以太坊智能合约编写最终版

由 郝昊天创建, 最终由 李源堃修改于 四月 16, 2018

前言

这份文档让你对以太坊以及智能合约有一个初步了解。

第一章是关于区块链与以太坊的简介，内容主要来自对ETHEREUM OVERVIEW一文的翻译（作者六级没过，要是觉得翻译不对请自行看原文）。

第二章是关于以太坊智能合约编写语言Solidity的介绍,内容主要来自Solidity与Solidity语言。

第三章介绍了以太坊智能合约编写框架truffle，并在官方demo基础上自己手动修改生成一个dapp。其中一部分来自手把手教创建你的第一个以太智能合约另一部分来自作者自己瞎编。

第四章介绍了如何从零开始创建一个项目，内容来自一个踩了无数坑的作者。

第五章介绍如何进行调试。

最后是作者遇到的一些坑。

作者是一名有着两星期以太坊智能合约开发经验的资(cai)深(ji)工(shi)程(xi)师(sheng)。如果文档有错误请告知我，如果你觉得我的代码可读性可维护性可扩展性差，那也没有办法，我的js、html、Solidity也是现学的。

另外框架不一定非要用nodejs也可以用react或者vue等，官方有demo。

git地址:git@git.n.xiaomi.com:haohaotian/votetest.git

1 区块链与以太坊基础

1.1 为什么使用区块链

区块链用在需要在多方之间共享数据以及传递价值而不需要信任彼此的场景下。

金融世界把这种信任叫做交易对手风险：这种风险下另一方不愿意承担风险。区块链通过一个具有数学，密码学，对等网络的革命性系统革除了这种风险。

第一个数据库

20世纪60年代第一个计算机数据库诞生了。随着硬件的分布式以及互联网历史进程的发展，数据自然而然地存在了中心化的物理环境。这种中心化的方法意味着数据的存储访问需要被一个中心化的权威控制。

中心化系统可以在内部或者外部被访问，因此我们不得不相信系统的拥有者有足够的意愿和资源来整合我们的数据并保证数据安全。中心化数据库现在仍然很普遍，支配者我们线上线下的大部分应用。

Self-hosted 日志是中心化数据库的一个常见例子。所有者可以在事后编辑帖子或对用户进行审查，而无需求助。相应地黑客也可以恶意攻击系统。如果没有数据库备份那么恢复被破坏的数据变得不可能。

数据共享的需求

共享大数据既昂贵又很不方便。我们可以通过分布式数据来减轻这种负担。读写数据被一个或多个实体控制，但是仍然存在中心化数据库中诸如腐败等问题。现代共享数据库采用技术来减少这种问题。其中的一些想法与区块链不谋而合。基于共享数据库的系统可能有以下特点：

- 不变性：复制一份旧数据作为历史记录而不是覆盖旧数据。可以访问数据来证明某些数据在特定时间存在过。

- 一致性：对于一份要被共享的数据，各方必须认同其内容。有很多方法达成一致性，比如将要讨论的POW。

区块链继承发扬这些方法来解决这些信任问题。

1.2 区块链是什么

根本上说区块链是一份共享数据，包含了交易的账本。很像银行，简单区块链的账簿记录了货币(在这种情况下，是加密货币)的所有权。每一个连接到链的设备都有一个账本的备份，这叫做节点。区块链通过以下方式消除了影响数据库信任的问题：

- 彻底的去中心化：读写数据库彻底去中心化并且安全。没有一个个人或者组织可以控制区块链。
- 高度的容错能力：容错能力不是区块链所独有的，区块链通过每个账户共享数据来实现这个逻辑。
- 独立验证：交易可以被每个人验证而不需要第三方。所谓的脱媒。

1.3 区块链如何工作

区块链网络账户之间的交互叫交易。可以是货币交易比如发送以太。也可以是数据交易比如一条评论。打捆的交易叫块(block)。

区块链上每一个账户都有一个独一无二的签名，让所有人知道谁创建的交易。在一个公链上每个人都可以读写数据，读数据免费写数据收费。这种消费叫做“gas”，用以太币购买，有助于网络安全。

挖矿

网络上的任何节点都可以通过一个名为“挖掘”的过程来参与保护网络。成为矿工的节点需要解决一个数学问题来获得记账权。因为挖矿需要算力，矿工会获得补偿。赢得记账权的矿工可以获得加密货币作为回报。这就激励了矿工的工作，以确保网络的安全，防止太多的控制权落入任何单一的采矿者手中。

哈希

一旦一个新块被挖出来，其他的矿工就会被通知到这个消息然后开始验证并把这个新块复制到自己链上。这通过密码学的哈希完成。哈希是一种单向函数，这种函数输入数据并返回一个固定长度的字符串。

然而并不能从哈希函数中推断出原始数据，相同的输入一定有相同的输出。这样可以哈希未经验证的数据并和原数据比较。如果匹配那么验证通过。

如果一半以上的矿工验证了新块，就认为网络达成了公式，这个块就永久成为区块链历史的一部分。现在这个数据可以被所有的节点下载，因为已经被核实有效。

1.4 以太坊是什么

以太坊是一个允许你跑程序的可信环境。比特币区块链的合约只能让你管理加密货币。

为了达到目的，以太坊有一个虚拟机，叫做EVM。EVM允许核实代码并且在区块链上执行，确保在每个人的机器上都以相同的方式执行。这段代码叫做“智能合约”。

除了追踪账户余额以外，以太坊维持区块链上EVM的状态。所有节点都处理智能合约来验证合约和输入的完整性。

1.5 智能合约是什么

智能合约是运行在EVM上的代码。智能合约可以接受存储以太币，数据或者二者的打包。然后，根据合同中的逻辑，它可以将该以太币分配给其他帐户，甚至是其他智能合约。

以太坊智能合约用solidity语言编写。Solidity是静态类型的，它支持继承、库和复杂的用户定义类型。Solidity语法类似于JavaScript。

1.6 以太坊网络

到现在为止我们已经以太坊公链的主要部分。在主网上，数据在链上——包含了账户余额和交易——这些是公开的，任何人都可以创造节点并开始验证交易。网络上的以太币有市场机制并且可以用其他加密货币或者诸如美元的发币交换。

但是也有别的网络。事实上每个人都可以创造自己的以太坊网络。

最新的测试网络

以太坊区块链可以在本地模拟一个环境。本地测试网络可以即时处理交易，以太坊可以按需分配。有一些以太坊模拟器，Ganache(http://truffleframework.com/ganache/)

公共测试网络

开发者可以使用公开测试网络来测试以太坊应用在最终部署到主网络之前。网络上的这些以太币只是用来测试但是没有什么价值。

有三种广泛使用的测试网络：

- Ropsten：官方测试网络，以太坊基金会发布，功能上类似主网络。
- Kovan：一个使用“POA”一致性算法的网络。这意味着交易可以被选出的其他成员验证，持续4秒。这个测试网络上的以太坊应用也用来减轻供给。
- Rinkeby：一个也用POA的测试网络，以太坊基金会发布。

私有网络/企业网络

私有的Ethereum网络允许各方共享数据而不使其公开访问。一个私有区块链是一个很好的选择:

- 共享敏感数据，比如健康记录。
- 由于网络规模较小，扩展处理更高的读/写吞吐量。

1.7 DApp

一个用智能合约处理程序的应用叫分布式应用，或者叫dapps。这些dapps的用户界面包括大家熟悉的语言，如HTML、CSS和JavaScript。应用程序本身可以托管在传统的web服务器上，也可以在分布式文件服务(如群集或IPFS)上。考虑到Ethereum区块链的好处，dapp可以成为许多行业的解决方案，包括但不限于：

- 记账
- 金融
- 供应链
- 不动产
- 市场

创建dapp的最好方法是测试一下，并且部署到你自己选的以太坊网络。当然是Truffle。

2 Solidity与开发环境搭建

2.1 Solidity简介

Solidity是一门编写智能合约的语言。用来在不同平台上实现智能合约。它由Gavin Wood, Christian Reitwiessner, Alex Beregszaszi, Liana Husikyan, Yoichi Hirai和几位以前的以太坊核心贡献者开发，可以在区块链平台(如Ethereum)上编写智能合约。语法略微清奇，类似 c++/java/python/js的结合体，还摒弃了很多语言的优点，集成了很多语言的缺点。

推荐几个学习的网站：

https://solidity.readthedocs.io/en/develop/ solidity的文档

http://solidity-cn.readthedocs.io/zh/latest/index.html 上边那个网站的中文版，但是翻译滞后

http://www.tryblockchain.org/ 这个也是中文版的文档

https://cryptozombies.io/ 通过做一个游戏来学习solidity，这个游戏如果学会了而且你还懂一点前端，那么你就可以自己开发一个加密兔而不用和别人抢兔子了。

