星云链Nebulas——3.编译和部署智能合约

在这篇教程中我们会学习怎样在Nebulas中编写、部署并执行智能合约。

1、准备工作

1.1 温习内容

在进入智能合约之前，先温习下之前学过的内容：

1. 安装、编译并启动neb应用
2. 创建钱包地址，设置coinbase，并开始挖矿
3. 查询neb节点信息、钱包地址余额等
4. 发送转账交易，并验证交易是否成功

1.2 学习目的

如果对上述的内容有疑惑的朋友可以重新去学习前面的章节，接下来我们会通过下面的步骤来学习和使用智能合约：

1. 编写智能合约
2. 部署智能合约
3. 调用智能合约，验证合约执行结果

2、编写智能合约

2.1 简要规范

跟以太坊类似，Nebulas实现了NVM虚拟机来运行智能合约，NVM的实现使用了JavaScript V8引擎，所以当前的开发版，我们可以使用JavaScript、TypeScript来编写智能合约。

编写智能合约的简要规范：

1. 智能合约代码必须是一个Prototype的对象；
2. 智能合约代码必须有一个init()的方法，这个方法只会在部署的时候被执行一次；
3. 智能合约里面的私有方法是以_开头的方法，私有方法不能被外部直接调用。

2.2 编写合约示例

下面我们使用JavaScript来编写第一个智能合约：银行保险柜。

这个智能合约需要实现以下功能：
1. 用户可以向这个银行保险柜存钱。
2. 用户可以从这个银行保险柜取钱。
3. 用户可以查询银行保险柜中的余额。

智能合约源代码：

```
var DepositeContent = function (text) {
  if (text) {
    var o = JSON.parse(text);
    this.balance = new BigNumber(o.balance);
    this.expiryHeight = new BigNumber(o.expiryHeight);
  } else {
    this.balance = new BigNumber(0);
    this.expiryHeight = new BigNumber(0);
  }
};

DepositeContent.prototype = {
  toString: function () {
    return JSON.stringify(this);
  }
};

var BankVaultContract = function () {
  LocalContractStorage.defineMapProperty(this, "bankVault", {
    parse: function (text) {
      return new DepositeContent(text);
    },
    stringify: function (o) {
      return o.toString();
    }
  });
};

// save value to contract, only after height of block, users can takeout
BankVaultContract.prototype = {
  init: function () {
    //TODO:
  },

  save: function (height) {
    var from = Blockchain.transaction.from;
    var value = Blockchain.transaction.value;
    var bk_height = new BigNumber(Blockchain.block.height);

    var orig_deposit = this.bankVault.get(from);
```

```
    if (orig_deposit) {
      value = value.plus(orig_deposit.balance);
    }

    var deposit = new DepositeContent();
    deposit.balance = value;
    deposit.expiryHeight = bk_height.plus(height);

    this.bankVault.put(from, deposit);
  },

  takeout: function (value) {
    var from = Blockchain.transaction.from;
    var bk_height = new BigNumber(Blockchain.block.height);
    var amount = new BigNumber(value);

    var deposit = this.bankVault.get(from);
    if (!deposit) {
      throw new Error("No deposit before.");
    }

    if (bk_height.lt(deposit.expiryHeight)) {
      throw new Error("Can not takeout before expiryHeight.");
    }

    if (amount.gt(deposit.balance)) {
      throw new Error("Insufficient balance.");
    }

    var result = Blockchain.transfer(from, amount);
    if (!result) {
      throw new Error("transfer failed.");
    }
    Event.Trigger("BankVault", {
      Transfer: {
        from: Blockchain.transaction.to,
        to: from,
        value: amount.toString()
      }
    });

    deposit.balance = deposit.balance.sub(amount);
    this.bankVault.put(from, deposit);
  },
  balanceOf: function () {
    var from = Blockchain.transaction.from;
    return this.bankVault.get(from);
  },
```

```
  verifyAddress: function (address) {
    // 1-valid, 0-invalid
    var result = Blockchain.verifyAddress(address);
    return {
      valid: result == 0 ? false : true
    };
  }
};
module.exports = BankVaultContract;
```

