持使用 Docker 在每个物理节点上，省去软件的编译和安装环节。

2.1. 依赖关系

需要注意的是于其他传统系统一样，Hyperledger Fabric 的启动也是有顺序的，这是因为他们之间存在着依赖关系。

2.2. 准备物理机

CentOS (Minimal ISO)

物理机

- ca 节点，域名：ca.example.com，IP地址：172.16.0.20，端口：7054
- orderer 节点，域名 orderer.example.com，IP地址：172.16.0.21，端口：7050
- peer 节点，域名：peer.example.com，IP地址：172.16.0.22，端口：7051、7053
- couchdb 节点，域名 couchdb.example.com，IP地址：172.16.0.25，端口：5984
- tools 节点，域名：tools.example.com，IP地址：172.16.0.20 与 CA 共用一台机器(这里为了节省资源)

在所有节点上运行下面脚本

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualization/docke
r/docker.centos7.ce.sh   | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualization/docke
r/docker-compose/docker-compose-1.19.0.sh | bash
```

3. cli 管理节点安装

Tools 在生成创世区块的时候我们就曾经使用，你可以沿用之前的 tools 节点，或者创建一个 cli 节点，这个节点主要是用于管理区块链集群，例如合约部署，调试等等。

3.1. 安装 Docker 镜像

```
docker pull hyperledger/fabric-tools:x86_64-1.1.0
docker tag hyperledger/fabric-tools:x86_64-1.1.0 hyperledger/fabric-tools
```

3.2. docker-compose-cli.yaml

```
version: '3'

networks:
  basic:

services:

  cli:
    container_name: cli
    image: hyperledger/fabric-tools
    tty: true
    environment:
      - GOPATH=/opt/gopath
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      - CORE_LOGGING_LEVEL=DEBUG
      - CORE_PEER_ID=cli
      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
      - CORE_PEER_LOCALMSPID=Org1MSP
      -
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOr
ganizations/org1.example.com/users/Admin@org1.example.com/msp
      - CORE_CHAINCODE_KEEPALIVE=10
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
    command: /bin/bash
    volumes:
        - /var/run/:/host/var/run/
        - ./chaincode/:/opt/gopath/src/github.com/
        - ./crypto-config:/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
        - ~/netkiller:/root/netkiller
    networks:
        - basic
    #depends_on:
    #  - orderer.example.com
    #  - peer0.org1.example.com
    #  - couchdb
    extra_hosts:
                - "ca.example.com:172.16.0.20"
                - "orderer.example.com:172.16.0.21"
                - "peer0.org1.example.com:172.16.0.22"
                - "couchdb.example.com:172.16.0.25"
```

3.3. 启动 cli 节点

```
[root@localhost netkiller]# docker-compose -f docker-compose-cli.yaml up -d
```

后面合约的部署将在 cli 节点上进行

3.4. 生成证书和创世区块

这里我们需要几个命令（configtxgen configtxlator cryptogen），官方的安装方式：

```
curl -sSL https://goo.gl/byy2Qj | bash -s 1.1.0
```

无论如何我都安装不成功，可能是（https://goo.gl/byy2Qj）被天朝给墙了。不过我发现 fabric-tools 里面有这个工具。

提示

经过翻墙发现 https://goo.gl/byy2Qj 地址是 301 到下面地址：

https://raw.githubusercontent.com/hyperledger/fabric/v1.1.0/scripts/bootstrap.sh

```
[root@localhost ~]# mkdir netkiller
[root@localhost ~]# cd netkiller/
[root@localhost netkiller]# mkdir -p {chaincode,crypto-config,config,artifacts}
```

3.4.1. 创建配置文件

3.4.1.1. crypto-config.yaml

创建证书

```
OrdererOrgs:
  - Name: Orderer
    Domain: example.com
    Specs:
      - Hostname: orderer
PeerOrgs:
  - Name: Org1
    Domain: org1.example.com
    Template:
      Count: 1
    Users:
      Count: 1
```

3.4.1.2. configtx.yaml

```
---
Profiles:

    OneOrgOrdererGenesis:
        Orderer:
            <<: *OrdererDefaults
            Organizations:
                - *OrdererOrg
        Consortiums:
            SampleConsortium:
                Organizations:
                    - *Org1
    OneOrgChannel:
        Consortium: SampleConsortium
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Org1

Organizations:

    - &OrdererOrg
        Name: OrdererOrg

        ID: OrdererMSP

        MSPDir: crypto-config/ordererOrganizations/example.com/msp

    - &Org1
        Name: Org1MSP

        ID: Org1MSP

        MSPDir: crypto-config/peerOrganizations/org1.example.com/msp

        AnchorPeers:
            - Host: peer0.org1.example.com
              Port: 7051

Orderer: &OrdererDefaults

    OrdererType: solo

    Addresses:
        - orderer.example.com:7050

    BatchTimeout: 2s

    BatchSize:

        MaxMessageCount: 10

        AbsoluteMaxBytes: 99 MB

        PreferredMaxBytes: 512 KB

    Kafka:
        Brokers:
```

```
            - 127.0.0.1:9092

    Organizations:

Application: &ApplicationDefaults

    Organizations:
```

3.4.2. 生成证书

命令

```
cryptogen generate --config=./crypto-config.yaml
```

演示

```
root@8f467a88de99:~/netkiller# cryptogen generate --config=./crypto-config.yaml
org1.example.com

root@8f467a88de99:~/netkiller# ls -1 crypto-config
ordererOrganizations
peerOrganizations
```

3.4.3. 生成创世区块

```
root@8f467a88de99:~/netkiller# export FABRIC_CFG_PATH=$PWD
root@8f467a88de99:~/netkiller# configtxgen -profile OneOrgOrdererGenesis -outputBlock
./config/genesis.block
2018-02-08 08:35:30.121 UTC [common/configtx/tool] main -> INFO 001 Loading
configuration
2018-02-08 08:35:30.236 UTC [common/configtx/tool] doOutputBlock -> INFO 002 Generating
genesis block
2018-02-08 08:35:30.238 UTC [common/configtx/tool] doOutputBlock -> INFO 003 Writing
genesis block
```

3.4.4. 生成通道配置文件

命令

```
CHANNEL_NAME=mychannel
configtxgen -profile OneOrgChannel -outputCreateChannelTx ./config/channel.tx -channelID
$CHANNEL_NAME
```

操作演示

```
root@8f467a88de99:~/netkiller# CHANNEL_NAME=mychannel
root@8f467a88de99:~/netkiller# configtxgen -profile OneOrgChannel -outputCreateChannelTx
./config/channel.tx -channelID $CHANNEL_NAME
2018-02-08 08:41:08.010 UTC [common/configtx/tool] main -> INFO 001 Loading
configuration
2018-02-08 08:41:08.020 UTC [common/configtx/tool] doOutputChannelCreateTx -> INFO 002
Generating new channel configtx
2018-02-08 08:41:08.020 UTC [common/configtx/tool] doOutputChannelCreateTx -> INFO 003
Writing new channel tx
```

3.4.5. generate anchor peer transaction

命令

```
CHANNEL_NAME=mychannel
configtxgen -profile OneOrgChannel -outputAnchorPeersUpdate ./config/Org1MSPanchors.tx -
channelID $CHANNEL_NAME -asOrg Org1MSP
```

操作演示

```
root@8f467a88de99:~/netkiller# CHANNEL_NAME=mychannel
root@8f467a88de99:~/netkiller# configtxgen -profile OneOrgChannel -
outputAnchorPeersUpdate ./config/Org1MSPanchors.tx -channelID $CHANNEL_NAME -asOrg
Org1MSP
2018-02-08 08:46:19.162 UTC [common/configtx/tool] main -> INFO 001 Loading
configuration
2018-02-08 08:46:19.176 UTC [common/configtx/tool] doOutputAnchorPeersUpdate -> INFO 002
Generating anchor peer update
2018-02-08 08:46:19.177 UTC [common/configtx/tool] doOutputAnchorPeersUpdate -> INFO 003
Writing anchor peer update
```

至此所有需要生成的配置文件全部生成完毕。

```
[root@localhost netkiller]# tree -L 4 crypto-config
crypto-config
|-- ordererOrganizations
|   `-- example.com
|       |-- ca
|       |   |-- ca.example.com-cert.pem
|       |   `-- de9204448c9c8e2a72d092f53e8ff069e12dea62001b7b8b9a83ae240d80ed57_sk
|       |-- msp
|       |   |-- admincerts
```

```
|       |   |-- cacerts
|       |   `-- tlscacerts
|       |-- orderers
|       |   `-- orderer.example.com
|       |-- tlsca
|       |   |-- c0b4dd42bd396d68f468aa07dae8ce944ab2d9832b2593cfafb27e53c69ec5e2_sk
|       |   `-- tlsca.example.com-cert.pem
|       `-- users
|           `-- Admin@example.com
`-- peerOrganizations
    `-- org1.example.com
        |-- ca
        |   |-- 74023bd84cc5e6957f9bc30b3ebcd6c5b7507016721702a014dd640df265b61a_sk
        |   `-- ca.org1.example.com-cert.pem
        |-- msp
        |   |-- admincerts
        |   |-- cacerts
        |   `-- tlscacerts
        |-- peers
        |   `-- peer0.org1.example.com
        |-- tlsca
        |   |-- 71bb82530580707aa20fa5955beab202f266aa4da4b435bef20741ce5e64abb9_sk
        |   `-- tlsca.org1.example.com-cert.pem
        `-- users
            |-- Admin@org1.example.com
            `-- User1@org1.example.com

25 directories, 8 files
```

将config和crypto-config文件加复制到ca,orderer,peer,cli等节点上去。

3.5. 清理 Docker 容器

至此所需的证书与创世区块都已生产完毕，fabric-tools 容易完成了它的使命，你可以继续保留或者清理干净。

```
[root@localhost netkiller]# docker-compose -f docker-compose-tools.yml down
Stopping tools ... done
Removing tools ... done
Removing network netkiller_basic
```

清理 tools 容器

```
docker rm -f $(docker ps -qa)
```

4. CA 节点安装

CA 节点需要我们之前生成 crypto-config

4.1. 安装 Docker 镜像

```
docker pull hyperledger/fabric-ca:x86_64-1.1.0
docker tag hyperledger/fabric-ca:x86_64-1.1.0
hyperledger/fabric-ca
```

4.2. docker-compose-ca.yml

```
version: '3'

networks:
  basic:

services:
  ca.example.com:
    image: hyperledger/fabric-ca
    environment:
      - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
      - FABRIC_CA_SERVER_CA_NAME=ca.example.com
      - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/fabric-ca-
server-config/ca.org1.example.com-cert.pem
      - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/fabric-ca-
server-
config/4239aa0dcd76daeeb8ba0cda701851d14504d31aad1b2ddddbac6a573
65e497c_sk
    ports:
      - "XXX.XXX.XXX.XXX:7054:7054"
    command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
    volumes:
      - ./crypto-
config/peerOrganizations/org1.example.com/ca/:/etc/hyperledger/f
abric-ca-server-config
    container_name: ca.example.com
```

```
    networks:
      - basic
```

4.3. 启动 CA 节点

```
docker-compose -f docker-compose-ca.yaml up -d
```

5. CouchDB 节点

整个 Hyperledger Fabric 技术栈中只有这个 CouchDB 是个外来户，看到 CouchDB 我就非常兴奋，这是一个NoSQL数据库(它与MongoDB十分类似)，所以CouchDB 100%可以独立运行，且最容易分离。

CouchDB 在这里有两个方案可以选择。

- 采用 Docker 运行 CouchDB的方案。
- 采用传统方式物理机上本地安装 CouchDB

理论两种方案对实际结果没有什么区别，只需提供IP地址，用户名与密码供其他节点访问即可。但实际我们看到 Hyperledger Fabric 使用的镜像是 hyperledger/fabric-couchdb 不清楚是否有修改过 CouchDB 数据库。

如果你对 Docker 比较熟悉就采用 Docker 方案。如果不熟悉就采用本地安装方式。总之选择一种你能Hold住（掌控）的方案，一旦出现故障，你能第一时间排查并处理。

5.1. 安装 Docker 镜像

```
docker pull hyperledger/fabric-couchdb:x86_64-1.1.0
docker tag hyperledger/fabric-couchdb:x86_64-1.1.0
hyperledger/fabric-couchdb
```

5.2. 安装 CouchDB

下面是 Docker 方案

```
[root@localhost netkiller]# vim docker-compose-couchdb.yml

version: '3'

networks:
  basic:

services:
  couchdb:
    container_name: couchdb
    image: hyperledger/fabric-couchdb
    # Populate the COUCHDB_USER and COUCHDB_PASSWORD to set an
admin user and password
    # for CouchDB.  This will prevent CouchDB from operating in
an "Admin Party" mode.
    environment:
      - COUCHDB_USER=admin
      - COUCHDB_PASSWORD=passw0rd
    ports:
      - 172.16.0.17:5984:5984
    networks:
      - basic
```

5.3. 启动 CouchDB

启动 Docker 容器

```
docker-compose -f docker-compose-couchdb.yml up -d
```

访问CouchDB管理界面，http://172.16.0.17:5984/_utils/ 请使用上面设置的密码进入。若想进入到容器内部可以使用下面命令:

```
docker-compose -f docker-compose-couchdb.yml exec couchdb bash
```

至此 CouchDB 节点部署完毕。

5.4. 备份与恢复 CouchDB

既然是运维区块链，对于运维工作我们最关心的就是如何备份数据，在出现故障的时候恢复数据。

```
npm install --save couchdb-backup-restore
```

```
var cbr = require('couchdb-backup-restore');

var config = {credentials: 'http://localhost:5984'};

function done(err) {
  if (err) {
    return console.error(err);
  }
  console.log('all done!');
}

// backup
cbr.backup(config, done).pipe(fs.createWriteStream('./db-
backup.tar.gz'))

// restore
fs.createReadStream('./db-
backup.tar.gz').pipe(cbr.restore(config, done));
```

6. Orderer 节点安装

6.1. 安装 Docker 镜像

```
docker pull hyperledger/fabric-orderer:x86_64-1.1.0
docker tag hyperledger/fabric-orderer:x86_64-1.1.0 hyperledger/fabric-
orderer
```

6.2. docker-compose-orderer.yml

```
version: '3'

networks:
  basic:

services:
  orderer.example.com:
    container_name: orderer.example.com
    image: hyperledger/fabric-orderer
    environment:
      - ORDERER_GENERAL_LOGLEVEL=debug
      - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
      - ORDERER_GENERAL_GENESISMETHOD=file
      -
ORDERER_GENERAL_GENESISFILE=/etc/hyperledger/configtx/genesis.block
      - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
      - ORDERER_GENERAL_LOCALMSPDIR=/etc/hyperledger/msp/orderer/msp
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric/orderer
    command: orderer
    ports:
      - 7050:7050
    volumes:
        - ./config/:/etc/hyperledger/configtx
        - ./crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/:/e
tc/hyperledger/msp/orderer
        - ./crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/:
/etc/hyperledger/msp/peerOrg1
    networks:
      - basic
```

6.3. 启动 Orderer 节点

```
docker-compose -f docker-compose-orderer.yaml up -d
```

7. Peer 节点安装

7.1. 安装 Docker 镜像

```
docker pull hyperledger/fabric-peer:x86_64-1.1.0
docker tag hyperledger/fabric-peer:x86_64-1.1.0
hyperledger/fabric-peer
```

7.2. docker-compose-peer.yml

```
version: '3'

networks:
  basic:

services:

  peer0.org1.example.com:
    container_name: peer0.org1.example.com
    image: hyperledger/fabric-peer
    environment:
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      - CORE_PEER_ID=peer0.org1.example.com
      - CORE_LOGGING_PEER=debug
      - CORE_CHAINCODE_LOGGING_LEVEL=DEBUG
      - CORE_PEER_LOCALMSPID=Org1MSP
      - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/peer/
      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
      # # the following setting starts chaincode containers on
the same
      # # bridge network as the peers
      # # https://docs.docker.com/compose/networking/
      -
CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}_b
asic
      - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
      -
```

```
CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=172.16.0.17:5984
      # The CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME and
CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD
      # provide the credentials for ledger to connect to
CouchDB.  The username and password must
      # match the username and password set for the associated
CouchDB.
      - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=admin
      - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=passw0rd
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric
    command: peer node start
    # command: peer node start --peer-chaincodedev=true
    ports:
      - 7051:7051
      - 7053:7053
    volumes:
        - /var/run/:/host/var/run/
        - ./crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.exam
ple.com/msp:/etc/hyperledger/msp/peer
        - ./crypto-
config/peerOrganizations/org1.example.com/users:/etc/hyperledge
r/msp/users
        - ./config:/etc/hyperledger/configtx
    #depends_on:
    #  - orderer.example.com
    #  - couchdb
    networks:
      - basic
```

Peer 需要连接到 CouchDB 注意配置项 CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=172.16.0.17:5984

同时连接CouchDB的用户与密码要正确

7.3. 启动 Peer 节点

```
[root@localhost netkiller]# docker-compose -f docker-compose-
peer.yaml up -d
```

7.4. 创建 Channel

进入 Peer 容器

```
docker-compose -f docker-compose-peer.yaml exec
peer0.org1.example.com bash
```

添加 Orderer 节点并创建 Channel

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.e
xample.com/msp
peer channel create -o orderer.example.com:7050 -c mychannel -f
/etc/hyperledger/configtx/channel.tx

peer channel create -o 172.16.0.17:7050 -c mychannel -f
/etc/hyperledger/configtx/channel.tx
```

加入到 mychannel

```
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.e
xample.com/msp
peer channel join -b mychannel.block
```

查看通道

```
st
t@f39764f58ff7:/opt/gopath/src/github.com/hyperledger/fabric#
peer channel list
2018-02-09 08:12:46.454 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-02-09 08:12:46.454 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-02-09 08:12:46.456 UTC [channelCmd] InitCmdFactory -> INFO
003 Endorser and orderer connections initialized
2018-02-09 08:12:46.457 UTC [msp/identity] Sign -> DEBU 004
Sign: plaintext:
0A8A070A5C08031A0C08FEAFF5D30510...631A0D0A0B4765744368616E6E65
6C73
2018-02-09 08:12:46.458 UTC [msp/identity] Sign -> DEBU 005
Sign: digest:
E27446498819AA4FE8EE835ADEF16195489975377A3C18D89C36D37AA24E5CA
2
2018-02-09 08:12:46.469 UTC [channelCmd] list -> INFO 006
Channels peers has joined to:
2018-02-09 08:12:46.469 UTC [channelCmd] list -> INFO 007
mychannel
2018-02-09 08:12:46.469 UTC [main] main -> INFO 008
Exiting.....
```

8. 验收与测试

8.1. 准备合约文件

```
[root@localhost netkiller]# cat chaincode/fabcar/fabcar.go
/*
 * Licensed to the Apache Software Foundation (ASF) under one
 * or more contributor license agreements.  See the NOTICE file
 * distributed with this work for additional information
 * regarding copyright ownership.  The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License.  You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing,
 * software distributed under the License is distributed on an
 * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 * KIND, either express or implied.  See the License for the
 * specific language governing permissions and limitations
 * under the License.
 */

/*
 * The sample smart contract for documentation topic:
 * Writing Your First Blockchain Application
 */

package main

/* Imports
 * 4 utility libraries for formatting, handling bytes, reading
and writing JSON, and string manipulation
 * 2 specific Hyperledger Fabric specific libraries for Smart
Contracts
 */
import (
        "bytes"
        "encoding/json"
        "fmt"
        "strconv"
```

```
        "github.com/hyperledger/fabric/core/chaincode/shim"
        sc "github.com/hyperledger/fabric/protos/peer"
)

// Define the Smart Contract structure
type SmartContract struct {
}

// Define the car structure, with 4 properties.  Structure tags
are used by encoding/json library
type Car struct {
        Make   string `json:"make"`
        Model  string `json:"model"`
        Colour string `json:"colour"`
        Owner  string `json:"owner"`
}

/*
 * The Init method is called when the Smart Contract "fabcar" is
instantiated by the blockchain network
 * Best practice is to have any Ledger initialization in
separate function -- see initLedger()
 */
func (s *SmartContract) Init(APIstub
shim.ChaincodeStubInterface) sc.Response {
        return shim.Success(nil)
}

/*
 * The Invoke method is called as a result of an application
request to run the Smart Contract "fabcar"
 * The calling application program has also specified the
particular smart contract function to be called, with arguments
 */
func (s *SmartContract) Invoke(APIstub
shim.ChaincodeStubInterface) sc.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := APIstub.GetFunctionAndParameters()
        // Route to the appropriate handler function to interact
with the ledger appropriately
        if function == "queryCar" {
                return s.queryCar(APIstub, args)
        } else if function == "initLedger" {
                return s.initLedger(APIstub)
        } else if function == "createCar" {
                return s.createCar(APIstub, args)
```

```
        } else if function == "queryAllCars" {
                return s.queryAllCars(APIstub)
        } else if function == "changeCarOwner" {
                return s.changeCarOwner(APIstub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}

func (s *SmartContract) queryCar(APIstub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        carAsBytes, _ := APIstub.GetState(args[0])
        return shim.Success(carAsBytes)
}

func (s *SmartContract) initLedger(APIstub
shim.ChaincodeStubInterface) sc.Response {
        cars := []Car{
                Car{Make: "Toyota", Model: "Prius", Colour:
"blue", Owner: "Tomoko"},
                Car{Make: "Ford", Model: "Mustang", Colour:
"red", Owner: "Brad"},
                Car{Make: "Hyundai", Model: "Tucson", Colour:
"green", Owner: "Jin Soo"},
                Car{Make: "Volkswagen", Model: "Passat", Colour:
"yellow", Owner: "Max"},
                Car{Make: "Tesla", Model: "S", Colour: "black",
Owner: "Adriana"},
                Car{Make: "Peugeot", Model: "205", Colour:
"purple", Owner: "Michel"},
                Car{Make: "Chery", Model: "S22L", Colour:
"white", Owner: "Aarav"},
                Car{Make: "Fiat", Model: "Punto", Colour:
"violet", Owner: "Pari"},
                Car{Make: "Tata", Model: "Nano", Colour:
"indigo", Owner: "Valeria"},
                Car{Make: "Holden", Model: "Barina", Colour:
"brown", Owner: "Shotaro"},
        }

        i := 0
```

```
        for i < len(cars) {
                fmt.Println("i is ", i)
                carAsBytes, _ := json.Marshal(cars[i])
                APIstub.PutState("CAR"+strconv.Itoa(i),
carAsBytes)
                fmt.Println("Added", cars[i])
                i = i + 1
        }

        return shim.Success(nil)
}

func (s *SmartContract) createCar(APIstub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 5 {
                return shim.Error("Incorrect number of
arguments. Expecting 5")
        }

        var car = Car{Make: args[1], Model: args[2], Colour:
args[3], Owner: args[4]}

        carAsBytes, _ := json.Marshal(car)
        APIstub.PutState(args[0], carAsBytes)

        return shim.Success(nil)
}

func (s *SmartContract) queryAllCars(APIstub
shim.ChaincodeStubInterface) sc.Response {

        startKey := "CAR0"
        endKey := "CAR999"

        resultsIterator, err :=
APIstub.GetStateByRange(startKey, endKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        defer resultsIterator.Close()

        // buffer is a JSON array containing QueryResults
        var buffer bytes.Buffer
        buffer.WriteString("[")

        bArrayMemberAlreadyWritten := false
        for resultsIterator.HasNext() {
```

```
                queryResponse, err := resultsIterator.Next()
                if err != nil {
                        return shim.Error(err.Error())
                }
                // Add a comma before array members, suppress it
for the first array member
                if bArrayMemberAlreadyWritten == true {
                        buffer.WriteString(",")
                }
                buffer.WriteString("{\"Key\":")
                buffer.WriteString("\"")
                buffer.WriteString(queryResponse.Key)
                buffer.WriteString("\"")

                buffer.WriteString(", \"Record\":")
                // Record is a JSON object, so we write as-is
                buffer.WriteString(string(queryResponse.Value))
                buffer.WriteString("}")
                bArrayMemberAlreadyWritten = true
        }
        buffer.WriteString("]")

        fmt.Printf("- queryAllCars:\n%s\n", buffer.String())

        return shim.Success(buffer.Bytes())
}

func (s *SmartContract) changeCarOwner(APIstub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        carAsBytes, _ := APIstub.GetState(args[0])
        car := Car{}

        json.Unmarshal(carAsBytes, &car)
        car.Owner = args[1]

        carAsBytes, _ = json.Marshal(car)
        APIstub.PutState(args[0], carAsBytes)

        return shim.Success(nil)
}

// The main function is only relevant in unit test mode. Only
```

```
included here for completeness.
func main() {

        // Create a new Smart Contract
        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
}
```

8.2. 安装 chaincode

安装合约在 tools 节点上进行。

```
docker-compose -f docker-compose-cli.yaml exec cli bash

CORE_PEER_ADDRESS=172.16.0.17:7051
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/f
abric/peer/crypto/peerOrganizations/org1.example.com/users/Admin
@org1.example.com/msp

peer chaincode install -n fabcar -v 1.0 -p github.com/fabcar
peer chaincode instantiate -o orderer.example.com:7050 -C
mychannel -n fabcar -v 1.0 -c '{"Args":[""]}' -P "OR
('Org1MSP.member','Org2MSP.member')"
peer chaincode invoke -o orderer.example.com:7050 -C mychannel -
n fabcar -c '{"function":"initLedger","Args":[""]}'
```

```
root@a90d0d869dd3:/opt/gopath/src/github.com/hyperledger/fabric/
peer# peer chaincode install -n fabcar -v 1.0 -p
github.com/fabcar
2018-02-09 11:26:28.025 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-02-09 11:26:28.025 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-02-09 11:26:28.025 UTC [chaincodeCmd]
```

```
checkChaincodeCmdParams -> INFO 003 Using default escc
2018-02-09 11:26:28.025 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-02-09 11:26:28.139 UTC [golang-platform] getCodeFromFS ->
DEBU 005 getCodeFromFS github.com/fabcar
2018-02-09 11:26:29.394 UTC [golang-platform] func1 -> DEBU 006
Discarding GOROOT package bytes
2018-02-09 11:26:29.395 UTC [golang-platform] func1 -> DEBU 007
Discarding GOROOT package encoding/json
2018-02-09 11:26:29.395 UTC [golang-platform] func1 -> DEBU 008
Discarding GOROOT package fmt
2018-02-09 11:26:29.395 UTC [golang-platform] func1 -> DEBU 009
Discarding provided package
github.com/hyperledger/fabric/core/chaincode/shim
2018-02-09 11:26:29.395 UTC [golang-platform] func1 -> DEBU 00a
Discarding provided package
github.com/hyperledger/fabric/protos/peer
2018-02-09 11:26:29.395 UTC [golang-platform] func1 -> DEBU 00b
Discarding GOROOT package strconv
2018-02-09 11:26:29.396 UTC [golang-platform]
GetDeploymentPayload -> DEBU 00c done
2018-02-09 11:26:29.406 UTC [msp/identity] Sign -> DEBU 00d
Sign: plaintext:
0A8A070A5C08031A0C08E58AF6D30510...939FFF060000FFFF9C08DC0700200
000
2018-02-09 11:26:29.406 UTC [msp/identity] Sign -> DEBU 00e
Sign: digest:
A504EE8048EEE8C77F9E1C39827124474638110FD3980017BCA6D644E3E7EC98
2018-02-09 11:26:29.426 UTC [chaincodeCmd] install -> DEBU 00f
Installed remotely response:<status:200 payload:"OK" >
2018-02-09 11:26:29.427 UTC [main] main -> INFO 010 Exiting.....
```

```
peer chaincode instantiate -o 172.16.0.17:7050 -C mychannel -n
fabcar -v 1.0 -c '{"Args":[""]}' -P "OR
('Org1MSP.member','Org2MSP.member')"
```

9. 总结

第 26 章 Chaincode 链码（智能合约）

1. 链码开发与测试

1.1. Docker 开发环境

环境安装

安装 fabric-samples

```
git clone https://github.com/hyperledger/fabric-samples.git
cd fabric-samples
```

```
[root@localhost ~]# cd fabric-samples/chaincode-docker-devmode/
[root@localhost chaincode-docker-devmode]# mkdir chaincode
[root@localhost chaincode-docker-devmode]# cp docker-compose-
simple.yaml docker-compose.yaml
[root@localhost chaincode-docker-devmode]# sed -i
"s|./../chaincode|./chaincode|g" docker-compose.yaml
```

chaincode 目录是开发目录

清理镜像和容器

```
docker rm -f $(docker ps -q -a)
docker rmi -f $(docker images -q)
```

安装所需镜像

```
[root@localhost chaincode-docker-devmode]# curl -s
https://raw.githubusercontent.com/oscm/shell/master/blockchain/h
yperledger/fabric/1.1.0/all-in-one.sh | bash

[root@localhost chaincode-docker-devmode]# docker images
REPOSITORY                    TAG                 IMAGE ID
CREATED             SIZE
hyperledger/fabric-tools      latest              6a8993b718c8
2 months ago        1.33GB
hyperledger/fabric-tools      x86_64-1.1.0        6a8993b718c8
2 months ago        1.33GB
hyperledger/fabric-couchdb    latest              9a58db2d2723
2 months ago        1.5GB
hyperledger/fabric-couchdb    x86_64-1.1.0        9a58db2d2723
2 months ago        1.5GB
hyperledger/fabric-orderer    latest              368c78b6f03b
2 months ago        151MB
hyperledger/fabric-orderer    x86_64-1.1.0        368c78b6f03b
2 months ago        151MB
hyperledger/fabric-peer       latest              c2ab022f0bdb
2 months ago        154MB
hyperledger/fabric-peer       x86_64-1.1.0        c2ab022f0bdb
2 months ago        154MB
hyperledger/fabric-ccenv      latest              33feadb8f7a6
2 months ago        1.28GB
hyperledger/fabric-ccenv      x86_64-1.1.0        33feadb8f7a6
2 months ago        1.28GB
hyperledger/fabric-ca         latest              002c9089e464
2 months ago        238MB
hyperledger/fabric-ca         x86_64-1.1.0        002c9089e464
2 months ago        238MB
```

1.2. chaincode 代码

```
[root@localhost chaincode-docker-devmode]# cat
chaincode/chaincode_example02.go
/*
Copyright IBM Corp. 2016 All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
```

```
You may obtain a copy of the License at

                 http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing,
software
distributed under the License is distributed on an "AS IS"
BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
See the License for the specific language governing permissions
and
limitations under the License.
*/

package main

//WARNING - this chaincode's ID is hard-coded in
chaincode_example04 to illustrate one way of
//calling chaincode from a chaincode. If this example is
modified, chaincode_example04.go has
//to be modified as well with the new ID of chaincode_example02.
//chaincode_example05 show's how chaincode ID can be passed in
as a parameter instead of
//hard-coding.

import (
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)

// SimpleChaincode example simple Chaincode implementation
type SimpleChaincode struct {
}

func (t *SimpleChaincode) Init(stub shim.ChaincodeStubInterface)
pb.Response {
        fmt.Println("ex02 Init")
        _, args := stub.GetFunctionAndParameters()
        var A, B string    // Entities
        var Aval, Bval int // Asset holdings
        var err error

        if len(args) != 4 {
                return shim.Error("Incorrect number of
```

```
arguments. Expecting 4")
        }

        // Initialize the chaincode
        A = args[0]
        Aval, err = strconv.Atoi(args[1])
        if err != nil {
                return shim.Error("Expecting integer value for
asset holding")
        }
        B = args[2]
        Bval, err = strconv.Atoi(args[3])
        if err != nil {
                return shim.Error("Expecting integer value for
asset holding")
        }
        fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

        // Write the state to the ledger
        err = stub.PutState(A, []byte(strconv.Itoa(Aval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        err = stub.PutState(B, []byte(strconv.Itoa(Bval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        return shim.Success(nil)
}

func (t *SimpleChaincode) Invoke(stub
shim.ChaincodeStubInterface) pb.Response {
        fmt.Println("ex02 Invoke")
        function, args := stub.GetFunctionAndParameters()
        if function == "invoke" {
                // Make payment of X units from A to B
                return t.invoke(stub, args)
        } else if function == "delete" {
                // Deletes an entity from its state
                return t.delete(stub, args)
        } else if function == "query" {
                // the old "Query" is now implemtned in invoke
                return t.query(stub, args)
        }

        return shim.Error("Invalid invoke function name.
```

```
Expecting \"invoke\" \"delete\" \"query\"")
}

// Transaction makes payment of X units from A to B
func (t *SimpleChaincode) invoke(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        var A, B string    // Entities
        var Aval, Bval int // Asset holdings
        var X int          // Transaction value
        var err error

        if len(args) != 3 {
                return shim.Error("Incorrect number of
arguments. Expecting 3")
        }

        A = args[0]
        B = args[1]

        // Get the state from the ledger
        // TODO: will be nice to have a GetAllState call to
ledger
        Avalbytes, err := stub.GetState(A)
        if err != nil {
                return shim.Error("Failed to get state")
        }
        if Avalbytes == nil {
                return shim.Error("Entity not found")
        }
        Aval, _ = strconv.Atoi(string(Avalbytes))

        Bvalbytes, err := stub.GetState(B)
        if err != nil {
                return shim.Error("Failed to get state")
        }
        if Bvalbytes == nil {
                return shim.Error("Entity not found")
        }
        Bval, _ = strconv.Atoi(string(Bvalbytes))

        // Perform the execution
        X, err = strconv.Atoi(args[2])
        if err != nil {
                return shim.Error("Invalid transaction amount,
expecting a integer value")
        }
        Aval = Aval - X
        Bval = Bval + X
```

```
        fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

        // Write the state back to the ledger
        err = stub.PutState(A, []byte(strconv.Itoa(Aval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        err = stub.PutState(B, []byte(strconv.Itoa(Bval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        return shim.Success(nil)
}

// Deletes an entity from state
func (t *SimpleChaincode) delete(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        A := args[0]

        // Delete the key from the state in ledger
        err := stub.DelState(A)
        if err != nil {
                return shim.Error("Failed to delete state")
        }

        return shim.Success(nil)
}

// query callback representing the query of a chaincode
func (t *SimpleChaincode) query(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        var A string // Entities
        var err error

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting name of the person to query")
        }

        A = args[0]
```

```
        // Get the state from the ledger
        Avalbytes, err := stub.GetState(A)
        if err != nil {
                jsonResp := "{\"Error\":\"Failed to get state
for " + A + "\"}"
                return shim.Error(jsonResp)
        }

        if Avalbytes == nil {
                jsonResp := "{\"Error\":\"Nil amount for " + A +
"\"}"
                return shim.Error(jsonResp)
        }

        jsonResp := "{\"Name\":\"" + A + "\",\"Amount\":\"" +
string(Avalbytes) + "\"}"
        fmt.Printf("Query Response:%s\n", jsonResp)
        return shim.Success(Avalbytes)
}

func main() {
        err := shim.Start(new(SimpleChaincode))
        if err != nil {
                fmt.Printf("Error starting Simple chaincode:
%s", err)
        }
}
```