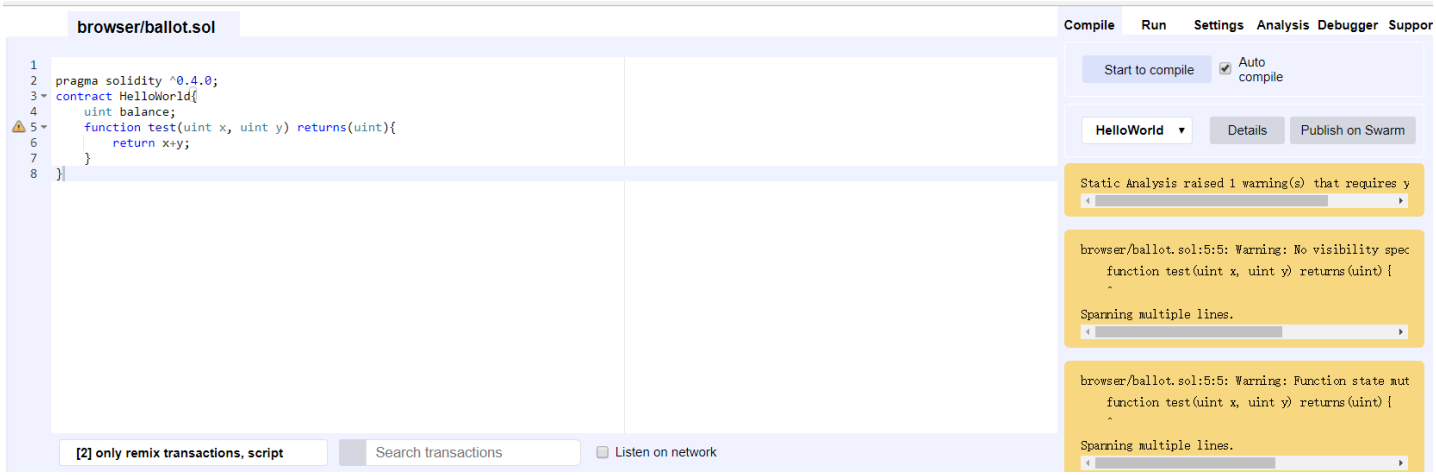
很多工具都可以用来开发Solidity比如vscode/eclipse/webstorm/pycharm……在官方文档上有详细列表。

官方推荐ide是remix，有在线版和离线版，在这里我们以在线版为例。

以下面这段代码为例：

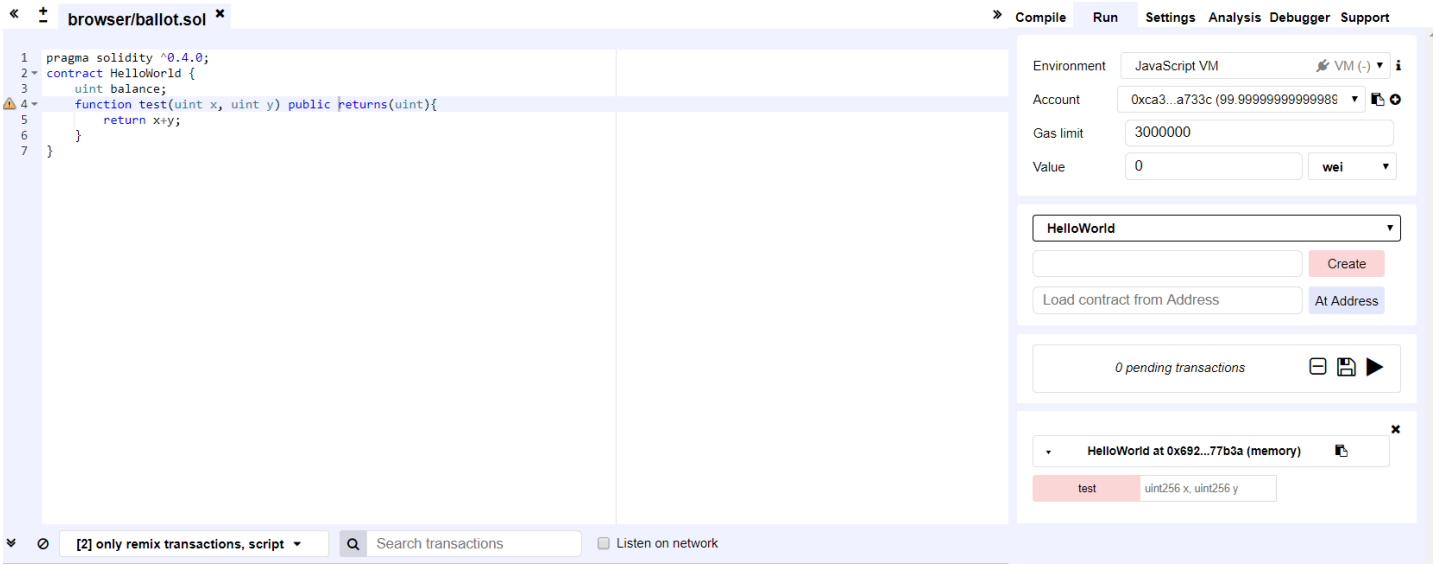
```
pragma solidity ^0.4.0;
contract HelloWorld{
    uint balance;
    function test(uint x, uint y) returns(uint){
        return x+y;
    }
}
```