上面智能合约的示例可以看到，`BankVaultContract`是一个prototype对象，这个对象有一个init()方法，满足了我们说的编写智能合约最基础的规范。

`BankVaultContract`实现了另外两个方法：

- save(): 用户可以通过调用save()方法向银行保险柜存钱；
- takeout(): 用户可以通过调用takeout()方法向银行保险柜取钱；
- balanceOf(): 用户可以通过调用balanceOf()方法向银行保险柜查询余额；

2.3 合约代码分析

上面的合约代码用到了内置的`Blockchain`对象和内置的`BigNumber()`方法，下面我们来逐行拆解分析合约代码：

- save方法

```
// Deposit the amount into the safe

save: function (height) {
  var from = Blockchain.transaction.from;
  var value = Blockchain.transaction.value;
  var bk_height = new BigNumber(Blockchain.block.height);

  var orig_deposit = this.bankVault.get(from);
  if (orig_deposit) {
    value = value.plus(orig_deposit.balance);
  }
  var deposit = new DepositeContent();
  deposit.balance = value;
  deposit.expiryHeight = bk_height.plus(height);

  this.bankVault.put(from, deposit);
},
```

- takeout方法

```
takeout: function (value) {
  var from = Blockchain.transaction.from;
  var bk_height = new BigNumber(Blockchain.block.height);
  var amount = new BigNumber(value);

  var deposit = this.bankVault.get(from);
  if (!deposit) {
    throw new Error("No deposit before.");
  }

  if (bk_height.lt(deposit.expiryHeight)) {
    throw new Error("Can not takeout before expiryHeight.");
  }

  if (amount.gt(deposit.balance)) {
    throw new Error("Insufficient balance.");
  }

  var result = Blockchain.transfer(from, amount);
  if (!result) {
    throw new Error("transfer failed.");
  }
  Event.Trigger("BankVault", {
    Transfer: {
      from: Blockchain.transaction.to,
      to: from,
      value: amount.toString()
    }
  });

  deposit.balance = deposit.balance.sub(amount);
  this.bankVault.put(from, deposit);
},
```

> **注意：进行以下操作时，请确保星云链节点已经启动。**

3、部署智能合约

上面介绍了在Nebulas中怎么去编写一个智能合约，现在我们需要把智能合约部署到链上。

3.1 发送交易

在Nebulas中部署一个智能合约其实也是发送一个transaction来实现，前面有介绍过用户如何在

Nebulas中进行转账交易，由于是本地测试，我们直接使用`解锁 & 发送`的方式来发送交易。

首先，我们在`conf/default/genesis.conf`中预分配过代币的账户里，选择账户`n1H4MYms9F55ehcvygwWE71J8tJC4CRr2so`作为发送者账号，并检查该账户的状态。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H Accept:application/json -X POST h
ttp://localhost:8685/v1/user/accountstate -d '{"address":"n1H4MYms9F55ehcvyg
wWE71J8tJC4CRr2so"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:10:36 GMT
Content-Length: 72

{"result":{"balance":"500000000000000000000000","nonce":"0","type":87}}
```

3.2 解锁发送者账户

该账户有足够的钱来支付手续费，接下来，我们解锁发送者账户`n1H4MYms9F55ehcvygwWE71J8tJC4CRr2so`，解锁12小时。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/admin/account/unlock -d '{"address":"n1H4MY
ms9F55ehcvygwWE71J8tJC4CRr2so","passphrase":"passphrase","duration":"4320000
0000000"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:18:04 GMT
Content-Length: 26

{"result":{"result":true}}
```