1.3. 启动容器部署chaincode

```
[root@localhost chaincode-docker-devmode]# docker-compose up -d
```

进入容器

```
[root@localhost ~]# docker exec -it chaincode bash
root@78d23b3b2b37:/opt/gopath/src/chaincode#
```

编译并部署chaincode

```
root@78d23b3b2b37:/opt/gopath/src/chaincode# ls
chaincode_example02.go
root@78d23b3b2b37:/opt/gopath/src/chaincode# go build
chaincode_example02.go
root@78d23b3b2b37:/opt/gopath/src/chaincode#
```

进入 chaincode 容器启动 chaincode 程序。

```
[root@localhost ~]# docker exec -it chaincode bash
root@78d23b3b2b37:/opt/gopath/src/chaincode# ls
chaincode_example02.go
root@78d23b3b2b37:/opt/gopath/src/chaincode# go build
chaincode_example02.go
root@78d23b3b2b37:/opt/gopath/src/chaincode#
CORE_PEER_ADDRESS=peer:7051
CORE_CHAINCODE_ID_NAME=chaincode_example02:1.0
./chaincode_example02
2018-02-17 04:07:31.156 UTC [shim] SetupChaincodeLogging -> INFO
001 Chaincode log level not provided; defaulting to: INFO
2018-02-17 04:07:31.156 UTC [shim] SetupChaincodeLogging -> INFO
002 Chaincode (build level: ) starting up ...
```

CORE_CHAINCODE_ID_NAME=chaincode_example02:1.0 记住这里的配置，下面会用到

1.4. 手工测试

进入 cli 容器

```
[root@localhost ~]# docker exec -it cli bash
root@33dcacfb6f81:/opt/gopath/src/chaincodedev#
```

```
root@33dcacfb6f81:/opt/gopath/src/chaincodedev# peer chaincode
install -v 1.0 -n chaincode_example02 -p chaincodedev/chaincode
2018-02-17 04:46:23.871 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-02-17 04:46:23.871 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-02-17 04:46:23.872 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 003 Using default escc
2018-02-17 04:46:23.872 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-02-17 04:46:23.986 UTC [golang-platform] getCodeFromFS ->
DEBU 005 getCodeFromFS chaincodedev/chaincode
2018-02-17 04:46:24.280 UTC [golang-platform] func1 -> DEBU 006
Discarding GOROOT package fmt
2018-02-17 04:46:24.281 UTC [golang-platform] func1 -> DEBU 007
Discarding provided package
github.com/hyperledger/fabric/core/chaincode/shim
2018-02-17 04:46:24.281 UTC [golang-platform] func1 -> DEBU 008
Discarding provided package
github.com/hyperledger/fabric/protos/peer
2018-02-17 04:46:24.281 UTC [golang-platform] func1 -> DEBU 009
Discarding GOROOT package strconv
2018-02-17 04:46:24.281 UTC [golang-platform]
GetDeploymentPayload -> DEBU 00a done
2018-02-17 04:46:24.287 UTC [msp/identity] Sign -> DEBU 00b
Sign: plaintext:
0AA4080A5C08031A0C08A0E79ED40510...0F19FF0E0000FFFFBA10C104001C0
000
2018-02-17 04:46:24.287 UTC [msp/identity] Sign -> DEBU 00c
Sign: digest:
BE51F2BBE156552DDF4CCBA3E01F890921FB965463683B0B636DAC53BAF2E7C4
2018-02-17 04:46:24.299 UTC [chaincodeCmd] install -> DEBU 00d
Installed remotely response:<status:200 payload:"OK" >
2018-02-17 04:46:24.299 UTC [main] main -> INFO 00e Exiting.....
```

```
root@33dcacfb6f81:/opt/gopath/src/chaincodedev# peer chaincode
instantiate -C myc -n chaincode_example02 -v 1.0 -c '{"Args":
["init","a","100","b","200"]}'
2018-02-17 04:52:42.395 UTC [msp] GetLocalMSP -> DEBU 001
```

```
Returning existing local MSP
2018-02-17 04:52:42.396 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-02-17 04:52:42.396 UTC [msp/identity] Sign -> DEBU 003
Sign: plaintext:
0AA4080A5C08011A0C089AEA9ED40510...436F6E666967426C6F636B0A036D7
963
2018-02-17 04:52:42.396 UTC [msp/identity] Sign -> DEBU 004
Sign: digest:
B2BE9C220FB3313901DBAAE578F428FB98B950ACB5E7519DBC5150D2D21562CB
2018-02-17 04:52:42.413 UTC [common/config] NewStandardValues ->
DEBU 005 Initializing protos for *config.ChannelProtos
2018-02-17 04:52:42.414 UTC [common/config]
initializeProtosStruct -> DEBU 006 Processing field:
HashingAlgorithm
2018-02-17 04:52:42.414 UTC [common/config]
initializeProtosStruct -> DEBU 007 Processing field:
BlockDataHashingStructure
2018-02-17 04:52:42.414 UTC [common/config]
initializeProtosStruct -> DEBU 008 Processing field:
OrdererAddresses
2018-02-17 04:52:42.414 UTC [common/config]
initializeProtosStruct -> DEBU 009 Processing field: Consortium
2018-02-17 04:52:42.415 UTC [common/configtx] addToMap -> DEBU
00a Adding to config map: [Groups] /Channel
2018-02-17 04:52:42.415 UTC [common/configtx] addToMap -> DEBU
00b Adding to config map: [Groups] /Channel/Orderer
2018-02-17 04:52:42.415 UTC [common/configtx] addToMap -> DEBU
00c Adding to config map: [Groups] /Channel/Orderer/SampleOrg
2018-02-17 04:52:42.415 UTC [common/configtx] addToMap -> DEBU
00d Adding to config map: [Values]
/Channel/Orderer/SampleOrg/MSP
2018-02-17 04:52:42.415 UTC [common/configtx] addToMap -> DEBU
00e Adding to config map: [Policy]
/Channel/Orderer/SampleOrg/Admins
2018-02-17 04:52:42.416 UTC [common/configtx] addToMap -> DEBU
00f Adding to config map: [Policy]
/Channel/Orderer/SampleOrg/Readers
2018-02-17 04:52:42.416 UTC [common/configtx] addToMap -> DEBU
010 Adding to config map: [Policy]
/Channel/Orderer/SampleOrg/Writers
2018-02-17 04:52:42.416 UTC [common/configtx] addToMap -> DEBU
011 Adding to config map: [Values] /Channel/Orderer/BatchSize
2018-02-17 04:52:42.416 UTC [common/configtx] addToMap -> DEBU
012 Adding to config map: [Values] /Channel/Orderer/BatchTimeout
2018-02-17 04:52:42.416 UTC [common/configtx] addToMap -> DEBU
013 Adding to config map: [Values]
/Channel/Orderer/ChannelRestrictions
```

```
2018-02-17 04:52:42.417 UTC [common/configtx] addToMap -> DEBU
014 Adding to config map: [Values]
/Channel/Orderer/ConsensusType
2018-02-17 04:52:42.418 UTC [common/configtx] addToMap -> DEBU
015 Adding to config map: [Policy] /Channel/Orderer/Admins
2018-02-17 04:52:42.419 UTC [common/configtx] addToMap -> DEBU
016 Adding to config map: [Policy]
/Channel/Orderer/BlockValidation
2018-02-17 04:52:42.419 UTC [common/configtx] addToMap -> DEBU
017 Adding to config map: [Policy] /Channel/Orderer/Readers
2018-02-17 04:52:42.419 UTC [common/configtx] addToMap -> DEBU
018 Adding to config map: [Policy] /Channel/Orderer/Writers
2018-02-17 04:52:42.419 UTC [common/configtx] addToMap -> DEBU
019 Adding to config map: [Groups] /Channel/Application
2018-02-17 04:52:42.420 UTC [common/configtx] addToMap -> DEBU
01a Adding to config map: [Groups]
/Channel/Application/SampleOrg
2018-02-17 04:52:42.420 UTC [common/configtx] addToMap -> DEBU
01b Adding to config map: [Values]
/Channel/Application/SampleOrg/MSP
2018-02-17 04:52:42.422 UTC [common/configtx] addToMap -> DEBU
01c Adding to config map: [Policy]
/Channel/Application/SampleOrg/Readers
2018-02-17 04:52:42.422 UTC [common/configtx] addToMap -> DEBU
01d Adding to config map: [Policy]
/Channel/Application/SampleOrg/Writers
2018-02-17 04:52:42.422 UTC [common/configtx] addToMap -> DEBU
01e Adding to config map: [Policy]
/Channel/Application/SampleOrg/Admins
2018-02-17 04:52:42.422 UTC [common/configtx] addToMap -> DEBU
01f Adding to config map: [Policy] /Channel/Application/Writers
2018-02-17 04:52:42.423 UTC [common/configtx] addToMap -> DEBU
020 Adding to config map: [Policy] /Channel/Application/Readers
2018-02-17 04:52:42.423 UTC [common/configtx] addToMap -> DEBU
021 Adding to config map: [Policy] /Channel/Application/Admins
2018-02-17 04:52:42.423 UTC [common/configtx] addToMap -> DEBU
022 Adding to config map: [Values] /Channel/Consortium
2018-02-17 04:52:42.423 UTC [common/configtx] addToMap -> DEBU
023 Adding to config map: [Values] /Channel/OrdererAddresses
2018-02-17 04:52:42.423 UTC [common/configtx] addToMap -> DEBU
024 Adding to config map: [Values] /Channel/HashingAlgorithm
2018-02-17 04:52:42.424 UTC [common/configtx] addToMap -> DEBU
025 Adding to config map: [Values]
/Channel/BlockDataHashingStructure
2018-02-17 04:52:42.424 UTC [common/configtx] addToMap -> DEBU
026 Adding to config map: [Policy] /Channel/Readers
2018-02-17 04:52:42.424 UTC [common/configtx] addToMap -> DEBU
027 Adding to config map: [Policy] /Channel/Writers
```

```
2018-02-17 04:52:42.424 UTC [common/configtx] addToMap -> DEBU
028 Adding to config map: [Policy] /Channel/Admins
2018-02-17 04:52:42.424 UTC [common/configtx] processConfig ->
DEBU 029 Beginning new config for channel myc
2018-02-17 04:52:42.425 UTC [common/config] NewStandardValues ->
DEBU 02a Initializing protos for *config.ChannelProtos
2018-02-17 04:52:42.425 UTC [common/config]
initializeProtosStruct -> DEBU 02b Processing field:
HashingAlgorithm
2018-02-17 04:52:42.425 UTC [common/config]
initializeProtosStruct -> DEBU 02c Processing field:
BlockDataHashingStructure
2018-02-17 04:52:42.425 UTC [common/config]
initializeProtosStruct -> DEBU 02d Processing field:
OrdererAddresses
2018-02-17 04:52:42.425 UTC [common/config]
initializeProtosStruct -> DEBU 02e Processing field: Consortium
2018-02-17 04:52:42.425 UTC [policies] ProposePolicy -> DEBU 02f
Proposed new policy Readers for Channel
2018-02-17 04:52:42.425 UTC [policies] ProposePolicy -> DEBU 030
Proposed new policy Writers for Channel
2018-02-17 04:52:42.426 UTC [policies] ProposePolicy -> DEBU 031
Proposed new policy Admins for Channel
2018-02-17 04:52:42.426 UTC [common/config] NewStandardValues ->
DEBU 032 Initializing protos for *config.OrdererProtos
2018-02-17 04:52:42.426 UTC [common/config]
initializeProtosStruct -> DEBU 033 Processing field:
ConsensusType
2018-02-17 04:52:42.426 UTC [common/config]
initializeProtosStruct -> DEBU 034 Processing field: BatchSize
2018-02-17 04:52:42.426 UTC [common/config]
initializeProtosStruct -> DEBU 035 Processing field:
BatchTimeout
2018-02-17 04:52:42.426 UTC [common/config]
initializeProtosStruct -> DEBU 036 Processing field:
KafkaBrokers
2018-02-17 04:52:42.426 UTC [common/config]
initializeProtosStruct -> DEBU 037 Processing field:
ChannelRestrictions
2018-02-17 04:52:42.426 UTC [policies] ProposePolicy -> DEBU 038
Proposed new policy Readers for Orderer
2018-02-17 04:52:42.426 UTC [policies] ProposePolicy -> DEBU 039
Proposed new policy Writers for Orderer
2018-02-17 04:52:42.426 UTC [policies] ProposePolicy -> DEBU 03a
Proposed new policy Admins for Orderer
2018-02-17 04:52:42.426 UTC [policies] ProposePolicy -> DEBU 03b
Proposed new policy BlockValidation for Orderer
2018-02-17 04:52:42.426 UTC [common/config] NewStandardValues ->
```

```
DEBU 03c Initializing protos for *config.OrganizationProtos
2018-02-17 04:52:42.426 UTC [common/config]
initializeProtosStruct -> DEBU 03d Processing field: MSP
2018-02-17 04:52:42.427 UTC [policies] ProposePolicy -> DEBU 03e
Proposed new policy Admins for SampleOrg
2018-02-17 04:52:42.427 UTC [policies] ProposePolicy -> DEBU 03f
Proposed new policy Readers for SampleOrg
2018-02-17 04:52:42.427 UTC [policies] ProposePolicy -> DEBU 040
Proposed new policy Writers for SampleOrg
2018-02-17 04:52:42.427 UTC [common/config] NewStandardValues ->
DEBU 041 Initializing protos for *struct {}
2018-02-17 04:52:42.427 UTC [policies] ProposePolicy -> DEBU 042
Proposed new policy Admins for Application
2018-02-17 04:52:42.427 UTC [policies] ProposePolicy -> DEBU 043
Proposed new policy Writers for Application
2018-02-17 04:52:42.427 UTC [policies] ProposePolicy -> DEBU 044
Proposed new policy Readers for Application
2018-02-17 04:52:42.427 UTC [common/config] NewStandardValues ->
DEBU 045 Initializing protos for *config.OrganizationProtos
2018-02-17 04:52:42.427 UTC [common/config]
initializeProtosStruct -> DEBU 046 Processing field: MSP
2018-02-17 04:52:42.427 UTC [common/config] NewStandardValues ->
DEBU 047 Initializing protos for *config.ApplicationOrgProtos
2018-02-17 04:52:42.427 UTC [common/config]
initializeProtosStruct -> DEBU 048 Processing field: AnchorPeers
2018-02-17 04:52:42.427 UTC [common/config] NewStandardValues ->
DEBU 049 Initializing protos for *config.OrganizationProtos
2018-02-17 04:52:42.428 UTC [common/config]
initializeProtosStruct -> DEBU 04a Processing field: MSP
2018-02-17 04:52:42.428 UTC [policies] ProposePolicy -> DEBU 04b
Proposed new policy Readers for SampleOrg
2018-02-17 04:52:42.428 UTC [policies] ProposePolicy -> DEBU 04c
Proposed new policy Writers for SampleOrg
2018-02-17 04:52:42.428 UTC [policies] ProposePolicy -> DEBU 04d
Proposed new policy Admins for SampleOrg
2018-02-17 04:52:42.428 UTC [common/config] validateMSP -> DEBU
04e Setting up MSP for org SampleOrg
2018-02-17 04:52:42.428 UTC [msp] NewBccspMsp -> DEBU 04f
Creating BCCSP-based MSP instance
2018-02-17 04:52:42.428 UTC [msp] Setup -> DEBU 050 Setting up
MSP instance DEFAULT
2018-02-17 04:52:42.429 UTC [msp/identity] newIdentity -> DEBU
051 Creating identity instance for ID -----BEGIN CERTIFICATE----
-
MIICYjCCAgmgAwIBAgIUB3CTDOU47sUC5K4kn/Caqnh114YwCgYIKoZIzj0EAwIw
fzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xHzAdBgNVBAoTFkludGVybmV0IFdpZGdldHMsIEluYy4xDDAK
BgNVBAsTA1dXVzEUMBIGA1UEAxMLZXhhbXBsZS5jb20wHhcNMTYxMDEyMTkzMTAw
```

```
WhcNMjExMDExMTkzMTAwWjB/MQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv
cm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEfMB0GA1UEChMWSW50ZXJuZXQg
V2lkZ2V0cywgSW5jLjEMMAoGA1UECxMDV1dXMRQwEgYDVQQDEwtleGFtcGxlLmNv
bTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABKIH5b2JaSmqiQXHyqC+cmknICcF
i5AddVjsQizDV6uZ4v6s+PWiJyzfA/rTtMvYAPq/yeEHpBUB1j053mxnpMujYzBh
MA4GA1UdDwEB/wQEAwIBBjAPBgNVHRMBAf8EBTADAQH/MB0GA1UdDgQWBBQXZ0I9
qp6CP8TFHZ9bw5nRtZxIEDAfBgNVHSMEGDAWgBQXZ0I9qp6CP8TFHZ9bw5nRtZxI
EDAKBggqhkjOPQQDAgNHADBEAiAHp5Rbp9Em1G/UmKn8WsCbqDfWecVbZPQj3RK4
oG5kQQIgQAe4OOKYhJdh3f7URaKfGTf492/nmRmtK+ySKjpHSrU=
-----END CERTIFICATE-----
2018-02-17 04:52:42.430 UTC [msp/identity] newIdentity -> DEBU
052 Creating identity instance for ID -----BEGIN CERTIFICATE----
-
MIICjDCCAjKgAwIBAgIUBEVwsSx0TmqdbzNwleNBBzoIT0wwCgYIKoZIzj0EAwIw
fzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xHzAdBgNVBAoTFkludGVybmV0IFdpZGdldHMsIEluYy4xDDAK
BgNVBAsTA1dXVzEUMBIGA1UEAxMLZXhhbXBsZS5jb20wHhcNMTYxMTExMTcwNzAw
WhcNMTcxMTExMTcwNzAwWjBjMQswCQYDVQQGEwJVUzEXMBUGA1UECBMOTm9ydGgg
Q2Fyb2xpbmExEDAOBgNVBAcTB1JhbGVpZ2gxGzAZBgNVBAoTEkh5cGVybGVkZ2Vy
IEZhYnJpYzEMMAoGA1UECxMDQ09QMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE
HBuKsAO43hs4JGpFfiGMkB/xsILTsOvmN2WmwpsPHZNL6w8HWe3xCPQtdG/XJJvZ
+C756KEsUBM3yw5PTfku8qOBpzCBpDAOBgNVHQ8BAf8EBAMCBaAwHQYDVR0lBBYw
FAYIKwYBBQUHAwEGCCsGAQUFBwMCMAwGA1UdEwEB/wQCMAAwHQYDVR0OBBYEFOFC
dcUZ4es3ltiCgAVDoyLfVpPIMB8GA1UdIwQYMBaAFBdnQj2qnoI/xMUdn1vDmdG1
nEgQMCUGA1UdEQQeMByCCm15aG9zdC5jb22CDnd3dy5teWhvc3QuY29tMAoGCCqG
SM49BAMCA0gAMEUCIDf9Hbl4xn3z4EwNKmilM9lX2Fq4jWpAaRVB97OmVEeyAiEA
25aDPQHGGq2AvhKT0wvt08cX1GTGCIbfmuLpMwKQj38=
-----END CERTIFICATE-----
2018-02-17 04:52:42.432 UTC [msp/identity] newIdentity -> DEBU
053 Creating identity instance for ID -----BEGIN CERTIFICATE----
-
MIICjDCCAjKgAwIBAgIUBEVwsSx0TmqdbzNwleNBBzoIT0wwCgYIKoZIzj0EAwIw
fzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xHzAdBgNVBAoTFkludGVybmV0IFdpZGdldHMsIEluYy4xDDAK
BgNVBAsTA1dXVzEUMBIGA1UEAxMLZXhhbXBsZS5jb20wHhcNMTYxMTExMTcwNzAw
WhcNMTcxMTExMTcwNzAwWjBjMQswCQYDVQQGEwJVUzEXMBUGA1UECBMOTm9ydGgg
Q2Fyb2xpbmExEDAOBgNVBAcTB1JhbGVpZ2gxGzAZBgNVBAoTEkh5cGVybGVkZ2Vy
IEZhYnJpYzEMMAoGA1UECxMDQ09QMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE
HBuKsAO43hs4JGpFfiGMkB/xsILTsOvmN2WmwpsPHZNL6w8HWe3xCPQtdG/XJJvZ
+C756KEsUBM3yw5PTfku8qOBpzCBpDAOBgNVHQ8BAf8EBAMCBaAwHQYDVR0lBBYw
FAYIKwYBBQUHAwEGCCsGAQUFBwMCMAwGA1UdEwEB/wQCMAAwHQYDVR0OBBYEFOFC
dcUZ4es3ltiCgAVDoyLfVpPIMB8GA1UdIwQYMBaAFBdnQj2qnoI/xMUdn1vDmdG1
nEgQMCUGA1UdEQQeMByCCm15aG9zdC5jb22CDnd3dy5teWhvc3QuY29tMAoGCCqG
SM49BAMCA0gAMEUCIDf9Hbl4xn3z4EwNKmilM9lX2Fq4jWpAaRVB97OmVEeyAiEA
25aDPQHGGq2AvhKT0wvt08cX1GTGCIbfmuLpMwKQj38=
-----END CERTIFICATE-----
2018-02-17 04:52:42.435 UTC [msp] Validate -> DEBU 054 MSP
DEFAULT validating identity
```

```
2018-02-17 04:52:42.435 UTC [msp] getCertificationChain -> DEBU
055 MSP DEFAULT getting certification chain
2018-02-17 04:52:42.436 UTC [common/config] Validate -> DEBU 056
Anchor peers for org SampleOrg are
2018-02-17 04:52:42.436 UTC [common/config] validateMSP -> DEBU
057 Setting up MSP for org SampleOrg
2018-02-17 04:52:42.436 UTC [msp] NewBccspMsp -> DEBU 058
Creating BCCSP-based MSP instance
2018-02-17 04:52:42.436 UTC [msp] Setup -> DEBU 059 Setting up
MSP instance DEFAULT
2018-02-17 04:52:42.437 UTC [msp/identity] newIdentity -> DEBU
05a Creating identity instance for ID -----BEGIN CERTIFICATE----
-
MIICYjCCAgmgAwIBAgIUB3CTDOU47sUC5K4kn/Caqnh114YwCgYIKoZIzj0EAwIw
fzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xHzAdBgNVBAoTFkludGVybmV0IFdpZGdldHMsIEluYy4xDDAK
BgNVBAsTA1dXVzEUMBIGA1UEAxMLZXhhbXBsZS5jb20wHhcNMTYxMDEyMTkzMTAw
WhcNMjExMDExMTkzMTAwWjB/MQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv
cm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEfMB0GA1UEChMWSW50ZXJuZXQg
V2lkZ2V0cywgSW5jLjEMMAoGA1UECxMDV1dXMRQwEgYDVQQDEwtleGFtcGxlLmNv
bTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABKIH5b2JaSmqiQXHyqC+cmknICcF
i5AddVjsQizDV6uZ4v6s+PWiJyzfA/rTtMvYAPq/yeEHpBUB1j053mxnpMujYzBh
MA4GA1UdDwEB/wQEAwIBBjAPBgNVHRMBAf8EBTADAQH/MB0GA1UdDgQWBBQXZ0I9
qp6CP8TFHZ9bw5nRtZxIEDAfBgNVHSMEGDAWgBQXZ0I9qp6CP8TFHZ9bw5nRtZxI
EDAKBggqhkjOPQQDAgNHADBEAiAHp5Rbp9Em1G/UmKn8WsCbqDfWecVbZPQj3RK4
oG5kQQIgQAe4OOKYhJdh3f7URaKfGTf492/nmRmtK+ySKjpHSrU=
-----END CERTIFICATE-----
2018-02-17 04:52:42.438 UTC [msp/identity] newIdentity -> DEBU
05b Creating identity instance for ID -----BEGIN CERTIFICATE----
-
MIICjDCCAjKgAwIBAgIUBEVwsSx0TmqdbzNwleNBBzoIT0wwCgYIKoZIzj0EAwIw
fzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xHzAdBgNVBAoTFkludGVybmV0IFdpZGdldHMsIEluYy4xDDAK
BgNVBAsTA1dXVzEUMBIGA1UEAxMLZXhhbXBsZS5jb20wHhcNMTYxMTExMTcwNzAw
WhcNMTcxMTExMTcwNzAwWjBjMQswCQYDVQQGEwJVUzEXMBUGA1UECBMOTm9ydGgg
Q2Fyb2xpbmExEDAOBgNVBAcTB1JhbGVpZ2gxGzAZBgNVBAoTEkh5cGVybGVkZ2Vy
IEZhYnJpYzEMMAoGA1UECxMDQ09QMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE
HBuKsAO43hs4JGpFfiGMkB/xsILTsOvmN2WmwpsPHZNL6w8HWe3xCPQtdG/XJJvZ
+C756KEsUBM3yw5PTfku8qOBpzCBpDAOBgNVHQ8BAf8EBAMCBaAwHQYDVR0lBBYw
FAYIKwYBBQUHAwEGCCsGAQUFBwMCMAwGA1UdEwEB/wQCMAAwHQYDVR0OBBYEFOFC
dcUZ4es3ltiCgAVDoyLfVpPIMB8GA1UdIwQYMBaAFBdnQj2qnoI/xMUdn1vDmdG1
nEgQMCUGA1UdEQQeMByCCm15aG9zdC5jb22CDnd3dy5teWhvc3QuY29tMAoGCCqG
SM49BAMCA0gAMEUCIDf9Hbl4xn3z4EwNKmilM9lX2Fq4jWpAaRVB97OmVEeyAiEA
25aDPQHGGq2AvhKT0wvt08cX1GTGCIbfmuLpMwKQj38=
-----END CERTIFICATE-----
2018-02-17 04:52:42.439 UTC [msp/identity] newIdentity -> DEBU
05c Creating identity instance for ID -----BEGIN CERTIFICATE----
-
```

MIICjDCCAjKgAwIBAgIUBEVwsSx0TmqdbzNwleNBBzoIT0wwCgYIKoZIzj0EAwIw
fzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh
biBGcmFuY2lzY28xHzAdBgNVBAoTFkludGVybmV0IFdpZGdldHMsIEluYy4xDDAK
BgNVBAsTA1dXVzEUMBIGA1UEAxMLZXhhbXBsZS5jb20wHhcNMTYxMTExMTcwNzAw
WhcNMTcxMTExMTcwNzAwWjBjMQswCQYDVQQGEwJVUzEXMBUGA1UECBMOTm9ydGgg
Q2Fyb2xpbmExEDAOBgNVBAcTB1JhbGVpZ2gxGzAZBgNVBAoTEkh5cGVybGVkZ2Vy
IEZhYnJpYzEMMAoGA1UECxMDQ09QMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE
HBuKsAO43hs4JGpFfiGMkB/xsILTsOvmN2WmwpsPHZNL6w8HWe3xCPQtdG/XJJvZ
+C756KEsUBM3yw5PTfku8qOBpzCBpDAOBgNVHQ8BAf8EBAMCBaAwHQYDVR0lBBYw
FAYIKwYBBQUHAwEGCCsGAQUFBwMCMAwGA1UdEwEB/wQCMAAwHQYDVR0OBBYEFOFC
dcUZ4es3ltiCgAVDoyLfVpPIMB8GA1UdIwQYMBaAFBdnQj2qnoI/xMUdn1vDmdG1
nEgQMCUGA1UdEQQeMByCCm15aG9zdC5jb22CDnd3dy5teWhvc3QuY29tMAoGCCqG
SM49BAMCA0gAMEUCIDf9Hbl4xn3z4EwNKmilM9lX2Fq4jWpAaRVB97OmVEeyAiEA
25aDPQHGGq2AvhKT0wvt08cX1GTGCIbfmuLpMwKQj38=
-----END CERTIFICATE-----
2018-02-17 04:52:42.442 UTC [msp] Validate -> DEBU 05d MSP
DEFAULT validating identity
2018-02-17 04:52:42.443 UTC [msp] getCertificationChain -> DEBU
05e MSP DEFAULT getting certification chain
2018-02-17 04:52:42.444 UTC [msp] Setup -> DEBU 05f Setting up
the MSP manager (1 msps)
2018-02-17 04:52:42.444 UTC [msp] Setup -> DEBU 060 MSP manager
setup complete, setup 1 msps
2018-02-17 04:52:42.444 UTC [policies] GetPolicy -> DEBU 061
Returning policy Admins for evaluation
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
062 In commit adding relative sub-policy SampleOrg/Admins to
Orderer
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 063
Returning policy Readers for evaluation
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
064 In commit adding relative sub-policy SampleOrg/Readers to
Orderer
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 065
Returning policy Writers for evaluation
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
066 In commit adding relative sub-policy SampleOrg/Writers to
Orderer
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 067
Returning policy Readers for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 068
Returning policy Writers for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 069
Returning policy Admins for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 06a
Returning policy Writers for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 06b
Returning policy Writers for evaluation

```
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
06c In commit adding relative sub-policy SampleOrg/Writers to
Application
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 06d
Returning policy Admins for evaluation
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
06e In commit adding relative sub-policy SampleOrg/Admins to
Application
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 06f
Returning policy Readers for evaluation
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
070 In commit adding relative sub-policy SampleOrg/Readers to
Application
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 071
Returning policy Admins for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 072
Returning policy Writers for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 073
Returning policy Readers for evaluation
2018-02-17 04:52:42.445 UTC [policies] GetPolicy -> DEBU 074
Returning policy SampleOrg/Writers for evaluation
2018-02-17 04:52:42.445 UTC [policies] CommitProposals -> DEBU
075 In commit adding relative sub-policy
Orderer/SampleOrg/Writers to Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 076
Returning policy Readers for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
077 In commit adding relative sub-policy Orderer/Readers to
Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 078
Returning policy Writers for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
079 In commit adding relative sub-policy Orderer/Writers to
Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 07a
Returning policy Admins for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
07b In commit adding relative sub-policy Orderer/Admins to
Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 07c
Returning policy BlockValidation for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
07d In commit adding relative sub-policy Orderer/BlockValidation
to Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 07e
Returning policy SampleOrg/Admins for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
07f In commit adding relative sub-policy
```

```
Orderer/SampleOrg/Admins to Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 080
Returning policy SampleOrg/Readers for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
081 In commit adding relative sub-policy
Orderer/SampleOrg/Readers to Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 082
Returning policy Admins for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
083 In commit adding relative sub-policy Application/Admins to
Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 084
Returning policy Writers for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
085 In commit adding relative sub-policy Application/Writers to
Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 086
Returning policy Readers for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
087 In commit adding relative sub-policy Application/Readers to
Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 088
Returning policy SampleOrg/Writers for evaluation
2018-02-17 04:52:42.446 UTC [policies] CommitProposals -> DEBU
089 In commit adding relative sub-policy
Application/SampleOrg/Writers to Channel
2018-02-17 04:52:42.446 UTC [policies] GetPolicy -> DEBU 08a
Returning policy SampleOrg/Admins for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
08b In commit adding relative sub-policy
Application/SampleOrg/Admins to Channel
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 08c
Returning policy SampleOrg/Readers for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
08d In commit adding relative sub-policy
Application/SampleOrg/Readers to Channel
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 08e
Returning policy Readers for evaluation
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 08f
Returning policy Readers for evaluation
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 090
Returning policy Writers for evaluation
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 091
Returning policy Writers for evaluation
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 092
Returning policy Admins for evaluation
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 093
Returning policy Admins for evaluation
```

```
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 094
Returning policy Readers for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
095 As expected, current configuration has policy
'/Channel/Readers'
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 096
Returning policy Writers for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
097 As expected, current configuration has policy
'/Channel/Writers'
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 098
Returning policy Application/Readers for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
099 As expected, current configuration has policy
'/Channel/Application/Readers'
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 09a
Returning policy Application/Writers for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
09b As expected, current configuration has policy
'/Channel/Application/Writers'
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 09c
Returning policy Application/Admins for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
09d As expected, current configuration has policy
'/Channel/Application/Admins'
2018-02-17 04:52:42.447 UTC [policies] GetPolicy -> DEBU 09e
Returning policy Orderer/BlockValidation for evaluation
2018-02-17 04:52:42.447 UTC [policies] CommitProposals -> DEBU
09f As expected, current configuration has policy
'/Channel/Orderer/BlockValidation'
2018-02-17 04:52:42.448 UTC [chaincodeCmd] InitCmdFactory ->
INFO 0a0 Get chain(myc) orderer endpoint: orderer:7050
2018-02-17 04:52:42.450 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 0a1 Using default escc
2018-02-17 04:52:42.450 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 0a2 Using default vscc
2018-02-17 04:52:42.450 UTC [msp/identity] Sign -> DEBU 0a3
Sign: plaintext:
0AA9080A6108031A0C089AEA9ED40510...30300A000A04657363630A0476736
363
2018-02-17 04:52:42.451 UTC [msp/identity] Sign -> DEBU 0a4
Sign: digest:
53C69F4D3E17BDC5F77DED0D002584A4ED9CE44A89C49F6CABBB03DED7E49A68
2018-02-17 04:52:42.477 UTC [msp/identity] Sign -> DEBU 0a5
Sign: plaintext:
0AA9080A6108031A0C089AEA9ED40510...8E2320C661BDAC91644166CF71555
68D
2018-02-17 04:52:42.477 UTC [msp/identity] Sign -> DEBU 0a6
```

```
Sign: digest:
17F14535D40E75D42A20AAD7A3BA9C93796FCB89AA50F80ADCDFAC77BECA0F65
2018-02-17 04:52:42.483 UTC [main] main -> INFO 0a7 Exiting.....
```