进入编译器ol页面。

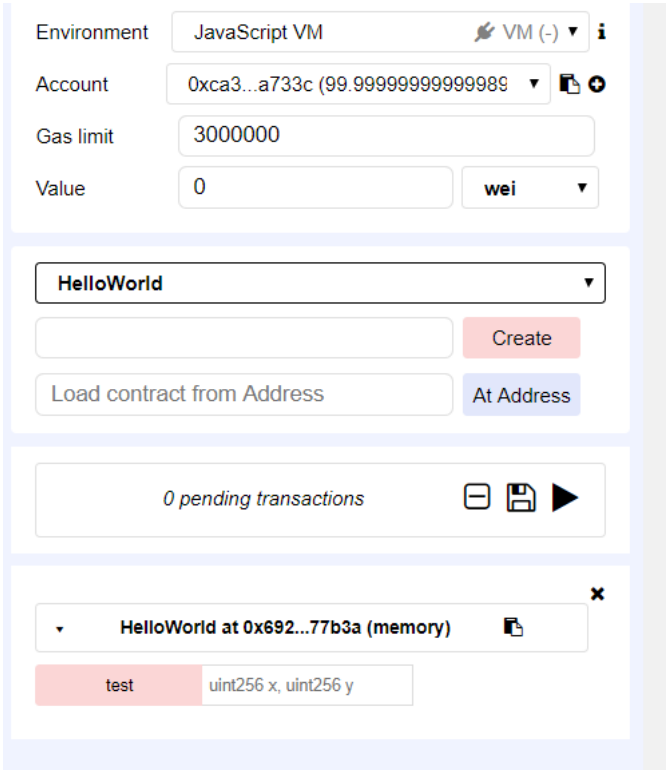


左边的框框输入代码，右边点击**start to compile。**

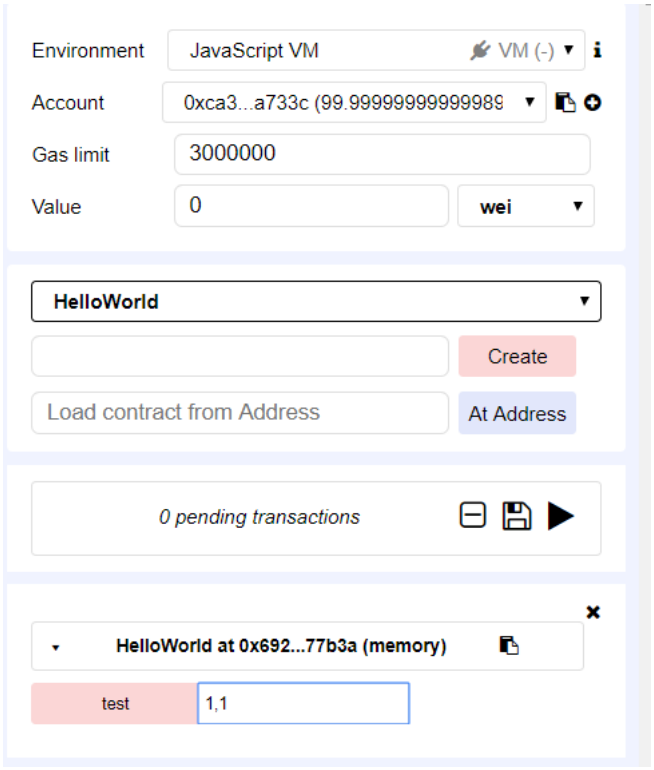
如果想要测试数据，选择**run**选项卡。点击**Create。**

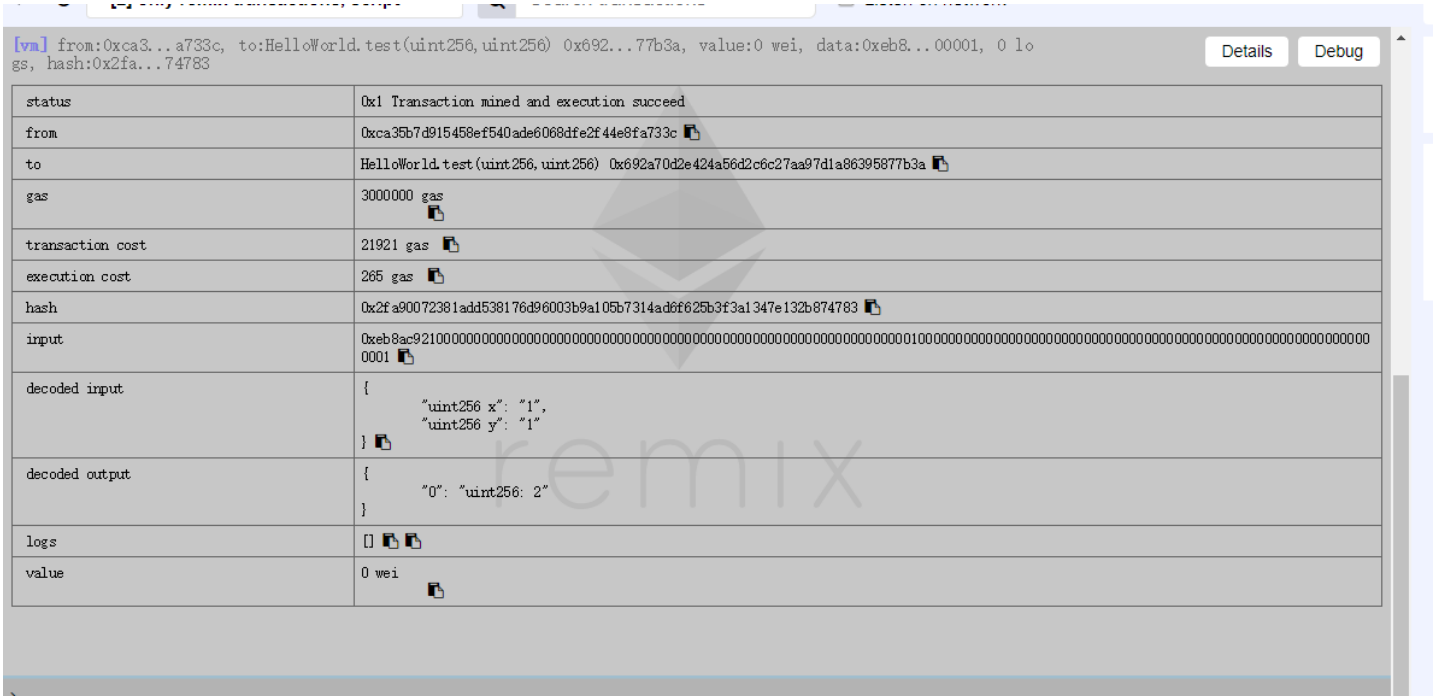


发现下边多了一个输入框 test，也就是代码中的test函数。



在test中输入数据，比如1,1，点击test。在底部控制台可以看到





点击**detail**就可以看到输出结果。如果修改了代码就把上述过程再执行一遍。

2.2 开发环境搭建

1. 安装最新版的node.js
2. 安装最新版的npm
3. 安装truffle和solc。

```
npm install -g truffle
truffle version//查看是否安装正确
npm install solc
```

（不同版本的truffle在使用方面存在着差异，本文用的是4.1.3。旧版本的有很多不完善的地方。）

2.3 truffle简介

Truffle是一个世界级的开发环境，测试框架，以太坊的资源管理通道，致力于让以太坊上的开发变得简单，truffle boxes是一些有助于你编写DApp的模板。Truffle Boxes包含了一些模块，比如Solidity 合约和库，前端后端，让你以一种easy的方式完成dapp。官方truffle boxes地址：http://truffleframework.com/boxes/。我们先从一个truffle box的模板开始。

3 一个基于truffle box模板的Dapp

3.1 项目创建

新建文件夹pet-shop

```
cd pet-shop/
truffle unbox pet-shop
```

在早期版本的truffle中如果用truffle init初始化一个工程，工程目录下是有一些demo的，但是现在的没有了，所以网上一些资料出现了偏差。新版的truffle 最好用unbox的方式初始化。等待项目unbox完成。看一下项目目录（我用的vsCode）

```
hk-journalist@ubuntu:~/work/code/pet-shop$ tree -L 1
.
├── box-img-lg.png
├── box-img-sm.png
├── bs-config.json
├── contracts
├── migrations
├── node_modules
├── package.json
├── package-lock.json
├── src
├── test
└── truffle.js
```

- contracts:智能合约
- migrate:移植合约
- src:和前端有关的资源
- test:测试代码
- truffle.js :truffle配置文件

3.2 合约编写

官方给的代码已经把一些代码写好了并且提供了前端资源，我们需要完成剩下的部分。这个DApp的作用是模拟领养宠物狗，如果你点击领养了某只狗，那么你变为他的领养者，别人不能领养他了。

合约功能是模拟领养宠物狗，如果你选择了领养，那么别人就不能再领养。一共16只，编号0-15.

在contracts下新建一个文件，文件名Adoption.sol。文件代码如下：

```
pragma solidity ^0.4.16;//指定版本

contract Adoption {
    address[16] public adopters;//领养者的address
    function adopt(uint petId) public returns (uint) {
        require(petId >= 0 && petId <= 15);
        adopters[petId]= msg.sender;//记录领养者address
        return  petId;
    }
    function getAdopters() public view returns (address[16]) {
        return adopters;
    }
}
```

3.3 合约编译与部署

Truffle Develop是Truffle的内置开发平台，可以用来测试部署合同，还可以在上边执行truffle命令。首先我们用truffle把solidity编译成EVM能读懂的东西。在工程目录下执行以下指令：truffle develop：

```
hk-journalist@ubuntu:~/work/code/pet-shop$ truffle develop
Connected to existing Truffle Develop session at http://localhost:9545/

truffle(develop)>
```

使用compile指令看到编译结果

```
truffle(develop)> compile
Compiling ./contracts/Adoption.sol...
Compiling ./contracts/Migrations.sol...

Compilation warnings encountered:

/home/hk-journalist/work/code/pet-shop/contracts/Migrations.sol:11:3: Warning: No visibility specified. Defaulting to "public".
  function Migrations() {
  ^
Spanning multiple lines.
,/home/hk-journalist/work/code/pet-shop/contracts/Migrations.sol:15:3: Warning: No visibility specified. Defaulting to "public".
  function setCompleted(uint completed) restricted {
  ^
Spanning multiple lines.
,/home/hk-journalist/work/code/pet-shop/contracts/Migrations.sol:19:3: Warning: No visibility specified. Defaulting to "public".
  function upgrade(address new address) restricted {
  ^
Spanning multiple lines.

Writing artifacts to ./build/contracts

truffle(develop)>
```

下面把合约迁移到链上。

在migrations下新建一个2_deploy_contracts.js。内容如下：

```
var Adoption = artifacts.require("Adoption");
    module.exports = function(deployer) {
    deployer.deploy(Adoption);
};
```

回到控制台，迁移一下，输入：migrate。看到输出。

```
truffle(develop)> migrate
Using network 'develop'.

Running migration: 1 initial migration.js
  Deploying Migrations...
  ... 0xcafa71ecd6b046777956367e31ff6e66b2c92ca18d1503edd5d33e812491f8e8
  Migrations: 0x4e72770760c011647d4873f60a3cf6cdea896cd8
Saving successful migration to network...
  ... 0x1280a1ee711d57c7eb1ebd42de8651e0fdb6df58cdb90aaaa0b108fd42ecbd82
Saving artifacts...
Running migration: 2 deploy contracts.js
  Deploying Adoption...
  ... 0xda0e5ad51741eb77eed2f9cac1d760d057d9c17634f71d1fa006de02bd5d3711
  Adoption: 0x13274fe19c0178208bcbee397af8167a7be27f6f
Saving successful migration to network...
  ... 0xc12ef018c969ae96ccbd73d5ffa8b0a118cc7bef8c202fb984c027e0a5a22602
Saving artifacts...
```

3.4 单元测试

在test下新建TestAdoption.sol，代码如下：

```
pragma solidity ^0.4.16;
import "truffle/Assert.sol";
import "truffle/DeployedAddresses.sol";
import "../contracts/Adoption.sol";
contract TestAdoption {
    Adoption adoption = Adoption(DeployedAddresses.Adoption());
```

```
    function testUserCanAdoptPet() {
        uint returnedId = adoption.adopt(8);
        uint expected = 8;
        Assert.equal(returnedId, expected, "Adoption of pet ID 8 should be recorded.");
    }
    function testGetAdopterAddressByPetIdInArray() {
        address expected = this;
        address[16] memory adopters = adoption.getAdopters();
        Assert.equal(adopters[8], expected, "Owner of pet ID 8 should be recorded.");
    }
}
```

导入的两个truffle文件是全局的，虽然在VSCode中报错但是不用管他。在import truffle的两个文件那里可能会提示找不到文件，不过没关系。这个东西是全局的，不是这个鼠目寸光的编辑器能够理解得了的。

再多说一句，有时候vscode会有奇怪的提示，比如什么空格tab不能混用，for与(之间要用空格，solidity版本太超前。通常情况下都可以当成warning处理，虽然有红线出没。

输入test，完成测试

```
TestAdoption
  ✓ testUserCanAdoptPet (44ms)
  ✓ testGetAdopterAddressByPetIdInArray (61ms)


  2 passing (587ms)

truffle(develop)>
```

当然如果你有信心代码不用测试也没问题。

3.5 前端交互

仅仅有黑框框是不够的，现在需要一个交互页面，不过官方已经给你写好了。我们需要把剩余的补充了。打开src/js/app.js，最后代码长这样（其中的代码逻辑一看就懂）：

App.contracts.Adoption = TruffleContract(AdoptionArtifact);这样做会导致提交运行不了，为此建议参考我们另一个demo（http://git.n.xiaomi.com/hackathon2018/hackathon-demo.git）里面wallet依赖：

```
12      "dependencies": {
13        "vue": "^2.5.2",
14        "vue-router": "^3.0.1",
15        "vue2-highcharts": "^1.1.9",
16        "wallet": "git+ssh://git@git.n.xiaomi.com:hackathon2018/wallet.git"
17      },
```

```
App = {
  web3Provider: null,
  contracts: {},

  init: function() {
    // Load pets.
    $.getJSON('../pets.json', function(data) {
      var petsRow = $('#petsRow');
      var petTemplate = $('#petTemplate');
      for (i = 0; i < data.length; i ++) {
        petTemplate.find('.panel-title').text(data[i].name);
        petTemplate.find('img').attr('src', data[i].picture);
        petTemplate.find('.pet-breed').text(data[i].breed);
        petTemplate.find('.pet-age').text(data[i].age);
        petTemplate.find('.pet-location').text(data[i].location);
        petTemplate.find('.btn-adopt').attr('data-id', data[i].id);
        petsRow.append(petTemplate.html());
      }
    });
    return App.initWeb3();
  },

  initWeb3: function() {
    // Is there is an injected web3 instance?
    if (typeof web3 !== 'undefined') {
      App.web3Provider = web3.currentProvider;
    } else {
      // If no injected web3 instance is detected, fallback to the TestRPC
      App.web3Provider = new Web3.providers.HttpProvider('http://localhost:8545');
    }
    web3 = new Web3(App.web3Provider);
    return App.initContract();
  },

  initContract: function() {
    $.getJSON('Adoption.json', function(data) {
      // Get the necessary contract artifact file and instantiate it with truffle-contract
      var AdoptionArtifact = data;
      App.contracts.Adoption = TruffleContract(AdoptionArtifact);
```

```
      // Set the provider for our contract
      App.contracts.Adoption.setProvider(App.web3Provider);
      // Use our contract to retrieve and mark the adopted pets
      return App.markAdopted();
    });
    return App.bindEvents();
  },

  bindEvents: function() {
    $(document).on('click', '.btn-adopt', App.handleAdopt);
  },

  markAdopted: function(adopters, account) {
    var adoptionInstance;
    App.contracts.Adoption.deployed().then(function(instance) {
      adoptionInstance = instance;
      return adoptionInstance.getAdopters.call();
    }).then(function(adopters) {
      for (i = 0; i < adopters.length; i++) {
        if (adopters[i] !== '0x0000000000000000000000000000000000000000') {
          $('.panel-pet').eq(i).find('button').text('Success').attr('disabled', true);
        }
      }
    }).catch(function(err) {
      console.log(err.message);
    });
  },

  handleAdopt: function(event) {
    event.preventDefault();
    var petId = parseInt($(event.target).data('id'));
    var adoptionInstance;
    web3.eth.getAccounts(function(error, accounts) {
      if (error) {
        console.log(error);
      }

      var account = accounts[0];
      App.contracts.Adoption.deployed().then(function(instance) {
        adoptionInstance = instance;
        // Execute adopt as a transaction by sending account
        return adoptionInstance.adopt(petId, {from: account});
      }).then(function(result) {
        return App.markAdopted();
      }).catch(function(err) {
        console.log(err.message);
      });
    });
  }

};

$(function() {
  $(window).load(function() {
    App.init();
  });
});
```

init:从pets.json读取数据并初始化页面。这一部分是unbox官方给写好的。

initWeb3：初始化web3。web3.js是以太坊提供的一个js库，封装了以太坊RPC API。提供了与区块链交互的js对象和函数。

initContract：初始化智能合约

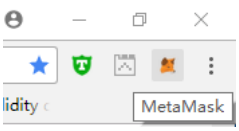
bindEvents：绑定事件，主要是那个adopt按钮的事件。

markAdopted：处理adopt的流程，主要是前端页面的更新

handleAdopt：处理adopt的流程，主要是与后端交互，并将数据提交前端。

3.6 metamask安装与使用

想在浏览器中看到dapp需要一些插件，我们这次采用metamask，基于Chrome。安装过程略，给出metamask地址：https://metamask.io/。在chrome右上角看到一只狐狸，这就是metamask。



点击启动metamask，假装阅读并被迫接受协议，然后看到这么一个页面。

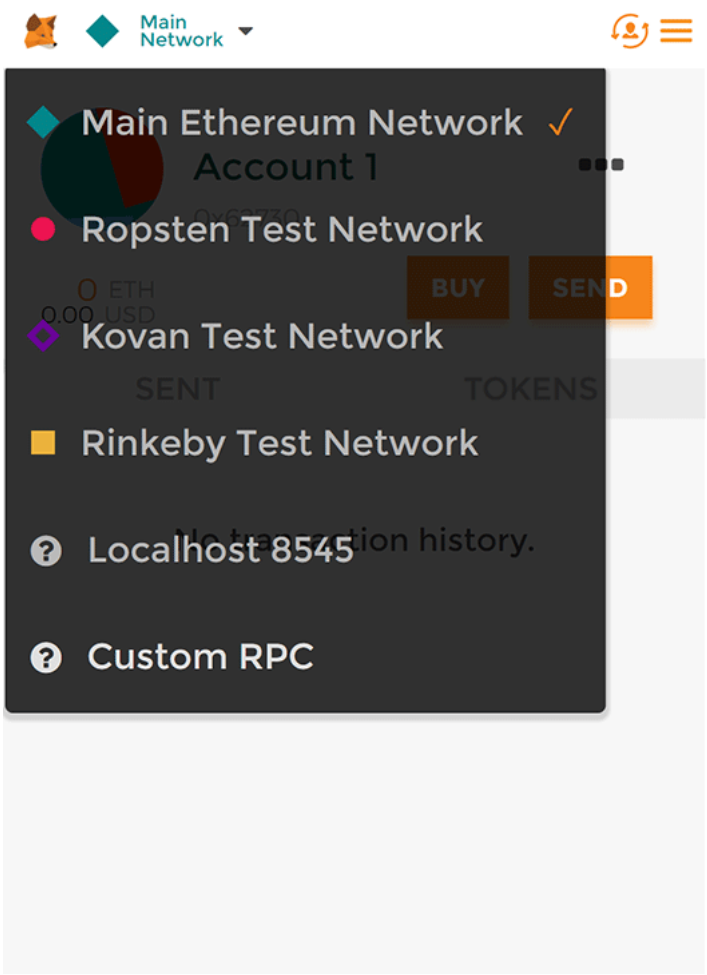
不要着急创建，点Import Existing DEN，在标有“Wallet Seed”的框中，输入登陆Truffle Develop时显示的助记词(**最好自己领搞一套助记词以免重复**):

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

在底下输入密码和确认密码，点击ok。

现在我们需要将MetaMask连接到由Truffle Develop创建的区块链。 点击显示“Main Network”的菜单并选择Custom RPC。

（tips：比赛时候我们会给你提供另一个地址）



在标题为“New RPC URL”的框中输入http://localhost:9545，(这个端口号具体多少看一下truffle develop后控制台有个输出，看看那个是多少)然后单击Save。点击"Settings"旁边的向左箭头关闭页面并返回到“帐户”页面。Truffle Develop .创建的每个账户都有100个eth。 你会注意到它在第一个账户上稍微少一些，因为当合约本身被部署时使用了一些gas。

现在回到linux终端（不是truffle终端），进入工程目录，启动代码：npm run dev 等待浏览器自动启动。每次执行交易的时候都会弹出一个页面要你提供一定的gas才能继续。有些时候console会提示invalid address，这是因为你没开metamask。

4 怎样从零开始编写一个Dapp

有些时候我们不想使用truffle box的模板，就想自己写一个。这一部分将写一个投票系统demo做演示。(tips:作者对前端和solidity一无所知，纯属自学成才)

4.1 创建工程

新建一个文件夹vote并用npm初始化一下

```
cd vote
truffle init
npm init
```

```
hk-journalist@ubuntu:~/work/code/vote$ la
contracts  migrations  package.json  test  truffle-config.js  truffle.js
```

看起来比pet-shop的东西要少一些。在package.json中添加一些依赖，最终效果:

```
{
  "name": "vote",
  "version": "1.0.0",
  "description": "",

  "directories": {
    "test": "test"
  },
  "scripts": {
    "dev": "lite-server",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
```

```
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "lite-server": "^2.3.0",
    "truffle-privatekey-provider": "0.0.5",
    "web3": "1.0.0-beta.33"
  },
  "dependencies": {
    "truffle-contract": "^3.0.4"
  }
}
```

npm install 安装下依赖。如果报错就改一下truffle版本。

```
$ npm uninstall -g truffle
$ npm install -g truffle@beta
```

还要在根目录下新建bs-config.json。

```
{
  "server": {
    "baseDir": ["./src", "./build/contracts"]
  }
}
```

配置下truffle.js

```
module.exports = {
  // See <http://truffleframework.com/docs/advanced/configuration>
  // for more about customizing your Truffle configuration!
  networks: {
    development: {
      host: "127.0.0.1",
      port: 9545,
      network_id: "*" // Match any network id
    }
  }
};
```

4.2 合约编写

contracts下新建三个合约,Legal.sol.Vote.sol.VoteDDL.sol.

legal检查当前时间是否在合法投票时间段内。

Vote实现了投票以及获取候选项票数的功能。

VoteDDL继承以上合约并增加了倒计时功能。

Legal.sol:

```
pragma experimental ABIEncoderV2;

contract Legal {
    //必须在某个时间节点以前投票
    modifier judgeLegalTimeIn(uint end) {
        require(now < end);
        _;
    }
    //必须在某个时间节点以后投票
    modifier judgeLegalTimeOut(uint begin) {
        require(now > begin);
        _;
    }
}
```

Vote.sol

```
pragma experimental ABIEncoderV2;//指定solidity版本

contract Vote {// 相当于 c++/java中的class

    mapping(address => uint) voteTo;
    //address:160bit,代表以太坊账户地址
    //mapping:映射,定义方式为mapping(key => value),key不能是mapping
```

```
    struct Candidate {//类似c中的结构体
        uint128 id;//无符号128位int,有uint8,uint16,uint24……uint256,uint=uint256
        uint128 votes;
        string name;
        //string是变长类型，在合约间不允许直接传递变长类型。但是前端可以直接获取合约中的变长
    }

    Candidate[] public candidates;
    string[5] names =["餐厅A", "餐厅B", "餐厅C", "餐厅D", "餐厅E"];

    function Vote() {
        for (uint128 i=0; i < 5; i++) {
            candidates.push(Candidate((i+1), (i+2), names[i]));
        }
    }

    function voteCandidate(uint id) public returns (uint) {
        require(id >= 1 && id <= candidates.length);
        candidates[id-1].votes++;
        voteTo[msg.sender] = id;
        return candidates[id-1].votes;
    }

    function getVotes(uint id) public returns (uint) {
        return candidates[id].votes;
    }

    function hasVoted() public returns (bool) {
        return !(voteTo[msg.sender] == 0);
    }

    function getCandidate(uint id) returns (uint, uint, string) {
        return(candidates[id].id, candidates[id].votes, candidates[id].name);
    }

    function getCandidateCounts() returns(uint) {
        return candidates.length;
    }

}
```

VoteDDL.sol

```
pragma experimental ABIEncoderV2;
import "./Vote.sol";
import "./Legal.sol";

contract VoteDDL is Vote, Legal {
    /**
     *  @notice  get the candidate you vote to
     *  @return the id of candidate you voted to
     */
    uint256 end = now + 1000 seconds;

    /**
    * override
    **/
    function vote(uint id) public  judgeLegalTimeIn(end) returns (uint) {
        require(id >= 1 && id <= candidates.length);
        candidates[id].votes++;
        voteTo[msg.sender] = id;
        return candidates[id].votes;
    }

    /**
    * 距离ddl还剩多少秒
    **/
    function getDDL() view returns(uint) {
        return 1000 * (end - now);
    }

}
```

migrations下新建2_deploy_contracts.js

```
var Vote = artifacts.require("Vote");
```

```
var VoteDDL = artifacts.require("VoteDDL");
var Legal = artifacts.require("Legal");
module.exports = function(deployer) {
  deployer.deploy(Vote);
  deployer.deploy(VoteDDL);
  deployer.deploy(Legal);
};
```

4.3 前端交互

根目录下新建src文件夹，src下新建文件夹：assets,css,js三个文件夹，以及两个html文件index.html.（代码在git上有，地址在本文前边）

assets下新建countdown文件夹，并导入jquery.countdown.css和jquery.coupntdown.js，这俩是有关前端页面倒计时的文件。不是我写的，我在网上下的，我也看不懂==

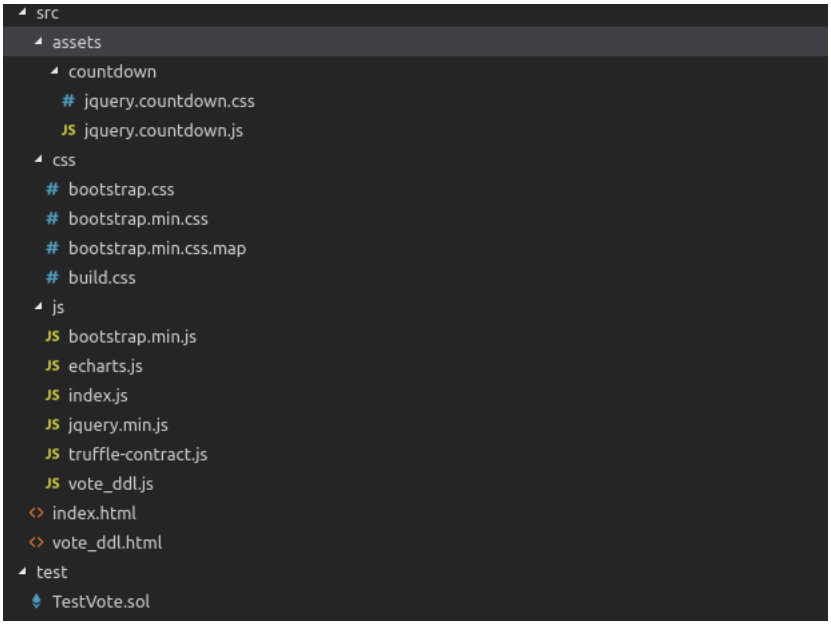
css下引入bootstrap.css,bootstrap.min.css,bootstrap.min.css.map,build.css.

js下引入bootstrap.min.js,echarts.js,jquery.min.js,truffle-contract.js。并新建index.js,vote_ddl.js

truffle-contract.js是一个神器，简化了很多操作，不知道比以前的truffle高到哪里去。index.js与vote_ddl.js分别与两个合约交互。

（其实assets文件夹下的资源可以放到css和js下的，但是最开始就这么放的我懒得改了）

最后如图所示。



着重说一下index.js和vote_ddl.js

首先看index.js.先上代码：

```
var voteData = [];
var hasVoted = false;
App = {
  web3Provider: null,
  contracts: {},

  initWeb3: function() {
    if (typeof web3 !== 'undefined') {
      App.web3Provider = web3.currentProvider;
    } else {
      App.web3Provider = new Web3.providers.HttpProvider('http://localhost:8545');
    }
    web3 = new Web3(App.web3Provider);

    return App.initContract();
  },

  initContract: function() {
    $.getJSON('Vote.json', function(data) {
      var VoteArtifact = data;
      App.contracts.Vote = TruffleContract(VoteArtifact);
      App.contracts.Vote.setProvider(App.web3Provider);
      return App.initPage();
    });

    return App.bindEvents();
  },

  bindEvents: function() {
    $(document).on("click","#submit",function() {
      if(hasVoted){
        alert("已经投过票了");
```

```
      }
      else{
        var id = 0;
        for(var i = 0; i < document.getElementsByName("option").length; i++){
          if(document.getElementsByName("option")[i].checked == true){
            App.handleVote(i+1);
            break;
          }
        }
      }
    });
  },

  initPage: function() {
    App.contracts.Vote.deployed().then(function(instance) {
      voteInstance = instance;
      return voteInstance.getCandidateCounts.call();
    }).then(function(count) {
        for(var i=0;i<count;i++){
          App.getCandidate(i,count-i);
        }
        return voteInstance.hasVoted.call();
    }).then(function(res){
        hasVoted = res;
    }).catch(function(err) {
      console.log(err.message);
    });
  },

  getCandidate: function(candidateId,end) {
    App.contracts.Vote.deployed().then(function(instance) {
      voteInstance = instance;
      return voteInstance.getCandidate.call(candidateId);
    }).then(function(res) {
      var id = res[0].c[0];
      var votes = res[1].c[0];
      var name = res[2];
      voteData.push({name:name,value:votes});
      var vote_panel =  $("#vote_panel");
      var tdName = $("<div class=\"radio radio-danger\"><input type=\"radio\" name=\"option\" id=\"radio3\" data-id="+id+" value=\"option1
      vote_panel.append(tdName);
      if(end == 1){
        var myChart = echarts.init(document.getElementById('show_panel'));
        var option={
            backgroundColor: '#D3D3D3',
            textStyle: {
                        color: 'rgba(0, 0, 0, 1)',
                        fontSize: 20
                    },
            series : [
                {
                    name: '统计数据',
                    type: 'pie',
                    radius: '90%',
                    data:(function(){
                      var res = [];
                      for(var i=0;i<voteData.length;i++) {
                        res.push({
                            name: voteData[i].name+":"+voteData[i].value+"票",
                            value: voteData[i].value
                        });
                      }
                      return res;
                   })(),

                    itemStyle: {
                        emphasis: {
                            shadowBlur: 200,
                            shadowColor: 'rgba(0, 0, 0, 1)'
                        },
                    },
                    label: {
                        normal: {
                            textStyle: {
                                color: 'rgba(0, 0, 0, 1)'
                            }
                        }
                    },
```

```
                    labelLine: {
                        normal: {
                            lineStyle: {
                                color: 'rgba(0, 0, 0, 1)'
                            },
                        }
                    }

                }
            ]
        }
        myChart.setOption(option);
        function eConsole(param) {
            var index = param.dataIndex;
           console.log(index);
        }
        myChart.on("click", eConsole);
      }

    }).catch(function(err) {
      console.log(err.message);
    });
  },

  notification:function(message){
    alert(message);
  },

  handleVote: function(id) {
    var voteInstance;
    web3.eth.getAccounts(function(error, accounts) {
      if (error) {
        console.log(error);
      }
      var account = accounts[0];
      App.contracts.Vote.deployed().then(function(instance) {
        voteInstance = instance;
        return voteInstance.voteCandidate(id, {from: account});
      }).then(function(result) {
        alert("投票成功");
        location.reload();
        App.initPage();
      }).catch(function(err) {
        console.log(err.message);
      });
    });
  }

};

$(function() {
  $(window).load(function() {
    App.initWeb3();
  });
});
```

解释下代码

两个全局变量voteData和hasVoted分别存储每个候选人的得票信息和当前address的用户是否已经投过票了。

initWeb3:看看有没有web3provider，没有就装一个

initContract:初始化合约

bindEvents:绑定点击事件，当用户提交投票时遍历radio找到选中项并交给handleVote处理。

initPage：初始化页面，先获取候选人数量再for循环获取每个候选人的信息，因为solidity暂时不方便传变长数据，故出此下策。

getCandidate:获取候选人信息，获取完毕后通过echarts进行显示。

notification:没有……

handleVote:处理投票事件

而vote_ddl.js与index.js大同小异，只是多了一个倒计时功能。代码如下：

```
var voteData = [];
var hasVoted = false;
var end =false;
App = {
  web3Provider: null,
  contracts: {},
```

```
init: function() {
  var main = $("#main");
  return App.initWeb3();
},

initWeb3: function() {
  if (typeof web3 !== 'undefined') {
    App.web3Provider = web3.currentProvider;
  } else {
    App.web3Provider = new Web3.providers.HttpProvider('http://localhost:8545');
  }
  web3 = new Web3(App.web3Provider);

  return App.initContract();
},

initContract: function() {
  $.getJSON('VoteDDL.json', function(data) {
    var VoteArtifact = data;
    App.contracts.VoteDDL = TruffleContract(VoteArtifact);
    App.contracts.VoteDDL.setProvider(App.web3Provider);
    return App.initPage();
  });

  return App.bindEvents();
},

bindEvents: function() {
  $(document).on("click","#submit",function() {
    if(end){
      alert("投票截至啦");
    }
    else if(hasVoted){
      alert("已经投过票了");
    }
    else{
      var id = 0;
      for(var i = 0; i < document.getElementsByName("option").length; i++){
        if(document.getElementsByName("option")[i].checked == true){
          App.handleVote(i+1);
          break;
        }
      }
    }
  });
},

initPage: function() {
  App.contracts.VoteDDL.deployed().then(function(instance) {
    voteInstance = instance;
    return voteInstance.getCandidateCounts.call();
  }).then(function(count) {
      for(var i=0;i<count;i++){
        App.getCandidate(i,count-i);
      }
      return voteInstance.hasVoted.call();
  }).then(function(res){
    App.initDDL();
    hasVoted = res;
  }).catch(function(err) {
    console.log(err.message);
  });
},

getCandidate: function(candidateId,end) {
  App.contracts.VoteDDL.deployed().then(function(instance) {
    voteInstance = instance;
    return voteInstance.getCandidate.call(candidateId);
  }).then(function(res) {
    var id = res[0].c[0];
    var votes = res[1].c[0];
    var name = res[2];
    voteData.push({name:name,value:votes});
    var vote_panel = $("#vote_panel");
    var tdName = $("<div class=\"radio radio-danger\"><input type=\"radio\" name=\"option\" id=\"radio3\" data-id="+id+" value=\"option1
    vote_panel.append(tdName);
    if(end == 1){
      var myChart = echarts.init(document.getElementById('show_panel'));
```

```
        var option = {
          color: ['#3398DB'],
          tooltip : {
              trigger: 'axis',
              axisPointer : {
                  type : 'shadow'
              }
          },
          grid: {
              left: '3%',
              right: '4%',
              bottom: '3%',
              containLabel: true
          },
          xAxis : [
              {
                  type : 'category',
                  data : (function(){
                    var res = [];
                    for(var i=0;i<voteData.length;i++) {
                      res.push(voteData[i].name);
                    }
                    return res;
                  })(),
                  axisTick: {
                      alignWithLabel: true
                  }
              }
          ],
          yAxis : [
              {
                  type : 'value'
              }
          ],
          textStyle: {
            color: 'rgba(0, 0, 0, 1)',
            fontSize: 200
          },
          series : [
              {
                  name:'票数',
                  type:'bar',
                  barWidth: '60%',
                  data: (function(){
                    var res = [];
                    for(var i=0;i<voteData.length;i++) {
                      res.push(voteData[i].value);
                    }
                    return res;
                  })(),
              }
          ]
      };
        myChart.setOption(option);
        function eConsole(param) {
            var index = param.dataIndex;
           console.log(index);
        }
        myChart.on("click", eConsole);
      }

      return voteInstance.hasVoted.call();
    }).catch(function(err) {
      console.log(err.message);
    });
  },

  notification:function(message){
    alert(message);
  },

  handleVote: function(id) {
    var voteInstance;
    web3.eth.getAccounts(function(error, accounts) {
      if (error) {
        console.log(error);
      }
      var account = accounts[0];
```

```
      App.contracts.VoteDDL.deployed().then(function(instance) {
        voteInstance = instance;
        return voteInstance.voteCandidate(id, {from: account});
      }).then(function(result) {
        alert("投票成功");
        location.reload();
        App.initPage();
      }).catch(function(err) {
        console.log(err.message);
      });
    });
  },

  initDDL:function(ddl){
    App.contracts.VoteDDL.deployed().then(function(instance) {
      voteInstance = instance;
      return voteInstance.getDDL.call();
    }).then(function(ddl) {
      ddl = parseInt(ddl);
      var note = $('#note'),
      ts = (new Date()).getTime() + ddl;
      $('#countdown').countdown({
          timestamp   : ts,
          callback    : function(days, hours, minutes, seconds){
              if (ts <= (new Date()).getTime()){
                end = true;
              }
              var message = "";
              message += days + " day" + ( days==1 ? '':'s' ) + ", ";
              message += hours + " hour" + ( hours==1 ? '':'s' ) + ", ";
              message += minutes + " minute" + ( minutes==1 ? '':'s' ) + " and ";
              message += seconds + " second" + ( seconds==1 ? '':'s' ) + " <br />";

          }
      });
    }).catch(function(err) {
      console.log(err.message);
    });
  }

};

$(function() {
  $(window).load(function() {
    App.init();
  });
});
```

最后是两个html 的代码。这个没什么解释的。

index.html:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Vote</title>
    <script src="js/echarts.js"></script>
    <script src="js/jquery.min.js"></script>
    <script src="js/bootstrap.min.js"></script>
    <script src="js/truffle-contract.js"></script>
    <script src="js/index.js"></script>
    <link rel="stylesheet" type="text/css" href="css/bootstrap.css" />
    <link rel="stylesheet" type="text/css" href="css/build.css">
</head>
<style>
    html,body{
        margin:0px;
        height:100%;
    }
    #bg{
        width:100%;
        height:100%;
        MARGIN: 0px auto;
    }
    </style>
```

```
<body>
    <div style="display:inline">
        <div id="show_panel" style="width: 80%;height:100%;float:left; background:#D3D3D3">
        </div>
        <div id="vote_panel" style="width: 20%;height:100%; background:#D3D3D3;float:left;">
            <h4 >投票表决中午吃什么</h4>
                <div id="main" class="main">
                    <tr>
                        <td>
                            <input type="submit" id="submit" value="确认投票" class="btn btn-primary"/>
                        </td>
                    </tr>
                </div>
        </div>
    </div>
</body>
</html>
```

vote_ddl.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Votes With DDL</title>
        <script src="http://cdn.static.runoob.com/libs/jquery/2.1.1/jquery.min.js"></script>
        <link rel="stylesheet" href="assets/countdown/jquery.countdown.css" />
        <script src="assets/countdown/jquery.countdown.js"></script>
        <link href="css/bootstrap.min.css" rel="stylesheet">
        <script src="js/truffle-contract.js"></script>
        <script src="js/bootstrap.min.js"></script>
        <script src="js/echarts.js"></script>
    </head>
    <style>
        html,body{
            margin:0px;
            height:100%;
        }
        #bg{
            width:100%;
            height:100%;
            MARGIN: 0px auto;
        }
        </style>
    <body>
        <div style="display:inline">
           <div id="show_panel" style="width: 74%;height:100%;float:left; background:#D3D3D3">
            </div>
            <div id="vote_panel" style="width: 26%;height:100%; background:#D3D3D3;float:left;">
               <div id="countdown"><h1>距离投票结束还有</h1></div>
                <h4 >投票表决中午吃什么</h4>
                    <div id="vote_panel" class="main">
                      <tr>
                            <td>
                               <input type="submit" id="submit" value="确认投票" class="btn btn-primary"/>
                            </td>
                        </tr>
                    </div>
            </div>
        </div>
        <script src="js/vote_ddl.js"></script>
    </body>
</html>
```

然后是truffle控制台编译部署，再执行npm run dev即可。

5 怎样调试solidity

truffle提供了简(fei)单(chang)易(nan)行(yong)的调试工具，本文举几个栗子。

5.1 新建项目

新建一个项目，随便起个名字，然后在项目目录下truffle init.等待建好后在contracts下新建一个合约。合约代码如下

```
pragma solidity ^0.4.16;
```

```
contract SimpleStorage {
    uint myVariable;

    function set(uint x) public {
        myVariable = x;
    }

    function get() public constant  returns (uint) {
        return myVariable;
    }
}
```

在migrations下新建2_deploy_contracts.js，代码如下：

```
var SimpleStorage = artifacts.require("SimpleStorage");

module.exports = function(deployer) {
  deployer.deploy(SimpleStorage);
};
```

进入truffle控制台， 编译一下，然后migrate。控制台输入：

```
SimpleStorage.deployed().then(function(instance){return instance.get.call();}).then(function(value){return value.toNumber()});
```

看到控制台输出：

```
truffle(develop)> SimpleStorage.deployed().then(function(instance){return instance.get.call();}).then(function(value){return value.toNumber()});
0
```

再换一个指令

```
SimpleStorage.deployed().then(function(instance){return instance.set(4);});
```

因为这个过程要写入数据，这样可以看到交易过程。再执行一次

```
truffle(develop)> SimpleStorage.deployed().then(function(instance){return instance.set(4);});
{ tx: '0xae1d57731e2fa44d801fef2b5401748ed81aac3db5c1e5fc38d809cb8a5b3ccf',
  receipt:
   { transactionHash: '0xae1d57731e2fa44d801fef2b5401748ed81aac3db5c1e5fc38d809cb8a5b3ccf',
     transactionIndex: 0,
     blockHash: '0xf915bc1d3c914532f6583ab7f3325424e7f925ef348c44a4f76df4d8390de21f',
     blockNumber: 27,
     gasUsed: 42487,
     cumulativeGasUsed: 42487,
     contractAddress: null,
     logs: [ [Object] ],
     status: '0x01',
     logsBloom: '0x000000000000000000000000000000000000000000000000000200000000000000000000000000000000000000040004000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000004000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000400000000000000000000
0000000000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000' },
  logs:
   [ { logIndex: 0,
       transactionIndex: 0,
       transactionHash: '0xae1d57731e2fa44d801fef2b5401748ed81aac3db5c1e5fc38d809cb8a5b3ccf',
       blockHash: '0xf915bc1d3c914532f6583ab7f3325424e7f925ef348c44a4f76df4d8390de21f',
       blockNumber: 27,
       address: '0xb529f14aa8096f943177c09ca294ad66d2e08b1f',
       type: 'mined',
       event: 'Odd',
       args: {} } ] }
```

```
SimpleStorage.deployed().then(function(instance){return instance.get.call();}).then(function(value){return value.toNumber()});
```

看到

```
truffle(develop)> SimpleStorage.deployed().then(function(instance){return instance.get.call();}).then(function(value){return value.toNumber()});
4
```

5.2 调试死循环

首先把合约代码改成死循环

```
pragma solidity ^0.4.16;


contract SimpleStorage {
    uint myVariable;

    function set(uint x) public {
        while (true) {
            myVariable = x;
        }
    }
```

```
    function get() public constant  returns (uint) {
        return myVariable;
    }
}
```

回到第一个终端，输入然后执行migrate --reset。为了追踪问题，我们需要再开一个终端看交易号（如果交易失败就用第二个终端），在第二个终端输入：truffle develop --log。

```
SimpleStorage.deployed().then(function(instance){return instance.set(4);});
```

看到控制台输出：

```
truffle(develop)> SimpleStorage.deployed().then(function(instance){return instance.set(4);});
Error: VM Exception while processing transaction: out of gas
    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(develop)>
```

gas用光了，然后看一下第二个终端，看到这样的输出

```
PROBLEMS 4   OUTPUT   DEBUG CONSOLE   TERMINAL                                  2: node

 develop:testrpc eth getBlockByNumber +118ms
 develop:testrpc eth getBlockByNumber +116ms
 develop:testrpc eth getBlockByNumber +117ms
 develop:testrpc eth getBlockByNumber +117ms
 develop:testrpc eth getBlockByNumber +115ms
 develop:testrpc eth getBlockByNumber +115ms
 develop:testrpc eth getBlockByNumber +117ms
 develop:testrpc eth getBlockByNumber +115ms
 develop:testrpc eth getBlockByNumber +116ms
 develop:testrpc eth sendTransaction +11ms
 develop:testrpc eth getBlockByNumber +103ms
 develop:testrpc  +1s
 develop:testrpc   Transaction: 0xcb696a18ceb6984d37edaeb6e41d0981fecdb2fbcc1ad835b1d88ff7723caecf +0ms
 develop:testrpc   Gas usage: 6721975 +0ms
 develop:testrpc   Block Number: 33 +0ms
 develop:testrpc   Block Time: Fri Mar 09 2018 11:36:01 GMT+0800 (CST) +0ms
 develop:testrpc   Runtime Error: out of gas +1ms
 develop:testrpc  +0ms
```

可以看到**+1s**，还能看到交易id。这个id要记下来。回到第一个终端，输入：

```
debug 0xe493340792ab92b95ac40e43dca6bc88fba7fd67191989d59ca30f79320e883f
```

注意用你刚才记录的交易id替换debug后边那一串数字。看到输出：

```
Gathering transaction data...

Addresses affected:
 0xb9b7e0cb2edf5ea031c8b297a5a1fa20379b6a0a - SimpleStorage

Commands:
(enter) last command entered (step next)
(o) step over, (i) step into, (u) step out, (n) step next
(;) step instruction, (p) print instruction, (h) print this help, (q) quit
(b) toggle breakpoint, (c) continue until breakpoint
(+) add watch expression (`+:<expr>`), (-) remove watch expression (-:<expr>)
(?) list existing watch expressions
(v) print variables and values, (:) evaluate expression - see `v`


SimpleStorage.sol:

2:
3:
4: contract SimpleStorage {
   ^^^^^^^^^^^^^^^^^^^^^^^^
```

按回车或者n执行下一步，最终来到：

```
SimpleStorage.sol:

6:
7:     function set(uint x) public {
8:         while (true) {
           ^^^^^^^^^^^^^^

debug(develop:0x748148f6...)>

SimpleStorage.sol:

6:
7:     function set(uint x) public {
8:         while (true) {
                  ^^^^

debug(develop:0x748148f6...)>

SimpleStorage.sol:

6:
7:     function set(uint x) public {
8:         while (true) {
           ^^^^^^^^^^^^^^

debug(develop:0x748148f6...)>

SimpleStorage.sol:

7:     function set(uint x) public {
8:         while (true) {
9:             myVariable = x;
                            ^
```

显然这里发生了死循环。

5.3 调试逻辑错误

有些时候程序不报错但是结果和预期不一样，那么可能发生了逻辑错误。

首先改一下合约代码

```
pragma solidity ^0.4.16;


contract SimpleStorage {
    uint myVariable;

    event Odd();
    event Even();

    function set(uint x) public {
        myVariable = x;
        if (x % 2 == 0) {
            Odd();
        } else {
            Even();
        }
    }
}
```

按照设想，x是偶数就执行even否则odd，但是代码中明显写反了。even函数和odd函数并没给出实现，但是不影响debug。

```
migrate --reset
SimpleStorage.deployed().then(function(instance){return instance.set(4);});
```

```
{ tx: '0x9eded7f9b4af3bd4b37adfbaaab24cc6ec67cca033666e4813bee244a67c5ea3',
  receipt:
   { transactionHash: '0x9eded7f9b4af3bd4b37adfbaaab24cc6ec67cca033666e4813bee244a67c5ea3',
     transactionIndex: 0,
     blockHash: '0xa7aed499180df1d9729f9a86484fe35e915b9bfb6ddeaf5c42a99ea8350a84d6',
     blockNumber: 43,
     gasUsed: 42487,
     cumulativeGasUsed: 42487,
     contractAddress: null,
     logs: [ [Object] ],
     status: '0x01',
     logsBloom: '0x000000000000000000000000000000000000000000000002000000000000000000000000000000000000040000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000000000000800000000000000000000000000000000000000000000000000000000000000000
000000000000004000000' },
  logs:
   [ { logIndex: 0,
       transactionIndex: 0,
       transactionHash: '0x9eded7f9b4af3bd4b37adfbaaab24cc6ec67cca033666e4813bee244a67c5ea3',
       blockHash: '0xa7aed499180df1d9729f9a86484fe35e915b9bfb6ddeaf5c42a99ea8350a84d6',
       blockNumber: 43,
       address: '0x1411cb266fced1587b0aa29e9d5a9ef3db64a9c5',
       type: 'mined',
       event: 'Odd',
       args: {} } ] }
```

看输出发现代码出现了偏差，不知道谁来负责。不过这个代码并没出现error，因此在当前窗口就能获取到交易id。

```
debug 0x9eded7f9b4af3bd4b37adfbaaab24cc6ec67cca033666e4813bee244a67c5ea3
```

一路回车，最后来到：

```
10:     function set(uint x) public {
11:         myVariable = x;
12:         if (x % 2 == 0) {
            ^^^^^^^^^^^^^^^^

debug(develop:0x9eded7f9...)>

SimpleStorage.sol:

11:         myVariable = x;
12:         if (x % 2 == 0) {
13:             Odd();
                ^^^^^

debug(develop:0x9eded7f9...)>

SimpleStorage.sol:

10:     function set(uint x) public {
11:         myVariable = x;
12:         if (x % 2 == 0) {
            ^^^^^^^^^^^^^^^^

debug(develop:0x9eded7f9...)>

SimpleStorage.sol:

 8:     event Even();
 9:
10:     function set(uint x) public {
        ^^^^^^^^^^^^^^^^^^^^^^^^^^^^

debug(develop:0x9eded7f9...)>

Transaction completed successfully.
```

显然错误发生在了这里。

6 遇到的一些问题

6.1 Assert找不到equal函数

报错输出：TypeError: Member "equal" is not available in type(library Assert) outside of storage.

原因：你所测试的函数的返回值是string，但是目前为止solidity不支持合约之间返回string。因为人呐就不知道就不可以预料一个字符串有多长。所以改成byte32.

6.2 error Invalid address

Metatask没登录……

6.3 Using network 'develop'.Network up to date.

migrate –reset

6.4 Attempting to run transaction which calls a contract function, but recipient address

migrate –reset

6.5 base fee exceeds gas limit

批准交易时在gas limit里设置一下gas，推荐的最低值不一定够，要续一下。

6.6 Contract has not been deployed to detected network (network/artifact mismatch)

Migrations文件夹下的2_deploy_contracts.js文件存在问题。修复并重新编译。

6.7 Uncaught Error: the tx doesn't have the correct nonce. account has nonce of: X tx has nonce of: Y

红色的X和Y是我自己打的，实际情况中可能是任何数字（X<Y）.

在metamask中切换到Main Ethereum Network再切回来。

6.8 VM Exception while processing transaction: invalid opcode

合约代码有问题……

6.9 metamask 提示 connecting to unknown private network

可能原因：

1 truffle没开

2 没切换到正确的网络

3 脸黑

4 其他原因

6.10 Source file requires different compiler version……

这个问题在vscode中出现了，可以忽略不计强行编译。。。。。。vscode是个智障编译器

6.11 npm报错

npm ERR! Command failed: /usr/bin/git clone -q https://github.com/frozeman/bignumber.js-nolookahead.git /home/hht/.npm/_cacache/tmp/git-clone-484532f1

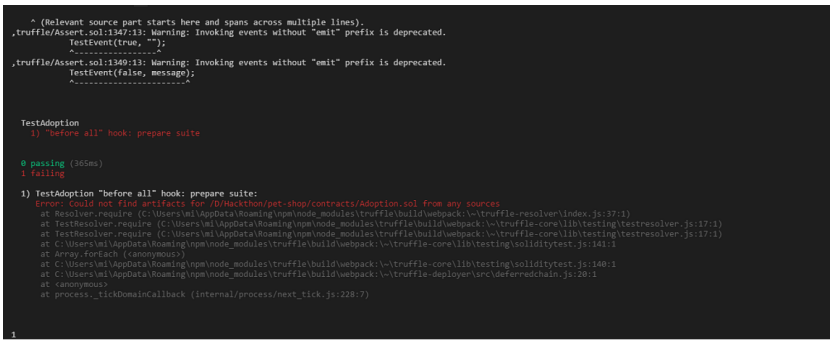
npm ERR! /home/hht/.npm/_cacache/tmp/git-clone-484532f1/.git: Permission denied

解决：

$ npm uninstall -g truffle

$ npm install -g truffle@beta

6.12 test出现 “before all” hook： prepare suite：



这个问题最近多次出现而以前没遇到过，可能是最近风水不太好……后多方查证有人用6.11那个办法解决了。原因我也不知道……猜测是truffle的版本问题。

6.13 如何使用library（存疑）

首先写一个library

```
pragma solidity ^0.4.16;

library N {
    function getN()returns(uint) {
        return 1;
    }
}
```

再写一个调用library的

```
pragma solidity ^0.4.16;
import "./N.sol";

contract TestN {
    function test() returns(uint){
        return N.getN();
    }
}
```

关键在2_deploy_contracts.js，注意有个link

```
var testN = artifacts.require("TestN");
var N = artifacts.require("N");
module.exports = function(deployer) {
  deployer.deploy(testN);
  deployer.deploy(N);
  deployer.link(N,testN);
};
```

6.14 调用sodity写的方法的时候 有时候要加 call 有时候又不加，这有啥区别啊？ （@信息部-苏瑞）

回答：这个问题主要是要理解call的作用，请先阅读下面的代码示例，以及他们的区别：

```
// Automatically determines the use of call or sendTransaction based on the method type
myContractInstance.myMethod(param1 [, param2, ...] [, transactionObject] [, defaultBlock] [, callback]);

// Explicitly calling this method
myContractInstance.myMethod.call(param1 [, param2, ...] [, transactionObject] [, defaultBlock] [, callback]);

// Explicitly sending a transaction to this method
myContractInstance.myMethod.sendTransaction(param1 [, param2, ...] [, transactionObject] [, callback]);

// Get the call data, so you can call the contract through some other means
var myCallData = myContractInstance.myMethod.getData(param1 [, param2, ...]);
```

```
// myCallData = '0x45ff3ff6000000000004545345345345..'
```

从示例的解释可以看到，myContractInstance.myMethod.call() 是直接调用myMethod方法；当不显示使用call时候，myContractInstance.myMethod()会自动根据方法 type 调用call 或者sendTransaction 方法。
sendTransaction方法是需要额外给出发起人账户的'from'，因此你提到的调用voteForCandidate方法时候，后面加call不可以，那是因为此时自动适配调用的是voteForCandidate.sendTransaction()方法，因此你可以尝试显示调用：contractInstance.voteForCandidate.**sendTransaction**('Rama', {from:web3.eth.accounts[0]}) 跟contractInstance.voteForCandidate('Rama', {from:web3.eth.accounts[0]})是一样的。

另外，关于call 还有额外的用法，供参考：https://solidity.readthedocs.io/en/latest/types.html#address

```
address nameReg = 0x72ba7d8e73fe8eb666ea66babc8116a41bfb10e2;
nameReg.call("register", "MyName");
nameReg.call(bytes4(keccak256("fun(uint256)")), a);
```

6.15 invalid address问题

重新编译部署

7 如何打包提交到比赛网站（@善禄）

打包前端，提交
在hackathon-demo目录：
1. npm run build
2. 编辑 hackathon-demo\dist\index.html 文件，将绝对路径替换为相对路径（将所有'=/static' 替换为 '=static'）
3. zip dist.zip -r dist
4. 到比赛报名网站(http://hackathon2018.ad.xiaomi.com/team/index) 提交dist.zip，完成

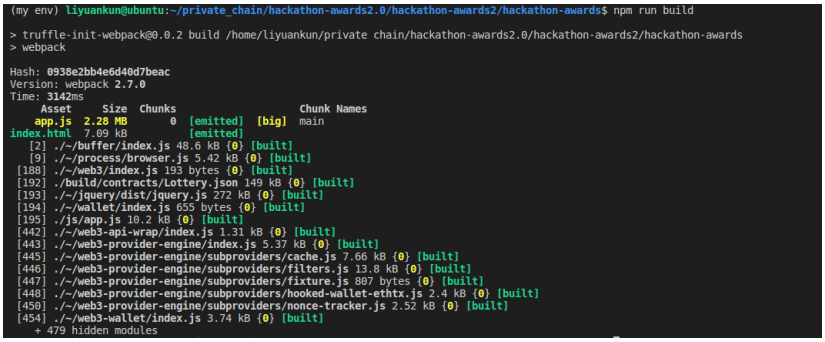
详情见 http://git.n.xiaomi.com/hackathon2018/hackathon-demo/tree/master

关于webpack打包前端及提交

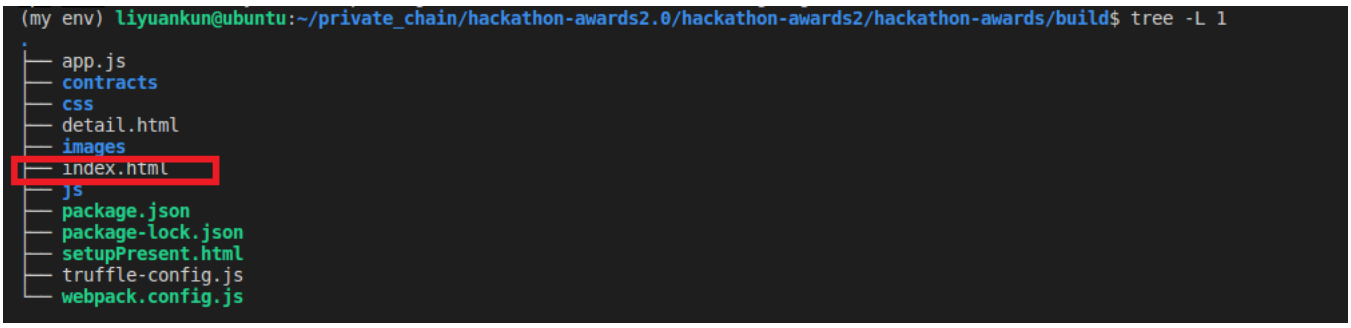
1. 在项目根目录运行 npm run build
2. 将和你网页相关的信息（包括所需.js，.css，图片，其他html文件等信息）一并放入build目录下
3. 在项目根目录下运行 zip dist.zip -r build
4. 到比赛报名网站(http://hackathon2018.ad.xiaomi.com/team/index) 提交压缩文件包，完成

举个栗子：

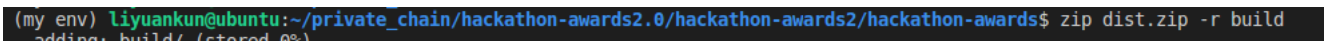
假设你的项目文件夹叫做：hackathon-awards。首先在根目录下运行npm run build，也就是这样：



然后将网页相关信息存放至build文件夹下，也就是这样：



然后返回上一层文件，运行zip dist.zip -r build，也就是这样：



最后到比赛报名网站上传，搞定！

PS：压缩前请务必注意将index.html放至build目录下！压缩前请务必注意将index.html放至build目录下！压缩前请务必注意将index.html放至build目录下！重要的事情说三遍。

由于上传以后，文件存放路径和原本项目会不一样，所以记得上传之前修改一下自己各个文件中的路径，以免上传后运行发生偏差。

无标签

评论

Di1 Yao 姚迪 发表:
好顶赞