3.3 部署智能合约

接下来，发送部署BankVault合约的交易。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Accept: application/json' -X POS
T http://localhost:8685/v1/admin/transactionWithPassphrase -H 'Content-Type:
 application/json' -d '{"transaction": {"from":"n1H4MYms9F55ehcvygwWE71J8tJC
4CRr2so","to":"n1H4MYms9F55ehcvygwWE71J8tJC4CRr2so", "value":"0","nonce":1,"
gasPrice":"1000000","gasLimit":"2000000","contract":{"source":"\"use strict\
";var DepositeContent=function(text){if(text){var o=JSON.parse(text);this.ba
```

```
lance=new BigNumber(o.balance);this.expiryHeight=new BigNumber(o.expiryHeigh
t);}else{this.balance=new BigNumber(0);this.expiryHeight=new BigNumber(0);}}
;DepositeContent.prototype={toString:function(){return JSON.stringify(this);
}};var BankVaultContract=function(){LocalContractStorage.defineMapProperty(t
his,\"bankVault\",{parse:function(text){return new DepositeContent(text);},s
tringify:function(o){return o.toString();}});};BankVaultContract.prototype={
init:function(){},save:function(height){var from=Blockchain.transaction.from
;var value=Blockchain.transaction.value;var bk_height=new BigNumber(Blockcha
in.block.height);var orig_deposit=this.bankVault.get(from);if(orig_deposit){
value=value.plus(orig_deposit.balance);} var deposit=new DepositeContent();d
eposit.balance=value;deposit.expiryHeight=bk_height.plus(height);this.bankVa
ult.put(from,deposit);},takeout:function(value){var from=Blockchain.transact
ion.from;var bk_height=new BigNumber(Blockchain.block.height);var amount=new
 BigNumber(value);var deposit=this.bankVault.get(from);if(!deposit){throw ne
w Error(\"No deposit before.\");} if(bk_height.lt(deposit.expiryHeight)){thr
ow new Error(\"Can not takeout before expiryHeight.\");} if(amount.gt(deposi
t.balance)){throw new Error(\"Insufficient balance.\");} var result=Blockcha
in.transfer(from,amount);if(!result){throw new Error(\"transfer failed.\");}
 Event.Trigger(\"BankVault\",{Transfer:{from:Blockchain.transaction.to,to:fr
om,value:amount.toString()}});deposit.balance=deposit.balance.sub(amount);th
is.bankVault.put(from,deposit);},balanceOf:function(){var from=Blockchain.tr
ansaction.from;return this.bankVault.get(from);},verifyAddress:function(addr
ess){var result=Blockchain.verifyAddress(address);return{valid:result==0?fal
se:true};}};module.exports=BankVaultContract;","sourceType":"js", "args":""}
}, "passphrase": "passphrase"}'
HTTP/1.1 100 Continue

HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:26:41 GMT
Content-Length: 145