查看 a,b 两个账号的余额

```
peer chaincode query -C myc -n chaincode_example02 -c '{"Args":
["query","a"]}'
peer chaincode query -C myc -n chaincode_example02 -c '{"Args":
["query","b"]}'
```

a 向 b 转账 50

```
peer chaincode invoke -C myc -n chaincode_example02 -c '{"Args":
["invoke","a","b","50"]}'
peer chaincode query -C myc -n chaincode_example02 -c '{"Args":
["query","a"]}'
peer chaincode query -C myc -n chaincode_example02 -c '{"Args":
["query","b"]}'
```

这是再查看 a, b 两个掌控的余额应该是 a: 50 , B: 250

```
root@33dcacfb6f81:/opt/gopath/src/chaincodedev# peer chaincode
query -C myc -n chaincode_example02 -c '{"Args":["query","a"]}'
2018-02-17 04:56:46.870 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-02-17 04:56:46.870 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-02-17 04:56:46.870 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 003 Using default escc
2018-02-17 04:56:46.870 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-02-17 04:56:46.871 UTC [msp/identity] Sign -> DEBU 005
Sign: plaintext:
```

```
0AB8080A7008031A0C088EEC9ED40510...6C6530321A0A0A0571756572790A0
161
2018-02-17 04:56:46.871 UTC [msp/identity] Sign -> DEBU 006
Sign: digest:
1F0EB0AD273AE083C5673CE73877AE3F09324BDCB6968E2477D3F947B37EF4C5
Query Result: 50
2018-02-17 04:56:46.903 UTC [main] main -> INFO 007 Exiting.....

root@33dcacfb6f81:/opt/gopath/src/chaincodedev# peer chaincode
query -C myc -n chaincode_example02 -c '{"Args":["query","b"]}'
2018-02-17 04:55:29.791 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-02-17 04:55:29.792 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-02-17 04:55:29.792 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 003 Using default escc
2018-02-17 04:55:29.792 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-02-17 04:55:29.793 UTC [msp/identity] Sign -> DEBU 005
Sign: plaintext:
0AB8080A7008031A0C08C1EB9ED40510...6C6530321A0A0A0571756572790A0
162
2018-02-17 04:55:29.793 UTC [msp/identity] Sign -> DEBU 006
Sign: digest:
36BED07CE9664E26478A993392B2869F10533506CA9BD5F0A9D48D02EDB1F1D0
Query Result: 250
2018-02-17 04:55:29.823 UTC [main] main -> INFO 007 Exiting.....
```

chaincode 终端也会显示下面内容。

```
root@78d23b3b2b37:/opt/gopath/src/chaincode#
CORE_PEER_ADDRESS=peer:7051
CORE_CHAINCODE_ID_NAME=chaincode_example02:1.0
./chaincode_example02
2018-02-17 04:51:25.481 UTC [shim] SetupChaincodeLogging -> INFO
001 Chaincode log level not provided; defaulting to: INFO
2018-02-17 04:51:25.482 UTC [shim] SetupChaincodeLogging -> INFO
002 Chaincode (build level: ) starting up ...
ex02 Init
Aval = 100, Bval = 200
ex02 Invoke
Query Response:{"Name":"a","Amount":"100"}
ex02 Invoke
```

```
Query Response:{"Name":"b","Amount":"200"}
ex02 Invoke
Aval = 50, Bval = 250
ex02 Invoke
Query Response:{"Name":"b","Amount":"250"}
ex02 Invoke
Query Response:{"Name":"a","Amount":"50"}
```

1.5. 代码测试

```
/*
Copyright IBM Corp. All Rights Reserved.
SPDX-License-Identifier: Apache-2.0
*/

package example02

import (
        "fmt"
        "testing"

        "github.com/hyperledger/fabric/core/chaincode/shim"
)

func checkInit(t *testing.T, stub *shim.MockStub, args [][]byte)
{
        res := stub.MockInit("1", args)
        if res.Status != shim.OK {
                fmt.Println("Init failed", string(res.Message))
                t.FailNow()
        }
}

func checkState(t *testing.T, stub *shim.MockStub, name string,
value string) {
        bytes := stub.State[name]
        if bytes == nil {
                fmt.Println("State", name, "failed to get
value")
                t.FailNow()
        }
        if string(bytes) != value {
                fmt.Println("State value", name, "was not",
```

```
value, "as expected")
                t.FailNow()
        }
}

func checkQuery(t *testing.T, stub *shim.MockStub, name string,
value string) {
        res := stub.MockInvoke("1", [][]byte{[]byte("query"),
[]byte(name)})
        if res.Status != shim.OK {
                fmt.Println("Query", name, "failed",
string(res.Message))
                t.FailNow()
        }
        if res.Payload == nil {
                fmt.Println("Query", name, "failed to get
value")
                t.FailNow()
        }
        if string(res.Payload) != value {
                fmt.Println("Query value", name, "was not",
value, "as expected")
                t.FailNow()
        }
}

func checkInvoke(t *testing.T, stub *shim.MockStub, args []
[]byte) {
        res := stub.MockInvoke("1", args)
        if res.Status != shim.OK {
                fmt.Println("Invoke", args, "failed",
string(res.Message))
                t.FailNow()
        }
}

func TestExample02_Init(t *testing.T) {
        scc := new(SimpleChaincode)
        stub := shim.NewMockStub("ex02", scc)

        // Init A=123 B=234
        checkInit(t, stub, [][]byte{[]byte("init"), []byte("A"),
[]byte("123"), []byte("B"), []byte("234")})

        checkState(t, stub, "A", "123")
        checkState(t, stub, "B", "234")
}
```

```
func TestExample02_Query(t *testing.T) {
        scc := new(SimpleChaincode)
        stub := shim.NewMockStub("ex02", scc)

        // Init A=345 B=456
        checkInit(t, stub, [][]byte{[]byte("init"), []byte("A"),
[]byte("345"), []byte("B"), []byte("456")})

        // Query A
        checkQuery(t, stub, "A", "345")

        // Query B
        checkQuery(t, stub, "B", "456")
}

func TestExample02_Invoke(t *testing.T) {
        scc := new(SimpleChaincode)
        stub := shim.NewMockStub("ex02", scc)

        // Init A=567 B=678
        checkInit(t, stub, [][]byte{[]byte("init"), []byte("A"),
[]byte("567"), []byte("B"), []byte("678")})

        // Invoke A->B for 123
        checkInvoke(t, stub, [][]byte{[]byte("invoke"),
[]byte("A"), []byte("B"), []byte("123")})
        checkQuery(t, stub, "A", "444")
        checkQuery(t, stub, "B", "801")

        // Invoke B->A for 234
        checkInvoke(t, stub, [][]byte{[]byte("invoke"),
[]byte("B"), []byte("A"), []byte("234")})
        checkQuery(t, stub, "A", "678")
        checkQuery(t, stub, "B", "567")
        checkQuery(t, stub, "A", "678")
        checkQuery(t, stub, "B", "567")
}
```

1.6. 在宿主主机上编译合约

编译 chaincode 可以在 docker 容器中进行，也可以在 docker 外面进行。方法如下:

安装依赖的库文件

```
yum install libtool-ltdl-devel
go get github.com/hyperledger/fabric
```

编译合约文件

```
[root@localhost ~]# cd
~/netkiller/chaincode/chaincode_example02/
[root@localhost chaincode_example02]#

[root@localhost chaincode_example02]# go build
chaincode_example02.go

[root@localhost chaincode_example02]# ls
chaincode_example02  chaincode_example02.go
chaincode_example02_test.go
```

提示

由于 docker 容器中没有 vim/nano 命令，无法编译 .go 文件，所以在宿主主机编译更为方便。

1.7. 链码升级

已经实例化的链码如果需要更新，可以通过 upgrade 升级。

```
peer chaincode upgrade -o orderer.example.com:7050 -C mychannel
-n token4 -v 1.1 -c '{"Args":[""]}' -P "OR
('Org1MSP.member','Org2MSP.member')"
```

操作演示

```
[root@localhost token]# docker exec -e
"CORE_PEER_LOCALMSPID=Org1MSP" -e
"CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/
fabric/peer/crypto/peerOrganizations/org1.example.com/users/Admi
n@org1.example.com/msp" cli peer chaincode upgrade -o
orderer.example.com:7050 -C mychannel -n token4 -v 1.1 -c
'{"Args":[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
2018-03-21 05:45:57.825 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-03-21 05:45:57.825 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-03-21 05:45:57.827 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 003 Using default escc
2018-03-21 05:45:57.828 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 004 Using default vscc
2018-03-21 05:45:57.829 UTC [chaincodeCmd] upgrade -> DEBU 005
Get upgrade proposal for chaincode <name:"token4" version:"1.1"
>
2018-03-21 05:45:57.829 UTC [msp/identity] Sign -> DEBU 006
Sign: plaintext:
0A91070A6708031A0C0895E3C7D50510...324D53500A04657363630A0476736
363
2018-03-21 05:45:57.830 UTC [msp/identity] Sign -> DEBU 007
Sign: digest:
D74025895DC08535F47FACF5AB3BE1ECFBCC30BA569E60C3DD45655BF5B89F89
2018-03-21 05:46:43.134 UTC [chaincodeCmd] upgrade -> DEBU 008
endorse upgrade proposal, get response <status:200 message:"OK"
payload:"\n\006token4\022\0031.1\032\004escc\"\004vscc*
(\022\014\022\n\010\001\022\002\010\000\022\002\010\001\032\013\
022\t\n\007Org1MSP\032\013\022\t\n\007Org2MSP2D\n
\260\010<\325\376\335\274W\266N\236N\221\241NP\257&\330\220\232\
355\022OB\334cE\021\374\343\222\022
r\2150\010R\nQ=\3135\314\312\334\312f\373\363\2646*hUT\362\227<Q
\356*\007\204f:
\362\366k\2363\033\314\177\226v\020\\KT\216\376\032\347\252\034L
\274\224G+\003GE\2673G\222B\031\022\010\022\006\010\001\022\002\
010\000\032\r\022\013\n\007Org1MSP\020\001" >
2018-03-21 05:46:43.166 UTC [msp/identity] Sign -> DEBU 009
Sign: plaintext:
0A91070A6708031A0C0895E3C7D50510...B547273E6CEACD8482A8F632F2681
093
2018-03-21 05:46:43.167 UTC [msp/identity] Sign -> DEBU 00a
Sign: digest:
```

```
D2F66B1F91AE4A6C138672975FBFC3D5D2A551419F74950B4129F5D4CA8B244D
2018-03-21 05:46:43.167 UTC [chaincodeCmd] upgrade -> DEBU 00b
Get Signed envelope
2018-03-21 05:46:43.167 UTC [chaincodeCmd] chaincodeUpgrade ->
DEBU 00c Send signed envelope to orderer
2018-03-21 05:46:43.179 UTC [main] main -> INFO 00d Exiting.....
```

2. Chaincode 结构

Chaincode 实现 shim.ChaincodeStubInterface 接口，有三个方法，分别是：Init、Query 和 Invoke

https://github.com/hyperledger-archives/fabric/blob/master/core/chaincode/shim/chaincode.go

2.1. 包

由于需要编译为可执行文件，所以需要 main 包

```
package main
```

2.2. 导入库

这里需要导入两个包 "github.com/hyperledger/fabric/core/chaincode/shim" 和 "github.com/hyperledger/fabric/protos/peer" 其他包，根据实际需要而定。

```
import (
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)
```

2.3. 定义类

```
type SimpleChaincode struct {
}
```

2.4. Init 方法

负责初始化工作，链码首次部署到区块链网络时调用，将由部署自己的链代码实例的每个对等节点执行。此方法可用于任何与初始化、引导或设置相关的任务。

```
func (t *SimpleChaincode) Init(stub
shim.ChaincodeStubInterface) pb.Response {
}
```

2.5. Query

只要在区块链状态上执行任何读取/获取/查询操作，就需要调用 Query 方法。如果尝试在 Query 方法内修改区块链的状态，将会抛出异常。

2.6. Invoke

此方法主要是做修改操作，但是很多例子中一些用户也在 Invoke 做查询。

put, get, del 等操作都在可以在 Invoke 中运行

```
func (t *SimpleChaincode) Invoke(stub
shim.ChaincodeStubInterface) pb.Response {
```

```
}
```

参考例子

```
func (s *SmartContract) Invoke(stub
shim.ChaincodeStubInterface) sc.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to
interact with the ledger appropriately
        if function == "balanceToken" {
                return s.balanceToken(stub, args)
        } else if function == "initLedger" {
                return s.initLedger(stub)
        } else if function == "transferToken" {
                return s.transferToken(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}
```

在 Invoke 函数中，首先使用 stub.GetFunctionAndParameters() 获取合约函数

```
function, args := stub.GetFunctionAndParameters()
```

然后判断函数名称，实现对应的逻辑关系。

```
if function == "balanceToken" {
return s.balanceToken(stub, args)
} else if function == "initLedger" {
return s.initLedger(stub)
```

```
} else if function == "transferToken" {
return s.transferToken(stub, args)
}
```

2.7. func main()

任何 Go 程序的都需要 main 函数，他是程序的入口，因此该函数被用于引导/启动链代码。当对peer节点部署chaincode并实例化时，就会执行 main 函数。

```
func main() {
        err := shim.Start(new(SimpleChaincode))
        if err != nil {
                fmt.Printf("Error starting Simple chaincode: %s", err)
        }
}
```

shim.Start(new(SampleChaincode)) 启动链代码并注册到peer 节点。

3. shim.ChaincodeStubInterface 接口

Hyperledger提供基于key/value的数据存储，其中key是字符串，value则是二进制字节数组，Hyperledger的Go API提供了三个方法用于数据存取：PutState（key, value）用于向Hyperledger中存储数据， GetState(key)用于从Hyperledger中提取数据，而DelState(key)则从Hyperledger中删除数据。

写入数据如果是 struct 结构体，需要序列化，通常使用 json,其他形式的序列化也可以，只要能反序列化即可。

3.1. State 数据库曾，删，查 操作

3.1.1. PutState（key, value）写入区块

写入区块联系

```
func (s *SmartContract) initLedger(stub shim.ChaincodeStubInterface) sc.Response
{

        token := &Token{
                Owner: "netkiller",
                TotalSupply: 10000,
                TokenName: "代币通正",
                TokenSymbol: "COIN",
                BalanceOf: map[string]uint{}}

        token.initialSupply()

        tokenAsBytes, _ := json.Marshal(token)
        stub.PutState("Token", tokenAsBytes)
        fmt.Println("Added", tokenAsBytes)

        return shim.Success(nil)
}
```

3.1.2. GetState(key) 读取区块

通过key获取区块信息

```
func (s *SmartContract) balanceToken(stub shim.ChaincodeStubInterface, args
[]string) sc.Response {

        if len(args) != 1 {
```

```
                return shim.Error("Incorrect number of arguments. Expecting 1")
        }

        tokenAsBytes, _ := stub.GetState(args[0])
        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)
        amount := token.balance(args[1])

        return shim.Success(amount)
}
```

3.1.3. DelState(key) 删除区块

删除区块信息

```
func (s *SmartContract) deleteData(stub shim.ChaincodeStubInterface, args
[]string) sc.Response {
        if len(args) != 1 {
                return shim.Error("Incorrect number of arguments. Expecting 1")
        }
        err= stub.DelState(args[0])
        if err != nil {
           return shim.Error("Failed to delete Student from DB, key is: "+key)
        }
        return shim.Success(nil)
}
```

3.1.4. 修改数据

State 数据库并没有提供修改功能，修改数据可以先读取，再修改，最后写入

```
func (s *SmartContract) transferToken(stub shim.ChaincodeStubInterface, args
[]string) sc.Response {

        if len(args) != 3 {
                return shim.Error("Incorrect number of arguments. Expecting 2")
        }

        tokenAsBytes, _ := stub.GetState(args[0])
        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)
        token.transfer(args[1],args[2],args[3])

        tokenAsBytes, _ = json.Marshal(token)
        stub.PutState(args[0], tokenAsBytes)
```

```
        return shim.Success(nil)
}
```

3.1.5. GetStateByRange(startKey, endKey) 范围查找

区块链是一个线性的数据结构，只要知道开始位置，结束位置，就能够取出中间部分的数据。

```
func (s *SmartContract) queryAllCars(APIstub shim.ChaincodeStubInterface)
sc.Response {

        startKey := "CAR0"
        endKey := "CAR999"

        resultsIterator, err := APIstub.GetStateByRange(startKey, endKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        defer resultsIterator.Close()

        // buffer is a JSON array containing QueryResults
        var buffer bytes.Buffer
        buffer.WriteString("[")

        bArrayMemberAlreadyWritten := false
        for resultsIterator.HasNext() {
                queryResponse, err := resultsIterator.Next()
                if err != nil {
                        return shim.Error(err.Error())
                }
                // Add a comma before array members, suppress it for the first
array member
                if bArrayMemberAlreadyWritten == true {
                        buffer.WriteString(",")
                }
                buffer.WriteString("{\"Key\":")
                buffer.WriteString("\"")
                buffer.WriteString(queryResponse.Key)
                buffer.WriteString("\"")

                buffer.WriteString(", \"Record\":")
                // Record is a JSON object, so we write as-is
                buffer.WriteString(string(queryResponse.Value))
                buffer.WriteString("}")
                bArrayMemberAlreadyWritten = true
        }
        buffer.WriteString("]")

        fmt.Printf("- queryAllCars:\n%s\n", buffer.String())

        return shim.Success(buffer.Bytes())
}
```

3.1.6. GetQueryResult(query string) CouchDB 查询

GetQueryResult 能查询 json 里面的数据

下面例子是 Name = Neo Chen 的所有数据。

```
func (t *SimpleChaincode) getQueryResult(stub shim.ChaincodeStubInterface, args
[]string) pb.Response{
   name:="Neo Chen"     //需要查询的名字
   queryString := fmt.Sprintf("{\"selector\":{\"Name\":\"%s\"}}", name)
   resultsIterator,err:= stub.GetQueryResult(queryString)//必须是CouchDB才行
   if err!=nil{
      return shim.Error("query failed")
   }
   person,err:=getListResult(resultsIterator)
   if err!=nil{
      return shim.Error("query failed")
   }
   return shim.Success(person)
}
```

3.1.7. stub.GetHistoryForKey

通过key获取历史数据

```
func (t *SimpleChaincode) historyQuery(stub shim.ChaincodeStubInterface, args
[]string) pb.Response{

        if len(args) != 1 {
                return shim.Error("Incorrect number of arguments. Expecting 1")
        }

   it,err:= stub.GetHistoryForKey(args[0])
   if err!=nil{
      return shim.Error(err.Error())
   }
   var result,_= getHistoryListResult(it)
   return shim.Success(result)
}
```

3.1.8. shim.HistoryQueryIteratorInterface 接口

```
func getHistoryListResult(resultsIterator shim.HistoryQueryIteratorInterface)
([]byte,error){

   defer resultsIterator.Close()
   // buffer is a JSON array containing QueryRecords
   var buffer bytes.Buffer
   buffer.WriteString("[")

   bArrayMemberAlreadyWritten := false
   for resultsIterator.HasNext() {
      queryResponse, err := resultsIterator.Next()
      if err != nil {
         return nil, err
      }
      // Add a comma before array members, suppress it for the first array
member
      if bArrayMemberAlreadyWritten == true {
         buffer.WriteString(",")
      }
      item,_:= json.Marshal( queryResponse)
      buffer.Write(item)
      bArrayMemberAlreadyWritten = true
   }
   buffer.WriteString("]")
   fmt.Printf("queryResult:\n%s\n", buffer.String())
   return buffer.Bytes(), nil
}
```

3.2. 复合键

3.2.1. 创建复合键

```
        // maintain the index
        indexName := "color~name"
        colorNameIndexKey, err := stub.CreateCompositeKey(indexName,
[]string{marbleJSON.Color, marbleJSON.Name})
        if err != nil {
                return shim.Error(err.Error())
        }

        //  Delete index entry to state.
        err = stub.DelState(colorNameIndexKey)
        if err != nil {
                return shim.Error("Failed to delete state:" + err.Error())
        }
```

3.2.2. 分解复合键

```
                // get the color and name from color~name composite key
                objectType, compositeKeyParts, err :=
stub.SplitCompositeKey(responseRange.Key)
                if err != nil {
                        return shim.Error(err.Error())
                }
                returnedColor := compositeKeyParts[0]
                returnedMarbleName := compositeKeyParts[1]
```

3.3. stub.SetEvent(key, value) 事件

Hyperledger Fabic 事件实现了发布/订阅消息队列。您可以自由地在链码中创建和发出自定义事件。例如，区块链的状态发生改变，就会生成一个事件。通过向区块链上的事件中心注册一个事件适配器，客户端应用程序可以订阅和使用这些事件。

```
func (t *SimpleChaincode) testEvent(stub shim.ChaincodeStubInterface, args
[]string) pb.Response{
   message := "Event send data is here!"
   err := stub.SetEvent("evtsender", []byte(message))
   if err != nil {
      return shim.Error(err.Error())
   }
   return shim.Success(nil)
}
```

```
func (t *SimpleChaincode) testEvent(stub shim.ChaincodeStubInterface, args
[]string) pb.Response{
        event := &Token{
                Owner: "netkiller",
                TotalSupply: 10000,
                TokenName: "代币通正",
                TokenSymbol: "COIN",
                BalanceOf: map[string]uint{}}

    eventBytes, err ;= json.Marshal(&event)
    if err != nil {
            return nil, err
    }
    err := stub.SetEvent("evtSender", eventBytes)
    if err != nil {
        fmt.Println("Could not set event for loan application creation", err)
    }
}
```

3.4. 调用其他链码

在当前连码中调用另一个连码，调用连码需要提供连码名和通道名
stub.InvokeChaincode("连码名",调用函数,"通道")

```
func (t *SimpleChaincode) testInvokeChainCode(stub shim.ChaincodeStubInterface,
args []string) pb.Response{
   trans:=[][]byte{[]byte("invoke"),[]byte("transfer"),[]byte("netkiller"),
[]byte("neo"),[]byte("100")}
   response:= stub.InvokeChaincode("token",trans,"mychannel")
   fmt.Println(response.Message)
   return shim.Success([]byte( response.Message))
}
```

```
parms1 := []string{"query","a"}
queryArgs := make([][]byte, len(parms1))
for i, arg := range parms1 {
   queryArgs[i] = []byte(arg)
}


response :=
stub.InvokeChaincode("cc_endfinlshed",queryArgs,"roberttestchannel12")

if response.Status != shim.OK {
   errStr := fmt.Sprintf("Failed to query chaincode. Got error: %s",
response.Payload)
   fmt.Printf(errStr)
return shim.Error(errStr)
}

result := string(response.Payload)



fmt.Printf(" invoke chaincode  %s " ,result)

return shim.Success([]byte("success InvokeChaincode  and Not opter !!!!!!!! " +
result))
```

```
import (

        "encoding/json"
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/common/util"
        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"

)
```

```
// Invoke
func (am *accountManagement) Invoke(stub shim.ChaincodeStubInterface)
peer.Response {
    actionName, params := stub.GetFunctionAndParameters()

    if actionName == "callAnotherCC" {
        chainCodeArgs := util.ToChaincodeArgs("anotherCCFunc", "paramA")
        response := stub.InvokeChaincode("anotherCCName", chainCodeArgs,
"channelName")

        if response.Status != shim.OK {
           return shim.Error(response.Message)
        }
        return shim.Success(nil)
    }

    // NOTE: This is an example, hence assuming only valid call is to call
another chaincode
    return shim.Error(fmt.Sprintf("[ERROR] No <%s> action defined", actionName))
}
```

3.5. stub.GetCreator() 获得证书资料

```
func (t *SimpleChaincode) certificate(stub shim.ChaincodeStubInterface, args
[]string) pb.Response{
   creatorByte,_:= stub.GetCreator()
   certStart := bytes.IndexAny(creatorByte, "-----BEGIN")
   if certStart == -1 {
      fmt.Errorf("No certificate found")
   }
   certText := creatorByte[certStart:]
   bl, _ := pem.Decode(certText)
   if bl == nil {
      fmt.Errorf("Could not decode the PEM structure")
   }

   cert, err := x509.ParseCertificate(bl.Bytes)
   if err != nil {
      fmt.Errorf("ParseCertificate failed")
   }
   uname:=cert.Subject.CommonName
   fmt.Println("Name:"+uname)
   return shim.Success([]byte("Called testCertificate "+uname))
}
```

4. 链码案例

4.1. 模仿以太坊 ERC20 规范的 Hyperledger Fabric 实现 Token 通证

借用以太坊思维，将以太坊代币合约搬到 hyperledger 上，一样可以实现代币的功能，这个代币除了不能上交易所，基本满足我们替代积分系统的需求，下面是我写了这样一个合约，在超级账本上实现类似以太坊的代币转账功能。

合约实现代币转账，额度查询，增发代币，冻结账号，锁仓等等服务器，功能与 ERC20 Token 相仿。

合约实例化所有代币打入了 coinbase 账号，分发代币需要使用转账功能从 coinbase 想普通账号转账

普通账号消费可以在将代币转到 coinbase 账号中，这样就完成了代币流通，形成一个闭环。

```
package main

import (
        "encoding/json"
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        sc "github.com/hyperledger/fabric/protos/peer"
)

// Define the Smart Contract structure
type SmartContract struct {
}

type Token struct {
        Owner           string  `json:"Owner"`
```

```
        TotalSupply     int     `json:"TotalSupply"`
        TokenName       string  `json:"TokenName"`
        TokenSymbol     string  `json:"TokenSymbol"`
        BalanceOf       map[string]int  `json:"BalanceOf"`
}

func (token *Token) initialSupply(){
        token.BalanceOf[token.Owner] = token.TotalSupply;
}

func (token *Token) transfer (_from string, _to string, _value
int){
        if(token.BalanceOf[_from] >= _value){
                token.BalanceOf[_from] -= _value;
                token.BalanceOf[_to] += _value;
        }
}

func (token *Token) balance (_from string) int{
        return token.BalanceOf[_from]
}

func (token *Token) burn(_value int) {
        if(token.BalanceOf[token.Owner] >= _value){
                token.BalanceOf[token.Owner] -= _value;
                token.TotalSupply -= _value;
        }
}

func (token *Token) burnFrom(_from string, _value int) {
        if(token.BalanceOf[_from] >= _value){
                token.BalanceOf[_from] -= _value;
                token.TotalSupply -= _value;
        }
}

func (token *Token) mint(_value int) {

        token.BalanceOf[token.Owner] += _value;
        token.TotalSupply += _value;

}

func (s *SmartContract) Init(stub shim.ChaincodeStubInterface)
sc.Response {
        return shim.Success(nil)
}
```

```
func (s *SmartContract) initLedger(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 3 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        symbol:= args[0]
        name  := args[1]
        supply,_:= strconv.Atoi(args[2])

        token := &Token{
                Owner: "coinbase",
                TotalSupply: supply,
                TokenName: name,
                TokenSymbol: symbol,
                BalanceOf: map[string]int{}}

        token.initialSupply()

        tokenAsBytes, _ := json.Marshal(token)
        err := stub.PutState(symbol, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Init %s \n", string(tokenAsBytes))

        return shim.Success(nil)
}

func (s *SmartContract) transferToken(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 4 {
                return shim.Error("Incorrect number of
arguments. Expecting 4")
        }
        _from   := args[1]
        _to     := args[2]
        _amount,_       := strconv.Atoi(args[3])
        if(_amount <= 0){
                return shim.Error("Incorrect number of amount")
        }

        tokenAsBytes,err := stub.GetState(args[0])
        if err != nil {
                return shim.Error(err.Error())
```

```
        }
        fmt.Printf("transferToken - begin %s \n",
string(tokenAsBytes))

        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)
        token.transfer(_from, _to, _amount)

        tokenAsBytes, err = json.Marshal(token)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(args[0], tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("transferToken - end %s \n",
string(tokenAsBytes))

        return shim.Success(nil)
}

func (s *SmartContract) balanceToken(stub
shim.ChaincodeStubInterface, args []string) sc.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        tokenAsBytes,err := stub.GetState(args[0])
        if err != nil {
                return shim.Error(err.Error())
        }
        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)
        amount := token.balance(args[1])
        value := strconv.Itoa(amount)
        fmt.Printf("%s balance is %s \n", args[1], value)
        //jsonVal, _ := json.Marshal(string(value))

        return shim.Success([]byte(value))
}

func (s *SmartContract) Invoke(stub shim.ChaincodeStubInterface)
sc.Response {
```

```
        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to interact
with the ledger appropriately
        if function == "balanceToken" {
                return s.balanceToken(stub, args)
        } else if function == "initLedger" {
                return s.initLedger(stub, args)
        } else if function == "transferToken" {
                return s.transferToken(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}

// The main function is only relevant in unit test mode. Only
included here for completeness.
func main() {

        // Create a new Smart Contract
        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
}
```

这个合约用户可以创建多套代币，Args":["Token" 的第一参数 Token就是代币名称

```
peer chaincode invoke -C myc -n token -c
'{"function":"initLedger","Args":["Apple","水果币","1000000"]}'
peer chaincode invoke -C myc -n token -c
'{"function":"initLedger","Args":["Token","蔬菜币","1000000"]}'
peer chaincode invoke -C myc -n token -c
'{"function":"initLedger","Args":["Oil","粮油币","1000000"]}'
```

这个方案仍有不足之处，作者还不清楚如果用户上线是多少，达到临界值后，Hyperledger Fabric 无法在提供服务。

可能 chaincode_example02 做法更靠谱，就是不用 map 保存数据，将每个用户存储在 State 数据上。这里需要创建多套代币，所以使用了一个 key 来存储所有账号。如果像 chaincode_example02 那样就需要部署多个 chaincode 在 channel 中。管理起来比较复杂。

```
[root@localhost fabric-samples]# cd chaincode-docker-devmode/
[root@localhost chaincode-docker-devmode]# docker-compose up -d
[root@localhost chaincode-docker-devmode]# docker exec -it cli
bash
root@765cbcd51fd7:/opt/gopath/src/chaincode#

CORE_PEER_ADDRESS=peer:7051 CORE_CHAINCODE_ID_NAME=token:1.0
./chaincode/chaincode/token/token

peer chaincode install -n token -v 1.0 -p
chaincodedev/chaincode/chaincode/token
peer chaincode instantiate -C myc -n token -v 1.0 -c '{"Args":
[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
peer chaincode instantiate -C myc -n token -v 1.0 -c '{"Args":
["init"]}'
sleep 10
peer chaincode invoke -C myc -n token -c
'{"function":"initLedger","Args":["Token","Netkiller
Coin","1000000"]}'

peer chaincode invoke -C myc -n token -c
'{"function":"transferToken","Args":
["Token","coinbase","netkiller","100"]}'
peer chaincode invoke -C myc -n token -c
'{"function":"balanceToken","Args":["Token","netkiller"]}'

peer chaincode invoke -C myc -n token -c
'{"function":"transferToken","Args":
["Token","netkiller","jerry","100"]}'

peer chaincode query -C myc -n token -c
'{"function":"balanceToken","Args":["Token","netkiller"]}'
```

测试不存在账号转账

```
peer chaincode invoke -C myc -n token -c
'{"function":"transferToken","Args":
["Token","neo","netkiller","100"]}'
```

以太坊和超级账本各有优势，虽然超级账本的Token功能无法和以太坊相比，但是使用超级账本实现的Token交易不用矿工费。同事超级账本还有一个优势，就是可以在合约中调用另一个合约，这样一来可以做出很多复杂的需求

