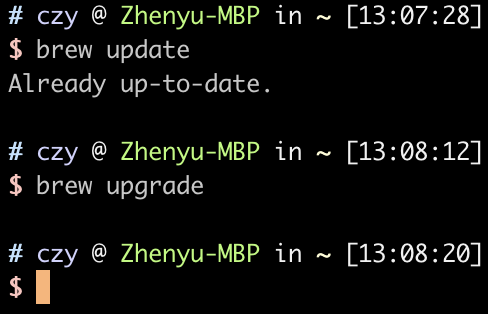
Mac 下搭建以太坊开发环境

**1.** 安装依赖

**1.1**安装**Homebrew**并确保它为最新版本

brew update

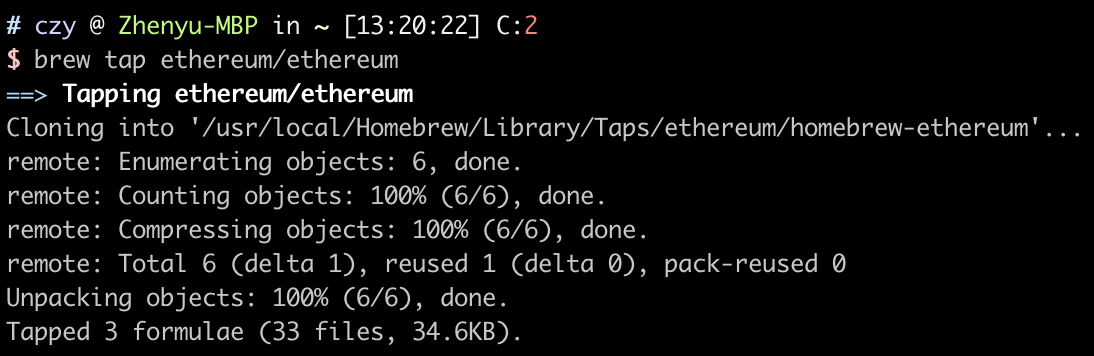
brew upgrade



**1.2** 安装**go**



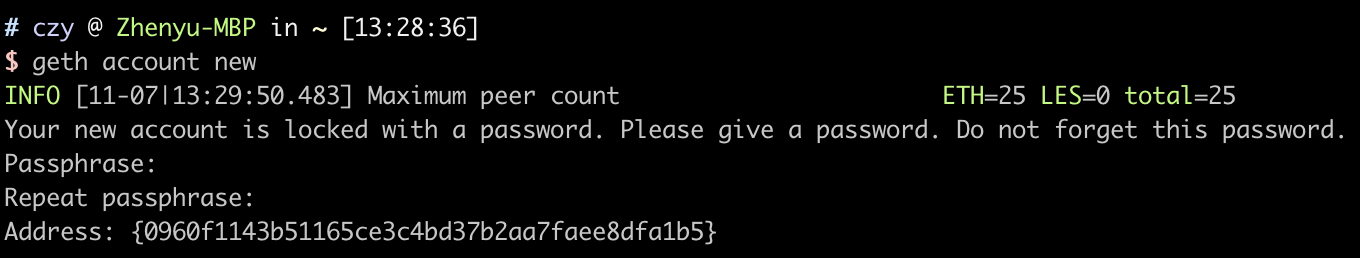
**1.3** 安装**ethereum**

1) brew tap ethereum/ethereum

2) brew install ethereum

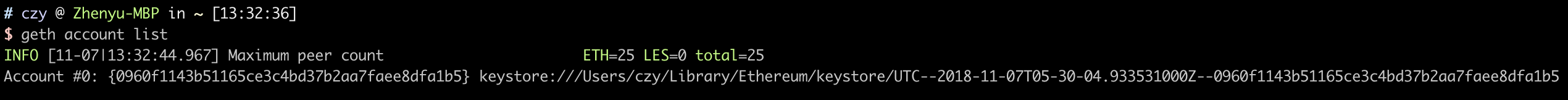
**2.**配置以太坊环境

**2.1** 创建账户

geth account new可以得到以太坊账户地址：0960f1143b51165ce3c4bd37b2aa7faee8dfa1b5