{"result":{"txhash":"4ebe3c6f2b9ad7bf2459140c594633937fd11e3159dfa04f735ba58
40a8f1037","contract_address":"n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM"}}
```


- from: 合约创建者账户地址
- to: 和from一致，同为合约创建者地址
- value：部署合约时为 `"0"`
- nonce: 比创建者当前的 `nonce` 多1，可以通过 `GetAccountState` 获取创建前当前 `nonce`
- gasPrice：部署智能合约用到的gasPrice，可以通过 `GetGasPrice` 获取，或者使用默认值: `"1000000"` ；
- gasLimit: 部署合约的gasLimit，通过 `EstimateGas` 可以获取部署合约的gas消耗，不能使用默认值，也可以设置一个较大值，执行时以实际使用计算。
- contract: 合约信息，部署合约时传入的参数

- `source`：合约代码
- `sourceType`：合约代码类型，支持 `js` 和 `ts`(对应javaScript和typeScript代码)
- `args`：合约初始化方法参数，无参数为空字符串，有参数时为JSON数组

部署智能合约的返回值是transaction的hash地址txhash和合约的部署地址contract_address。 得到返回值并不能保证合约已经部署成功，因为发送交易是一个异步的过程，需要经过矿工打包，正如之前的转账交易一样，转账并不能实时到账，依赖矿工打包的速度，所以需要等待一段时间，然后可以通过查询合约地址的信息或者调用智能合约来验证合约是否部署成功。

3.4 验证合约是否部署成功

在部署智能合约的时候得到了transaction的hash地址txhash，我们可以很方便的使用console控制台查询transaction的hash信息来验证合约是否部署成功。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/user/getTransactionReceipt -d '{"hash":"4eb
e3c6f2b9ad7bf2459140c594633937fd11e3159dfa04f735ba5840a8f1037"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:27:08 GMT
Transfer-Encoding: chunked

{"result":{"hash":"4ebe3c6f2b9ad7bf2459140c594633937fd11e3159dfa04f735ba5840
a8f1037","chainId":100,"from":"n1H4MYms9F55ehcvygwWE71J8tJC4CRr2so","to":"n1
H4MYms9F55ehcvygwWE71J8tJC4CRr2so","value":"0","nonce":"1","timestamp":"1528
271021","type":"deploy","data":"eyJTb3VyY2VUeXBlIjoianMiLCJTb3VyY2UiOiJcInVz
ZSBzdHJpY3RcIjt2YXIgRGVwb3NpdGVDb250ZW50PWZ1bmN0aW9uKHRleHQpe2lmKHRleHQpe3Zh
ciBvPUpTT04ucGFyc2UodGV4dCk7dGhpcy5iYWxhbmNlPW5ldyBCaWdOdW1iZXIoby5iYWxhbmNl
KTt0aGlzLmV4cGlyeUhlaWdodD1uZXcgQmlnTnVtVyKG8uZXhwaXJ5SGVpZ2h0KTt9ZWxzZXt0
aGlzLmJhbGFuY2U9bmV3IEJpZ051bWJlcigwKTt0aGlzLmV4cGlyeUhlaWdodD1uZXcgQmlnTnVt
YmVyKDApO319O0RlcG9zaXRlQ29udGVudC5wcm90b3R5cGU9e3RvU3RyaW5nOmZ1bmN0aW9uKCl7
cmV0dXJuIEpTT04uc3RyaW5naWZ5KHRoaXMpO319O3ZhciBCYW5rVmF1bHRDb250cmFjdD1mdW5j
dGlvbigpe0xvY2FsQ29udHJhY3RTdG9yYWdlLmRlZmluZU1hcFByb3BlcnR5KHRoaXMsXCJiYW5r
VmF1bHRcIix7cGFyc2U6ZnVuY3Rpb24odGV4dCl7cmV0dXJuIG5ldyBEZXBvc2l0ZUNvbnRlbnQo
dGV4dCk7fSxzdHJpbmdpZnk6ZnVuY3Rpb24obyl7cmV0dXJuIG8udG9TdHJpbmcoKTt9fSk7fTtC
YW5rVmF1bHRDb250cmFjdC5wcm90b3R5cGU9e2luaXQ6ZnVuY3Rpb24oKXt9LHNhdmU6ZnVuY3Rp
b24oKXt2YXIgZnJvbT1CbG9ja2NoYWluLnRyYW5zYWN0aW9uLmZyb207dmFyIHZhbHVl
PUJsb2NrY2hhaW4udHJhbnNhY3Rpb24udmFsdWU7dmFyIGJrX2hlaWdodD1uZXcgQmlnTnVtYmVy
KEJsb2NrY2hhaW4uYmxvY2suaGVpZ2h0KTt2YXIgb3JpZ19kZXBvc2l0PXRoaXMuYmFua1ZhdWx0
LmdldChmcm9tKTtpZihvcmlnX2RlcG9zaXQpe3ZhbHVlPXZhbHVlLnBsdXMob3JpZ19kZXBvc2l0
LmJhbGFuY2UpO30gdmFyIGRlcG9zaXQ9bmV3IERlcG9zaXRlQ29udGVudCgpO2RlcG9zaXQuYmFs
YW5jZT12Ywx1ZTtkZXBvc2l0LmV4cGlyeUhlaWdodD1ia19oZWlnaHQucGx1cyhoZWlnaHQpO3Ro
aXMuYmFua1ZhdWx0LnB1dChmcm9tLGRlcG9zaXQpO30gdGFrZW91dDpmdW5jdGlvbih2Ywx1ZSl7
dmFyIGZyb209QmxvY2tjaGFpbi50cmFuc2FjdGlvbi5mcm9tO3ZhciBia19oZWlnaHQ9bmV3IEJp
```