例如我们在订票的合约中，就可以直接从Token合约中直接扣款。

4.2. 万能的通用合约

我们一般会在合约中定义结构体，然后序列化后存入 state 数据库中。一旦数据结构变化，就需要升级 chaincode。

下面我们只实现了 create, find, update, delete 四个方法，没有数据结构，用户自行提交 json 格式或者其他序列化后的字符串数据。

```
package main

import (
        "fmt"
        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)

type SmartContract struct {}

func (s *SmartContract) Init(stub shim.ChaincodeStubInterface)
pb.Response {
        return shim.Success(nil)
}
```

```
func (s *SmartContract) Query(stub shim.ChaincodeStubInterface)
pb.Response {
        return shim.Success(nil)
}

func (s *SmartContract) Invoke(stub shim.ChaincodeStubInterface)
pb.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to interact
with the ledger appropriately
        if function == "create" {
                return s.create(stub, args)
        } else if function == "find" {
                return s.find(stub, args)
        } else if function == "update" {
                return s.update(stub, args)
        } else if function == "delete" {
                return s.delete(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}

func (s *SmartContract) create(stub shim.ChaincodeStubInterface,
args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _key    := args[0]
        _data   := args[1]

        if(_data == ""){
                return shim.Error("Incorrect string of data")
        }

        existAsBytes,err := stub.GetState(_key)
        if string(existAsBytes) != "" {
                fmt.Println("Failed to create account, Duplicate
key.")
                return shim.Error("Failed to create account,
```

```
Duplicate key.")
        }

        err = stub.PutState(_key, []byte(_data))
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("create %s %s \n", _key, string(_data))

        return shim.Success([]byte(_data))
}

func (s *SmartContract) find(stub shim.ChaincodeStubInterface,
args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        _key    := args[0]
        _data, err := stub.GetState(_key)
        if err != nil {
                return shim.Error(err.Error())
        }
        if string(_data) == "" {
                return shim.Error("The key isn't exist.")
        }else{
                fmt.Printf("query %s %s \n", _key,
string(_data))
        }

        return shim.Success(_data)
}

func (s *SmartContract) update(stub shim.ChaincodeStubInterface,
args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _key    := args[0]
        _data   := args[1]

        if(_data == ""){
                return shim.Error("Incorrect string of data")
```

```
        }

        err := stub.PutState(_key, []byte(_data))
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("update %s %s \n", _key,
string(_data))
        }

        return shim.Success([]byte(_data))
}

// Deletes an entity from state
func (t *SmartContract) delete(stub shim.ChaincodeStubInterface,
args []string) pb.Response {
        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        _key := args[0]

        // Delete the key from the state in ledger
        err := stub.DelState(_key)
        if err != nil {
                return shim.Error("Failed to delete state")
        }

        return shim.Success(nil)
}

func main() {

        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
}
```

测试环境 Hyperledger Fabric v1.2.0

```
[root@localhost ~]# docker exec -it cli bash

root@8cb899359aec:/opt/gopath/src/chaincodedev# peer chaincode
install -v 1.0 -n art -p chaincodedev/chaincode/art
root@8cb899359aec:/opt/gopath/src/chaincodedev# peer chaincode
instantiate -C myc -n art -v 1.0 -c '{"Args":["init",""]}'

root@8cb899359aec:/opt/gopath/src/chaincodedev# peer chaincode
instantiate -C myc -n art -v 1.0 -c '{"Args":[]}'
```

操作演示

```
root@8cb899359aec:/opt/gopath/src/chaincodedev# peer chaincode
install -v 1.0 -n art -p chaincodedev/chaincode/art
2018-07-20 02:39:48.091 UTC [viperutil] getKeysRecursively ->
DEBU 001 Found map[string]interface{} value for peer.BCCSP
2018-07-20 02:39:48.092 UTC [viperutil] unmarshalJSON -> DEBU
002 Unmarshal JSON: value cannot be unmarshalled: invalid
character 'S' looking for beginning of value
2018-07-20 02:39:48.092 UTC [viperutil] getKeysRecursively ->
DEBU 003 Found real value for peer.BCCSP.Default setting to
string SW
2018-07-20 02:39:48.093 UTC [viperutil] getKeysRecursively ->
DEBU 004 Found map[string]interface{} value for peer.BCCSP.SW
2018-07-20 02:39:48.093 UTC [viperutil] unmarshalJSON -> DEBU
005 Unmarshal JSON: value cannot be unmarshalled: invalid
character 'S' looking for beginning of value
2018-07-20 02:39:48.093 UTC [viperutil] getKeysRecursively ->
DEBU 006 Found real value for peer.BCCSP.SW.Hash setting to
string SHA2
2018-07-20 02:39:48.094 UTC [viperutil] unmarshalJSON -> DEBU
007 Unmarshal JSON: value is not a string: 256
2018-07-20 02:39:48.094 UTC [viperutil] getKeysRecursively ->
DEBU 008 Found real value for peer.BCCSP.SW.Security setting to
int 256
2018-07-20 02:39:48.095 UTC [viperutil] getKeysRecursively ->
DEBU 009 Found map[string]interface{} value for
peer.BCCSP.SW.FileKeyStore
2018-07-20 02:39:48.095 UTC [viperutil] unmarshalJSON -> DEBU
00a Unmarshal JSON: value cannot be unmarshalled: unexpected end
of JSON input
2018-07-20 02:39:48.095 UTC [viperutil] getKeysRecursively ->
DEBU 00b Found real value for
```

```
peer.BCCSP.SW.FileKeyStore.KeyStore setting to string
2018-07-20 02:39:48.096 UTC [viperutil] getKeysRecursively ->
DEBU 00c Found map[string]interface{} value for
peer.BCCSP.PKCS11
2018-07-20 02:39:48.096 UTC [viperutil] getKeysRecursively ->
DEBU 00d Found map[string]interface{} value for
peer.BCCSP.PKCS11.FileKeyStore
2018-07-20 02:39:48.097 UTC [viperutil] unmarshalJSON -> DEBU
00e Unmarshal JSON: value is not a string: <nil>
2018-07-20 02:39:48.098 UTC [viperutil] getKeysRecursively ->
DEBU 00f Found real value for
peer.BCCSP.PKCS11.FileKeyStore.KeyStore setting to <nil> <nil>
2018-07-20 02:39:48.098 UTC [viperutil] unmarshalJSON -> DEBU
010 Unmarshal JSON: value is not a string: <nil>
2018-07-20 02:39:48.099 UTC [viperutil] getKeysRecursively ->
DEBU 011 Found real value for peer.BCCSP.PKCS11.Library setting
to <nil> <nil>
2018-07-20 02:39:48.099 UTC [viperutil] unmarshalJSON -> DEBU
012 Unmarshal JSON: value is not a string: <nil>
2018-07-20 02:39:48.100 UTC [viperutil] getKeysRecursively ->
DEBU 013 Found real value for peer.BCCSP.PKCS11.Label setting to
<nil> <nil>
2018-07-20 02:39:48.101 UTC [viperutil] unmarshalJSON -> DEBU
014 Unmarshal JSON: value is not a string: <nil>
2018-07-20 02:39:48.101 UTC [viperutil] getKeysRecursively ->
DEBU 015 Found real value for peer.BCCSP.PKCS11.Pin setting to
<nil> <nil>
2018-07-20 02:39:48.102 UTC [viperutil] unmarshalJSON -> DEBU
016 Unmarshal JSON: value is not a string: <nil>
2018-07-20 02:39:48.103 UTC [viperutil] getKeysRecursively ->
DEBU 017 Found real value for peer.BCCSP.PKCS11.Hash setting to
<nil> <nil>
2018-07-20 02:39:48.103 UTC [viperutil] unmarshalJSON -> DEBU
018 Unmarshal JSON: value is not a string: <nil>
2018-07-20 02:39:48.104 UTC [viperutil] getKeysRecursively ->
DEBU 019 Found real value for peer.BCCSP.PKCS11.Security setting
to <nil> <nil>
2018-07-20 02:39:48.104 UTC [viperutil]
EnhancedExactUnmarshalKey -> DEBU 01a
map[peer.BCCSP:map[Default:SW SW:map[Security:256
FileKeyStore:map[KeyStore:] Hash:SHA2] PKCS11:map[Pin:<nil>
Hash:<nil> Security:<nil> FileKeyStore:map[KeyStore:<nil>]
Library:<nil> Label:<nil>]]]
2018-07-20 02:39:48.105 UTC [bccsp_sw] openKeyStore -> DEBU 01b
KeyStore opened at [/etc/hyperledger/msp/keystore]...done
2018-07-20 02:39:48.105 UTC [bccsp] initBCCSP -> DEBU 01c
Initialize BCCSP [SW]
2018-07-20 02:39:48.106 UTC [msp] getPemMaterialFromDir -> DEBU
```

```
01d Reading directory /etc/hyperledger/msp/signcerts
2018-07-20 02:39:48.106 UTC [msp] getPemMaterialFromDir -> DEBU
01e Inspecting file /etc/hyperledger/msp/signcerts/peer.pem
2018-07-20 02:39:48.106 UTC [msp] getPemMaterialFromDir -> DEBU
01f Reading directory /etc/hyperledger/msp/cacerts
2018-07-20 02:39:48.106 UTC [msp] getPemMaterialFromDir -> DEBU
020 Inspecting file /etc/hyperledger/msp/cacerts/cacert.pem
2018-07-20 02:39:48.107 UTC [msp] getPemMaterialFromDir -> DEBU
021 Reading directory /etc/hyperledger/msp/admincerts
2018-07-20 02:39:48.107 UTC [msp] getPemMaterialFromDir -> DEBU
022 Inspecting file
/etc/hyperledger/msp/admincerts/admincert.pem
2018-07-20 02:39:48.107 UTC [msp] getPemMaterialFromDir -> DEBU
023 Reading directory /etc/hyperledger/msp/intermediatecerts
2018-07-20 02:39:48.107 UTC [msp] getMspConfig -> DEBU 024
Intermediate certs folder not found at
[/etc/hyperledger/msp/intermediatecerts]. Skipping. [stat
/etc/hyperledger/msp/intermediatecerts: no such file or
directory]
2018-07-20 02:39:48.107 UTC [msp] getPemMaterialFromDir -> DEBU
025 Reading directory /etc/hyperledger/msp/tlscacerts
2018-07-20 02:39:48.107 UTC [msp] getPemMaterialFromDir -> DEBU
026 Inspecting file /etc/hyperledger/msp/tlscacerts/tlsroot.pem
2018-07-20 02:39:48.107 UTC [msp] getPemMaterialFromDir -> DEBU
027 Reading directory /etc/hyperledger/msp/tlsintermediatecerts
2018-07-20 02:39:48.108 UTC [msp] getPemMaterialFromDir -> DEBU
028 Inspecting file
/etc/hyperledger/msp/tlsintermediatecerts/tlsintermediate.pem
2018-07-20 02:39:48.108 UTC [msp] getPemMaterialFromDir -> DEBU
029 Reading directory /etc/hyperledger/msp/crls
2018-07-20 02:39:48.108 UTC [msp] getMspConfig -> DEBU 02a crls
folder not found at [/etc/hyperledger/msp/crls]. Skipping. [stat
/etc/hyperledger/msp/crls: no such file or directory]
2018-07-20 02:39:48.108 UTC [msp] getMspConfig -> DEBU 02b MSP
configuration file not found at
[/etc/hyperledger/msp/config.yaml]: [stat
/etc/hyperledger/msp/config.yaml: no such file or directory]
2018-07-20 02:39:48.109 UTC [msp] newBccspMsp -> DEBU 02c
Creating BCCSP-based MSP instance
2018-07-20 02:39:48.109 UTC [msp] New -> DEBU 02d Creating
Cache-MSP instance
2018-07-20 02:39:48.109 UTC [msp] loadLocaMSP -> DEBU 02e
Created new local MSP
2018-07-20 02:39:48.110 UTC [msp] Setup -> DEBU 02f Setting up
MSP instance DEFAULT
2018-07-20 02:39:48.111 UTC [msp/identity] newIdentity -> DEBU
030 Creating identity instance for cert -----BEGIN CERTIFICATE--
---
```

```
MIICYjCCAgigAwIBAgIRAL1fEAnz5zp4moJ8MdSb/lYwCgYIKoZIzj0EAwIwgYEx
CzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g
RnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMQwwCgYDVQQLEwND
T1AxHDAaBgNVBAMTE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcxMTEyMTM0MTEx
WhcNMjcxMTEwMTM0MTExWjCBgTELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlm
b3JuaWExFjAUBgNVBAcTDVNhbiBGcmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhh
bXBsZS5jb20xDDAKBgNVBAsTA0NPUDEcMBoGA1UEAxMTY2Eub3JnMS5leGFtcGxl
LmNvbTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABGrsQ6oJpk6hDWf63HU3OSNd
bou9KNw/VIee1IngPDI4YJU7O+Xa/XLJuwnFv7BpR8Ytl3f+njC8i/RZP2/svO+j
XzBdMA4GA1UdDwEB/wQEAwIBpjAPBgNVHSUECDAGBgRVHSUAMA8GA1UdEwEB/wQF
MAMBAf8wKQYDVR0OBCIEIIpzkSIZzxBWVIV5unlgZJuyu2XPEeP8+y1uB6LLA5Qr
MAoGCCqGSM49BAMCA0gAMEUCIQDUh/+CC2dAICnYtACXspwUaaEbiyZxYIx+XDvW
o8VVcgIgGz5S4iC5+xkxgeaISPfxKTTVy6yzTdYGzCw1vPppjzo=
-----END CERTIFICATE-----
2018-07-20 02:39:48.112 UTC [msp/identity] newIdentity -> DEBU
031 Creating identity instance for cert -----BEGIN CERTIFICATE--
---
MIICNjCCAd2gAwIBAgIRAMnf9/dmV9RvCCVw9pZQUfUwCgYIKoZIzj0EAwIwgYEx
CzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g
RnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMQwwCgYDVQQLEwND
T1AxHDAaBgNVBAMTE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcxMTEyMTM0MTEx
WhcNMjcxMTEwMTM0MTExWjBpMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv
cm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEMMAoGA1UECxMDQ09QMR8wHQYD
VQQDExZwZWVyMC5vcmcxLmV4YW1wbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0D
AQcDQgAEZ8S4V71OBJpyMIVZdwYdFXAckItrpvSrCf0HQg40WW9XSoOOO76I+Umf
EkmTlIJXP7/AyRRSRU38oI8Ivtu4M6NNMEswDgYDVR0PAQH/BAQDAgeAMAwGA1Ud
EwEB/wQCMAAwKwYDVR0jBCQwIoAginORIhnPEFZUhXm6eWBkm7K7Zc8R4/z7LW4H
ossDlCswCgYIKoZIzj0EAwIDRwAwRAIgVikIUZzgfuFsGLQHWJUVJCU7pDaETkaz
PzFgsCiLxUACICgzJYlW7nvZxP7b6tbeu3t8mrhMXQs956mD4+BoKuNI
-----END CERTIFICATE-----
2018-07-20 02:39:48.181 UTC [msp/identity] newIdentity -> DEBU
032 Creating identity instance for cert -----BEGIN CERTIFICATE--
---
MIICNjCCAd2gAwIBAgIRAMnf9/dmV9RvCCVw9pZQUfUwCgYIKoZIzj0EAwIwgYEx
CzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g
RnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMQwwCgYDVQQLEwND
T1AxHDAaBgNVBAMTE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcxMTEyMTM0MTEx
WhcNMjcxMTEwMTM0MTExWjBpMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv
cm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEMMAoGA1UECxMDQ09QMR8wHQYD
VQQDExZwZWVyMC5vcmcxLmV4YW1wbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0D
AQcDQgAEZ8S4V71OBJpyMIVZdwYdFXAckItrpvSrCf0HQg40WW9XSoOOO76I+Umf
EkmTlIJXP7/AyRRSRU38oI8Ivtu4M6NNMEswDgYDVR0PAQH/BAQDAgeAMAwGA1Ud
EwEB/wQCMAAwKwYDVR0jBCQwIoAginORIhnPEFZUhXm6eWBkm7K7Zc8R4/z7LW4H
ossDlCswCgYIKoZIzj0EAwIDRwAwRAIgVikIUZzgfuFsGLQHWJUVJCU7pDaETkaz
PzFgsCiLxUACICgzJYlW7nvZxP7b6tbeu3t8mrhMXQs956mD4+BoKuNI
-----END CERTIFICATE-----
2018-07-20 02:39:48.183 UTC [msp/identity] newIdentity -> DEBU
033 Creating identity instance for cert -----BEGIN CERTIFICATE--
```

```
---
MIICNjCCAd2gAwIBAgIRAMnf9/dmV9RvCCVw9pZQUfUwCgYIKoZIzj0EAwIwgYEx
CzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1TYW4g
RnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMQwwCgYDVQQLEwND
T1AxHDAaBgNVBAMTE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTcxMTEyMTM0MTEx
WhcNMjcxMTEwMTM0MTExWjBpMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv
cm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEMMAoGA1UECxMDQ09QMR8wHQYD
VQQDExZwZWVyMC5vcmcxLmV4YW1wbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0D
AQcDQgAEZ8S4V71OBJpyMIVZdwYdFXAckItrpvSrCf0HQg40WW9XSoOOO76I+Umf
EkmTlIJXP7/AyRRSRU38oI8Ivtu4M6NNMEswDgYDVR0PAQH/BAQDAgeAMAwGA1Ud
EwEB/wQCMAAwKwYDVR0jBCQwIoAginORIhnPEFZUhXm6eWBkm7K7Zc8R4/z7LW4H
ossDlCswCgYIKoZIzj0EAwIDRwAwRAIgVikIUZzgfuFsGLQHWJUVJCU7pDaETkaz
PzFgsCiLxUACICgzJYlW7nvZxP7b6tbeu3t8mrhMXQs956mD4+BoKuNI
-----END CERTIFICATE-----
2018-07-20 02:39:48.183 UTC [msp] setupSigningIdentity -> DEBU
034 Signing identity expires at 2027-11-10 13:41:11 +0000 UTC
2018-07-20 02:39:48.185 UTC [msp] Validate -> DEBU 035 MSP
DEFAULT validating identity
2018-07-20 02:39:48.187 UTC [grpc] Printf -> DEBU 036 parsed
scheme: ""
2018-07-20 02:39:48.188 UTC [grpc] Printf -> DEBU 037 scheme ""
not registered, fallback to default scheme
2018-07-20 02:39:48.188 UTC [grpc] Printf -> DEBU 038
ccResolverWrapper: sending new addresses to cc: [{peer:7051 0
<nil>}]
2018-07-20 02:39:48.188 UTC [grpc] Printf -> DEBU 039 ClientConn
switching balancer to "pick_first"
2018-07-20 02:39:48.188 UTC [grpc] Printf -> DEBU 03a
pickfirstBalancer: HandleSubConnStateChange: 0xc4203d7c40,
CONNECTING
2018-07-20 02:39:48.190 UTC [grpc] Printf -> DEBU 03b
pickfirstBalancer: HandleSubConnStateChange: 0xc4203d7c40, READY
2018-07-20 02:39:48.192 UTC [grpc] Printf -> DEBU 03c parsed
scheme: ""
2018-07-20 02:39:48.192 UTC [grpc] Printf -> DEBU 03d scheme ""
not registered, fallback to default scheme
2018-07-20 02:39:48.192 UTC [grpc] Printf -> DEBU 03e
ccResolverWrapper: sending new addresses to cc: [{peer:7051 0
<nil>}]
2018-07-20 02:39:48.192 UTC [grpc] Printf -> DEBU 03f ClientConn
switching balancer to "pick_first"
2018-07-20 02:39:48.192 UTC [grpc] Printf -> DEBU 040
pickfirstBalancer: HandleSubConnStateChange: 0xc420452130,
CONNECTING
2018-07-20 02:39:48.194 UTC [grpc] Printf -> DEBU 041
pickfirstBalancer: HandleSubConnStateChange: 0xc420452130, READY
2018-07-20 02:39:48.197 UTC [msp] GetDefaultSigningIdentity ->
DEBU 042 Obtaining default signing identity
```

```
2018-07-20 02:39:48.197 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 043 Using default escc
2018-07-20 02:39:48.198 UTC [chaincodeCmd]
checkChaincodeCmdParams -> INFO 044 Using default vscc
2018-07-20 02:39:48.198 UTC [chaincodeCmd] getChaincodeSpec ->
DEBU 045 java chaincode disabled
2018-07-20 02:39:48.266 UTC [golang-platform] getCodeFromFS ->
DEBU 046 getCodeFromFS chaincodedev/chaincode/art
2018-07-20 02:39:49.372 UTC [golang-platform] func1 -> DEBU 047
Discarding GOROOT package fmt
2018-07-20 02:39:49.372 UTC [golang-platform] func1 -> DEBU 048
Discarding provided package
github.com/hyperledger/fabric/core/chaincode/shim
2018-07-20 02:39:49.372 UTC [golang-platform] func1 -> DEBU 049
Discarding provided package
github.com/hyperledger/fabric/protos/peer
2018-07-20 02:39:49.373 UTC [golang-platform]
GetDeploymentPayload -> DEBU 04a done
2018-07-20 02:39:49.373 UTC [container] WriteFileToPackage ->
DEBU 04b Writing file to tarball:
src/chaincodedev/chaincode/art/art.go
2018-07-20 02:39:49.378 UTC [msp/identity] Sign -> DEBU 04c
Sign: plaintext:
0AC4070A5C08031A0C08F596C5DA0510...97DB3F010000FFFFD8DB7F9600140
000
2018-07-20 02:39:49.379 UTC [msp/identity] Sign -> DEBU 04d
Sign: digest:
EFE78D9C254C9A5795FAC35A6AC5A543AAD70322E6F756D1822E9BBEC11AD7E0
2018-07-20 02:39:49.459 UTC [chaincodeCmd] install -> INFO 04e
Installed remotely response:<status:200 payload:"OK" >
```

第 27 章 Hyperledger Fabric Client SDK for Node.js

1. package.json

```
[root@localhost chaincode-docker-devmode]# mkdir token
[root@localhost chaincode-docker-devmode]# cd token/
[root@localhost devel]#
```

```
[root@localhost token]# cat package.json
{
    "name": "token",
    "version": "1.0.0",
    "description": "Hyperledger Fabric 'token' Sample
Application by netkiller<netkiller@msn.com>",
    "main": "token.js",
    "scripts": {
        "test": "echo \"Error: no test specified\" && exit 1"
    },
    "dependencies": {
        "fabric-ca-client": "^1.0.2",
        "fabric-client": "^1.0.2",
        "grpc": "^1.6.0"
    },
    "author": "netkiller <netkiller@msn.com>",
    "license": "Apache-2.0",
    "keywords": [
        "Hyperledger",
        "Fabric",
        "token"
    ]
}
```

```
[root@localhost token]# npm install
```

2. Node.js 测试程序

```
package main

/*
-------------------------------------------------
Author: netkiller <netkiller@msn.com>
Home: http://www.netkiller.cn
Data: 2018-03-20 11:00 PM
-------------------------------------------------

CORE_PEER_ADDRESS=peer:7051 CORE_CHAINCODE_ID_NAME=token3:1.0
chaincode/token/token3

peer chaincode install -n token3 -v 1.0 -p
chaincodedev/chaincode/token
peer chaincode instantiate -C myc -n token3 -v 1.0 -c '{"Args":
[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"


peer chaincode invoke -C myc -n token3 -c
'{"function":"createAccount","Args":["coinbase"]}'
peer chaincode invoke -C myc -n token3 -v 1.0 -c
'{"function":"showAccount","Args":["coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"balanceAll","Args":["coinbase"]}'

peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":["Netkiller
Token","NKC","1000000","coinbase"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"initCurrency","Args":["NEO
Token","NEC","1000000","coinbase"]}'

peer chaincode invoke -C myc -n token3 -c
'{"function":"setLock","Args":["true"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"setLock","Args":["false"]}'

peer chaincode invoke -C myc -n token3 -c
'{"function":"mintToken","Args":["NKC","5000","coinbase"]}'

peer chaincode invoke -C myc -n token3 -c
```

```
'{"function":"createAccount","Args":["netkiller"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"transferToken","Args":
["coinbase","netkiller","NKC","100"]}'
peer chaincode invoke -C myc -n token3 -c
'{"function":"balance","Args":["netkiller","NKC"]}'

peer chaincode invoke -C myc -n token3 -c
'{"function":"frozenAccount","Args":["netkiller","true"]}'

------------------------------------------------

*/

import (
        "encoding/json"
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)

type Msg struct{
        Status  bool    `json:"Status"`
        Code    int             `json:"Code"`
        Message string  `json:"Message"`
}

type Currency struct{
        TokenName               string  `json:"TokenName"`
        TokenSymbol     string  `json:"TokenSymbol"`
        TotalSupply     float64 `json:"TotalSupply"`
}

type Token struct {
        Lock            bool    `json:"Lock"`
        Currency        map[string]Currency
`json:"Currency"`
}

func (token *Token) transfer (_from *Account, _to *Account,
_currency string, _value float64) []byte{

        var rev []byte
        if (token.Lock){
                msg := &Msg{Status: false, Code: 0, Message:
```

```
"锁仓状态，停止一切转账活动"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if(_from.Frozen ) {
                msg := &Msg{Status: false, Code: 0, Message:
"From 账号冻结"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if( _to.Frozen) {
                msg := &Msg{Status: false, Code: 0, Message:
"To 账号冻结"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if(!token.isCurrency(_currency)){
                msg := &Msg{Status: false, Code: 0, Message:
"货币符号不存在"}
                rev, _ = json.Marshal(msg)
                return rev
        }
        if(_from.BalanceOf[_currency] >= _value){
                _from.BalanceOf[_currency] -= _value;
                _to.BalanceOf[_currency] += _value;

                msg := &Msg{Status: true, Code: 0, Message: "转
账成功"}
                rev, _ = json.Marshal(msg)
                return rev
        }else{
                msg := &Msg{Status: false, Code: 0, Message:
"余额不足"}
                rev, _ = json.Marshal(msg)
                return rev
        }

}
func (token *Token) initialSupply(_name string, _symbol string,
_supply float64, _account *Account) []byte{
        if _,ok := token.Currency[_symbol]; ok {
                msg := &Msg{Status: false, Code: 0, Message:
"代币已经存在"}
                rev, _ := json.Marshal(msg)
                return rev
        }
```

```
        if _account.BalanceOf[_symbol] > 0 {
                msg := &Msg{Status: false, Code: 0, Message:
"账号中存在代币"}
                rev, _ := json.Marshal(msg)
                return rev
        }else{
                token.Currency[_symbol] = Currency{TokenName:
_name, TokenSymbol: _symbol, TotalSupply: _supply}
                _account.BalanceOf[_symbol] = _supply

                msg := &Msg{Status: true, Code: 0, Message: "代
币初始化成功"}
                rev, _ := json.Marshal(msg)
                return rev
        }

}

func (token *Token) mint(_currency string, _amount float64,
_account *Account) []byte{
        if(!token.isCurrency(_currency)){
                msg := &Msg{Status: false, Code: 0, Message:
"货币符号不存在"}
                rev, _ := json.Marshal(msg)
                return rev
        }
        cur := token.Currency[_currency]
        cur.TotalSupply += _amount;
        token.Currency[_currency] = cur
        _account.BalanceOf[_currency] += _amount;

        msg := &Msg{Status: true, Code: 0, Message: "代币增发成
功"}
        rev, _ := json.Marshal(msg)
        return rev

}
func (token *Token) burn(_currency string, _amount float64,
_account *Account) []byte{
        if(!token.isCurrency(_currency)){
                msg := &Msg{Status: false, Code: 0, Message:
"货币符号不存在"}
                rev, _ := json.Marshal(msg)
                return rev
        }
```

```
        if(token.Currency[_currency].TotalSupply >= _amount){
                cur := token.Currency[_currency]
                cur.TotalSupply -= _amount;
                token.Currency[_currency] = cur
                _account.BalanceOf[_currency] -= _amount;

                msg := &Msg{Status: false, Code: 0, Message:
"代币回收成功"}
                rev, _ := json.Marshal(msg)
                return rev
        }else{
                msg := &Msg{Status: false, Code: 0, Message:
"代币回收失败，回收额度不足"}
                rev, _ := json.Marshal(msg)
                return rev
        }

}
func (token *Token) isCurrency(_currency string) bool {
        if _, ok := token.Currency[_currency]; ok {
                return true
        }else{
                return false
        }
}
func (token *Token) setLock(_look bool) bool {
        token.Lock = _look
        return token.Lock
}
type Account struct {
        Name                    string  `json:"Name"`
        Frozen                  bool    `json:"Frozen"`
        BalanceOf               map[string]float64
`json:"BalanceOf"`
}
func (account *Account) balance (_currency string)
map[string]float64{
        bal     :=
map[string]float64{_currency:account.BalanceOf[_currency]}
        return bal
}

func (account *Account) balanceAll() map[string]float64{
        return account.BalanceOf
}

// -----------
```

```
const TokenKey = "Token"

// Define the Smart Contract structure
type SmartContract struct {

}

func (s *SmartContract) Init(stub shim.ChaincodeStubInterface)
pb.Response {

        token := &Token{Currency: map[string]Currency{}}

        tokenAsBytes, err := json.Marshal(token)
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Init Token %s \n",
string(tokenAsBytes))
        }
        return shim.Success(nil)
}

func (s *SmartContract) Query(stub shim.ChaincodeStubInterface)
pb.Response {
        function, args := stub.GetFunctionAndParameters()
        if function == "balance" {
                return s.balance(stub, args)
        } else if function == "balanceAll" {
                return s.balanceAll(stub, args)
        } else if function == "showAccount" {
                return s.showAccount(stub, args)
        }
        return shim.Error("Invalid Smart Contract function
name.")
}

func (s *SmartContract) Invoke(stub
shim.ChaincodeStubInterface) pb.Response {

        // Retrieve the requested Smart Contract function and
arguments
        function, args := stub.GetFunctionAndParameters()
        // Route to the appropriate handler function to
interact with the ledger appropriately
        if function == "initLedger" {
                return s.initLedger(stub, args)
        } else if function == "createAccount" {
```

```
                return s.createAccount(stub, args)
        } else if function == "initCurrency" {
                return s.initCurrency(stub, args)
        } else if function == "setLock" {
                return s.setLock(stub, args)
        } else if function == "transferToken" {
                return s.transferToken(stub, args)
        } else if function == "frozenAccount" {
                return s.frozenAccount(stub, args)
        } else if function == "mintToken" {
                return s.mintToken(stub, args)
        } else if function == "balance" {
                return s.balance(stub, args)
        } else if function == "balanceAll" {
                return s.balanceAll(stub, args)
        } else if function == "showAccount" {
                return s.showAccount(stub, args)
        } else if function == "showToken" {
                return s.showToken(stub, args)
        }

        return shim.Error("Invalid Smart Contract function
name.")
}

func (s *SmartContract) createAccount(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }

        key  := args[0]
        name  := args[0]
        existAsBytes,err := stub.GetState(key)
        fmt.Printf("GetState(%s) %s \n", key,
string(existAsBytes))
        if string(existAsBytes) != "" {
                fmt.Println("Failed to create account,
Duplicate key.")
                return shim.Error("Failed to create account,
Duplicate key.")
        }

        account := Account{
                Name: name,
                Frozen: false,
```

```
                BalanceOf: map[string]float64{}}

        accountAsBytes, _ := json.Marshal(account)
        err = stub.PutState(key, accountAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("createAccount %s \n",
string(accountAsBytes))

        return shim.Success(accountAsBytes)
}
func (s *SmartContract) initLedger(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        return shim.Success(nil)
}
func (s *SmartContract) showToken(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("GetState(%s)) %s \n", TokenKey,
string(tokenAsBytes))
        }
        return shim.Success(tokenAsBytes)
}

func (s *SmartContract) initCurrency(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        if len(args) != 4 {
                return shim.Error("Incorrect number of
arguments. Expecting 4")
        }

        _name  := args[0]
        _symbol:= args[1]
        _supply,_:= strconv.ParseFloat(args[2], 64)
        _account := args[3]

        coinbaseAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Coinbase before %s \n",
string(coinbaseAsBytes))

        coinbase := &Account{}
```

```
        json.Unmarshal(coinbaseAsBytes, &coinbase)

        token := Token{}
        existAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("GetState(%s)) %s \n", TokenKey,
string(existAsBytes))
        }
        json.Unmarshal(existAsBytes, &token)

        result := token.initialSupply(_name,_symbol,_supply,
coinbase)

        tokenAsBytes, _ := json.Marshal(token)
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Init Token %s \n",
string(tokenAsBytes))
        }

        coinbaseAsBytes, _ = json.Marshal(coinbase)
        err = stub.PutState(_account, coinbaseAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Coinbase after %s \n",
string(coinbaseAsBytes))



        return shim.Success(result)
}

func (s *SmartContract) transferToken(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 4 {
                return shim.Error("Incorrect number of
arguments. Expecting 4")
        }
        _from           := args[0]
        _to                     := args[1]
        _currency       := args[2]
```

```
        _amount,_       := strconv.ParseFloat(args[3], 32)

        if(_amount <= 0){
                return shim.Error("Incorrect number of amount")
        }

        fromAsBytes,err := stub.GetState(_from)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("fromAccount %s \n", string(fromAsBytes))
        fromAccount := &Account{}
        json.Unmarshal(fromAsBytes, &fromAccount)

        toAsBytes,err := stub.GetState(_to)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("toAccount %s \n", string(toAsBytes))
        toAccount := &Account{}
        json.Unmarshal(toAsBytes, &toAccount)

        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Token %s \n", string(toAsBytes))
        token := Token{Currency: map[string]Currency{}}
        json.Unmarshal(tokenAsBytes, &token)

        result := token.transfer(fromAccount, toAccount,
_currency, _amount)
        fmt.Printf("Result %s \n", string(result))

        fromAsBytes, err = json.Marshal(fromAccount)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(_from, fromAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("fromAccount %s \n",
string(fromAsBytes))
        }

        toAsBytes, err = json.Marshal(toAccount)
        if err != nil {
```

```
                return shim.Error(err.Error())
        }
        err = stub.PutState(_to, toAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("toAccount %s \n",
string(toAsBytes))
        }

        return shim.Success(result)
}
func (s *SmartContract) mintToken(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 3 {
                return shim.Error("Incorrect number of
arguments. Expecting 3")
        }
        _currency       := args[0]
        _amount,_       := strconv.ParseFloat(args[1], 32)
        _account        := args[2]

        coinbaseAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Coinbase before %s \n",
string(coinbaseAsBytes))
        }

        coinbase := &Account{}
        json.Unmarshal(coinbaseAsBytes, &coinbase)

        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Token before %s \n", string(tokenAsBytes))

        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)

        result := token.mint(_currency, _amount, coinbase)

        tokenAsBytes, err = json.Marshal(token)
        if err != nil {
```

```
                return shim.Error(err.Error())
        }
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Token after %s \n", string(tokenAsBytes))

        coinbaseAsBytes, _ = json.Marshal(coinbase)
        err = stub.PutState(_account, coinbaseAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Coinbase after %s \n",
string(coinbaseAsBytes))
        }

        fmt.Printf("mintToken %s \n", string(tokenAsBytes))

        return shim.Success(result)
}

func (s *SmartContract) setLock(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _look := args[0]

        tokenAsBytes,err := stub.GetState(TokenKey)
        if err != nil {
                return shim.Error(err.Error())
        }
        // fmt.Printf("setLock - begin %s \n",
string(tokenAsBytes))

        token := Token{}

        json.Unmarshal(tokenAsBytes, &token)

        if(_look == "true"){
                token.setLock(true)
        }else{
                token.setLock(false)
        }
```

```
        tokenAsBytes, err = json.Marshal(token)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(TokenKey, tokenAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("setLock - end %s \n", string(tokenAsBytes))

        return shim.Success(nil)
}
func (s *SmartContract) frozenAccount(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 2")
        }

        _account        := args[0]
        _status         := args[1]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }
        // fmt.Printf("setLock - begin %s \n",
string(tokenAsBytes))

        account := Account{}

        json.Unmarshal(accountAsBytes, &account)

        var status bool
        if(_status == "true"){
                status = true;
        }else{
                status = false
        }

        account.Frozen = status

        accountAsBytes, err = json.Marshal(account)
        if err != nil {
                return shim.Error(err.Error())
        }
```

```
        err = stub.PutState(_account, accountAsBytes)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("frozenAccount - end %s \n",
string(accountAsBytes))
        }

        return shim.Success(nil)
}

func (s *SmartContract) showAccount(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }
        _account        := args[0]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Account balance %s \n",
string(accountAsBytes))
        }
        return shim.Success(accountAsBytes)
}

func (s *SmartContract) balance(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 2 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }
        _account        := args[0]
        _currency       := args[1]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Account balance %s \n",
string(accountAsBytes))
        }
```

```
        account := Account{}
        json.Unmarshal(accountAsBytes, &account)
        result := account.balance(_currency)

        resultAsBytes, _ := json.Marshal(result)
        fmt.Printf("%s balance is %s \n", _account,
string(resultAsBytes))

        return shim.Success(resultAsBytes)
}

func (s *SmartContract) balanceAll(stub
shim.ChaincodeStubInterface, args []string) pb.Response {

        if len(args) != 1 {
                return shim.Error("Incorrect number of
arguments. Expecting 1")
        }
        _account        := args[0]

        accountAsBytes,err := stub.GetState(_account)
        if err != nil {
                return shim.Error(err.Error())
        }else{
                fmt.Printf("Account balance %s \n",
string(accountAsBytes))
        }

        account := Account{}
        json.Unmarshal(accountAsBytes, &account)
        result := account.balanceAll()
        resultAsBytes, _ := json.Marshal(result)
        fmt.Printf("%s balance is %s \n", _account,
string(resultAsBytes))

        return shim.Success(resultAsBytes)
}

// The main function is only relevant in unit test mode. Only
included here for completeness.
func main() {

        // Create a new Smart Contract
        err := shim.Start(new(SmartContract))
        if err != nil {
                fmt.Printf("Error creating new Smart Contract:
%s", err)
        }
```

```
}
```