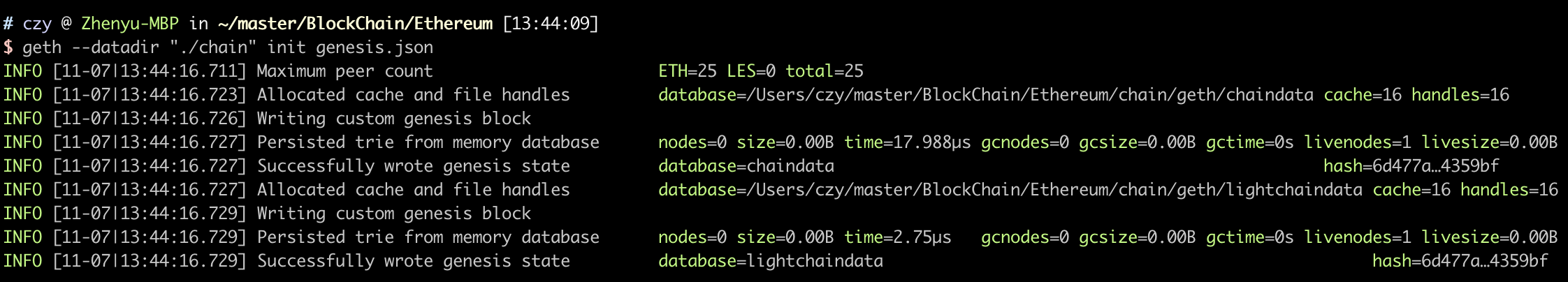
**2.2** 查看账户

geth account list



**2.3** 生成创世块

创建一个文件夹Ethereum，用于存储区块链相关数据，并在该目录下建立genesis.json文件，之后在目录chain下生成创世块。

geth --datadir "./chain" init genesis.json

**2.4** 启动以太坊

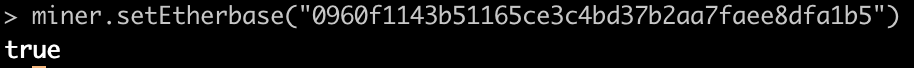
geth --datadir "./chain" console



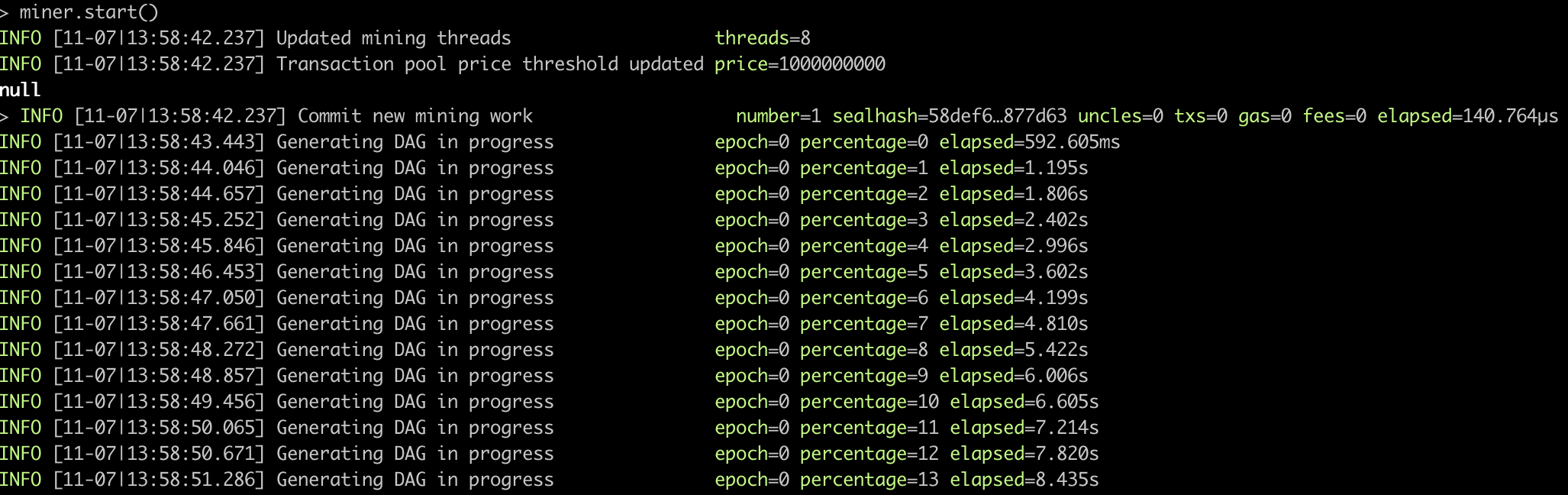
**2.5** 挖矿

1) 设置挖矿账号

miner.setEtherbase("0960f1143b51165ce3c4bd37b2aa7faee8dfa1b5")

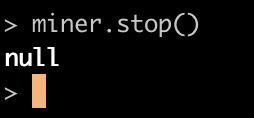


2) 启动挖矿

miner.start()

3) 停止挖矿

miner.stop()



**3.** 以太坊编程接口

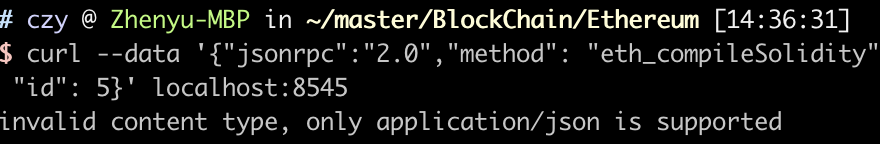
**3.1** 通过**JSON RPC**编译部署合约

1) 开启RPC端口

geth --datadir "./chain" --rpcport "8545" --rpc console

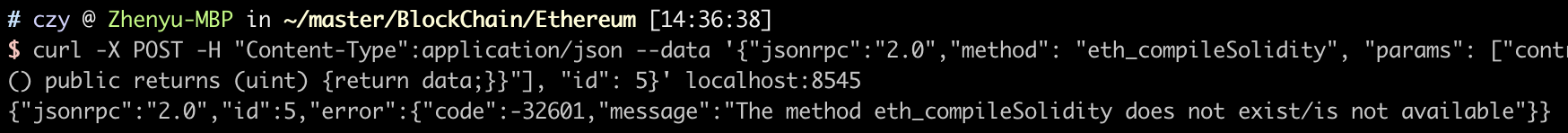


2) 执行编译命令

curl --data '{"jsonrpc":"2.0","method": "eth\_compileSolidity", "params": ["contract SimpleTest {uint data;function set(uint n) public {data = n\*3;}function get() public returns (uint) {return data;}}"], "id": 5}' localhost:8545

提示invalid content type, only application/json is supported。

3) 添加头部后执行编译指令

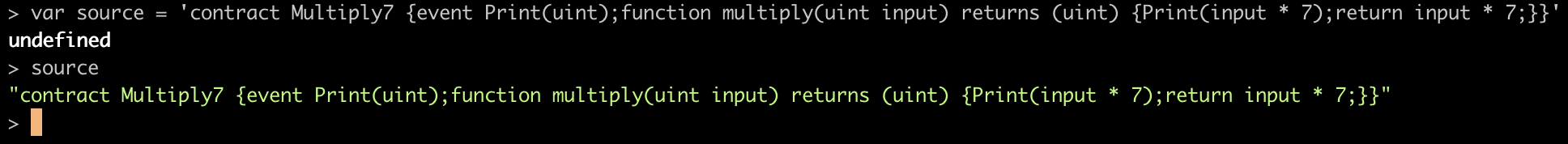
curl -X POST -H "Content-Type":application/json --data '{"jsonrpc":"2.0","method": "eth\_compileSolidity", "params": ["contract SimpleTest {uint data;function set(uint n) public {data = n\*3;}function get() public returns (uint) {return data;}}"], "id": 5}' localhost:8545

提示The method eth\_compileSolidity does not exist/is not available。

3.2 通过JavaScript API编译部署合约

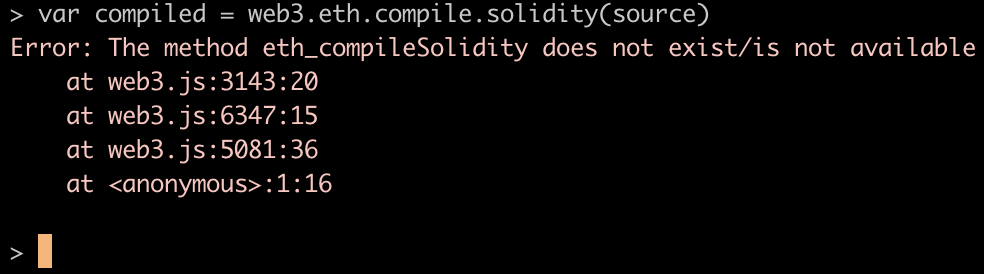
1) 定义合约

var source = 'contract Multiply7 {event Print(uint);function multiply(uint input) returns (uint) {Print(input \* 7);return input \* 7;}}’



2) 编译合约

var compiled = web3.eth.compile.solidity(source)



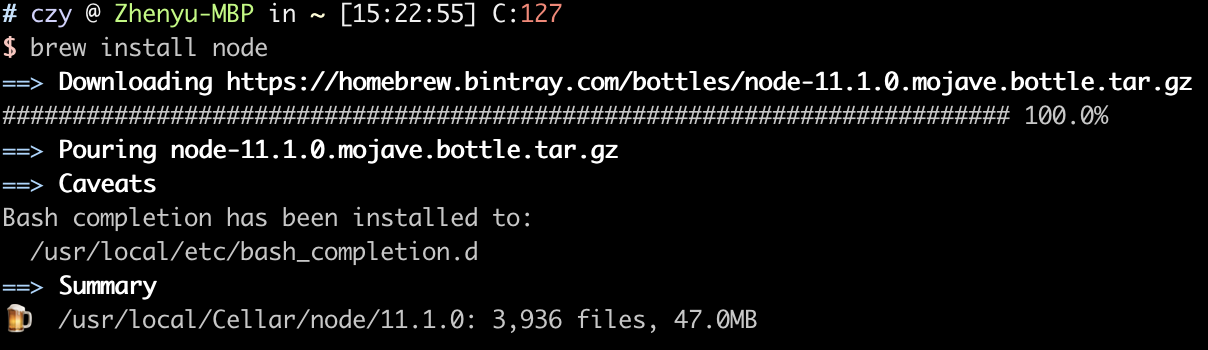
依旧提示The method eth\_compileSolidity does not exist/is not available。

因为在geth 1.6版之后就废弃了“eth\_compileSolidity”这个函数，所以无法通过上述两种方法编译合约。

**3.3** 通过在线**Remix**编译部署合约

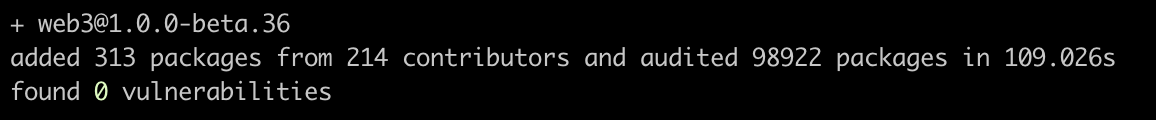
1) 安装node

brew install node



2) 安装web3

npm install web3

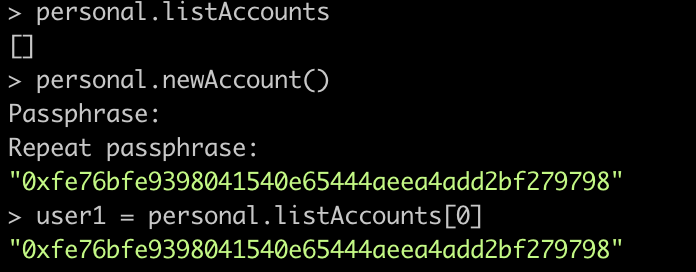


3) 新建账号

personal.listAccounts

personal.newAccount()

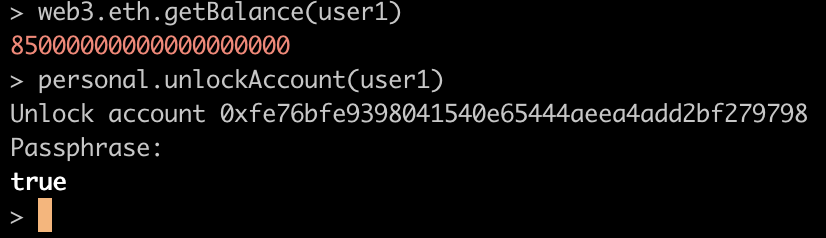
user1 = personal.listAccounts[0]



由于是新账号，没有余额，所以需要先挖矿。在挖坑之后，检查账号的余额，并解锁该账号

web3.eth.getBalance(user1)

personal.unlockAccount(user1)



