法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

关注 小象学院

区块链编程：Solidity以太坊智能合约

王亮

第五课 合约调用与web3.js

5.3 web3.js的使用与案例

合约调用基本流程

- 初始化web3，连接以太坊节点rpc服务，获得一个provider对象
- 初始化合约对象。
- 合约对象的provider设置为已经初始化的web3对象。
- 调用合约。
- 监听合约。

合约调用

- 合约调用可以使用call或者send。
- myContract.methods.myMethod([param1[, param2[, ...]]]).call/send(options[, callback])
- options 可以包括from, gasPrice,gas,value。分别代表调用者地址，gas价格，消耗的最低gas,发送的以太币数量。

Web3调用合约的例子

```
var Web3 = require('web3');
console.log(Web3.version);
//设置web3对象
var web3 = new Web3('http://localhost:8545');
var json = require("../build/contracts/Hello.json");
var abi = json["abi"];

var address = "0x91ab99f3f2210944164ef9b2d6e07e23e0fe49cd";

//合约实例
var contract = new web3.eth.Contract(abi,address);

//callback
contract.methods.helloWorld().call(function(error, result){
    console.log(result);
});
```

truffle对web3的封装

```
//1. 引入编译好的合约文件结果
var json = require("./build/contracts/MyContract.json");

// 2. 将合约转为合约抽象层实例
var contract = require("truffle-contract");
var MyContract = contract(json);

// 3. 设置合约抽象层实例的web3 provider
MyContract.setProvider(new Web3.providers.HttpProvider("http://localhost:8545"));

// 4. 现在你可以使用啦
MyContract.deployed().then(function(deployed) {
  return deployed.someFunction();
});
```

Truffle 封装web3的优点

- 对以太坊的智能合约做了更好的抽象，使用简单。
- 同步的交易：可以确保在交易生效之后再继续执行其他操作。
- 返回Promise：每个封装的合约函数会返回Promise，可以对它进行.then操作，避免了回调地狱（callback hell）问题。
- 为交易提供了默认参数：例如from或gas。
- (注: web3 1.0以后，以上优点也具有)

案例

- 课程积分案例
- 奖品竞品案例

流程演示

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：**小象学院**