3. 创建 package.json 文件

```
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e
"CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger
/fabric/peer/crypto/peerOrganizations/org1.example.com/users/Ad
min@org1.example.com/msp" cli peer chaincode install -n token3
-v 1.0 -p github.com/token
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e
"CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger
/fabric/peer/crypto/peerOrganizations/org1.example.com/users/Ad
min@org1.example.com/msp" cli peer chaincode instantiate -o
orderer.example.com:7050 -C mychannel -n token3 -v 1.0 -c
'{"Args":[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
sleep 10
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e
"CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger
/fabric/peer/crypto/peerOrganizations/org1.example.com/users/Ad
min@org1.example.com/msp" cli peer chaincode invoke -o
orderer.example.com:7050 -C mychannel -n token3 -c
'{"function":"createAccount","Args":["coinbase"]}'
```

4. 查询操作

```
[root@localhost token]# cat queryToken.js
'use strict';

var Fabric_Client = require('fabric-client');
var path = require('path');
var util = require('util');
var os = require('os');

//
var fabric_client = new Fabric_Client();

// setup the fabric network
var channel = fabric_client.newChannel('mychannel');
var peer = fabric_client.newPeer('grpc://localhost:7051');
channel.addPeer(peer);


var member_user = null;
var store_path = path.join(__dirname, 'hfc-key-store');
console.log('Store path:'+store_path);
var tx_id = null;

Fabric_Client.newDefaultKeyValueStore({ path: store_path
}).then((state_store) => {

        fabric_client.setStateStore(state_store);
        var crypto_suite = Fabric_Client.newCryptoSuite();

        var crypto_store =
Fabric_Client.newCryptoKeyStore({path: store_path});
        crypto_suite.setCryptoKeyStore(crypto_store);
        fabric_client.setCryptoSuite(crypto_suite);

        return fabric_client.getUserContext('user1', true);
}).then((user_from_store) => {
        if (user_from_store && user_from_store.isEnrolled()) {
                console.log('Successfully loaded user1 from
persistence');
                member_user = user_from_store;
        } else {
                throw new Error('Failed to get user1.... run
```

```
registerUser.js');
        }

        const request = {
                chaincodeId: 'token3',
                fcn: 'showAccount',
                args: ['coinbase']
        };

        // send the query proposal to the peer
        return channel.queryByChaincode(request);
}).then((query_responses) => {
        console.log("Query has completed, checking results");
        if (query_responses && query_responses.length == 1) {
                if (query_responses[0] instanceof Error) {
                        console.error("error from query = ",
query_responses[0]);
                } else {
                        console.log("Response is ",
query_responses[0].toString());
                }
        } else {
                console.log("No payloads were returned from
query");
        }
}).catch((err) => {
        console.error('Failed to query successfully :: ' +
err);
});
```

核心代码

```
        const request = {
                chaincodeId: 'token3',  // 链码名称
                fcn: 'showAccount',     //调用方法
                args: ['coinbase']      // 传递参数
        };
```

运行结果

```
[root@localhost token]# node queryToken.js
Store path:/root/fabric-samples/fabcar/hfc-key-store
Successfully loaded user1 from persistence
Query has completed, checking results
Response is  {"BalanceOf":{},"Frozen":false,"Name":"coinbase"}
```

5. Event

```
chain.eventHubConnect(peerEventHosts[0],{pem:pem});
setupEvents();

/**
 * Sample method to showcase how to subscribe and consume
events emitted from blockchain
 */
function setupEvents(){
    try{
            var eh = chain.getEventHub();
            var cid = config['chaincode']['id'];
            var regid = eh.registerChaincodeEvent(cid,
"^eventSender$", function(event) {
                console.log(event);
                var buffer = new Buffer(event.payload);
                console.log(buffer.toString());
            });
            console.log("EVENT SETUP DONE");
        }
        catch(err){
            console.log(err);
            console.log("Could not setup events");
        }
}

process.on('exit', function (){
    console.log('exit called');
    chain.eventHubDisconnect();
});
```

6.

第 28 章 fabric-sdk-java

1. Maven

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <groupId>cn.netkiller</groupId>
        <artifactId>fabric-sdk-java</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <dependencies>
                <!--
https://mvnrepository.com/artifact/org.hyperledger.fabric-sdk-
java/fabric-sdk-java -->
                <dependency>
                        <groupId>org.hyperledger.fabric-sdk-
java</groupId>
                        <artifactId>fabric-sdk-
java</artifactId>
                        <version>1.0.1</version>
                </dependency>

        </dependencies>
</project>
```

第 29 章 Hyperledger Explorer

克隆代码

```
git clone https://github.com/hyperledger/blockchain-
explorer.git
cd blockchain-explorer
```

将 fabricexplorer.sql 导入到数据中

```
mysql -u<username> -p < db/fabricexplorer.sql
```

设置 channel 和 数据库连接

```
 "channel": "mychannel",
 "mysql":{
      "host":"127.0.0.1",
      "database":"fabricexplorer",
      "username":"root",
      "passwd":"123456"
   }
```

启动

```
npm install
./start.sh
```

进入 Hyperledger Explorer http://localhost:8080/

第 30 章 已知 Hyperledger 落地案例

1. 莱茨狗

莱茨狗是山寨版以太猫，底层采用 Hyperledger

http://pet-chain.baidu.com

第 31 章 Fabric Command

1. peer

1.1. channel

1.1.1. list

列出当前 Channel

```
root@61814b46e7ee:/opt/gopath/src/github.com/chaincode/token#
peer channel list
2018-03-19 08:41:25.289 UTC [msp] GetLocalMSP -> DEBU 001
Returning existing local MSP
2018-03-19 08:41:25.290 UTC [msp] GetDefaultSigningIdentity ->
DEBU 002 Obtaining default signing identity
2018-03-19 08:41:25.293 UTC [channelCmd] InitCmdFactory -> INFO
003 Endorser and orderer connections initialized
2018-03-19 08:41:25.294 UTC [msp/identity] Sign -> DEBU 004
Sign: plaintext:
0A86070A5C08031A0C08B5EFBDD50510...631A0D0A0B4765744368616E6E65
6C73
2018-03-19 08:41:25.294 UTC [msp/identity] Sign -> DEBU 005
Sign: digest:
E4E3AB1CFD295AA881EA2AB08781EF8989CA4271FC3E2A0E6DFDDB991ECEE7F
6
2018-03-19 08:41:25.305 UTC [channelCmd] list -> INFO 006
Channels peers has joined to:
2018-03-19 08:41:25.305 UTC [channelCmd] list -> INFO 007
mychannel
2018-03-19 08:41:25.305 UTC [main] main -> INFO 008
Exiting.....
```

第 32 章 Fabric FAQ

1. ERROR: manifest for hyperledger/fabric-ca:latest not found

```
ERROR: manifest for hyperledger/fabric-orderer:latest not found
ERROR: manifest for hyperledger/fabric-couchdb:latest not found
ERROR: manifest for hyperledger/fabric-ca:latest not found
ERROR: manifest for hyperledger/fabric-tools:latest not found
```

解决方案

```
docker pull hyperledger/fabric-membersrvc:latest \
&& docker pull hyperledger/fabric-peer:x86_64-1.0.5 \
&& docker pull hyperledger/fabric-orderer:x86_64-1.0.5 \
&& docker pull hyperledger/fabric-couchdb:x86_64-1.0.5 \
&& docker pull hyperledger/fabric-ca:x86_64-1.0.5 \
&& docker pull hyperledger/fabric-tools:x86_64-1.0.5

docker tag hyperledger/fabric-peer:x86_64-1.0.5 hyperledger/fabric-peer
\
&& docker tag hyperledger/fabric-orderer:x86_64-1.0.5
hyperledger/fabric-orderer \
&& docker tag hyperledger/fabric-couchdb:x86_64-1.0.5
hyperledger/fabric-couchdb \
&& docker tag hyperledger/fabric-ca:x86_64-1.0.5 hyperledger/fabric-ca \
&& docker tag hyperledger/fabric-tools:x86_64-1.0.5 hyperledger/fabric-
tools
```

2. 卸载 hyperledger 环境

```
docker rmi -f $(docker images -q)
```

3. dseasb33srnrn.cloudfront.net 无法连接

```
error pulling image configuration: Get
https://dseasb33srnrn.cloudfront.net/registry-
v2/docker/registry/v2/blobs/sha256/72/72617b4fa9b4be7a594627504
8f9468320220cf7bd316f0ab77946d68d24e970/data?
Expires=1521480730&Signature=GKdv-Qc-
hU1QLjsw5h7DrHCp11oWGZ3paa17L4CvMHx0893u-hk-
klBKn7DWHnoEf4d3k9zp~OszcbVjsJsq87KlVFqWam17l7V3YvmXTPIpXEG5j4w
heS5CxKRKIw8vChnm6vRwCQ3FoTEkiugXfTfLpxLOUIpB7yca3VRDbbw_&Key-
Pair-Id=APKAJECH5M7VWIS5YZ6Q: dial tcp: lookup
dseasb33srnrn.cloudfront.net on 211.162.77.77:53: read udp
172.16.0.17:61941->211.162.77.77:53: i/o timeout
Error response from daemon: No such image: hyperledger/fabric-
ca:x86_64-1.1.0
```

解决方案，设置docker mirror 站点

```
curl -s
https://raw.githubusercontent.com/oscm/shell/master/virtualizat
ion/docker/registry-mirror.sh | bash
```

4. 超级账本的硬伤

在使用超级账本的过程中我发现一个问题，超级账本无法并发操作一个 key，stub.PutState 是异步执行，我们无法确认它是否执行完成，在没有执行完成之前再发起操作，就会产生覆盖。这个问题限制了超级账本的很多场景应用，这是超级账本的硬伤。

下面举一个例子来说明超级账本的问题

```
func (s *SmartContract) counter(stub
shim.ChaincodeStubInterface, args []string) pb.Response {
        key  := "counter"
        count,err = stub.GetState(key)
        count = count + 1
        stub.PutState(key,count)
        return shim.Success(count)
}
```

使用多线程请求chaincode中的counter函数100次。你会发现最终 count 并不等于 100。学习过多线程的朋友一定很清楚出了什么问题。

问题出在 stub.PutState 函数count还没有被写入，其他线程就开始读取 stub.GetState(key)，导致读取旧数据，最终计数器数字混乱。

很多场景需要更新区块中的数据，如果频繁操作，就会产生覆盖，目前Hyperledger Fabirc 并没有提供解决方案。

1. 我们不知道 stub.PutState是否执行完成，因为存储过程需要共识排序。

2. 超级账本没有提供事物处理或者互斥锁。

golang 提供的 mutex 也无法解决上面的问题，因为 mutex 锁只能工作在一个进程中。Peer / Orderer 节点不止一个。

使用 redis实现分布式锁或许能实现，但思考过后决定放弃，转为传统数据库。

总结，超级账本只适合一次写，多次读的场景，和极低频修改的场景

部分 IV. IPFS（InterPlanetary File System 星际文件系统）

第 33 章 IPFS（InterPlanetary File System，星际文件系统）

1. 什么是 IPFS

IPFS “星际文件系统” 是去中心化保存和共享文件的分布式文件系统，它是一种内容可寻址、版本化、点对点超媒体的分布式协议，IPFS旨在替代HTTP。它希望将所有的计算设备都连接到同一个文件系统中。

由于 IPFS 去中心化特点，使得很多区块链项目做文件存储的时候首选 IPFS。

1.1. 传统的中心化HTTP服务

举个例子，服务器上有一个文件https://www.netkiller.cn/video/av.mp4，按照HTTP协议标准流程，浏览器首先会解析域名并查找到服务器的IP地址，随后与服务器建立链接，并根据路径向服务器请求该文件/video/av.mp4。在这种体系下用户是否能获取该文件取决于文件是否被移动或删除，并且服务器没有关闭。

另一个问题就是文件服务器管理，中心化管理是很复杂的，如果访问量比较大，我们通常会使用负载均衡来解决，负载均衡需要每个节点上的数据是完整的，这就需要大量的数据同步。如果从深圳数据中心同步到上海数据中心 100T的数据可能要数个工作日。

HTTP过度依赖Internet主干网，同时网络通信遭遇DDoS攻击的风险也大大增加。

CDN 只解决最后一公里的性能问题，数据回源仍然是中心化，仍然无法彻底解决。

1.2. IPFS 解决方案

IPFS的做解决方案是不再关心中心服务器的位置，也不考虑文件的名字和路径，只关注文件中的内容。例如将文件av.mp4添加到IPFS节点，会得到一个新名字QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN，通过文件内容计算出的哈希值，访问文件不在使用文件名而是使用这个统一的 hash 值。

当用户被请求一个文件哈希时，它会根据这个分布式哈希表找到文件所在的节点，并取回文件内容并验证文件数据。大文件会被切分成小的分块，下载的时候可以从多个服务器同时获取。

IPFS 可以理解为内容被永久缓存到节点上的CDN。

2. 安装 IPFS

2.1. go get 方式

```
$ go get -u -d github.com/ipfs/go-ipfs

$ cd $GOPATH/src/github.com/ipfs/go-ipfs
$ make install
```

2.2. 安装 ipfs-update

ipfs-update 是一个 ipfs 版本管理工具，可以通过这个工具安装ipfs 和升级 ipfs.

```
# go get -u github.com/ipfs/ipfs-update

# ipfs-update install latest
```

2.3. Ubuntu

```
neo@netkiller ~ % sudo apt install golang1.9
```

2.4. Netkiller OSCM

2.4.1. 源码安装

```
curl -s https://raw.githubusercontent.com/oscm/shell/master/distributed/ipfs/go-
ipfs-0.4.14.sh | bash
curl -s
https://raw.githubusercontent.com/oscm/shell/master/distributed/ipfs/ipfs-
update-1.5.2.sh | bash
```

2.4.2. ipfs-update

```
[root@netkiller ~]# curl -s
https://raw.githubusercontent.com/oscm/shell/master/lang/golang/golang-1.11.sh |
bash
[root@netkiller ~]# curl -s
https://raw.githubusercontent.com/oscm/shell/master/distributed/ipfs/ipfs-
update.sh | bash

[root@netkiller ~]# ./go/bin/ipfs-update versions
[root@netkiller ~]# ./go/bin/ipfs-update install latest
```

2.5. Mac OS

```
neo@MacBook-Pro ~ % brew install ipfs
```

第 34 章 IPFS 命令

创建 ipfs 用户

```
[root@netkiller ~]# adduser ipfs
[root@netkiller ~]# su - ipfs
[ipfs@netkiller ~]$
```

1. help

```
neo@MacBook-Pro ~ % ipfs --help
USAGE
  ipfs - Global p2p merkle-dag filesystem.

SYNOPSIS
  ipfs [--config=<config> | -c] [--debug=<debug> | -D] [--help=
<help>] [-h=<h>] [--local=<local> | -L] [--api=<api>] <command>
...

OPTIONS

  -c,              --config   string - Path to the
configuration file to use.
  -D,              --debug    bool   - Operate in debug mode.
  --help                      bool   - Show the full command
help text.
  -h                          bool   - Show a short version of
the command help text.
  -L,              --local    bool   - Run the command locally,
instead of using the daemon.
  --api                       string - Use a specific API
instance (defaults to /ip4/127.0.0.1/tcp/5001).
  --enc,           --encoding string - The encoding type the
output should be encoded with (json, xml, or text). Default:
text.
  --stream-channels           bool   - Stream channel output.
  --timeout                   string - set a global timeout on
```

```
the command.

SUBCOMMANDS
  BASIC COMMANDS
    init          Initialize ipfs local configuration
    add <path>    Add a file to IPFS
    cat <ref>     Show IPFS object data
    get <ref>     Download IPFS objects
    ls <ref>      List links from an object
    refs <ref>    List hashes of links from an object

  DATA STRUCTURE COMMANDS
    block         Interact with raw blocks in the datastore
    object        Interact with raw dag nodes
    files         Interact with objects as if they were a unix
filesystem
    dag           Interact with IPLD documents (experimental)

  ADVANCED COMMANDS
    daemon        Start a long-running daemon process
    mount         Mount an IPFS read-only mountpoint
    resolve       Resolve any type of name
    name          Publish and resolve IPNS names
    key           Create and list IPNS name keypairs
    dns           Resolve DNS links
    pin           Pin objects to local storage
    repo          Manipulate the IPFS repository
    stats         Various operational stats
    p2p           Libp2p stream mounting
    filestore     Manage the filestore (experimental)

  NETWORK COMMANDS
    id            Show info about IPFS peers
    bootstrap     Add or remove bootstrap peers
    swarm         Manage connections to the p2p network
    dht           Query the DHT for values or peers
    ping          Measure the latency of a connection
    diag          Print diagnostics

  TOOL COMMANDS
    config        Manage configuration
    version       Show ipfs version information
    update        Download and apply go-ipfs updates
    commands      List all available commands

  Use 'ipfs <command> --help' to learn more about each command.

  ipfs uses a repository in the local file system. By default,
```

```
the repo is
  located at ~/.ipfs. To change the repo location, set the
$IPFS_PATH
  environment variable:

    export IPFS_PATH=/path/to/ipfsrepo

  EXIT STATUS

  The CLI will exit with one of the following values:

  0     Successful execution.
  1     Failed executions.

  Use 'ipfs <subcmd> --help' for more information about each
command.
```

3. 基本命令

3.1. 初始化节点

```
[ipfs@netkiller ~]$ ipfs init
initializing IPFS node at /home/ipfs/.ipfs
generating 2048-bit RSA keypair...done
peer identity: QmZakCF5czhP53KPvMi8XQcYtVrQohw5N71Xce4eC1rWz3
to get started, enter:

        ipfs cat
/ipfs/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv/readme
```

目录结构

```
[ipfs@netkiller ~]$ ls .ipfs/
blocks  config  datastore  datastore_spec  keystore  version
```

3.2. 添加文件或文本到 IPFS

3.2.1. 添加文件

```
[ipfs@netkiller ~]$ echo Helloworld > helloworld.txt
[ipfs@netkiller ~]$ ipfs add helloworld.txt
added QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN
helloworld.txt

[ipfs@netkiller ~]$ ipfs cat
QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN
Helloworld

[ipfs@netkiller ~]$ ipfs cat
```

```
/ipfs/QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN
Helloworld
```

3.2.2. 添加文本

添加一段字符串到 IPFS

```
[ipfs@netkiller ~]$ echo 'Look! Things have changed!' | ipfs add
added QmSb8DSVmu4Qip56jcqPVz1Cx9RJ3vTf3d1Gf9ixaG2tWg
QmSb8DSVmu4Qip56jcqPVz1Cx9RJ3vTf3d1Gf9ixaG2tWg

[ipfs@netkiller ~]$ ipfs cat
QmSb8DSVmu4Qip56jcqPVz1Cx9RJ3vTf3d1Gf9ixaG2tWg
Look! Things have changed!
```

3.2.3. 安静模式，仅返回 Hash

安静模式

```
[ipfs@netkiller ~]$ ipfs add -q /tmp/1536896811406807.mp4
QmcA1Fsrt6jGTVqAUNZBqaprMEdFaFkmkzA5s2M6mF85UC
```

3.2.4. 尝试修改内容

修改内容后 Hash 变化

```
[root@netkiller ~]# echo "version 1 of my text" | ipfs add
```

```
added QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy
QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy

[root@netkiller ~]# echo "version 2 of my text" | ipfs add
added QmTudJSaoKxtbEnTddJ9vh8hbN84ZLVvD5pNpUaSbxwGoa
QmTudJSaoKxtbEnTddJ9vh8hbN84ZLVvD5pNpUaSbxwGoa

[root@netkiller ~]# ipfs cat
QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy
version 1 of my text

[root@netkiller ~]# ipfs cat
QmTudJSaoKxtbEnTddJ9vh8hbN84ZLVvD5pNpUaSbxwGoa
version 2 of my text


[root@netkiller ~]# echo "version 1" > version.txt
[root@netkiller ~]# ipfs add version.txt
added QmQBcXurY2QBpv7sg8zyS4UXQeHGCqx4DBp86kBLPDzS18 version.txt

[root@netkiller ~]# echo "version 2" > version.txt
[root@netkiller ~]# ipfs add version.txt
added QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2 version.txt

[root@netkiller ~]# ipfs cat
QmQBcXurY2QBpv7sg8zyS4UXQeHGCqx4DBp86kBLPDzS18
version 1

[root@netkiller ~]# ipfs cat
QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2
version 2
```

3.2.5. 递归添加一个目录

```
[ipfs@netkiller ~]$ ipfs add -r /etc/nginx/
added QmTa5RvPS9GEgbR8KUy36Fkx7y8Z7LpFM47pbAWJ89tUoi
nginx/conf.d/default.conf
added QmSGZtdGvLd64eYqXNyDraegz5Tm5evyTrX7GMAhE4L1KB
nginx/conf.d/default.conf.backup
added Qmbh35NHRNYfpaXaj1bQF4gcVYKRjjBvD7eys4dK4iwNrY
nginx/fastcgi_params
added QmaHUhxr4NPTsxc2iubLAVGAvJ6hRKfZsbDPYZDT6Bdcb1 nginx/koi-
```

```
utf
added QmRpsaigHjE4udpVDXZ8T4YtA477M7YTox6xxxkbYjisgN nginx/koi-
win
added QmbJThNSiHPy1bDTkLq2Rp8uU4cDLbX8Nzgxrf79ujQ6D9
nginx/mime.types
added QmWDHS75NyRXZZxe3ZYAaETE4Z4mPx1htahBZBYx4cAYEM
nginx/modules
added QmNjAKtPe3AoUSSiLF1onA8fVbogcQ4cJHD2bvj38udpJQ
nginx/nginx.conf
added QmPXqmW4jpg5HbqqToJnzpqamSSNwHEkSxcGqiGmE9UaGM
nginx/nginx.conf.original
added QmTiu69XGzh1iqK6JuJNzoTiQ4SJEeQSso3uCHRwbXFba8
nginx/scgi_params
added QmW2nvZqf6fwmcgBeeUEv2UyePizsFjjkNCubqRjmCZHNV
nginx/uwsgi_params
added Qmen15DKJhF5ngxiBEzmpMyPA3HpMfXNqgogeyeati2sEx nginx/win-
utf
added QmZ1cs8uxMSRcd2UX6ei4a2DYanQSBNV8PieRWcKzXbKXY
nginx/conf.d
added Qmbw5SZyAKkccUTq5N5jLTXzRk21jZThmmMGsX5Fh2BPic nginx
 19.35 KiB / 27.35 KiB
[=============================================================>--------
---]  70.75%
```

注意最后面的 nginx 是目录 Hash，我们可以使用这个 Hash 访问文件

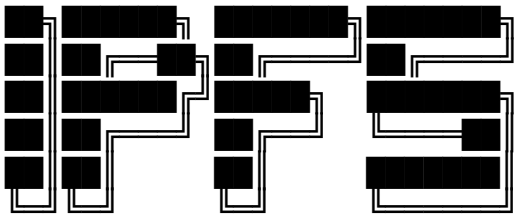
```
http://ipfs.netkiller.cn/ipfs/Qmbw5SZyAKkccUTq5N5jLTXzRk21jZThmm
MGsX5Fh2BPic/conf.d/default.conf
```

你也可以直接访问 Hash 值

```
http://ipfs.netkiller.cn/ipfs/QmSGZtdGvLd64eYqXNyDraegz5Tm5evyTr
X7GMAhE4L1KB
```

3.3. 查看文件

```
[ipfs@netkiller ~]$ ipfs cat
/ipfs/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv/readme
Hello and Welcome to IPFS!
```



```
If you're seeing this, you have successfully installed
IPFS and are now interfacing with the ipfs merkledag!

 ------------------------------------------------------
| Warning:                                             |
|   This is alpha software. Use at your own discretion! |
|   Much is missing or lacking polish. There are bugs.  |
|   Not yet secure. Read the security notes for more.   |
 ------------------------------------------------------

Check out some of the other files in this directory:

  ./about
  ./help
  ./quick-start     <-- usage examples
  ./readme          <-- this file
  ./security-notes
```

3.4. 下载文件

```
[root@netkiller ~]# ipfs get
/ipfs/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv/readme
Saving file(s) to readme
 1.08 KB / 1.08 KB
[=============================================================
=======] 100.00% 0s
[root@netkiller ~]# ls
readme
```

```
[root@netkiller ~]# ipfs add readme.txt
added QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn readme.txt

[root@netkiller ~]# ipfs cat
QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn
Helloworld

[root@netkiller ~]# ipfs get
QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn
Saving file(s) to QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn
 880 B / 880 B
[=============================================================
==========] 100.00% 0s

[root@netkiller ~]# ls
QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn  readme.txt
```

3.5. 列出文件或目录

```
[root@netkiller ~]# ipfs ls
/ipfs/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv/
QmZTR5bcpQD7cFgTorqxZDYaew1Wqgfbd2ud9QqGPAkK2V 1688 about
QmYCvbfNbCwFR45HiNP45rwJgvatpiW38D961L5qAhUM5Y 200  contact
QmY5heUM5qgRubMDD1og9fhCPA6QdkMp3QCwd4s7gJsyE7 322  help
QmejvEPop4D7YUadeGqYWmZxHhLc4JBUCzJJHWMzdcMe2y 12   ping
QmXgqKTbzdh83pQtKFb19SpMCpDDcKR2ujqk3pKph9aCNF 1692 quick-start
QmPZ9gcCEpqKTo6aq61g2nXGUhM4iCL3ewB6LDXZCtioEB 1102 readme
QmQ5vhrL7uv6tuoN9KeVBwd4PwfQkXdVVmDLUZuTNxqgvm 1173 security-
notes
```

```
[root@netkiller ~]# ipfs add -r /etc/rc.d
added QmP4m7YRN25kcRgoJgn95yFR4GsVgbQppnpMGh3AxPzUbc
rc.d/init.d/README
```

```
added QmemkmPhud9hBWhTDgnYfascZcXjX4H6j4oyqhKdSvFq8G
rc.d/init.d/aegis
added QmR2zvwDZYQPw4arpaJQwGESPhmz6qBt8MqyrV5UR72JUy
rc.d/init.d/agentwatch
added QmevSkKJVSF4amfHux7bMZCvaQQjuFsDxU1j7j7uvvJzBh
rc.d/init.d/functions
added QmXCmPYb1wWcmAWk8tm1jgYsFgF16mTe2PsnNebw1gzWDw
rc.d/init.d/jexec
added QmeQZxv7Fui5Ah9w9E9QuiyTrbm4XMKyJMthGLvbbtd5q4
rc.d/init.d/netconsole
added QmeDfRGHoLuSTZqk4zcwGKJQ1NUk9VhEYxFy1HC5d9hKxH
rc.d/init.d/network
added QmYWStNUPtn9TgUT39D7vXMjXc5y917W7SsjPTZvCMK136
rc.d/rc.local
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc0.d/K01agentwatch
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc0.d/K05jexec
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc0.d/K50aegis
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc0.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc0.d/K90network
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc1.d/K01agentwatch
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc1.d/K50aegis
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc1.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc1.d/K90network
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc1.d/S95jexec
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc2.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc2.d/S10network
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc2.d/S50aegis
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc2.d/S95jexec
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc2.d/S98agentwatch
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc3.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc3.d/S10network
```

```
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc3.d/S50aegis
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc3.d/S95jexec
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc3.d/S98agentwatch
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc4.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc4.d/S10network
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc4.d/S50aegis
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc4.d/S95jexec
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc4.d/S98agentwatch
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc5.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc5.d/S10network
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc5.d/S50aegis
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc5.d/S95jexec
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc5.d/S98agentwatch
added QmS92vH7JY9CnP3DVZiQcHxqyi9FQXc1RTcyQvhCJnTCon
rc.d/rc6.d/K01agentwatch
added QmabzShX1WxbdMfLmNK16M1edR5TP95iM5FBURSFwfdj4t
rc.d/rc6.d/K05jexec
added QmcuWaJdg5JCckgdh5BomTwYjrn3ChrC2xa3yY61jQ1pGV
rc.d/rc6.d/K50aegis
added QmYnJ5yxouhrbu81Nod3o1ZWKuwdHYYsbr7z4BgLCxxR2P
rc.d/rc6.d/K50netconsole
added QmPNCeB2yS9EZ3cG3ousMtPkLNemkQaDfKt63S4NTBJg3z
rc.d/rc6.d/K90network
added QmUy33heBFE59gc6PB2sevQ88gB7Lv3NDASFJF3S8f1qHK rc.d/init.d
added QmaDZHSDdtNyryWNwtxuMtJ8NduQdfhMMHk7JuTpNW5sTs rc.d/rc0.d
added QmX6D9eP1oHMg92rrVYarGAe4w3HQ9vURcncBRvPqtXSyG rc.d/rc1.d
added QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 rc.d/rc2.d
added QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 rc.d/rc3.d
added QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 rc.d/rc4.d
added QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 rc.d/rc5.d
added QmaDZHSDdtNyryWNwtxuMtJ8NduQdfhMMHk7JuTpNW5sTs rc.d/rc6.d
added QmRYPNdKdyCr6R7fE63g2u3sMD8SQh8TW6yQNAk9mT9Pay rc.d

[root@netkiller ~]# ipfs ls -v
QmRYPNdKdyCr6R7fE63g2u3sMD8SQh8TW6yQNAk9mT9Pay
```

```
Hash                                                Size  Name
QmUy33heBFE59gc6PB2sevQ88gB7Lv3NDASFJF3S8f1qHK 36013 init.d/
QmYWStNUPtn9TgUT39D7vXMjXc5y917W7SsjPTZvCMK136 484   rc.local
QmaDZHSDdtNyryWNwtxuMtJ8NduQdfhMMHk7JuTpNW5sTs 383   rc0.d/
QmX6D9eP1oHMg92rrVYarGAe4w3HQ9vURcncBRvPqtXSyG 383   rc1.d/
QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 383   rc2.d/
QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 383   rc3.d/
QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 383   rc4.d/
QmTEuKk4acWJcsCDYJViMNoKZ2BQDY8NSHraV9gB9nAWm8 383   rc5.d/
QmaDZHSDdtNyryWNwtxuMtJ8NduQdfhMMHk7JuTpNW5sTs 383   rc6.d/