Z051bWJlcihCbG9ja2NoYWluLmJsb2NrLmhlaWdodCk7dmFyIGFtb3VudD1uZXcgQmlnTnVtYmVy
KHZhbHVlKTt2YXIgZGVwb3NpdD10aGlzLmJhbmtWYXVsdC5nZXQoZnJvbSk7aWYoIWRlcG9zaXQp
e3Rocm93IG5ldyBFcnJvcihcIk5vIGRlcG9zaXQgYmVmb3JlLlwiKTt9IGlmKGJrX2hlaWdodC5s
dChkZXBvc2l0LmV4cGlyeUhlaWdodCkpe3Rocm93IG5ldyBFcnJvcihcIkNhbiBub3QgdGFrZW91
dCBiZWZvcmUgZXhwaXJ5SGVpZ2h0LlwiKTt9IGlmKGFtb3VudC5ndChkZXBvc2l0LmJhbGFuY2Up
KXt0aHJvdyBuZXcgRXJyb3IoXCJJbnN1ZmZpY2llbnQgYmFsYW5jZS5cIik7fSB2YXIgcmVzdWx0
PUJsb2NrY2hhaW4udHJhbnNmZXIoZnJvbSxhbW91bnQpO2lmKCFyZXN1bHQpe3Rocm93IG5ldyBF
cnJvcihcIkNhbiBub3QgdGFrZW91dCBiZWZvcmUgZXhwaXJ5SGVpZ2h0LlwiKTt9IGlmKGJrX2hl
bW91bnQudG9TdHJpbmcoKX19KTtkZXBvc2l0LmJhbGFuY2U9ZGVwb3NpdC5iYWxhbmNlLnN1Yihh
bW91bnQpO3RoaXMuYmFua1ZhdWx0LnB1dChmcm9tLGRlcG9zaXQpO30sYmFsYW5jZU9mOmZ1bmN0
aW9uKCl7dmFyIGZyb209QmxvY2tjaGFpbi50cmFuc2FjdGlvbi5mcm9tO3JldHVybiB0aGlzLmJh
bmtWYXVsdC5nZXQoZnJvbSk7fSx2ZXJpZnlBZGRyZXNzOmZ1bmN0aW9uKGFkZHJlc3Mpe3ZhciBy
ZXN1bHQ9QmxvY2tjaGFpbi52ZXJpZnlBZGRyZXNzKGFkZHJlc3MpO3JldHVybnt2YWxpZDpyZXN1
bHQ9PTA/ZmFsc2U6dHJ1ZX07fX07bW9kdWxlLmV4cG9ydHM9QmFua1ZhdWx0Q29udHJhY3Q7Iiwi
QXJncyI6IiJ9","gas_price":"1000000","gas_limit":"2000000","contract_address"
:"n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM","status":2,"gas_used":"","execute_err
or":"","execute_result":""}}

如上所示，部署合约的交易的状态变成了2，表示交易待定，当前交易还没有被打包，合约还没部署成功，过段时间再查询。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/user/getTransactionReceipt -d '{"hash":"4eb
e3c6f2b9ad7bf2459140c594633937fd11e3159dfa04f735ba5840a8f1037"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:30:03 GMT
Transfer-Encoding: chunked