4) 编译合约

使用下面的小程序编译部署合约

pragma solidity ^0.4.0;

contract Multiply6 {

event Print(uint);

function multiply(uint input) returns (uint) {

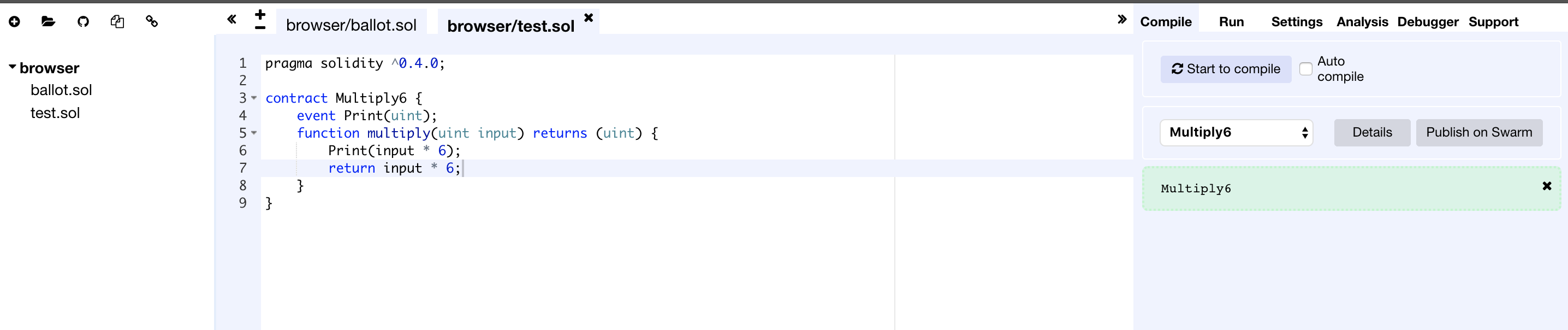
Print(input \* 6);

return input \* 6;

}

}

将程序放入[Solidity语言的在线编译器](https://link.jianshu.com/?t=http://ethereum.github.io/browser-solidity/)进行编译



复制WEB3DEPLOY



将WEB3程序全部粘贴到控制台，运行完程序之后开始挖坑，获取合约地址和此次交易的hash值

var multiply6Contract = web3.eth.contract([{"constant":false,"inputs":[{"name":"input","type":"uint256"}],"name":"multiply","outputs":[{"name":"","type":"uint256"}],"payable":false,"type":"function","stateMutability":"nonpayable"},{"anonymous":false,"inputs":[{"indexed":false,"name":"","type":"uint256"}],"name":"Print","type":"event"}]);

var multiply6 = multiply6Contract.new(

{

from: web3.eth.accounts[0],

data: '',

gas: '4700000'

}, function (e, contract){

console.log(e, contract);

if (typeof contract.address !== 'undefined') {

console.log('Contract mined! address: ' + contract.address + ' transactionHash: ' + contract.transactionHash);

}

})

miner.start()



在获取合约地址和此次交易的散列值之后停止挖坑，并查看当前的合约

miner.stop()

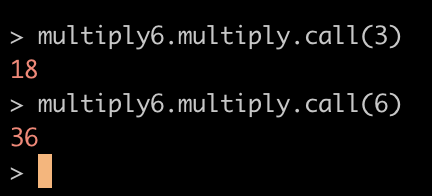
multiply6



对合约进行简单的测试

multiply6.multiply.call(3)

multiply6.multiply.call(3)



解锁当前帐号之后，发起交易，再通过交易的hash值查看当前交易

personal.unlockAccount(user1)

multiply6.multiply.sendTransaction(6, {from:user1, gas:200000})

eth.getTransaction(“0x44c65903c719522125a289286029c5976c3783d6c96cf73196aa602238415bbd")



References

<https://github.com/astaxie/build-web-application-with-golang/blob/master/zh/01.1.md>

https://blog.csdn.net/weixin\_36759405/article/details/83189004

https://www.ethereum.org/cli

<https://github.com/ethereum/go-ethereum/wiki/Installation-Instructions-for-Mac>

<https://gitter.im/ethereum/welcome/archives/2018/04/02>

http://liyuechun.org/2017/10/16/eth-private-blockchain/

<https://github.com/ethereum/wiki/wiki/JSON-RPC>

<https://github.com/ethereum/go-ethereum/wiki/JavaScript-Console>

https://juejin.im/post/5ad95da5f265da0b8a672a21