[root@netkiller ~]# ipfs ls -v
QmUy33heBFE59gc6PB2sevQ88gB7Lv3NDASFJF3S8f1qHK
Hash                                                Size  Name
QmP4m7YRN25kcRgoJgn95yFR4GsVgbQppnpMGh3AxPzUbc 1171  README
QmemkmPhud9hBWhTDgnYfascZcXjX4H6j4oyqhKdSvFq8G 2266  aegis
QmR2zvwDZYQPw4arpaJQwGESPhmz6qBt8MqyrV5UR72JUy 3015  agentwatch
QmevSkKJVSF4amfHux7bMZCvaQQjuFsDxU1j7j7uvvJzBh 17514 functions
QmXCmPYb1wWcmAWk8tm1jgYsFgF16mTe2PsnNebw1gzWDw 41    jexec
QmeQZxv7Fui5Ah9w9E9QuiyTrbm4XMKyJMthGLvbbtd5q4 4345  netconsole
QmeDfRGHoLuSTZqk4zcwGKJQ1NUk9VhEYxFy1HC5d9hKxH 7304  network
```

4. 数据结构命令

4.1. 块

4.1.1. 写入块

```
[ipfs@netkiller ~]$ echo "hello world" | ipfs block put
QmZjTnYw2TFhn9Nn7tjmPSoTBoY7YRkwPzwSrSbabY24Kp
```

4.1.2. 读取块

```
[ipfs@netkiller ~]$ ipfs block get
QmZjTnYw2TFhn9Nn7tjmPSoTBoY7YRkwPzwSrSbabY24Kp
hello world
```

4.1.3. 块状态

```
[ipfs@netkiller ~]$ ipfs block stat
QmZjTnYw2TFhn9Nn7tjmPSoTBoY7YRkwPzwSrSbabY24Kp
Key: QmZjTnYw2TFhn9Nn7tjmPSoTBoY7YRkwPzwSrSbabY24Kp
Size: 12
```

4.2. 对象

```
[ipfs@netkiller ~]$ ipfs object get
QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN
{"Links":[],"Data":"\u0008\u0002\u0012\u000bHelloworld\n\u0018\u000b"}
```

5. 高级命令

5.1. 守护进程

```
$ ipfs daemon
```

http://localhost:5001/webui

5.2. 发布并解析IPNS

```
[root@netkiller ~]# ipfs add readme.txt
added QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn readme.txt
[root@netkiller ~]# ipfs name publish
QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn

Published to QmPNH1AKXNidUgsW6MaMABiNRhdXnqpTbthbYoZzhZE9ve:
/ipfs/QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn

[root@netkiller ~]# ipfs cat
/ipfs/QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn
```

验证发布

```
[root@netkiller ~]# ipfs add version.txt
added QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2
version.txt

[root@netkiller ~]# ipfs name publish
QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2
Published to QmPNH1AKXNidUgsW6MaMABiNRhdXnqpTbthbYoZzhZE9ve:
/ipfs/QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2
```

```
[root@netkiller ~]# ipfs name resolve
QmPNH1AKXNidUgsW6MaMABiNRhdXnqpTbthbYoZzhZE9ve
/ipfs/QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2
```

5.3. 将 Pin 对象存储到本地

Pinning 是 IPFS 中一个非常重要的概念。你可以从任何一个节点访问IPFS网络上的文件，如果本地没有就会去整个IPFS网络上检索。

你希望将有些文件永久保存在你的服务器中，就需要用到 pin 功能。

IPFS 中有三种 pin 类型

- direct pins 仅仅pin住一个和其它区块没有关联的独立block
- recursive pins pin住给出的block以及递归pin住它的所有children
- indirect pins recursive pin的children，不是直接被pin住的，是被递归pin住的

5.3.1. 演示 Pin 操作

```
[ipfs@netkiller ~]$ echo "helloworld" > foo

[ipfs@netkiller ~]$ ipfs add foo
added QmUU2HcUBVSXkfWPUc3WUSeCMrWWeEJTuAgR9uyWBhh9Nf foo
 11 B / 11 B
[=============================================================
===============] 100.00%

 [ipfs@netkiller ~]$ ipfs pin ls --type=all | grep
QmUU2HcUBVSXkfWPUc3WUSeCMrWWeEJTuAgR9uyWBhh9Nf
QmUU2HcUBVSXkfWPUc3WUSeCMrWWeEJTuAgR9uyWBhh9Nf recursive

[ipfs@netkiller ~]$ ipfs pin rm -r
QmUU2HcUBVSXkfWPUc3WUSeCMrWWeEJTuAgR9uyWBhh9Nf
unpinned QmUU2HcUBVSXkfWPUc3WUSeCMrWWeEJTuAgR9uyWBhh9Nf
```

```
[ipfs@netkiller ~]$ ipfs pin ls --type=all | grep
QmUU2HcUBVSXkfWPUc3WUSeCMrWWeEJTuAgR9uyWBhh9Nf
```

5.3.2. 查看 pin

```
[root@netkiller ~]# ipfs pin ls
QmQBcXurY2QBpv7sg8zyS4UXQeHGCqx4DBp86kBLPDzS18 recursive
QmTudJSaoKxtbEnTddJ9vh8hbN84ZLVvD5pNpUaSbxwGoa recursive
QmXgqKTbzdh83pQtKFb19SpMCpDDcKR2ujqk3pKph9aCNF indirect
QmcUcr4nbCDE4V1fPNZXoZa3YkCwjCv8Wd9EqLJeysAt48 indirect
QmcUnyBqYgAR5rxVQ1Pzw2DBh9evqSRBRogjshb9khJA9T indirect
QmejvEPop4D7YUadeGqYWmZxHhLc4JBUCzJJHWMzdcMe2y indirect
QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv recursive
QmTiu69XGzh1iqK6JuJNzoTiQ4SJEeQSso3uCHRwbXFba8 indirect
QmW2nvZqf6fwmcgBeeUEv2UyePizsFjjkNCubqRjmCZHNV indirect
QmY5heUM5qgRubMDD1og9fhCPA6QdkMp3QCwd4s7gJsyE7 indirect
QmYCvbfNbCwFR45HiNP45rwJgvatpiW38D961L5qAhUM5Y indirect
QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy recursive
QmPZ9gcCEpqKTo6aq61g2nXGUhM4iCL3ewB6LDXZCtioEB indirect
QmQ5vhrL7uv6tuoN9KeVBwd4PwfQkXdVVmDLUZuTNxqgvm indirect
QmRpsaigHjE4udpVDXZ8T4YtA477M7YTox6xxxkbYjisgN indirect
QmWkLpc4aCRCgPYokdXGZU3zuupRognVGVccALY4aJH31W indirect
QmWzK72EZJJhW96x1tgaz8yU3G6okJ9MfMxbLianzFLhY2 recursive
QmaHUhxr4NPTsxc2iubLAVGAvJ6hRKfZsbDPYZDT6Bdcb1 indirect
QmSLNhoZCHiRUyAsPRsLRMscu2AromEYJgJJWw229DypXT indirect
QmUNLLsPACCz1vLxQVkXqqLX5R1X345qqfHbsf67hvA3Nn recursive
QmVbXqY6kY7iFjPNw1vNGggtRmNxTVLGAziLvC9GEsMLGR recursive
QmZTR5bcpQD7cFgTorqxZDYaew1Wqgfbd2ud9QqGPAkK2V indirect
Qmbh35NHRNYfpaXaj1bQF4gcVYKRjjBvD7eys4dK4iwNrY indirect
Qmen15DKJhF5ngxiBEzmpMyPA3HpMfXNqgogeyeati2sEx indirect
```

5.4. 查看状态

5.4.1. 仓库状态

```
[root@netkiller ~]# ipfs stats repo
```

```
NumObjects: 46
RepoSize:   3004401
StorageMax: 10000000000
RepoPath:   /root/.ipfs
Version:    fs-repo@6
```

5.4.2. 带宽状态

```
[root@netkiller ~]# ipfs stats bw
Total Up    Total Down  Rate Up     Rate Down
Bandwidth
TotalIn: 20 MB
TotalOut: 4.9 MB
RateIn: 4.4 kB/s
RateOut: 3.4 kB/s
```

6. 网络命令

6.1. 显示 IPFS 信息

```
[ipfs@netkiller ~]$ ipfs id
{
        "ID": "QmZakCF5czhP53KPvMi8XQcYtVrQohw5N71Xce4eC1rWz3",
        "PublicKey":
"CAASpgIwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDYWmvcJrJsIj
gP+n/EwZyLify7KgZ3SnIrsN9k2z9iF1X9uq9tebH1Dx6tPFIoBndYCvZlrdafH
kbL5XRjJ0j0W2ISVKC75I+u9bS4WHSD7gPI82AdlWEp6elL5Muw7DnyN8flOQyx
tJqhjkMY5pSnCcLXuOmXZJ2YY+4jZSQ58QmsA17wLcn01so4impSOxjDkjuOuda
y9SxK5vmstq9Qp/h2neyUsDtx1+q0M0kjlkInRtfXCmtd14EHqcv4VgG+aCcF8U
uWgNXBTLc/R5xqtxFx8KacWjf5xUOe7Sjat7tgaY5eBOphupy5zzXfP2YigTY0c
z5jIKkPoFK+lppjAgMBAAE=",
        "Addresses": null,
        "AgentVersion": "go-ipfs/0.4.14/",
        "ProtocolVersion": "ipfs/0.1.0"
}
```

6.2. 节点

```
[ipfs@netkiller ~]$ ipfs bootstrap list
/dnsaddr/bootstrap.libp2p.io/ipfs/QmNnooDu7bfjPFoTZYxMNLWUQJyrV
wtbZg5gBMjTezGAJN
/dnsaddr/bootstrap.libp2p.io/ipfs/QmQCU2EcMqAqQPR2i9bChDtGNJchT
bq5TbXJJ16u19uLTa
/dnsaddr/bootstrap.libp2p.io/ipfs/QmbLHAnMoJPWSCR5Zhtx6BHJX9KiK
NN6tpvbUcqanj75Nb
/dnsaddr/bootstrap.libp2p.io/ipfs/QmcZf59bWwK5XFi76CZX8cbJ4BhTz
zA3gU1ZjYZcYW3dwt
/ip4/104.131.131.82/tcp/4001/ipfs/QmaCpDMGvV2BGHeYERUEnRQAwe3N8
SzbUtfsmvsqQLuvuJ
/ip4/104.236.179.241/tcp/4001/ipfs/QmSoLPppuBtQSGwKDZT2M73ULpjv
fd3aZ6ha4oFGL1KrGM
/ip4/104.236.76.40/tcp/4001/ipfs/QmSoLV4Bbm51jM9C4gDYZQ9Cy3U6aX
MJDAbzgu2fzaDs64
```

```
/ip4/128.199.219.111/tcp/4001/ipfs/QmSoLSafTMBsPKadTEgaXctDQVcq
N88CNLHXMkTNwMKPnu
/ip4/178.62.158.247/tcp/4001/ipfs/QmSoLer265NRgSp2LA3dPaeykiS1J
6DifTC88f5uVQKNAd
/ip6/2400:6180:0:d0::151:6001/tcp/4001/ipfs/QmSoLSafTMBsPKadTEg
aXctDQVcqN88CNLHXMkTNwMKPnu
/ip6/2604:a880:1:20::203:d001/tcp/4001/ipfs/QmSoLPppuBtQSGwKDZT
2M73ULpjvfd3aZ6ha4oFGL1KrGM
/ip6/2604:a880:800:10::4a:5001/tcp/4001/ipfs/QmSoLV4Bbm51jM9C4g
DYZQ9Cy3U6aXMJDAbzgu2fzaDs64
/ip6/2a03:b0c0:0:1010::23:1001/tcp/4001/ipfs/QmSoLer265NRgSp2LA
3dPaeykiS1J6DifTC88f5uVQKNAd
```

6.3. 管理P2P网络链接

```
[root@netkiller ~]# ipfs swarm peers
/ip4/104.131.131.82/tcp/4001/ipfs/QmaCpDMGvV2BGHeYERUEnRQAwe3N8
SzbUtfsmvsqQLuvuJ
```

6.4. 查看节点端口详情

```
[ipfs@netkiller ~]$ ipfs dht findpeer
QmUHKXqq4e56BRu16D8wKHsqc8Ere3w6gLLqrfMF7djTwe
/ip4/10.14.0.16/tcp/4001
/ip4/95.85.40.97/tcp/4001
/ip4/127.0.0.1/tcp/4001
/ip4/10.129.26.130/tcp/4001
/ip6/::1/tcp/4001
```

7. 配置

"StorageMax": "10GB"   存储空间配额，默认为 10G

7.1. 显示配置

```
[ipfs@netkiller ~]$ ipfs config show
{
  "API": {
    "HTTPHeaders": {
      "Server": [
        "go-ipfs/0.4.14"
      ]
    }
  },
  "Addresses": {
    "API": "/ip4/127.0.0.1/tcp/5001",
    "Announce": [],
    "Gateway": "/ip4/127.0.0.1/tcp/8080",
    "NoAnnounce": [],
    "Swarm": [
      "/ip4/0.0.0.0/tcp/4001",
      "/ip6/::/tcp/4001"
    ]
  },
  "Bootstrap": [

"/dnsaddr/bootstrap.libp2p.io/ipfs/QmNnooDu7bfjPFoTZYxMNLWUQJyr
VwtbZg5gBMjTezGAJN",

"/dnsaddr/bootstrap.libp2p.io/ipfs/QmQCU2EcMqAqQPR2i9bChDtGNJch
Tbq5TbXJJ16u19uLTa",

"/dnsaddr/bootstrap.libp2p.io/ipfs/QmbLHAnMoJPWSCR5Zhtx6BHJX9Ki
KNN6tpvbUcqanj75Nb",

"/dnsaddr/bootstrap.libp2p.io/ipfs/QmcZf59bWwK5XFi76CZX8cbJ4BhT
zzA3gU1ZjYZcYW3dwt",
```

```
"/ip4/104.131.131.82/tcp/4001/ipfs/QmaCpDMGvV2BGHeYERUEnRQAwe3N
8SzbUtfsmvsqQLuvuJ",

"/ip4/104.236.179.241/tcp/4001/ipfs/QmSoLPppuBtQSGwKDZT2M73ULpj
vfd3aZ6ha4oFGL1KrGM",

"/ip4/128.199.219.111/tcp/4001/ipfs/QmSoLSafTMBsPKadTEgaXctDQVc
qN88CNLHXMkTNwMKPnu",

"/ip4/104.236.76.40/tcp/4001/ipfs/QmSoLV4Bbm51jM9C4gDYZQ9Cy3U6a
XMJDAbzgu2fzaDs64",

"/ip4/178.62.158.247/tcp/4001/ipfs/QmSoLer265NRgSp2LA3dPaeykiS1
J6DifTC88f5uVQKNAd",

"/ip6/2604:a880:1:20::203:d001/tcp/4001/ipfs/QmSoLPppuBtQSGwKDZ
T2M73ULpjvfd3aZ6ha4oFGL1KrGM",

"/ip6/2400:6180:0:d0::151:6001/tcp/4001/ipfs/QmSoLSafTMBsPKadTE
gaXctDQVcqN88CNLHXMkTNwMKPnu",

"/ip6/2604:a880:800:10::4a:5001/tcp/4001/ipfs/QmSoLV4Bbm51jM9C4
gDYZQ9Cy3U6aXMJDAbzgu2fzaDs64",

"/ip6/2a03:b0c0:0:1010::23:1001/tcp/4001/ipfs/QmSoLer265NRgSp2L
A3dPaeykiS1J6DifTC88f5uVQKNAd"
  ],
  "Datastore": {
    "BloomFilterSize": 0,
    "GCPeriod": "1h",
    "HashOnRead": false,
    "Spec": {
      "mounts": [
        {
          "child": {
            "path": "blocks",
            "shardFunc": "/repo/flatfs/shard/v1/next-to-
last/2",
            "sync": true,
            "type": "flatfs"
          },
          "mountpoint": "/blocks",
          "prefix": "flatfs.datastore",
          "type": "measure"
        },
        {
          "child": {
```

```
          "compression": "none",
          "path": "datastore",
          "type": "levelds"
        },
        "mountpoint": "/",
        "prefix": "leveldb.datastore",
        "type": "measure"
      }
    ],
    "type": "mount"
  },
  "StorageGCWatermark": 90,
  "StorageMax": "10GB"
},
"Discovery": {
  "MDNS": {
    "Enabled": true,
    "Interval": 10
  }
},
"Experimental": {
  "FilestoreEnabled": false,
  "Libp2pStreamMounting": false,
  "ShardingEnabled": false
},
"Gateway": {
  "HTTPHeaders": {
    "Access-Control-Allow-Headers": [
      "X-Requested-With",
      "Range"
    ],
    "Access-Control-Allow-Methods": [
      "GET"
    ],
    "Access-Control-Allow-Origin": [
      "*"
    ]
  },
  "PathPrefixes": [],
  "RootRedirect": "",
  "Writable": false
},
"Identity": {
  "PeerID": "QmZakCF5czhP53KPvMi8XQcYtVrQohw5N71Xce4eC1rWz3"
},
"Ipns": {
  "RecordLifetime": "",
  "RepublishPeriod": "",
```

```
    "ResolveCacheSize": 128
  },
  "Mounts": {
    "FuseAllowOther": false,
    "IPFS": "/ipfs",
    "IPNS": "/ipns"
  },
  "Reprovider": {
    "Interval": "12h",
    "Strategy": "all"
  },
  "Swarm": {
    "AddrFilters": null,
    "ConnMgr": {
      "GracePeriod": "20s",
      "HighWater": 900,
      "LowWater": 600,
      "Type": "basic"
    },
    "DisableBandwidthMetrics": false,
    "DisableNatPortMap": false,
    "DisableRelay": false,
    "EnableRelayHop": false
  }
}
```

7.2. 修改配置

```
[ipfs@netkiller ~]$ export EDITOR=/usr/bin/vim
[ipfs@netkiller ~]$ ipfs config edit

{
  "Identity": {
    "PeerID": "QmZakCF5czhP53KPvMi8XQcYtVrQohw5N71Xce4eC1rWz3",
    "PrivKey":
"CAASpwkwggSjAgEAAoIBAQDYWmvcJrJsIjgP+n/EwZyLify7KgZ3SnIrsN9k2z
9iF1X9uq9tebH1Dx6tPFIoBndYCvZlrdafHkbL5XRjJ0j0W2ISVKC75I+u9bS4W
HSD7gPI82AdlWEp6elL5Muw7DnyN8flOQyxtJqhjkMY5pSnCcLXuOmXZJ2YY+4j
ZSQ58QmsA17wLcn01so4impSOxjDkjuOuday9SxK5vmstq9Qp/h2neyUsDtx1+q
0M0kjlkInRtfXCmtd14EHqcv4VgG+aCcF8UuWgNXBTLc/R5xqtxFx8KacWjf5xU
Oe7Sjat7tgaY5eBOphupy5zzXfP2YigTY0cz5jIKkPoFK+lppjAgMBAAECggEAE
A6l8rDsjRn9DzKISRIVjEWxfDKiSDg+QP/flJyxF5ajyzEP1BA0JPv6SuEvN8lD
EkW+A83jH+wfVQKyoKlJwNkHblTZmRhdkZ6qywPFogUIQuHNQGTV0UaLChbxBzC
```

```
BHHkHXPve9VFyKItmb3Ktlbgjvd77d0EAcU75XackCSi3mQkm73w+9YquTlJtIc
pE9RX5k4ZRrdbY3waGd5ZzGCpzUeok009wbpPOxKBObyUdzSEhLRDS5gExwGRP8
hGdxpU3Er1x6A+ojEkG3vjqCk/FoUiNyUXADKzLifjFDqFD8Ek4jIdM712YB2F8
ERfVQo/AIt3c06Gzbc+DSXJ4AQKBgQD00k08k68bHF63NsD9qbfIY8iSge6D6k7
h1USVRoQcGzeZsMHmSBUNr+sWH+nV8NZSOiej/KLoZcVlHhjqUouajZj2Ej7X4B
CWF9TYt1DsKAjCOwqImmbSoISMZWtnWuqhemI1d+dEkOaYFL2oiqmGjx00Vo5eY
9DaX9zRpREuzQKBgQDiO1swOPWAb4pl5s0IOc6/kOWFLnvkP8efM/tdwv3NYwq2
iwurfGG395jCc35XcAlccIvsXoJ1+mGzaGeZu5ppSRKu2Vqg5YOVxSzBeoBRxl/
/KK2+wOwrHaEgIbN9vcjBZVYKR87VdwLJb4PlnaSyZac71ZQMjNzo0UsGhdmN7w
KBgDoHdwM6xjCY4uJuegQmLEe1Tx9a6Nwft57T3DO9ySaYVO969BrPTx41anWOD
vEE6ugGnMrD4SFQrh8vqRwxgKGbmnwJCxhEJepNr8fGe8neG2VedTq3zlNydLiK
eZC//glUZt7hktGvvtihYesHIvOgDH4RXiGFa0W3nzGZ/J6pAoGAZSEDclsD45X
41/SEUtkEgr3S2+Ybm7ynD5O9GfzAV7+eWlttrAq94+7aapIWOB/tD1WANvlIeF
Skt/5D0YT7UXVI1MB0stfmKl0p1JNeKS/0Watlf4/eAqgMDsEB64Oa6ljSTWYsH
2BD7qfa3hnKNbUbPLQMqk+NsMVeNFxBFCECgYEA0l9L2nLelP/osGt2iQ1yRlAQ
/EdaiHe9/RE3f+EQ2tFkGDw0sj8eC/vvLhrh9fDRco46bt3Jw5JGEfW+cJ/9u7U
LvsAiWiMes0ENqGu0GE/yLD0guFrPdSJfzlWS5+mVxxOXMyHnp5xo0Vz5A3aoGt
3Lq0neaKZAYXRwYkxcQd8="
  },
  "Datastore": {
    "StorageMax": "10GB",
    "StorageGCWatermark": 90,
    "GCPeriod": "1h",
    "Spec": {
      "mounts": [
        {
          "child": {
            "path": "blocks",
            "shardFunc": "/repo/flatfs/shard/v1/next-to-
last/2",
            "sync": true,
            "type": "flatfs"
          },
          "mountpoint": "/blocks",
          "prefix": "flatfs.datastore",
          "type": "measure"
        },
        {
          "child": {
            "compression": "none",
            "path": "datastore",
            "type": "levelds"
          },
          "mountpoint": "/",
```

7.3. API 配置

查看 API 配置

```
[ipfs@netkiller ~]$ ipfs config Addresses.API
/ip4/127.0.0.1/tcp/5001
```

修改配置

```
[ipfs@netkiller ~]$ ipfs config Addresses.API
/ip4/0.0.0.0/tcp/5001
```

7.4. CORS

```
ipfs config --json API.HTTPHeaders.Access-Control-Allow-Origin
"[\"http://example.com\"]"
ipfs config --json API.HTTPHeaders.Access-Control-Allow-
Credentials "[\"true\"]"
ipfs config --json API.HTTPHeaders.Access-Control-Allow-Methods
"[\"PUT\", \"POST\", \"GET\"]"
```

7.5. 配置 API 网关

默认 API 网关是 127.0.0.1 8080 端口

```
[ipfs@netkiller ~]$ ipfs config Addresses.Gateway
/ip4/127.0.0.1/tcp/8080
```

如需远程访问设置为 0.0.0.0

```
[ipfs@netkiller ~]$ ipfs config Addresses.Gateway
/ip4/0.0.0.0/tcp/8080
[ipfs@netkiller ~]$ ipfs config Addresses.Gateway
/ip4/0.0.0.0/tcp/8080
```

8. ipfs mount

挂载前提是必须安装 fusermount

```
[root@netkiller test]# yum install fuse
```

如果没有安装 fusermount 会出如下提示

```
17:35:29.129 ERROR       node: error mounting: fusermount:
exec: "fusermount": executable file not found in $PATH
mount_unix.go:89
```

将 ipfs 挂在到本地文件系统，默认目录 /ipfs 和 /ipns。

```
> mkdir /ipfs /ipns
> sudo chown `whoami` /ipfs /ipns
> ipfs daemon &
> ipfs mount
```

挂载到指定目录

```
[root@netkiller ~]# mkdir test
[root@netkiller test]# cd test
[root@netkiller test]# mkdir ipfs ipns
[root@netkiller test]# ipfs mount -f ipfs -n ipns
IPFS mounted at: ipfs
IPNS mounted at: ipns
```

检查挂载情况

```
[root@netkiller test]# mount | grep fuse
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
/dev/fuse on /root/test/ipfs type fuse
(rw,nosuid,nodev,relatime,user_id=0,group_id=0)
/dev/fuse on /root/test/ipns type fuse
(rw,nosuid,nodev,relatime,user_id=0,group_id=0)
```

卸载

```
umount /ipfs
umount /ipns
```

9. 守护进程

启动守护进程

```
[ipfs@netkiller ~]$ ipfs daemon
Initializing daemon...
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/172.31.180.30/tcp/4001
Swarm listening on /p2p-
circuit/ipfs/QmZakCF5czhP53KPvMi8XQcYtVrQohw5N71Xce4eC1rWz3
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip4/172.31.180.30/tcp/4001
API server listening on /ip4/127.0.0.1/tcp/5001
Gateway (readonly) server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready
```

10. ipfs-update

```
[root@netkiller ~]# ipfs-update versions
v0.3.2
v0.3.4
v0.3.5
v0.3.6
v0.3.7
v0.3.8
v0.3.9
v0.3.10
v0.3.11
v0.4.0
v0.4.1
v0.4.2
v0.4.3
v0.4.4
v0.4.5
v0.4.6
v0.4.7
v0.4.8
v0.4.9
v0.4.10
v0.4.11
v0.4.12
v0.4.13
v0.4.14-rc1
v0.4.14-rc2
v0.4.14-rc3
v0.4.14
```

安装指定版本

```
[root@netkiller ~]# ipfs-update install v0.4.14
```

11. DNS 解析

IPFS 允许用户使用现有的域名系统替代HASH值，这样更便于记忆。

```
[root@netkiller ~]# ipfs cat
/ipfs/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv/readme
```

DNS设置，只需要在 DNS 解析加入一条 TXT 记录：

```
记录类型            主机记录            记录值
TXT                         ipfs
dnslink=/ipns/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv
```

IPFS 允许用户使用现有的域名系统替代HASH值，这样更便于记忆。

```
[root@netkiller ~]# ipfs cat /ipns/ipfs.netkiller.cn/readme
```

第 35 章 IPFS WebUI

1. 配置 CORS

如果你远程访问 webui 会议都 403 问题,需要配置 IPFS 然后重启 ipfs daemon

```
ipfs config show
ipfs config --json API.HTTPHeaders.Access-Control-Allow-Methods
'["PUT", "GET", "POST", "OPTIONS"]'
ipfs config --json API.HTTPHeaders.Access-Control-Allow-Origin
'["*"]'
ipfs config --json API.HTTPHeaders.Access-Control-Allow-
Credentials '["true"]'
ipfs config show
```

修改后如下

```
  "HTTPHeaders": {
      "Access-Control-Allow-Credentials": [
        "true"
      ],
      "Access-Control-Allow-Methods": [
        "PUT",
        "GET",
        "POST",
        "OPTIONS"
      ],
      "Access-Control-Allow-Origin": [
        "*"
      ],
      "Server": [
        "go-ipfs/0.4.17"
      ]
    }
  },
```

生产环境需要严格配置 CORS

```
ipfs config --json API.HTTPHeaders.Access-Control-Allow-Origin
"[\"http://example.com\"]"
ipfs config --json API.HTTPHeaders.Access-Control-Allow-
Credentials "[\"true\"]"
ipfs config --json API.HTTPHeaders.Access-Control-Allow-Methods
"[\"PUT\", \"POST\", \"GET\"]"
```

3. HTTP 网关

默认 IPFS 监听 127.0.0.1 使用下面命令启动IPFS守护进程，然后使用 nginx 代理 127.0.0.1

3.1. 查看网关地址

```
[ipfs@netkiller ~]$ ipfs config Addresses.Gateway
/ip4/127.0.0.1/tcp/8080
```

3.2. 添加测试文件

```
[root@netkiller ~]# hash=$(cat /etc/centos-release | ipfs add -
q)
[root@netkiller ~]# curl "http://127.0.0.1:8080/ipfs/$hash"
CentOS Linux release 7.4.1708 (Core)
```

3.3. 配置代理服务器

Nginx 配置

```
    server {
        listen       80 default_server;
        listen       [::]:80 default_server;
        server_name  _;

        # Load configuration files for the default server
block.
        include /etc/nginx/default.d/*.conf;
```

```
        location / {
                        proxy_pass      http://127.0.0.1:8080;
        }

        error_page 404 /404.html;
            location = /40x.html {
        }

        error_page 500 502 503 504 /50x.html;
            location = /50x.html {
        }
    }
```

```
[root@netkiller ~]# hash=$(cat /etc/centos-release | ipfs add -q)
[root@netkiller ~]# curl "http://localhost/ipfs/$hash"
CentOS Linux release 7.4.1708 (Core)
```

使用浏览器访问文件

http://localhost/ipfs/QmdoPoadYA5HYvSzgUrgXYdEVRNL1T7pY38GaWabZ3KLgn

ipfs.io 节点

```
[root@netkiller ~]# hash=$(cat /etc/centos-release | ipfs add -q)
[root@netkiller ~]# curl "https://ipfs.io/ipfs/$hash"
CentOS Linux release 7.4.1708 (Core)
```

3.5. 监听地址

或者修改配置文件直接暴漏地址出去

```
[ipfs@netkiller ~]$ export EDITOR=/usr/bin/vim
[ipfs@netkiller ~]$ ipfs config edit

"API": "/ip4/127.0.0.1/tcp/5001",
"Gateway": "/ip4/127.0.0.1/tcp/8080"

"API": "/ip4/你的IP地址/tcp/5001",
"Gateway": "/ip4/你的IP地址/tcp/8080"
```

第 36 章 IPFS 集群配置

1. 手工添加节点

实验内容，链接两个节点 A 和 B，在 A 节点上添加文件，然后 B 节点下载该文件。

实验目的，证明两个节点互通，且文件已同步两份。

准备工作，两台主机，分别安装好 go、ipfs、关闭防火墙和 SELINUX

1.1.

1.2.

第 37 章 IPFS API

1. 启动 IPFS API

查看 api 监听地址，默认只能链接 /ip4/127.0.0.1/tcp/5001

```
[ipfs@netkiller ~]$ ipfs config Addresses.API
/ip4/127.0.0.1/tcp/5001
```

修改配置，如果你的开发机需要远程链接，请设置为 /ip4/0.0.0.0/tcp/5001

```
[ipfs@netkiller ~]$ ipfs config Addresses.API
/ip4/0.0.0.0/tcp/5001
```

启动 IPFS

```
[ipfs@netkiller ~]$ ipfs daemon
```

2. 原始 HTTP API

2.1. 查看节点

```
[root@netkiller ~]# curl http://127.0.0.1:5001/api/v0/swarm/peers
```

2.2. 上传文件

```
curl -F "image=@/home/bar.jpg" 127.0.0.1:5001/api/v0/add
```

如果你熟悉 curl 命令，可以使用网页版

```
<form action="http://127.0.0.1:5001/api/v0/add">
  <input type="file" name="image" accept="image/*">
  <input type="submit">
</form>
```

3. Infura IPFS API

3.1. 查看文件

```
curl "https://ipfs.infura.io:5001/api/v0/cat?
arg=QmToBaNbJQ1uNYHr93Z7RTjdPRmUWmenhUFXhBAx2sSke5"
```

3.2. 下载文件

请求地址

```
GET https://ipfs.infura.io:5001/api/v0/get?arg=<ipfs-path>&output=
<value>&archive=false&compress=false&compression-level=-1
```

请求参数

```
arg [required] - IPFS Hash 值
output [optional] - 存储路径
archive [optional] - 打包程 tar 文件. 默认: “false” 不打包.
compress [optional] - 开启 GZIP 压缩. 默认 “false” 不开启.
compression-level [optional] - 压缩级别 (1-9). 默认是: “-1”.
```

演示

```
neo@MacBook-Pro ~ % cd /tmp
neo@MacBook-Pro /tmp % wget -q "https://ipfs.infura.io:5001/api/v0/get?
arg=QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy&archive=true" -O test.tar
neo@MacBook-Pro /tmp % tar zxvf test.tar
x QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy
neo@MacBook-Pro /tmp % cat QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy
version 1 of my text
```

4. java-ipfs-api

4.1. Maven 配置

```
  <repositories>
    <repository>
        <id>jitpack.io</id>
        <url>https://jitpack.io</url>
    </repository>
  </repositories>

        <dependencies>
                <dependency>
                        <groupId>com.github.ipfs</groupId>
                        <artifactId>java-ipfs-api</artifactId>
                        <version>v1.2.1</version>
                </dependency>
                <dependency>
                        <groupId>com.github.multiformats</groupId>
                        <artifactId>java-multibase</artifactId>
                        <version>v1.0.1</version>
                </dependency>
                <dependency>
                        <groupId>com.github.multiformats</groupId>
                        <artifactId>java-multiaddr</artifactId>
                        <version>v1.3.1</version>
                </dependency>
                <dependency>
                        <groupId>com.github.multiformats</groupId>
                        <artifactId>java-multihash</artifactId>
                        <version>v1.2.1</version>
                </dependency>
                <dependency>
                        <groupId>com.github.ipld</groupId>
                        <artifactId>java-cid</artifactId>
                        <version>v1.1.1</version>
                </dependency>
        </dependencies>
```

4.2. 查看版本号

