2019-03-25工作记录3-玩家入场

上一篇工作记录完成了网页元素显示，本篇记录了按钮的提交动作是如何完成转账操作的，最后判断输赢并发送彩金。

1 改造App.js状态变量，增加value、message。

2 设置表单的onSubmit事件，避免每次点击“提交按钮”刷新页面。

import React, { Component } from 'react';

import logo from './logo.svg';

import './App.css';

import web3 from './web3';

import lottery from './lottery'; //导入合约对象实例

class App extends Component {

//定义状态变量

state = {

manager:'',

players:[],

balance:'',

value:'', //记录用户的下注金额

message:'' //操作过程信息

}

//生命周期函数，将在render()执行完后自动执行。注意名字千万不要拼写错了，错了就不会被调用执行

async componentDidMount(){

//lottery对象有一个public的状态变量manager，现在通过调用同名getter方法得到该变量

const manager = await lottery.methods.manager().call();

const players = await lottery.methods.getPlayers().call(); //数组变量比较特殊，必须要单独编写get函数

const balance = await web3.eth.getBalance(lottery.options.address);

console.log("players===>",players);

console.log("000 balance = ",balance);

this.setState({manager,players,balance}); //为获得manager变量后保存在状态变量中。

}

//send函数提供了玩家账户和下注金额，转账给合约账户，entry()函数调用时就存入玩家账号，接受下注金额。

onSubmit = async event =>{

event.preventDefault();

const accounts = await web3.eth.getAccounts();

this.setState({message:' 等待交易完成......'});

/\* 我运行send这句话就无法返回，执行不下去，出现错误提示：

Uncaught (in promise) Error: Transaction has been reverted by the EVM:

但是交易在网站查询时正确成功的，

（1）查明的原因是： reactDemo工程中安装的web3库是最新的1.0.0-beta.50，这是有问题的。

必须换成1.0.0-beta.36版本，就不会出现上述错误，切记切记！！！

（2）如果坚持用1.0.0-beta.50版本，我找到的方法是添加catch，捕捉错误,这样就能返回，

可以执行下一行语句了，但是导致entry函数无法被执行。这样挽救方法不可取。

\*/

await lottery.methods.entry()

.send({from:accounts[0], value:web3.utils.toWei(this.state.value,'ether')})

.catch(err => {

console.log(err) // 这里catch到错误，也许是服务器的异常问题，作为一个预防措施

});

this.setState({message:'玩家入场成功！'});

const players = await lottery.methods.getPlayers().call(); //数组变量比较特殊，必须要单独编写get函数

const balance = await web3.eth.getBalance(lottery.options.address);

//更新界面上的数据

console.log("111 players = ",players);

console.log("111 balance = ",balance);

this.setState({players,balance}); //为获得manager变量后保存在状态变量中。

//如何清空输入框？？？

console.log("9999999");

}

render() {

console.log(this.state.value);

return (

<div>

<h1>lottery管理者地址:</h1>

<p>this is manager by {this.state.manager}</p>

<p>当前参与者数量:{this.state.players.length}</p>

<p>当前资金池: {web3.utils.fromWei(this.state.balance,'ether')}</p>

<hr/>

<form onSubmit={this.onSubmit}>

<h4>参与到博彩项目？</h4>

<div>

<label>您的投注金额：</label>

<input id='stake'

value = {this.state.value}

onChange = {event=>{this.setState({value:event.target.value})}}

/>

</div>

<button>提交</button>

</form>

<hr/>

<p>{this.state.message}</p>

</div>

);

}

}

export default App;

3 重点是这句话，把用户输入的下注金额转账到了博彩账户中，注意下注最小金额要大于0.01ether，否则会出错，交易失败。

await lottery.methods.entry()

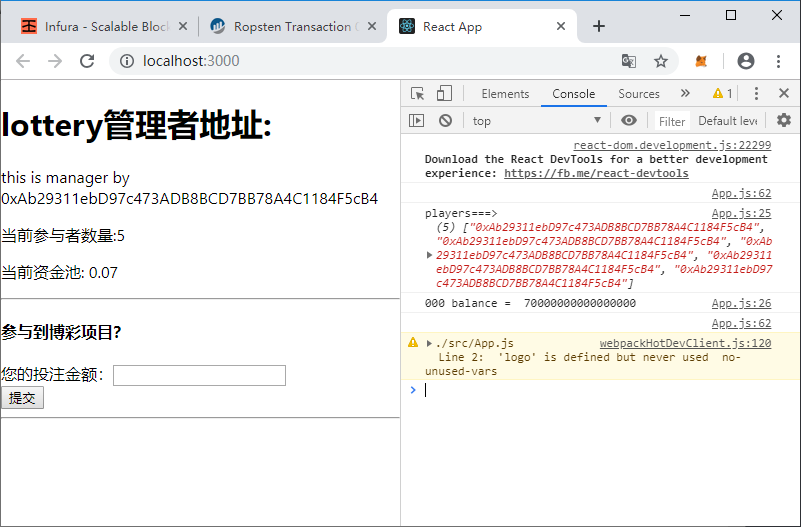
.send({from:accounts[0], value:web3.utils.toWei(this.state.value,'ether')})

.catch(err => {

console.log(err) // 这里catch到错误，作为保护措施

});

4 界面如下： 完美成功~~~~



5 现在就剩下最后一步了，只要调用合约中的判断赢家函数，给赢家发送奖金就全部完工了。

在 App.js中增加开奖按钮处理代码：

onClick = async event =>{

const accounts = await web3.eth.getAccounts();

this.setState({message:' 等待交易完成......'});

await lottery.methods.pickWinner().send({from:accounts[0]});

this.setState({message:'赢家产生了......'});

}

网页上增加开奖按钮：

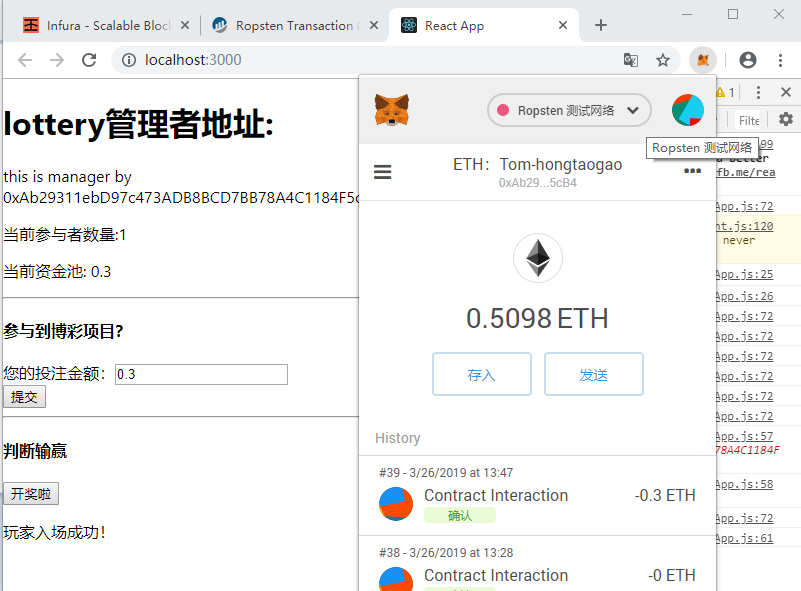
<h4>判断输赢</h4>

<button onClick={this.onClick}>开奖啦</button>

<p>{this.state.message}</p>

最后的运行结果如下图：

下注成功，注意玩家的账户金额：



只有一个玩家开奖，那么将会获得奖池中的0.3ether，账户金额增加。

