2019-03-25 博彩项目工作记录

第一部分：安装及测试react框架

1 安装react框架

全局安装create-react-app工具

D:\code-nodejs>npm install -g create-react-app

C:\Users\dell\AppData\Roaming\npm\create-react-app -> C:\Users\dell\AppData\Roaming\npm\node\_modules\create-react-app\index.js

+ create-react-app@2.1.8

added 63 packages from 20 contributors in 7.232s

D:\code-nodejs>

2 新建工程目录reactDemo，进入reactDemo目录， 执行创建项目命令：

create-react-app lotterydemo

这里的lotterydemo就是项目名称，注意的是不能有大写字母，必须用小写。

D:\code-nodejs>mkdir reactDemo

D:\code-nodejs>cd reactDemo

D:\code-nodejs\reactDemo>create-react-app lotterydemo

Creating a new React app in D:\code-nodejs\reactDemo\lotterydemo.

Installing packages. This might take a couple of minutes.

Installing react, react-dom, and react-scripts...

+ react@16.8.5

+ react-dom@16.8.5

+ react-scripts@2.1.8

added 1842 packages from 718 contributors and audited 36240 packages in 306.209s

found 63 low severity vulnerabilities

run `npm audit fix` to fix them, or `npm audit` for details

Success! Created lotterydemo at D:\code-nodejs\reactDemo\lotterydemo

Inside that directory, you can run several commands:

npm start

Starts the development server.

npm run build

Bundles the app into static files for production.

npm test

Starts the test runner.

npm run eject

Removes this tool and copies build dependencies, configuration files

and scripts into the app directory. If you do this, you can’t go back!

We suggest that you begin by typing:

cd lotterydemo

npm start

Happy hacking!

D:\code-nodejs\reactDemo>

3 启动react 服务器

D:\code-nodejs\reactDemo>cd lotterydemo

D:\code-nodejs\reactDemo\lotterydemo>npm start

> lotterydemo@0.1.0 start D:\code-nodejs\reactDemo\lotterydemo

> react-scripts start

Starting the development server...

Compiled successfully!

You can now view lotterydemo in the browser.

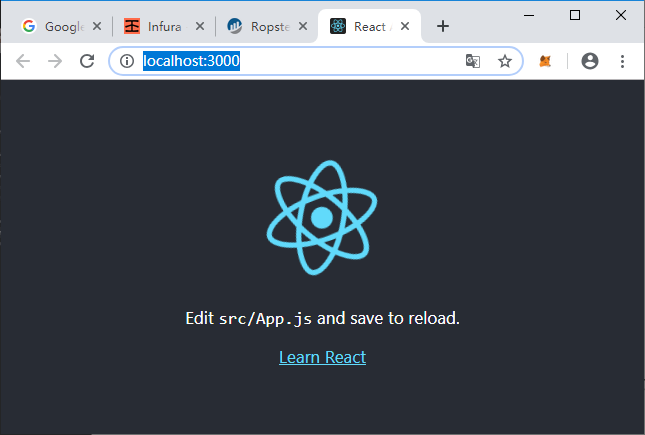
Local: http://localhost:3000/

On Your Network: http://192.168.202.1:3000/

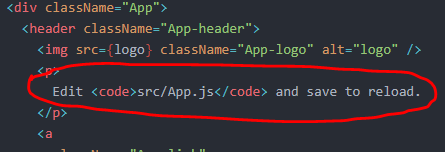
Note that the development build is not optimized.

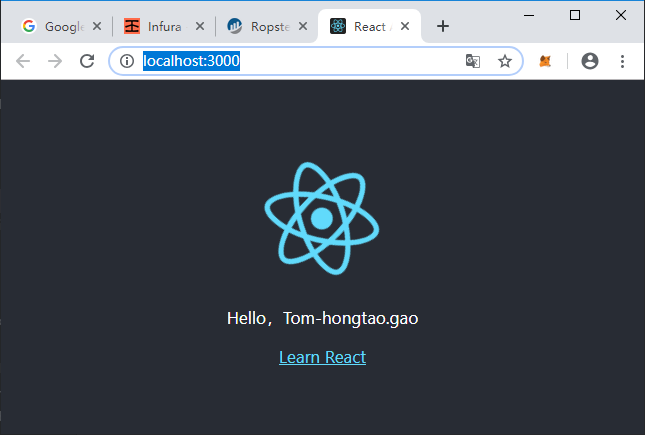
To create a production build, use npm run build.

启动后自动打开一个网页，地址是<http://localhost:3000/>；



中间的Edit…这句话提示我们可以修改src/app.js内容后重新加载显示出来。我们就在Atom修改这个文件：改成“Hello，Tom-hongtao.gao”， 页面马上就变更过来了。

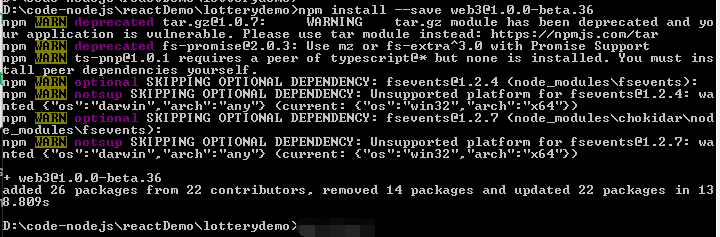




4 在lotterydemo目录下安装web3库

在reactDemo工程中安装web3时必须要指定版本1.0.0-beta.36，之后运行就正常。如果不指定版本，自动安装的是1.0.0-beta.50， 以后就会运行错误（例如读取的players错误，send命令发送后引发error等稀奇古怪问题）。

D:\code-nodejs\reactDemo\lotterydemo>npm install --save web3@1.0.0-beta.36



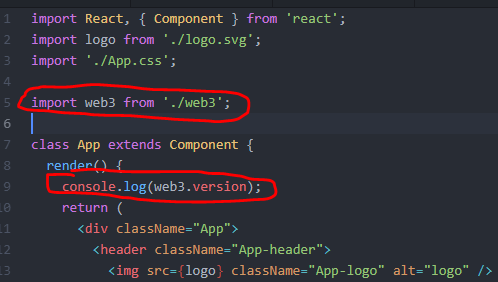
5 创建文件src/web3.js：

import Web3 from 'web3';

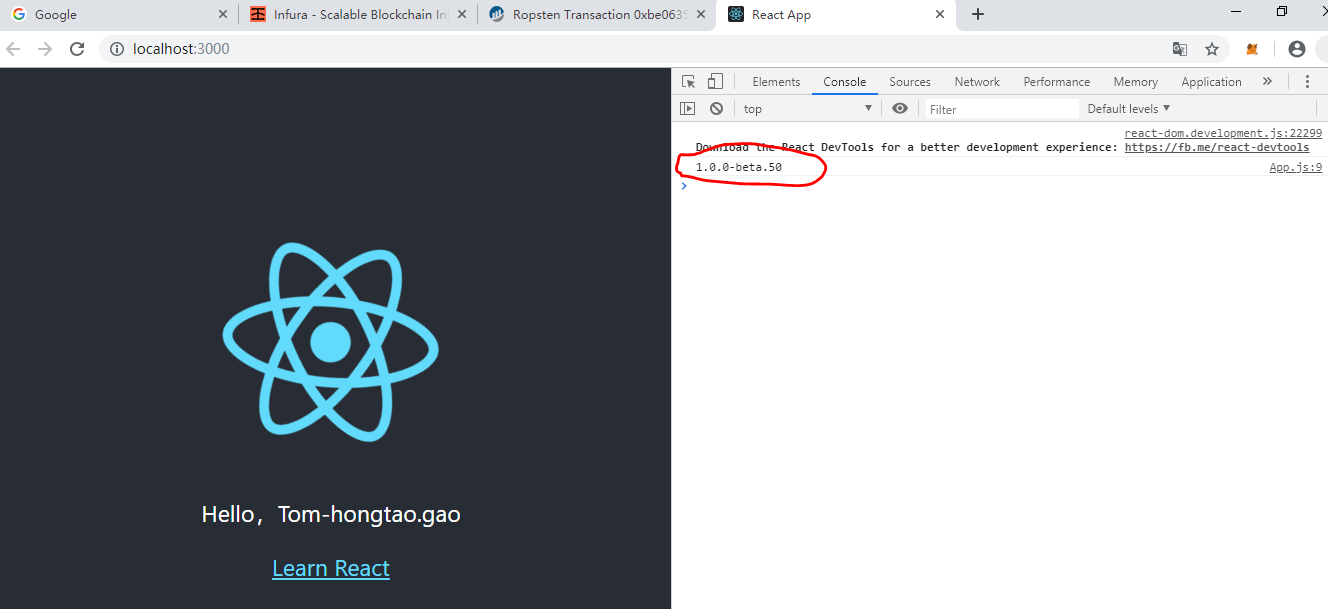
const web3 = new Web3(window.web3.currentProvider);

export default web3;

修改src/App.js,添加红圈中的语句，导入本地的web3.js库，在浏览器的控制台上输出版本号。



重新npm start启动，网页按F12查看控制台，看到下图的红圈中显示的版本号，大功告成!



第二部分 部署lottery合约及交互操作

2.1 之前在mytest工程中已经可以部署lottery合约了（new web3.eth.Contract(JSON.parse(interface)) ），现在更进一步，要求利用abi和address创建出合约对象。就修改mytest/deploy.ts,打印出interface和address。

//输出交易地址

console.log("contract deployed to: ",result.options.address);

console.log(interface);

2.2 在reactDemo工程中创建文件src/lottery.js, 内容如下：

import web3 from './web3';

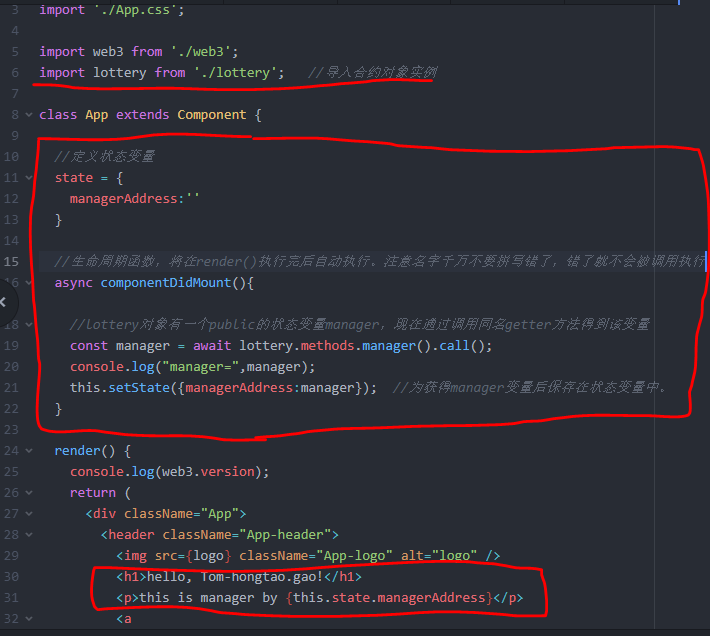
const address = '0x348D4e2606C1fbE8E9de058F6Ea67c96c50fa6F0';

const abi = [{"constant":true,"inputs":[],"name":"manager","outputs":[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":false,"inputs":[],"name":"pickWinner","outputs":[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":false,"inputs":[],"name":"entry","outputs":[],"payable":true,"stateMutability":"payable","type":"function"},{"constant":true,"inputs":[{"name":"","type":"uint256"}],"name":"players","outputs":[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","type":"function"},{"inputs":[],"payable":false,"stateMutability":"nonpayable","type":"constructor"}];

export default new web3.eth.Contract(abi,address);

这里就是使用mytest项目部署的合约abi接口和合约地址重新创建出的合约实例，输出该合约实例给App.js文件使用。

2.3 修改src/App.js文件，注意红色圆圈内容：



2.4 上述文件修改好了，再次执行npm start，将在网页上显示manager的地址。