```
package cn.netkiller.ipfs;

import io.ipfs.api.IPFS;
```

```
public class Demo {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		try {
			IPFS ipfs = new IPFS("/ip4/127.0.0.1/tcp/5001");
			System.out.println(ipfs.version());
		} catch (Exception e) {
			e.printStackTrace();
		}
	}

}
```

运行后显示 ipfs 版本号表示链接成功

4.3. 添加文件到 IPFS

首先创建一个测试文件

```
neo@MacBook-Pro ~ % echo "Helloworld" >  /tmp/hello.txt
neo@MacBook-Pro ~ % cat /tmp/hello.txt
Helloworld
```

添加本地文件到IPFS

```
	NamedStreamable.FileWrapper file = new
NamedStreamable.FileWrapper(new File("/tmp/hello.txt"));
	System.out.println(ipfs.add(file).get(0).toJSON());
```

输出结果

```
{Hash=QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN, Links=[],
Name=hello.txt, Size=19}
```

输出字符串 toJSONString()

```
                NamedStreamable.FileWrapper file = new
NamedStreamable.FileWrapper(new File("/tmp/hello.txt"));
                MerkleNode addResult = ipfs.add(file).get(0);
                System.out.println(addResult.toJSONString());
```

```
{"Hash":"QmS8R3nSbDHjQ7WRTjtX1pkiQ6BUpti9qTjweZkBgGPKiN","Links":
[],"Name":"hello.txt","Size":"19"}
```

提示

注意：文件名不会影响 Hash 值得变化

4.4. 添加文本到 IPFS

```
        public Object put() throws IOException {
                NamedStreamable.ByteArrayWrapper text = new
NamedStreamable.ByteArrayWrapper("url.txt",
"http://www.netkiller.cn".getBytes());
                MerkleNode addResult = ipfs.add(text).get(0);
                return addResult.toJSON();
        }
```

返回结果

```
{Hash=QmY1ZUDzCH7J2QcVPLWT6FKwhaVvnkFRY89GMvbD27Eyde, Links=[],
Name=url.txt, Size=31}
```

提示

注意：文件名不会影响 Hash 值得变化

4.5.

5. js-ipfs-api

5.1. 开发环境

```
mkdir projectdir
cd projectdir

npm install --save ipfs-api
```

5.2. 链接到 IPFS

```
const ipfsAPI = require('ipfs-api');
const ipfs = ipfsAPI({host: 'localhost', port: '5001',
protocol: 'http'});
```

第 38 章 IPFS Faq

1. 一个大文件将会被分块存储

创建测试文件

```
[ipfs@netkiller ~]$ dd if=/dev/zero of=test bs=258k count=1
记录了1+0 的读入
记录了1+0 的写出
264192字节(264 kB)已复制，0.000655979 秒，403 MB/秒
```

将测试文件添加到 IPFS 文件系统

```
[ipfs@netkiller ~]$ ipfs add test
added QmbZHydHNsDVdQJ5hYbCGAJCUPLF8vLdQDgSdxgHNjVKNY test
 258.00 KiB / 258.00 KiB
[============================================] 100.00%
```

查看 test 的文件块。

```
[ipfs@netkiller ~]$ ipfs ls
QmbZHydHNsDVdQJ5hYbCGAJCUPLF8vLdQDgSdxgHNjVKNY
QmRk1rduJvo5DfEYAaLobS2za9tDszk35hzaNSDCJ74DA7 262158
Qmcpc1Kk8Hhj3rVSnZGSrxzP4Fcp48cEARxMXxkHryUNJw 2059
```

第 39 章 BaaS (Blockchain as a Service) 平台

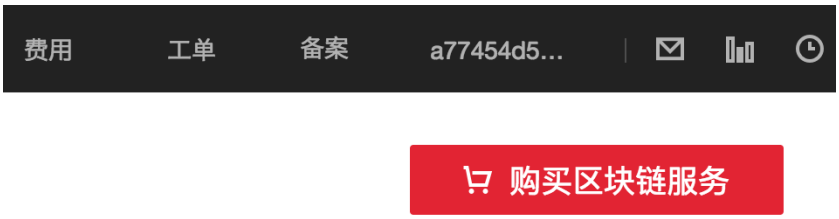
1. Huawei BCS

华为的 Hyperledger Fabric BaaS 尚处在测试阶段，申请地址是 https://www.huaweicloud.com/product/bcs.html 服务可以免费申请，但是需要

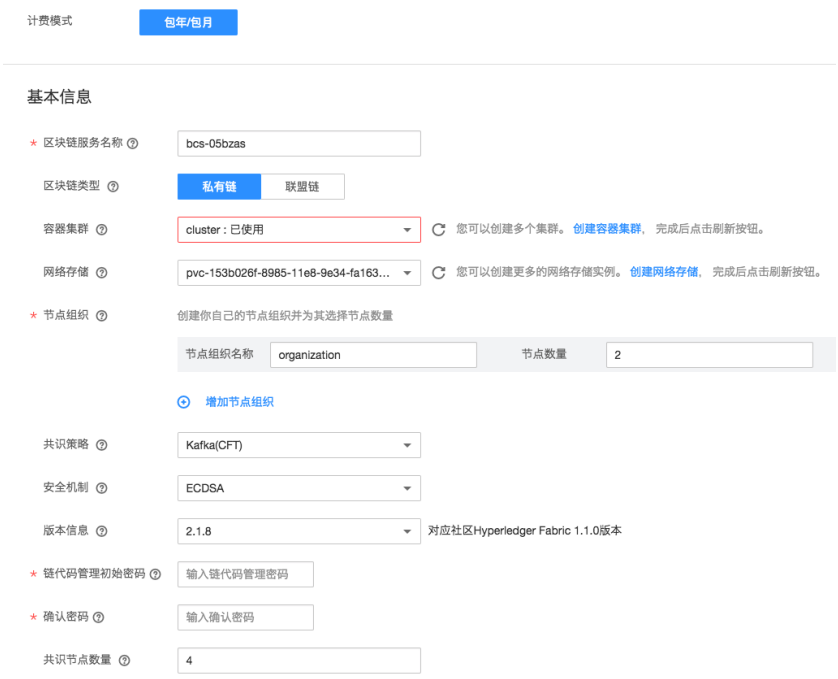
1.1. 创建 BCS 服务

打开网址 https://www.huaweicloud.com/product/bcs.html

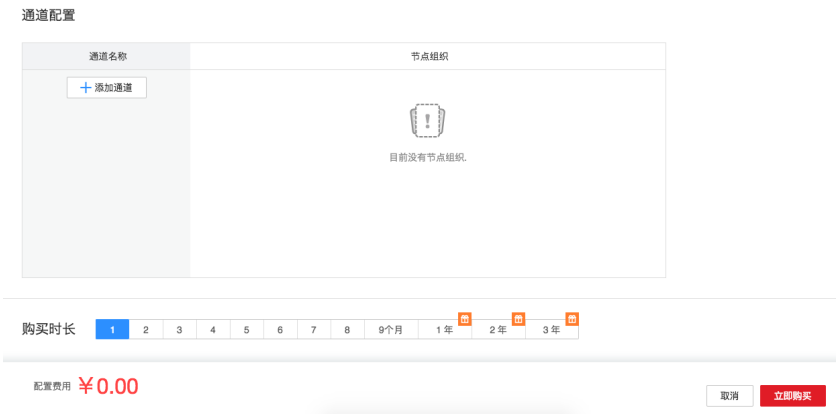
点击按钮 “立即体验”



点击“购买区块链服务”按钮



共识策略选择 Kafka(CFT), 输入链代码初始管理密码。



这里可以创建通道，也可以略过，后面可以创建。

经过一段时间初始化云主机，最终完成 BCS 创建。

1.2. 管理通道

创建通道，输入通道名称和描述，点击确定按钮。

创建好的通道，会显示“暂无节点加入”，点击右边 “加入节点” 连接

选择加入组织的数量

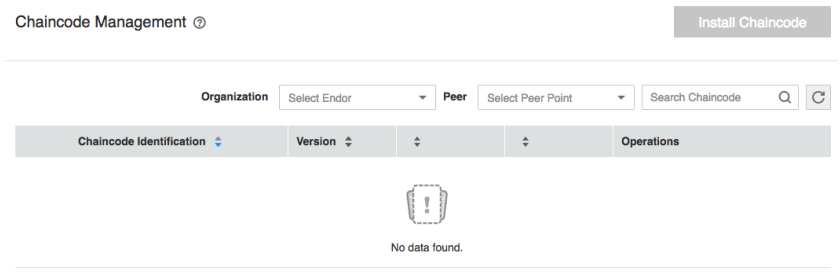
节点添加完毕

1.3. 安装链码

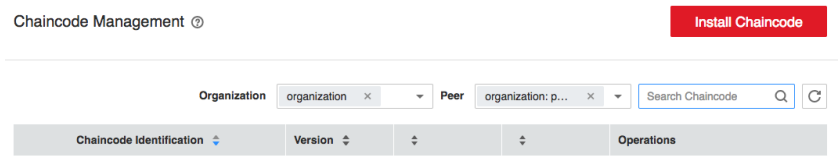
首先将 chaincode 源码压缩成 zip 文件

```
neo@MacBook-Pro ~/chaincode % zip token.zip token.go
  adding: token.go (deflated 82%)

neo@MacBook-Pro ~/chaincode % ls token.*
token.go  token.zip
```

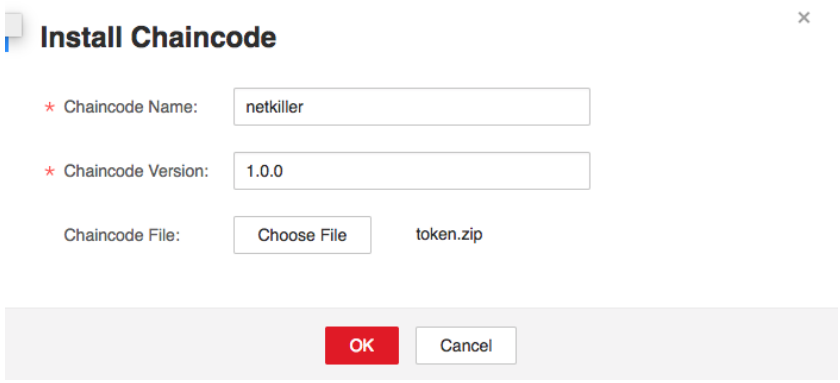
点击右边 “链代码管理” 按钮



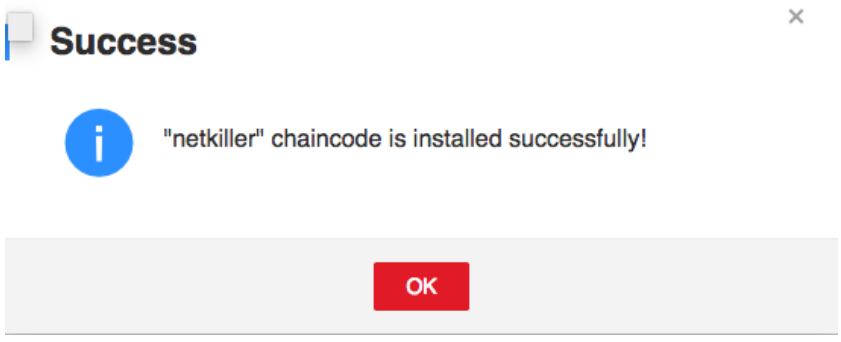
进入链码管理界面



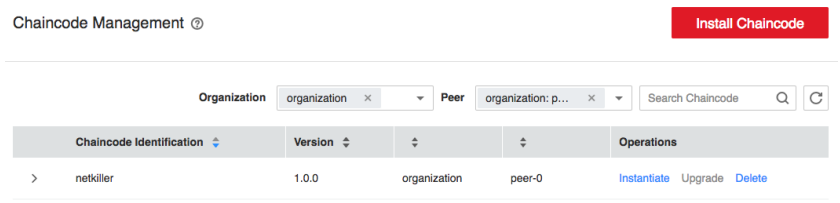
Organization 选择组织， Peer 选择节点，点击 “Install Chaincode” 按钮。



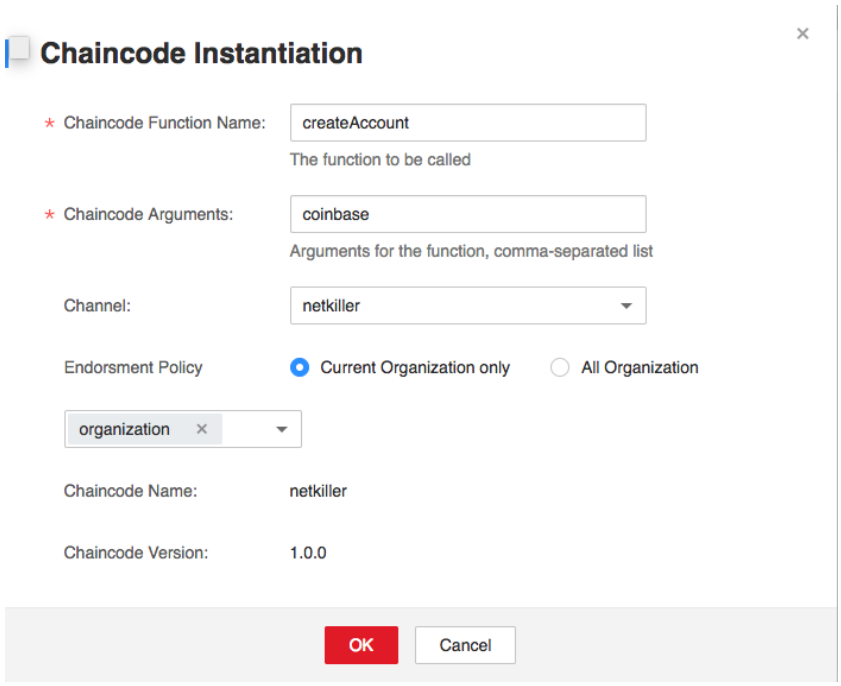
输入连码名称、版本、并上传 链码对应的 zip 文件。



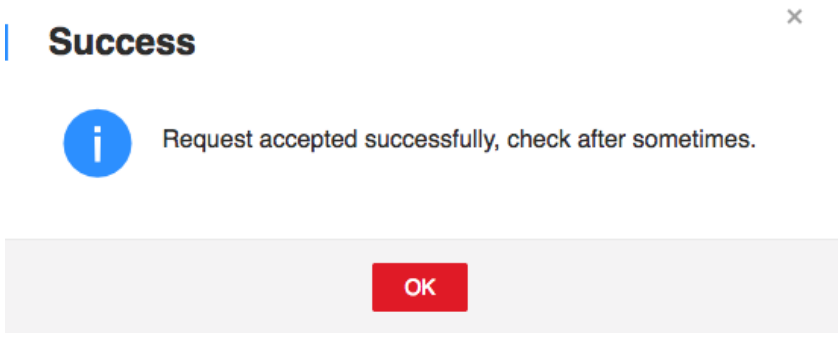
链码安装成功



实例化链码



输入调用的函数，和传递的参数，点击 OK 按钮



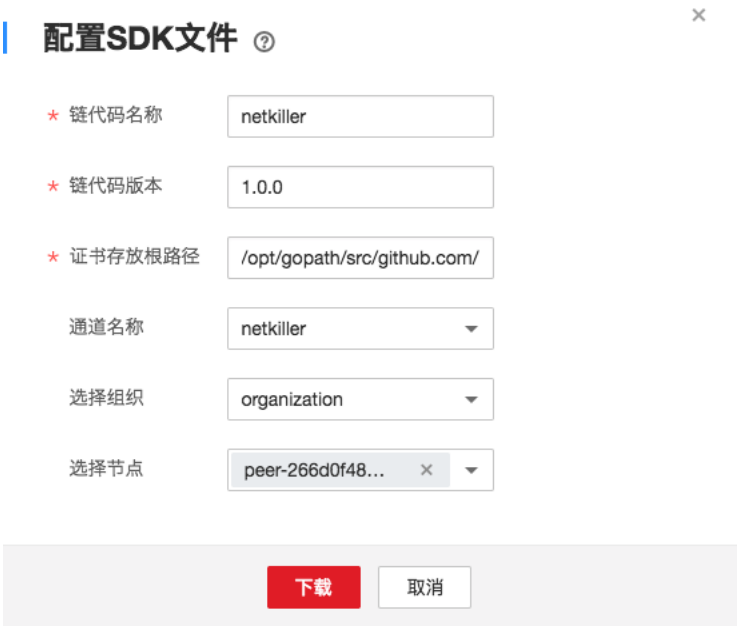
执行成功

1.4. 下载 SDK 配置

首先下载证书

点击左侧 V 符号，展开项目可以看到下载证书。点击 order 和 organization 后面的 "下载管理员证书"

选择 “下载 SDK 配置”



链代码名称是之前安装链码时输入的名称

链代码版本是当前的链码版本

证书存放根路径填写: /Users/neo/fabric/crypto (这里是本地证书路径，java / golang sdk 程序会访问这些证书。)

点击下载按钮后得到一个 bcs-itetze-sdk-config.zip 文件，解压开 bcs-itetze-sdk-config.yaml

1.5. 配置 SDK 文件

准备证书，下载 bcs-itetze-orderer-admin.zip 和 organization-admin.zip 两个证书压缩包。

bcs-itetze-orderer-admin.zip 文件解压后放置到 /Users/neo/fabric/crypto/ 目录中

organization-admin.zip 解压放置到 /Users/neo/fabric/crypto/ 目录

```
neo@MacBook-Pro ~ % cd fabric/crypto

neo@MacBook-Pro ~/fabric/crypto % pwd
/Users/neo/fabric/crypto

neo@MacBook-Pro ~/fabric/crypto % ls
266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
```

打开 bcs-itetze-sdk-config.yaml 文件，下面是 SDK 配置文件的内容。

```
name: "global-trade-network"

x-type: "hlfv1"
x-loggingLevel: info

description: "The network to be in if you want to stay in the global trade business"

version: 1.0.0

client:

  organization: 266d0f487933503a48f0ab728b85d5b469cb2b79

  logging:
```

```
    level: info

  peer:
    timeout:
      connection: 10s
      queryResponse: 45s
      executeTxResponse: 120s
  eventService:
    timeout:
      connection: 10s
      registrationResponse: 50s
  orderer:
    timeout:
      connection: 10s
      response: 45s

  cryptoconfig:
    path: /opt/gopath/src/github.com/hyperledger/fabric

  credentialStore:
    path: "/tmp/hfc-kvs"

    cryptoStore:
      path: /tmp/msp

    wallet: wallet-name

  BCCSP:
    security:
     enabled: true
     default:
      provider: "SW"
     hashAlgorithm: "SHA2"
     softVerify: true
     ephemeral: false
     level: 256

channels:

  example:
    orderers:

      - orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-0.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local

      - orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-1.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local

      - orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-2.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local

      - orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-3.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local

      - orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-4.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local

    peers:

      peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local:
        endorsingPeer: true
        chaincodeQuery: true
        ledgerQuery: true
        eventSource: true

    chaincodes:
      - example02:1.0

organizations:

  266d0f487933503a48f0ab728b85d5b469cb2b79:
```

```
    mspid: 266d0f487933503a48f0ab728b85d5b469cb2b79MSP

    cryptoPath:  /Users/neo/fabric/crypto/266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local/msp
    tlsCryptoKeyPath: /Users/neo/fabric/crypto/266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local/tls/server.key
    tlsCryptoCertPath: /Users/neo/fabric/crypto/266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local/tls/server.crt

    peers:

      - peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local

    certificateAuthorities:
      - ca-org1

    adminPrivateKey:
      pem: "-----BEGIN PRIVATE KEY----- <etc>"
    signedCert:
      path: "/tmp/somepath/signed-cert.pem"

  ordererorg:
      mspID: "7d8abfe15c3f1389f0468d90e27a382d0bd90b3fMSP"

      cryptoPath: /Users/neo/fabric/crypto/7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local/msp
#orderer eip: 49.4.85.126
orderers:

  orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-0.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:
    url: grpcs://orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-0.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:30805

    grpcOptions:
      ssl-target-name-override: orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-0.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
      grpc-max-send-message-length: 15

    tlsCACerts:
      path: /Users/neo/fabric/crypto/7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local/msp/tlscacerts/tlsca.7d8abfe1
5c3f1389f0468d90e27a382d0bd90b3f-cert.pem

  orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-1.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:
    url: grpcs://orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-1.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:30806

    grpcOptions:
      ssl-target-name-override: orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-1.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
      grpc-max-send-message-length: 15

    tlsCACerts:
      path: /Users/neo/fabric/crypto/7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local/msp/tlscacerts/tlsca.7d8abfe1
5c3f1389f0468d90e27a382d0bd90b3f-cert.pem

  orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-2.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:
    url: grpcs://orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-2.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:30807

    grpcOptions:
      ssl-target-name-override: orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-2.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
      grpc-max-send-message-length: 15

    tlsCACerts:
      path: /Users/neo/fabric/crypto/7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
```

```
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local/msp/tlscacerts/tlsca.7d8abfe1
5c3f1389f0468d90e27a382d0bd90b3f-cert.pem

  orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-3.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:
    url: grpcs://orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-3.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:30808

    grpcOptions:
      ssl-target-name-override: orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-3.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
      grpc-max-send-message-length: 15

    tlsCACerts:
      path: /Users/neo/fabric/crypto/7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local/msp/tlscacerts/tlsca.7d8abfe1
5c3f1389f0468d90e27a382d0bd90b3f-cert.pem

  orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-4.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:
    url: grpcs://orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-4.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:30809

    grpcOptions:
      ssl-target-name-override: orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-4.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
      grpc-max-send-message-length: 15

    tlsCACerts:
      path: /Users/neo/fabric/crypto/7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local/msp/tlscacerts/tlsca.7d8abfe1
5c3f1389f0468d90e27a382d0bd90b3f-cert.pem

#peer eip: 49.4.85.126
peers:

  peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local:
    url: grpcs://peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local:30608

    eventUrl: grpcs://peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local:30708

    grpcOptions:
      ssl-target-name-override: peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local
      grpc.http2.keepalive_time: 15

    tlsCACerts:
      path: /Users/neo/fabric/crypto/266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local/msp/tlscacerts/tlsca.266d0f48
7933503a48f0ab728b85d5b469cb2b79-cert.pem

certificateAuthorities:
  ca-org1:
    url: https://ca_peerOrg1:7054
    httpOptions:
      verify: true
    tlsCACerts:
      path: $GOPATH/src/github.com/hyperledger/fabric-sdk-go/test/api-
server/tls/fabricca/certs/ca_root.pem
      client:
       keyfile: $GOPATH/src/github.com/hyperledger/fabric-sdk-go/test/api-
server/tls/fabricca/certs/client/client_fabric_client-key.pem
        certfile: $GOPATH/src/github.com/hyperledger/fabric-sdk-go/test/api-
server/tls/fabricca/certs/client/client_fabric_client.pem

    registrar:
      enrollId: admin
      enrollSecret: adminpw
    caName: ca-org1
```

配置 organizations 证书

```
    adminPrivateKey:
      pem: "-----BEGIN PRIVATE KEY----- <etc>"
      path: "/Users/neo/fabric/crypto/266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local/msp/keystore"
    signedCert:
      path: "/Users/neo/fabric/crypto/266d0f487933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local/msp/signcerts/Admin@266d0f487
933503a48f0ab728b85d5b469cb2b79.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local-cert.pem"
```

找到 peers 和 orderers 的主机名

```
#peer eip: 49.4.85.126
peers:

  peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local:

#orderer eip: 49.4.85.126
orderers:

  orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-0.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local:
```

打开 /etc/hosts 文件，添加主机解析地址

```
49.4.85.126     peer-266d0f487933503a48f0ab728b85d5b469cb2b79-3.peer-
266d0f487933503a48f0ab728b85d5b469cb2b79.default.svc.cluster.local

49.4.85.126     orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-0.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
49.4.85.126     orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-1.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
49.4.85.126     orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-2.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
49.4.85.126     orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-3.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
49.4.85.126     orderer-7d8abfe15c3f1389f0468d90e27a382d0bd90b3f-4.orderer-
7d8abfe15c3f1389f0468d90e27a382d0bd90b3f.default.svc.cluster.local
```

1.6. Fabric Java SDK Demo

1.6.1. Maven pom.xml 文件

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
```

```
	<groupId>cn.netkiller</groupId>
	<artifactId>fabric-sdk-java</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<dependencies>
		<!-- https://mvnrepository.com/artifact/org.hyperledger.fabric-sdk-java/fabric-
sdk-java -->
		<dependency>
			<groupId>org.hyperledger.fabric-sdk-java</groupId>
			<artifactId>fabric-sdk-java</artifactId>
			<version>1.2.1</version>
		</dependency>

		<dependency>
			<groupId>junit</groupId>
			<artifactId>junit</artifactId>
			<version>4.11</version>
			<scope>test</scope>
		</dependency>
		<dependency>
			<groupId>log4j</groupId>
			<artifactId>log4j</artifactId>
			<version>1.2.17</version>
		</dependency>
		<dependency>
			<groupId>log4j</groupId>
			<artifactId>apache-log4j-extras</artifactId>
			<version>1.1</version>
			<scope>compile</scope>
		</dependency>
		<dependency>
			<groupId>org.slf4j</groupId>
			<artifactId>slf4j-api</artifactId>
			<version>1.7.16</version>
		</dependency>
		<dependency>
			<groupId>org.slf4j</groupId>
			<artifactId>slf4j-log4j12</artifactId>
			<version>1.7.16</version>
		</dependency>
		<!-- <dependency> <groupId>net.sf.json-lib</groupId> <artifactId>json-
lib</artifactId> <version>2.4</version> </dependency> -->
		<dependency>
			<groupId>org.junit.jupiter</groupId>
			<artifactId>junit-jupiter-api</artifactId>
			<version>RELEASE</version>
		</dependency>

	</dependencies>
	<build>
		<plugins>
			<plugin>
				<groupId>org.apache.maven.plugins</groupId>
				<artifactId>maven-compiler-plugin</artifactId>
				<version>3.6.1</version>
				<configuration>
					<source>1.8</source>
					<target>1.8</target>
				</configuration>
			</plugin>
			<plugin>
				<artifactId>maven-assembly-plugin</artifactId>
				<version>2.3</version>
				<configuration>
					<descriptorRefs>
						<descriptorRef>jar-with-
dependencies</descriptorRef>
					</descriptorRefs>
				</configuration>
				<executions>
					<execution>
						<phase>package</phase>
						<goals>
```

```
                                                        <goal>single</goal>
                                                    </goals>
                                                </execution>
                                            </executions>
                                        </plugin>
                                    </plugins>
                            </build>
</project>
```

1.6.2. chaincode_example02.go

Chaincode 是 Hyperledger Fabric官方提供的

```
/*
Copyright IBM Corp. 2016 All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

                 http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
*/

package main

//WARNING - this chaincode's ID is hard-coded in chaincode_example04 to illustrate one way of
//calling chaincode from a chaincode. If this example is modified, chaincode_example04.go has
//to be modified as well with the new ID of chaincode_example02.
//chaincode_example05 show's how chaincode ID can be passed in as a parameter instead of
//hard-coding.

import (
        "fmt"
        "strconv"

        "github.com/hyperledger/fabric/core/chaincode/shim"
        pb "github.com/hyperledger/fabric/protos/peer"
)

// SimpleChaincode example simple Chaincode implementation
type SimpleChaincode struct {
}

func (t *SimpleChaincode) Init(stub shim.ChaincodeStubInterface) pb.Response {
        fmt.Println("ex02 Init")
        _, args := stub.GetFunctionAndParameters()
        var A, B string    // Entities
        var Aval, Bval int // Asset holdings
        var err error

        if len(args) != 4 {
                return shim.Error("Incorrect number of arguments. Expecting 4")
        }

        // Initialize the chaincode
        A = args[0]
        Aval, err = strconv.Atoi(args[1])
        if err != nil {
                return shim.Error("Expecting integer value for asset holding")
        }
```

```
        B = args[2]
        Bval, err = strconv.Atoi(args[3])
        if err != nil {
                return shim.Error("Expecting integer value for asset holding")
        }
        fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

        // Write the state to the ledger
        err = stub.PutState(A, []byte(strconv.Itoa(Aval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        err = stub.PutState(B, []byte(strconv.Itoa(Bval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        return shim.Success(nil)
}

func (t *SimpleChaincode) Invoke(stub shim.ChaincodeStubInterface) pb.Response {
        fmt.Println("ex02 Invoke")
        function, args := stub.GetFunctionAndParameters()
        if function == "invoke" {
                // Make payment of X units from A to B
                return t.invoke(stub, args)
        } else if function == "delete" {
                // Deletes an entity from its state
                return t.delete(stub, args)
        } else if function == "query" {
                // the old "Query" is now implemtned in invoke
                return t.query(stub, args)
        }

        return shim.Error("Invalid invoke function name. Expecting \"invoke\" \"delete\"
\"query\"")
}

// Transaction makes payment of X units from A to B
func (t *SimpleChaincode) invoke(stub shim.ChaincodeStubInterface, args []string) pb.Response {
        var A, B string    // Entities
        var Aval, Bval int // Asset holdings
        var X int          // Transaction value
        var err error

        if len(args) != 3 {
                return shim.Error("Incorrect number of arguments. Expecting 3")
        }

        A = args[0]
        B = args[1]

        // Get the state from the ledger
        // TODO: will be nice to have a GetAllState call to ledger
        Avalbytes, err := stub.GetState(A)
        if err != nil {
                return shim.Error("Failed to get state")
        }
        if Avalbytes == nil {
                return shim.Error("Entity not found")
        }
        Aval, _ = strconv.Atoi(string(Avalbytes))

        Bvalbytes, err := stub.GetState(B)
        if err != nil {
                return shim.Error("Failed to get state")
        }
        if Bvalbytes == nil {
                return shim.Error("Entity not found")
        }
        Bval, _ = strconv.Atoi(string(Bvalbytes))
```

```
        // Perform the execution
        X, err = strconv.Atoi(args[2])
        if err != nil {
                return shim.Error("Invalid transaction amount, expecting a integer value")
        }
        Aval = Aval - X
        Bval = Bval + X
        fmt.Printf("Aval = %d, Bval = %d\n", Aval, Bval)

        // Write the state back to the ledger
        err = stub.PutState(A, []byte(strconv.Itoa(Aval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        err = stub.PutState(B, []byte(strconv.Itoa(Bval)))
        if err != nil {
                return shim.Error(err.Error())
        }

        return shim.Success(nil)
}

// Deletes an entity from state
func (t *SimpleChaincode) delete(stub shim.ChaincodeStubInterface, args []string) pb.Response {
        if len(args) != 1 {
                return shim.Error("Incorrect number of arguments. Expecting 1")
        }

        A := args[0]

        // Delete the key from the state in ledger
        err := stub.DelState(A)
        if err != nil {
                return shim.Error("Failed to delete state")
        }

        return shim.Success(nil)
}

// query callback representing the query of a chaincode
func (t *SimpleChaincode) query(stub shim.ChaincodeStubInterface, args []string) pb.Response {
        var A string // Entities
        var err error

        if len(args) != 1 {
                return shim.Error("Incorrect number of arguments. Expecting name of the person
to query")
        }

        A = args[0]

        // Get the state from the ledger
        Avalbytes, err := stub.GetState(A)
        if err != nil {
                jsonResp := "{\"Error\":\"Failed to get state for " + A + "\"}"
                return shim.Error(jsonResp)
        }

        if Avalbytes == nil {
                jsonResp := "{\"Error\":\"Nil amount for " + A + "\"}"
                return shim.Error(jsonResp)
        }

        jsonResp := "{\"Name\":\"" + A + "\",\"Amount\":\"" + string(Avalbytes) + "\"}"
        fmt.Printf("Query Response:%s\n", jsonResp)
        return shim.Success(Avalbytes)
}

func main() {
        err := shim.Start(new(SimpleChaincode))
```

```
        if err != nil {
                fmt.Printf("Error starting Simple chaincode: %s", err)
        }
}
```