{"result":{"hash":"4ebe3c6f2b9ad7bf2459140c594633937fd11e3159dfa04f735ba5840
a8f1037","chainId":100,"from":"n1H4MYms9F55ehcvygwWE71J8tJC4CRr2so","to":"n1
H4MYms9F55ehcvygwWE71J8tJC4CRr2so","value":"0","nonce":"1","timestamp":"1528
271021","type":"deploy","data":"eyJTb3VyY2VUeXBlIjoianMiLCJTb3VyY2UiOiJcInVz
ZSBzdHJpY3RcIjt2YXIgRGVwb3NpdGVDb250ZW50PWZ1bmN0aW9uKHRleHQpe2lmKHRleHQpe3Zh
ciBvPUpTT04ucGFyc2UodGV4dCk7dGhpcy5iYWxhbmNlPW5ldyBCaWdOdW1iZXIoby5iYWxhbmNl
KTt0aGlzLmV4cGlyeUhlaWdodD1uZXcgQmlnTnVtYmVyKG8uZXhwaXJ5SGVpZ2h0KTt9ZWxzZXt0
aGlzLmJhbGFuY2U9bmV3IEJpZ051bWJlcigwKTt0aGlzLmV4cGlyeUhlaWdodD1uZXcgQmlnTnVt
YmVyKDApO319O319O0RlcG9zaXRlQ29udGVudC5wcm90b3R5cGU9e3RvU3RyaW5nOmZ1bmN0aW9uKCl7
cmV0dXJuIEpTT04uc3RyaW5naWZ5KHRoaXMpO319O3ZhciBCYW5rVmF1bHRDb250cmFjdD1mdW5j
dGlvbigpe0xvY2FsQ29udHJhY3RTdG9yYWdlLmRlZmluZU1hcFByb3BlcnR5KHRoaXMsXCJiYW5r
VmF1bHRcIix7cGFyc2U6ZnVuY3Rpb24odGV4dCl7cmV0dXJuIG5ldyBEZXBvc2l0ZUNvbnRlbnQo
dGV4dCk7fSxzdHJpbmdpZnk6ZnVuY3Rpb24obyl7cmV0dXJuIG8udG9TdHJpbmcoKTt9fSk7fTtC
YW5rVmF1bHRDb250cmFjdC5wcm90b3R5cGU9e2luaXQ6ZnVuY3Rpb24oKXt9LHNhdmU6ZnVuY3Rp
b24oaGVpZ2h0KXt2YXIgZnJvbT1CbG9ja2NoYWluLnRyYW5zYWN0aW9uLmZyb207dmFyIHZhbHVl
PUJsb2NrY2hhaW4udHJhbnNhY3Rpb24udmFsdWU7dmFyIGJrX2hlaWdodD1uZXcgQmlnTnVtYmVy
```

```
KEJsb2NrY2hhaW4uYmxvY2suaGVpZ2h0KTt2YXIgb3JpZ19kZXBvc2l0PXRoaXMuYmFua1ZhdWx0
LmdldChmcm9tKTtpZihvcmlnX2RlcG9zaXQpe3ZhbHVlPXZhbHVlLnBsdXMob3JpZ19kZXBvc2l0
LmJhbGFuY2UpO30gdmFyIGRlcG9zaXQ9bmV3IERlcG9zaXRlQ29udGVudCgpO2RlcG9zaXQuYmFs
YW5jZT12YWx1ZTtkZXBvc2l0LmV4cGlyeUhlaWdodD1ia19oZWlnaHQucGx1cyhoZWlnaHQpO3Ro
aXMuYmFua1ZhdWx0LnB1dChmcm9tLGRlcG9zaXQpO30sdGFrZW91dDpmdW5jdGlvbih2YWx1ZSl7
dmFyIGZyb209QmxvY2tjaGFpbi50cmFjdGlvbi5mcm9tO3ZhciBia19oZWlnaHQ9bmV3IEJp
Z051bWJlcihCbG9ja2NoYWluLmJsb2NrLmhlaWdodCk7dmFyIGFtb3VudD1uZXcgQmlnTnVtYmVy
KHZhbHVlKTt2YXIgZGVwb3NpdD10aGlzLmJhbmtWYXVsdC5nZXQoZnJvbSk7aWYoIWRlcG9zaXQp
e3Rocm93IG5ldyBFcnJvcihcIk5vIGRlcG9zaXQgYmVmb3JlLlwiKTt9IGlmKGJrX2hlaWdodC5s
dChkZXBvc2l0LmV4cGlyeUhlaWdodCkpe3Rocm93IG5ldyBFcnJvcihcIkNhbiBub3QgdGFrZW91
dCBiZWZvcmUgZXhwaXJ5SGVpZ2h0LlwiKTt9IGlmKGFtb3VudC5ndChkZXBvc2l0LmJhbGFuY2Up
KXt0aHJvdyBuZXcgRXJyb3IoXCJJbnN1ZmZpY2llbnQgYmFsYW5jZS5cIik7fSB2YXIgcmVzdWx0
PUJsb2NrY2hhaW4udHJhbnNmZXIoZnJvbSxhbW91bnQpO2lmKCFyZXN1bHQpe3Rocm93IG5ldyBF
cnJvcihcInRyYW5zZmVyIGZhaWxlZC5cIik7fSBFdmVudC5UcmlnZ2VyKFwiQmFua1ZhdWx0XCIs
e1RyYW5zZmVyOntmcm9tOkJsb2NrY2hhaW4udHJhbnNhY3Rpb24udG8sdG86ZnJvbSx2YWx1ZTph
bW91bnQudG9TdHJpbmcoKX19KTtkZXBvc2l0LmJhbGFuY2U9ZGVwb3NpdC5iYWxhbmNlLnN1Yihh
bW91bnQpO3RoaXMuYmFua1ZhdWx0LnB1dChmcm9tLGRlcG9zaXQpO30sYmFsYW5jZU9mOmZ1bmN0
aW9uKCl7dmFyIGZyb209QmxvY2tjaGFpbi50cmFuc2FjdGlvbi5mcm9tO3JldHVybiB0aGlzLmJh
bmtWYXVsdC5nZXQoZnJvbSk7fSx2ZXJpZnlBZGRyZXNzOmZ1bmN0aW9uKGFkZHJlc3Mpe3ZhciBy
ZXN1bHQ9QmxvY2tjaGFpbi52ZXJpZnlBZGRyZXNzKGFkZHJlc3MpO3JldHVybnt2YWxpZDpyZXN1
bHQ9PTA/ZmFsc2U6dHJ1ZX07fX07bW9kdWxlLmV4cG9ydHM9QmFua1ZhdWx0Q29udHJhY3Q7Iiwi
QXJncyI6IiJ9","gas_price":"1000000","gas_limit":"2000000","contract_address"
:"n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM","status":1,"gas_used":"22016","execut
e_error":"","execute_result":"\"\""}}
```


如上所示，部署合约的交易的状态变成了1，表示交易成功，合约部署成功了。

4、调用智能合约

在Nebulas中调用智能合约的方式也很简单，同样是通过发送交易来调用智能合约。

4.1 调用智能合约的`save`方法

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Accept: application/json' -X POS
T http://localhost:8685/v1/admin/transactionWithPassphrase -H 'Content-Type:
 application/json' -d '{"transaction":{"from":"n1LkDi2gGMqPrjYcczUiweyP4RxTB
6Go1qS","to":"n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM", "value":"100","nonce":1,
"gasPrice":"1000000","gasLimit":"2000000","contract":{"function":"save","arg
s":"[0]"}}, "passphrase": "passphrase"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:44:47 GMT
Content-Length: 110

{"result":{"txhash":"6defe45b1e1ee8c505758864df0666000fc53ca24b02a2c94dbcb5f
```

89b6935b8","contract_address":""}}

- from: 用户的账户地址
- to: 智能合约地址
- value: 调用save()方法时想要存入智能合约代币数量
- nonce: 比创建者当前的 `nonce` 多1，可以通过 `GetAccountState` 获取创建前当前 `nonce`
- gasPrice：部署智能合约用到的gasPrice，可以通过 `GetGasPrice` 获取，或者使用默认值: `"1000000"` ;
- gasLimit: 部署合约的gasLimit，通过 `EstimateGas` 可以获取部署合约的gas消耗，不能使用默认值，也可以设置一个较大值，执行时以实际使用计算。
- contract: 合约信息，部署合约时传入的参数
  - `function` : 调用合约方法
  - `args` : 合约方法参数，无参数为空字符串，有参数时为JSON数组

4.2 验证调用合约 `save` 方法的交易

由于执行合约方法本质是提交一个交易，所以我们可以通过验证交易是否提交成功来判断合约方法是否执行成功。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/user/getTransactionReceipt -d '{"hash":"6de
fe45b1e1ee8c505758864df0666000fc53ca24b02a2c94dbcb5f89b6935b8"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:49:16 GMT
Content-Length: 446

{"result":{"hash":"6defe45b1e1ee8c505758864df0666000fc53ca24b02a2c94dbcb5f89
b6935b8","chainId":100,"from":"n1LkDi2gGMqPrjYcczUiweyP4RxTB6Go1qS","to":"n1
rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM","value":"100","nonce":"1","timestamp":"15
28271927","type":"call","data":"eyJGdW5jdGlvbiI6InNhdmUiLCJBcmdzIjoiWzBdIn0=
","gas_price":"1000000","gas_limit":"2000000","contract_address":"","status"
:1,"gas_used":"20361","execute_error":"","execute_result":"\"\""}}
```

如上所示，交易状态变为了1，表示执行save方法成功了。

4.3 调用智能合约的 `takeout` 方法

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Accept: application/json' -X POS
T http://localhost:8685/v1/admin/transactionWithPassphrase -H 'Content-Type:
 application/json' -d '{"transaction":{"from":"n1LkDi2gGMqPrjYcczUiweyP4RxTB
```

```
6Go1qS","to":"n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM", "value":"0","nonce":2,"g
asPrice":"1000000","gasLimit":"2000000","contract":{"function":"takeout","ar
gs":"[50]"}}, "passphrase": "passphrase"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:52:39 GMT
Content-Length: 110

{"result":{"txhash":"d2c1c4f48839483509e930edbaf0b4b24334ed24e0cf54286c2742b
c4c061863","contract_address":""}}
```

4.4 验证调用合约 `takeout` 方法的交易

在上面save方法的执行中，我们在合约n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM中存了100NAS。此时，我们执行takeout函数，从中取出50NAS。合约里应该还有50NAS。我们检测下合约账户的余额来验证takeout方法执行是否成功。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/user/accountstate -d '{"address":"n1rVLTRxQ
EXscTgThmbTnn2NqdWFEKwpYUM"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:54:37 GMT
Content-Length: 49

{"result":{"balance":"50","nonce":"0","type":88}}
```

结果和预期相符。

5、查询智能合约数据

在智能合约中，我们有一些方法并不更改合约的状态，这些方法被设计来帮助我们获取合约数据，它们是只读的。在星云链上，我们在API Module中为用户提供了 `Call` 接口来帮助用户来执行这些只读的方法，使用 `Call` 接口不会发送交易，也就无需支付上链手续费。

我们调用合约 `n1rVLTRxQEXscTgThmbTnn2NqdWFEKwpYUM` 中的 `balanceOf` 方法来查询该合约的余额。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Accept: application/json' -X POS
T http://localhost:8685/v1/user/call -H 'Content-Type: application/json' -d
'{"from":"n1LkDi2gGMqPrjYcczUiweyP4RxTB6Go1qS","to":"n1rVLTRxQEXscTgThmbTnn2
```

```
NqdWFEKwpYUM","value":"0","nonce":3,"gasPrice":"1000000","gasLimit":"2000000
","contract":{"function":"balanceOf","args":""}}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 11:58:51 GMT
Content-Length: 108

{"result":{"result":"{\"balance\":\"50\",\"expiryHeight\":\"24\"}","execute_
err":"","estimate_gas":"20209"}}
```

本文参考：星云链Nebulas官方Github

下一篇预告

星云链Nebulas——4.智能合约存储区