1.6.3. bcs-whbsxu-sdk-config.yaml

```
name: "global-trade-network"

x-type: "hlfv1"
x-loggingLevel: info

description: "The network to be in if you want to stay in the global trade business"

version: 1.0.0

client:

  organization: 23e50b60552eb5b6f32d3c1563305dd4530dc2f0

  logging:
    level: info

  peer:
    timeout:
      connection: 10s
      queryResponse: 45s
      executeTxResponse: 120s
  eventService:
    timeout:
      connection: 10s
      registrationResponse: 50s
  orderer:
    timeout:
      connection: 10s
      response: 45s

  cryptoconfig:
    path: /opt/gopath/src/github.com/hyperledger/fabric

  credentialStore:
    path: "/tmp/hfc-kvs"

    cryptoStore:
      path: /tmp/msp

    wallet: wallet-name

  BCCSP:
    security:
     enabled: true
     default:
      provider: "SW"
     hashAlgorithm: "SHA2"
     softVerify: true
     ephemeral: false
     level: 256

channels:

  artbank:
    orderers:

      - orderer-c18fb08fe08fca399fa6baf66bee2f59e50fcea5-0.orderer-
c18fb08fe08fca399fa6baf66bee2f59e50fcea5.default.svc.cluster.local
```

```
    peers:

      peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local:
        endorsingPeer: true
        chaincodeQuery: true
        ledgerQuery: true
        eventSource: true

      peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-1.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local:
        endorsingPeer: true
        chaincodeQuery: true
        ledgerQuery: true
        eventSource: true

    chaincodes:
      - example:1.0

organizations:

  23e50b60552eb5b6f32d3c1563305dd4530dc2f0:
    mspid: 23e50b60552eb5b6f32d3c1563305dd4530dc2f0MSP

    cryptoPath:  /opt/fabric/23e50b60552eb5b6f32d3c1563305dd4530dc2f0.peer/msp
    tlsCryptoKeyPath: /opt/fabric/23e50b60552eb5b6f32d3c1563305dd4530dc2f0.peer/tls/server.key
    tlsCryptoCertPath: /opt/fabric/23e50b60552eb5b6f32d3c1563305dd4530dc2f0.peer/tls/server.crt

    peers:

      - peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local

      - peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-1.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local

    certificateAuthorities:
      - ca-org1

  ordererorg:
      mspID: "c18fb08fe08fca399fa6baf66bee2f59e50fcea5MSP"

      cryptoPath: /opt/fabric/c18fb08fe08fca399fa6baf66bee2f59e50fcea5.orderer/msp
orderer-eip: 49.4.15.203
orderers:

  orderer-c18fb08fe08fca399fa6baf66bee2f59e50fcea5-0.orderer-
c18fb08fe08fca399fa6baf66bee2f59e50fcea5.default.svc.cluster.local:
    url: grpcs://49.4.15.203:30805

    grpcOptions:
      ssl-target-name-override: orderer-c18fb08fe08fca399fa6baf66bee2f59e50fcea5-0.orderer-
c18fb08fe08fca399fa6baf66bee2f59e50fcea5.default.svc.cluster.local
      grpc-max-send-message-length: 15
      sslProvider: openSSL
      negotiationType: TLS
      hostnameOverride: orderer-c18fb08fe08fca399fa6baf66bee2f59e50fcea5-0.orderer-
c18fb08fe08fca399fa6baf66bee2f59e50fcea5.default.svc.cluster.local

    tlsCACerts:
      path:
/opt/fabric/c18fb08fe08fca399fa6baf66bee2f59e50fcea5.orderer/msp/tlscacerts/tlsca.c18fb08fe08fca
399fa6baf66bee2f59e50fcea5-cert.pem

peer-eip: 49.4.15.203
peers:

  peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local:
    url: grpcs://49.4.15.203:30605
```

```
    eventUrl: grpcs://49.4.15.203:30705

    grpcOptions:
      ssl-target-name-override: peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local
      grpc.http2.keepalive_time: 15
      sslProvider: openSSL
      negotiationType: TLS
      hostnameOverride: peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local

    tlsCACerts:
      path:
/opt/fabric/23e50b60552eb5b6f32d3c1563305dd4530dc2f0.peer/msp/tlscacerts/tlsca.23e50b60552eb5b6f
32d3c1563305dd4530dc2f0-cert.pem

  peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-1.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local:
    url: grpcs://49.4.15.203:30606

    eventUrl: grpcs://49.4.15.203:30706

    grpcOptions:
      ssl-target-name-override: peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-1.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local
      grpc.http2.keepalive_time: 15
      sslProvider: openSSL
      negotiationType: TLS
      hostnameOverride: peer-23e50b60552eb5b6f32d3c1563305dd4530dc2f0-1.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local

    tlsCACerts:
      path:
/opt/fabric/23e50b60552eb5b6f32d3c1563305dd4530dc2f0.peer/msp/tlscacerts/tlsca.23e50b60552eb5b6f
32d3c1563305dd4530dc2f0-cert.pem

certificateAuthorities:
  ca-org1:
    url: https://ca_peerOrg1:7054
    httpOptions:
      verify: true
    tlsCACerts:
      path: $GOPATH/src/github.com/hyperledger/fabric-sdk-go/test/api-
server/tls/fabricca/certs/ca_root.pem
      client:
       keyfile: $GOPATH/src/github.com/hyperledger/fabric-sdk-go/test/api-
server/tls/fabricca/certs/client/client_fabric_client-key.pem
       certfile: $GOPATH/src/github.com/hyperledger/fabric-sdk-go/test/api-
server/tls/fabricca/certs/client/client_fabric_client.pem

    registrar:
      enrollId: admin
      enrollSecret: adminpw
    caName: ca-org1
```

1.6.4. FabricHelper.java

```
package cn.netkiller.fabric;


import org.apache.commons.io.IOUtils;
import org.bouncycastle.asn1.pkcs.PrivateKeyInfo;
import org.bouncycastle.openssl.PEMParser;
import org.bouncycastle.openssl.jcajce.JcaPEMKeyConverter;
import org.hyperledger.fabric.sdk.*;
import org.hyperledger.fabric.sdk.NetworkConfig.OrgInfo;
```

```
import org.hyperledger.fabric.sdk.exception.InvalidArgumentException;
import org.hyperledger.fabric.sdk.security.CryptoSuite;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.yaml.snakeyaml.Yaml;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.io.Reader;
import java.io.StringReader;
import java.security.PrivateKey;
import java.util.*;
import java.util.concurrent.TimeUnit;

import javax.json.Json;
import javax.json.JsonObject;
import javax.json.JsonValue;

import static java.nio.charset.StandardCharsets.UTF_8;

public class FabricHelper {
    private Logger logger = LoggerFactory.getLogger(FabricHelper.class);

        private String ymlFileName = "./huawei.yaml";
        private String channelName;
        private String chaincodeName;
        private String accessKey;
    private Map<String, HFClient> clientMap;
    private Map<String, Channel> channelMap;
    private FabricHelper() {
        channelMap = new HashMap<>();
        clientMap = new HashMap<>();
    }

    private static class Holder {
        private static FabricHelper instance = new FabricHelper();
    }

    public static FabricHelper getInstance(){
        return Holder.instance;
    }

    public void setConfigCtx(String configPath) {
        if (configPath == null || configPath.equals("")) {
            logger.error("config path is empty! please input correct ymal path!");
        }
        this.ymlFileName = configPath;
        try {
            InputStream stream = new FileInputStream(new File(this.ymlFileName));
            Yaml yaml = new Yaml();
            Map<String, Object> confYaml = yaml.load(stream);
            JsonObject confJson = Json.createObjectBuilder(confYaml).build();

            this.accessKey = confJson.getJsonObject("client").getString("organization");

            JsonObject channels = confJson.getJsonObject("channels");
            //因为只有一个channel 所以只需获取键名即可
            String chanNameTemp = channels.keySet().toString();
            this.channelName = chanNameTemp.substring(chanNameTemp.indexOf("[") + 1,
chanNameTemp.indexOf("]"));

            // 因为只有一个chaincode 所以只需取chaincode数组的第一个
            String codeNameTemp =
channels.getJsonObject(this.channelName).getJsonArray("chaincodes").getString(0);
            this.chaincodeName = codeNameTemp.substring(0, codeNameTemp.indexOf(":"));
        }
        catch (Exception e)
        {
```

```
            e.printStackTrace();
            logger.error(e.getMessage());
        }
    }

    public FabricHelper setConfigPath(String configPath){
        this.ymlFileName=configPath;
        return this;
    }

    public FabricHelper setChaincodeName(String chaincodeName) {
        this.chaincodeName=chaincodeName;
        return this;
    }

    public FabricHelper setChannelName(String channelName) {
        this.channelName=channelName;
        return this;
    }

    public FabricHelper setAccessKey(String accesskey) {
        this.accessKey=accesskey;
        return this;
    }

    private NetworkConfig loadfromYamlFile(String fileName) {
        try {
            return NetworkConfig.fromYamlFile(new File(fileName));

        } catch (Exception e) {
            String msg = "can't load yaml file: " + fileName;
            logger.error(msg, e);
            return null;
        }
    }

    private Channel buildChannel(String channelName, NetworkConfig networkConfig, HFClient
client) {
        try {
            FabricUser user = genFabricUser(accessKey);
            client.setUserContext(user);
            Channel channel = client.loadChannelFromConfig(channelName, networkConfig);
            channel.initialize();
            return channel;
        } catch (Exception e) {
            String msg = "can't construct channel: " + networkConfig.getClientOrganization();
            logger.error(msg, e);
            return null;
        }
    }

    private HFClient getClient(String orgName) {
        HFClient client = clientMap.get(orgName);
        if (client == null) {
            synchronized (clientMap) {
                client = clientMap.get(orgName);
                if (client != null) {
                        return client;
                }

                client = HFClient.createNewInstance();
                try {
                    client.setCryptoSuite(getCryptoSuite());
                    clientMap.put(orgName, client);
                } catch (Exception e) {
                    String msg = "can't construct client: " + orgName;
                    logger.error(msg, e);
                    System.out.println(msg);
                    return null;
                }
            }
        }
```

```
        return client;
    }

    private CryptoSuite getCryptoSuite() throws java.lang.IllegalAccessException,
java.lang.InstantiationException, java.lang.ClassNotFoundException,
org.hyperledger.fabric.sdk.exception.CryptoException,
org.hyperledger.fabric.sdk.exception.InvalidArgumentException, java.lang.NoSuchMethodException,
java.lang.reflect.InvocationTargetException, FileNotFoundException{
        CryptoSuite cs = null;
        InputStream stream = new FileInputStream(this.ymlFileName);
                Map<String, Object> map = new Yaml().load(stream);
                JsonObject root = Json.createObjectBuilder(map).build();
        String hashAlgo
=root.getJsonObject("client").getJsonObject("BCCSP").getJsonObject("security").getString("hashAl
gorithm");
        //for sm Algorithm
        if (hashAlgo.equals("SM3")){
                Properties properties = new Properties();
            properties.setProperty("org.hyperledger.fabric.sdk.hash_algorithm", "SM3");
            properties.setProperty("org.hyperledger.fabric.sdk.crypto.default_signature_userid",
"1234567812345678");
                cs = CryptoSuite.Factory.getCryptoSuite(properties);
        }else{
                cs = CryptoSuite.Factory.getCryptoSuite();
        }
        return cs;
    }

    private Channel getChannel(String accessKey, HFClient client) {
        Channel channel = channelMap.get(accessKey);
        if (channel == null) {
            synchronized (channelMap) {
                channel = channelMap.get(accessKey);
                if (channel != null) {
                   return channel;
                }

                NetworkConfig networkConfig = loadfromYamlFile(ymlFileName);
                if (networkConfig == null) {
                    return null;
                }

                try {

                    networkConfig.getOrdererNames().forEach(item -> {
                        try {
                                Properties p = networkConfig.getOrdererProperties(item);
                                p.setProperty("hostnameOverride", item);
                            p.setProperty("clientCertFile", GetTlsCert(ymlFileName,
"ordererorg"));
                            p.setProperty("clientKeyFile",  GetTlsKey(ymlFileName,
"ordererorg"));
                            networkConfig.setOrdererProperties(item, p);
                        } catch (InvalidArgumentException e) {
                            throw new RuntimeException(e);
                        }
                    });

                    networkConfig.getPeerNames().forEach(item -> {
                        try {
                                Properties p = networkConfig.getPeerProperties(item);
                                String orgId = getOrgIdByPeer(networkConfig,item);
                                p.setProperty("hostnameOverride", item);
                            p.setProperty("clientCertFile", GetTlsCert(ymlFileName, orgId));
                            p.setProperty("clientKeyFile",  GetTlsKey(ymlFileName, orgId));
                            networkConfig.setPeerProperties(item, p);
                        } catch (InvalidArgumentException e) {
                            throw new RuntimeException(e);
                        }
                    });
```

```
                networkConfig.getEventHubNames().forEach(item -> {
                    try {
                            Properties p = networkConfig.getEventHubsProperties(item);
                            String orgId = getOrgIdByPeer(networkConfig,item);
                            p.setProperty("hostnameOverride", item);
                        p.setProperty("clientCertFile", GetTlsCert(ymlFileName, orgId));
                        p.setProperty("clientKeyFile",  GetTlsKey(ymlFileName, orgId));
                        networkConfig.setEventHubProperties(item, p);
                    } catch (InvalidArgumentException e) {
                        throw new RuntimeException(e);
                    }
                });

                networkConfig.getChannelNames().forEach( item -> {
                            if (channelName!="" ){
                                        channelName = item;
                            }
                });

            } catch (Exception e) {
                String msg = "can't get channel: " + accessKey;
                logger.error(msg, e);
                return null;
            }

            channel = buildChannel(channelName, networkConfig, client);
            if (channel != null) {
                channelMap.put(accessKey, channel);
            }
        }
    }

    return channel;
}

private String getCryptoPath(String configFile, String orgId) {
            JsonObject root = null;
            try {
                    InputStream stream = new FileInputStream(configFile);
                    Yaml yaml = new Yaml();
                    Map<String, Object> map = yaml.load(stream);
                    root = Json.createObjectBuilder(map).build();
            } catch (FileNotFoundException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
            }

            JsonObject orgs = root.getJsonObject("organizations");
            for (Map.Entry<String, JsonValue> o : orgs.entrySet()) {
                    if (orgId.equals(o.getKey())) {
                            JsonObject v = (JsonObject) o.getValue();
                            return v.getString("cryptoPath");
                    }
            }
            return "";
    }

private String GetTlsCert(String configFile, String orgId){
            String msp = getCryptoPath(configFile, orgId);
            int index = msp.lastIndexOf("msp");
            String ret = msp.substring(0, index)+"tls/server.crt";
            logger.debug("tls cert for " + orgId +",path:"+ ret);
            return ret;
    }

    private String GetTlsKey(String configFile, String orgId){
            String msp = getCryptoPath(configFile, orgId);
            int index = msp.lastIndexOf("msp");
            String ret = msp.substring(0, index)+"tls/server.key";
            logger.debug("tls key for " + orgId +",path:"+ ret);
            return ret;
```

```
    }

    private String getOrgIdByPeer(NetworkConfig config, String peerName){
            for ( OrgInfo o : config.getOrganizationInfos()) {
                    for (String p : o.getPeerNames()){
                            if ( p.equals(peerName) ){
                                    return o.getName();
                            }
                    }
            }
            return "";
    }

    private FabricUser genFabricUser(String accessKey) {
            FabricUser user = new FabricUser(accessKey);
            String msp = getCryptoPath(this.ymlFileName, accessKey);
            String adminPrivateKeyString = extractPemString(msp,"keystore");
            String signedCert = extractPemString(msp, "signcerts");

            PrivateKey privateKey = null;
    try {
        privateKey = getPrivateKeyFromString(adminPrivateKeyString);
    } catch (IOException ioe) {
        ioe.printStackTrace();
    }

    final PrivateKey privateKeyFinal = privateKey;
            user.setEnrollment(new Enrollment() {
        @Override
        public PrivateKey getKey() {
            return privateKeyFinal;
        }

        @Override
        public String getCert() {
            return signedCert;
        }
    });
            return user;
    }

    private String extractPemString(String path, String sub){
                String pemString = "";
        File dir = new File(path + "/" + sub );
        if (!dir.exists()){
            logger.error("directory is not exist. path:" + dir);
            return "";
        }

        for (File f : dir.listFiles()){
                            try {
                                    FileInputStream  stream = new FileInputStream(f);
                                    pemString = IOUtils.toString(stream, "UTF-8");
                    return pemString;
                            } catch (Exception e) {
                                    // TODO Auto-generated catch block
                                    e.printStackTrace();
                            }
        }
    return pemString;
}
    private static PrivateKey getPrivateKeyFromString(String data)
        throws IOException {
    final Reader pemReader = new StringReader(data);
    final PrivateKeyInfo pemPair;
    try (PEMParser pemParser = new PEMParser(pemReader)) {
        pemPair = (PrivateKeyInfo) pemParser.readObject();
    }
    return new JcaPEMKeyConverter().getPrivateKey(pemPair);
}
```

```
    public boolean invokeBlockchain(String method, String[] args) {
        HFClient client = getClient(accessKey);
        if (client == null) {
                return false;
        }

        Channel channel = getChannel(accessKey, client);
        if (channel == null) {
            return false;
        }

        Collection<ProposalResponse> successful = new LinkedList<>();
        Collection<ProposalResponse> failed = new LinkedList<>();

        try {
            TransactionProposalRequest req = client.newTransactionProposalRequest();
            ChaincodeID cid = ChaincodeID.newBuilder().setName(chaincodeName).build();
            req.setChaincodeID(cid);
            req.setFcn(method);
            req.setArgs(args);

            Map<String, byte[]> tm2 = new HashMap<>();
            tm2.put("HyperLedgerFabric", "TransactionProposalRequest:JavaSDK".getBytes(UTF_8));
            tm2.put("method", "TransactionProposalRequest".getBytes(UTF_8));
            tm2.put("result", ":)".getBytes(UTF_8));  /// This should be returned see chaincode.
            req.setTransientMap(tm2);

            Collection<ProposalResponse> resps = channel.sendTransactionProposal(req);

            for (ProposalResponse response : resps) {
                if (response.getStatus() == ProposalResponse.Status.SUCCESS) {
                    successful.add(response);
                } else {
                    failed.add(response);
                }
            }

            // Check that all the proposals are consistent with each other. We should have only
one set
            // where all the proposals above are consistent.
            Collection<Set<ProposalResponse>> proposalConsistencySets =
SDKUtils.getProposalConsistencySets(resps);
            if (proposalConsistencySets.size() != 1) {
                logger.error("Expected only one set of consistent proposal responses but got {}:
{}" + proposalConsistencySets.size(), args.toString());
                return false;
            }

            if (failed.size() > 0) {
                ProposalResponse firstTransactionProposalResponse = failed.iterator().next();
                logger.error("Not enough endorsers for {}: {}. endorser error: {}, Was verified:
{}",  args, failed.size(),
                        firstTransactionProposalResponse.getMessage(),
firstTransactionProposalResponse.isVerified());
                return false;
            }

            BlockEvent.TransactionEvent transactionEvent =
channel.sendTransaction(successful).get(30, TimeUnit.SECONDS);
            if (transactionEvent.isValid()) {
                logger.info("Finished transaction with transaction id {}: {}",
transactionEvent.getTransactionID(), args);
                return true;
            } else {
                logger.error("can't commit result: {}", args.toString());
                return false;
            }
        } catch (Exception e) {
            String msg = "can't put record to blockchain: " + args;
            logger.error(msg, e);
            return false;
        }
```

```
    }

    public String queryBlockchain(String method, String[] params) {
        HFClient client = getClient(accessKey);
        if (client == null) {
            return "{1}";
        }

        Channel channel = getChannel(accessKey, client);
        if (channel == null)  {
            return "{2}";
        }

        try {

            QueryByChaincodeRequest queryByChaincodeRequest = client.newQueryProposalRequest();
            queryByChaincodeRequest.setArgs(params);
            queryByChaincodeRequest.setFcn(method);

queryByChaincodeRequest.setChaincodeID(ChaincodeID.newBuilder().setName(chaincodeName).build());

            Map<String, byte[]> tm2 = new HashMap<>();
            tm2.put("HyperLedgerFabric", "QueryByChaincodeRequest:JavaSDK".getBytes(UTF_8));
            tm2.put("method", "QueryByChaincodeRequest".getBytes(UTF_8));
            queryByChaincodeRequest.setTransientMap(tm2);

            String payload = null;
            Collection<ProposalResponse> queryProposals =
channel.queryByChaincode(queryByChaincodeRequest, channel.getPeers());
            for (ProposalResponse proposalResponse : queryProposals) {
                if (!proposalResponse.isVerified() || proposalResponse.getStatus() !=
ProposalResponse.Status.SUCCESS) {
                    logger.error("Failed query proposal from peer " +
proposalResponse.getPeer().getName() + " status: " + proposalResponse.getStatus() +
                            ". Messages: " + proposalResponse.getMessage()
                            + ". Was verified : " + proposalResponse.isVerified());
                } else {
                    payload =
proposalResponse.getProposalResponse().getResponse().getPayload().toStringUtf8();
                    logger.info("Query payload from peer {} returned {}",
proposalResponse.getPeer().getName(), payload);
                    break;
                }
            }

            return !payload.equals("null") ? payload : "{}";
        } catch (Exception e) {
            String msg = "can't query record from blockchain. condition: " +  accessKey + "," +
params;
            logger.error(msg);
            return "{3}";
        }
    }
}
```

1.6.5. FabricUser.java

```
package cn.netkiller.fabric;

import java.util.Set;

import org.hyperledger.fabric.sdk.Enrollment;
import org.hyperledger.fabric.sdk.User;

public class FabricUser implements User {
```

```
public void setName(String name) {
        this.name = name;
}

protected String name;
protected String enrollSecret;
protected String mspid;
private Set<String> roles;
private String account;
private String affiliation;
private Enrollment enrollment;

public void setEnrollSecret(String enrollSecret) {
        this.enrollSecret = enrollSecret;
}

public String getMspid() {
        return mspid;
}

public void setMspid(String mspid) {
        this.mspid = mspid;
}

public void setRoles(Set<String> roles) {
        this.roles = roles;
}

public void setAccount(String account) {
        this.account = account;
}

public void setAffiliation(String affiliation) {
        this.affiliation = affiliation;
}

public void setEnrollment(Enrollment enrollment) {
        this.enrollment = enrollment;
}

public FabricUser(String accessKey) {
        this.name = accessKey;
        this.mspid = accessKey + "MSP";
}

public String getEnrollSecret() {
        return enrollSecret;
}

@Override
public String getName() {
        return name;
}

@Override
public Set<String> getRoles() {
        return roles;
}

@Override
public String getAccount() {
        return account;
}

@Override
public String getAffiliation() {
        return affiliation;
}

@Override
public Enrollment getEnrollment() {
        return enrollment;
```

```
        }

        public String getMspId() {
                return mspid;
        }

}
```

1.6.6. Main.java

```
package cn.netkiller.fabric;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import java.util.concurrent.CountDownLatch;

/**
 * run this main method and check result
 */
public class Main {
        private static Logger logger = LoggerFactory.getLogger(Main.class);

        public static void main(String args[]) throws Exception {
                FabricHelper helper = FabricHelper.getInstance();
                helper.setConfigCtx("/Users/neo/workspace/fabric-sdk-
java/src/main/resources/fixture/config/bcs-whbsxu-sdk-config.yaml");
                LoopInvoke(1);
                // StartMultiTask(1,1);
        }

        public static void LoopInvoke(int loop) throws Exception {
                FabricHelper helper = FabricHelper.getInstance();
                for (int i = 0; i < loop; i++) {
                        helper.invokeBlockchain("invoke", new String[] { "a", "b", "100" });
                        String a = helper.queryBlockchain("query", new String[] { "a" });
                        String b = helper.queryBlockchain("query", new String[] { "b" });
                        logger.info("after invoke  a=" + a + ", invoke b=" + b);
                }
        }

        // StartMultiTask(1, 1);
        public static void StartMultiTask(int threadNumber, int loop) throws
InterruptedException {
                final CountDownLatch countDownLatch = new CountDownLatch(threadNumber);
                for (int i = 0; i < threadNumber; i++) {
                        final int threadID = i;
                        new Thread() {
                                public void run() {
                                        try {
                                                LoopInvoke(loop);
                                        } catch (Exception e) {
                                                // TODO Auto-generated catch block
                                                e.printStackTrace();
                                        }
                                        logger.info("threadID:[%s] finished!!", threadID);
                                        countDownLatch.countDown();
                                }
                        }.start();
                }

                countDownLatch.await();
                logger.info("main thread finished!!");
        }
}
```

1.6.7. 运行结果

```
2018-09-26 09:58:19,065 WARN [org.hyperledger.fabric.sdk.helper.Config] - Failed to load any
configuration from: config.properties. Using toolkit defaults
2018-09-26 09:58:24,254 INFO [cn.netkiller.fabric.FabricHelper] - Finished transaction with
transaction id f969f5785acaeab87c8471190ece5231adb54575410293bdcd5f00d227eb24b7: [a, b, 100]
2018-09-26 09:58:24,322 INFO [cn.netkiller.fabric.FabricHelper] - Query payload from peer peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local returned -500
2018-09-26 09:58:24,386 INFO [cn.netkiller.fabric.FabricHelper] - Query payload from peer peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0-0.peer-
23e50b60552eb5b6f32d3c1563305dd4530dc2f0.default.svc.cluster.local returned 800
2018-09-26 09:58:24,386 INFO [cn.netkiller.fabric.Main] - after invoke  a=-500, invoke b=800
```

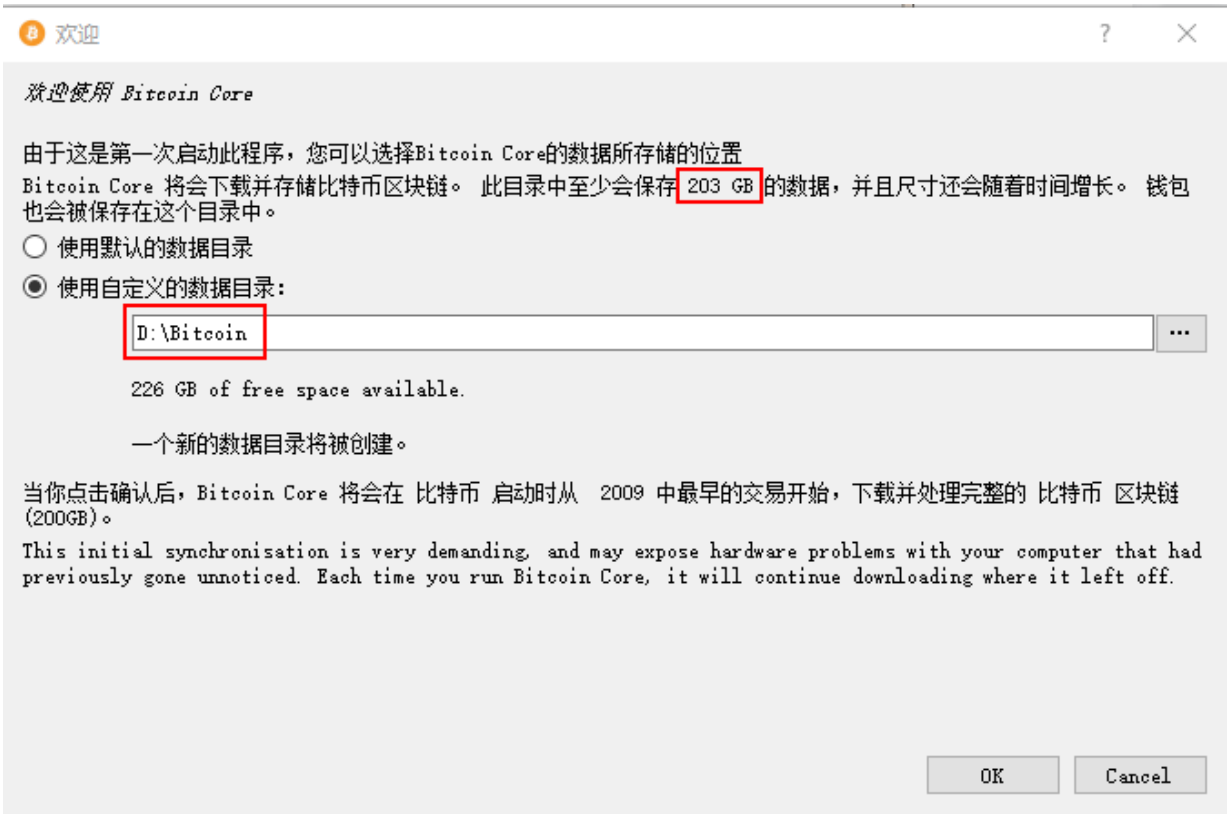
第 40 章 BitCoin

1. 私钥

它一个256bit的随机数，一般随机生成，范围在0x1到0xFFFF FFFF FFFF FFFF FFFF FFFF FFFE BAAE DCE6 AF48 A03B BFD2 5E8C D036 4140之间（这是由ECDSA spec256k1算法限定的）

2. 比特币钱包

2.1. Bitcoin Core




Bitcoin Core [测试网络]
[测试网络]
Bitcoin Core
Version v0.16.0
© 2009-2018 Bitcoin Core 开发人员
正在加载区块索引...


Bitcoin Core - 钱包 [测试网络]
文件(F) 设置(S) 帮助(H)
概况(O) 发送(S) 接收(R) 交易记录(T)
余额
最近交易记录
近期交易可能尚未显示，因此当前余额可能不准确。以上信息将在与比特币网络完全同步后更正。详情如下
尝试使用受未可见交易影响的余额将不被网络接受。
剩余区块数量 未知
上一数据块时间 周四 2月 3 07:16:42 2011
进度 0.00%
每小时进度增加 正在计算
预计剩余同步时间 正在计算
隐藏
同步区块头 (41.6%)...
落后 7 year(s) 和 13 week(s)
BTC HD

获得测试比特币 https://testnet.manu.backend.hamburg/faucet

2.2. 网页钱包

2.3. Coin.Space

https://coin.space/?network=bitcoin

response enter major zero net chief universe liquid ignore bean crash stand

2.4. BitGo

https://www.bitgo.com/login

2.5. GreenAddress

https://greenaddress.it/en/

3. bcoin

A Fullnode Bitcoin Implementation for Miners, Wallets, and Exchanges

bcoin.io/

```
$ npm install bcoin
```

设置比特币测试网节点

```
'use strict';

var bcoin = require('bcoin');

var node = new bcoin.fullnode({
  network: 'testnet',
  db: 'memory'
});

(async function() {
  await node.open();
  await node.connect();

  node.on('connect', function(entry, block) {
    console.log('%s (%d) added to chain.', entry.rhash(),
entry.height);
  });

  node.on('tx', function(tx) {
    console.log('%s added to mempool.', tx.txid());
  });

  node.startSync();
})();
```

4. HD Wallet

```
npm install bip39 bitcoinjs-lib
```

```
        const bip39 = require('bip39');
    const bitcoin = require('bitcoinjs-lib')

    const mnemonic = bip39.generateMnemonic();
    const seed = bip39.mnemonicToSeed(mnemonic);
        const root = bitcoin.HDNode.fromSeedBuffer(seed)
        //const root =
bitcoin.HDNode.fromSeedHex(seed.toString('hex'))
    const wallet = root.derivePath("m/44'/0'/0'/0/0");

    const address = wallet.getAddress();
    const wif = wallet.keyPair.toWIF();

        console.log(mnemonic)
    console.log(address)
    console.log(wif)
```

第 41 章 其他区块链相关

1. FISCO BCOS

早在2016年，微众银行、上海万向区块链、矩阵元三家公司达成战略合作，致力于共同进行区块链技术的探索，且在2017年7月，三方将顺利完成的区块链底层平台BCOS（取BlockChain OpenSource涵义命名）完全开源，以便更多的开发者加入，共同完善技术，构建真正根植中国的区块链生态。

2. 量子链（QTUM）

https://qtum.org/

量子链致力于开发比特币和以太坊之外的第三种区块链生态系统，通过价值传输协议（“Value Transfer Protocol”）来实现点对点的价值转移，并根据此协议，构建一个支持多个行业（包括金融、物联网、供应链、社交、游戏等）的去中心化的应用开发平台（DApp Platform）

2.1. BeeChat

https://beechat.io

BeeChat是基于区块链的手机通讯应用。以独有的去中心化分布式技术，高度安全的加密服务确保免费环球畅聊，群人数最高可达10000人。它还基于量子链区块链技术，支持BTC,ETH,Qtum的移动钱包，发红包转账和聊天一样简单快捷。

3. asch

https://www.asch.io

4. K-Line 开发库

https://www.tradingview.com

https://cn.tradingview.com

5. 数字货币行情

https://coinmarketcap.com

https://coinmarketcap.com/currencies/eos/

附录 A. 附录

1. Hyperledger Fabric

http://hyperledger-fabric.readthedocs.io/en/latest/

2. Ethereum

2.1. web3.js Document

https://web3js.readthedocs.io/en/1.0/index.html

2.2. Standardizing of HD wallet derivation paths

https://github.com/ethereum/EIPs/issues/84

3. NXP（恩智浦）相关产品

3.1. MifareUltralight

艺术品数字身份证

ID

将NFC标签的存储区域分为16个页，每一个页可以存储4个字节，一个可存储64个字节（512位）。页码从0开始（0至15）。前4页（0至3）存储了NFC标签相关的信息（如NFC标签的序列号、控制位等）。从第5页开始存储实际的数据（4至15页）。

NTAG21x系列容量大小

```
NTAG210 48byte
NTAG213 144byte
NTAG215 504byte
NTAG216, 888byte
```

3.2. MifareClassic

3.3. Mifare

4. NFC 数据格式

4.1. NFC 标准

历史记录

```
ISO14443-4
ISO14443-3A
ISO14443-3B
```

4.2. NDEF (NFC Data Exchange Format)

```
Value    Protocol
-----    --------
0x00     No prepending is done ... the entire URI is contained
in the URI Field
0x01     http://www.
0x02     https://www.
0x03     http://
0x04     https://
0x05     tel:
0x06     mailto:
0x07     ftp://anonymous:anonymous@
0x08     ftp://ftp.
0x09     ftps://
0x0A     sftp://
0x0B     smb://
0x0C     nfs://
0x0D     ftp://
0x0E     dav://
0x0F     news:
0x10     telnet://
0x11     imap:
0x12     rtsp://
0x13     urn:
0x14     pop:
0x15     sip:
```

0x16	sips:
0x17	tftp:
0x18	btspp://
0x19	btl2cap://
0x1A	btgoep://
0x1B	tcpobex://
0x1C	irdaobex://
0x1D	file://
0x1E	urn:epc:id:
0x1F	urn:epc:tag:
0x20	urn:epc:pat:
0x21	urn:epc:raw:
0x22	urn:epc:
0x23	urn:nfc: