智能合约编程（一）感受智能合约代码

产品名称	智能合约编程（一）感受智能合约代码
公司名称	深圳漫云网络科技有限公司
价格	.00/件
规格参数	漫云科技:app开发 漫云网络:源码交付 app定制:售后一对一
公司地址	深圳市南山区粤海街道麻岭社区科研路9号比克科技大厦1701D
联系电话	18638029017 18638029017

产品详情

这个系列是笔者自己学习***时的笔记，基于**新版的Solidity技术文档（0.8.7）。

参考：

Solidity-Solidity0.8.7documentation

docs.soliditylang.org/en/latest/index.html

Solidity*新(0.8.0)中文文档

learnblockchain.cn/docs/solidity/index.html

另外还有许多其他medium的技术博客以及相关书籍

《深入以太坊智能合约**》、《精通以太坊》、《智能合约安全分析和审计指南》

感受代码

铸币代码

pragmasolidity^0.4;

contractCoin{

//setthe"address"typevariableminter

```
addresspublicminter;
/*convert"address"(forstoringaddressorkey)
tothetypeof"uint"whichisassubscripofobjectbalances*/
mapping(address=>uint)publicbalances;
//setaneventsoastobeseenpublicly
eventSent(addressfrom,addressto,uintamount);
//constructoronlyrunoncewhencreatingcontract,unabletoinvoke
//"msg"istheaddressofcreator."msg.sender"is
constructor()public{
minter=msg.sender;
}
//铸币
//canonlybecalledbycreator
functionmint(addressreceiver,uintamount)public{
require(msg.sender==minter);
balances[receiver]+=amount;
}
//转账
functionsend(addressreceiver,uintamount)public{
require(balances[msg.sender]>=amount);
balances[msg.sender]-=amount;
balances[receiver]+=amount;
emitSent(msg.sender,receiver,amount);
}


}
```

源文件结构

版本标识

pragma

版本标识，是pragmaticinformation的简称，用于启动编译器检查，避免因为solidity更新后造成的不兼容和语法变动的错误。只对本文件有效，如果导入其他文件，版本标识不会被导入，而是采用工作的文件自身的版本标识

pragmasolidity^0.5.2;

这里^表示从0.5.2到0.6（不含）的版本

导入其他文件

语法如下

import"filename";

这种导入方式会把导入文件的所有全局符号都导入到工作文件的全局作用域，会污染命名空间，不建议这么使用。

import*assymbolNamefrom"filename";

//等价于

import"filename"assymbolName;

这样所有的全局符号都以symbolName.symbol的格式提供。

我们还可以设置别名，别名和重定义的符号名，都可以表示导入的文件里的全局符号。

import{symbol1asalias,symbol2}from"filename";

路径

路径的形式和Linux下的完全一致，但是要避免使用..。我们可以引入指定路径的文件，如import"./filename"assimbolName，是当前目录下的文件。引用的文件除了本地文件，git,http发现的其他资源。

实际solc编译器使用的时候可以指定路径的重映射，编译器可以从重映射的位置读取文件。尤其是使用网络文件的时候例如，可以使github.com/ethereum/dapp-bin/library会被重映射到/usr/local/dapp-bin/library,格式如下。

solcgithub.com/ethereum/dapp-bin/=/usr/local/dapp-bin/source.sol

更具体地会在solc编译器地部分说明。而truffle框架和remix就相对智能，可以通过网络获取文件。

注释

单行注释//,多行注释/*.....*/

一种natspec注释，他是用///或者/**.....*/，它里面可以使用Doxygen样式来给出相关地信息。

Doxygen样式地注释可以使特殊地注释形式变得可识别，方便读取和自动提取信息。主要有

```
//SPDX-License-Identifier:GPL-3.0

pragmasolidity>=0.4.21<0.9.0;


/**@titleShapecalculator.

*@file（文件名）

*@authorJohnDoe

*@version1.0（版本）

*@details（细节）

*@date（年-月-日）

*@license（版权协议）

*@brief（类的简单概述）

*@sectionLICENSE*/（这一段的主要内容）

*@paramDescriptionofmethod'sorfunction'sinputparameter（形式参数说明）

*@returnDescriptionofthereturnvalue（返回说明）

*@retval（返回值说明）

*@attention（注意）

*@warning（警告）

*@var（变量声明）

*@bug（代码缺陷）

*@exception（异常）

contractShapeCalculator{

///@devCalculatesarectangle'ssurfaceandperimeter.

///@paramwWidthoftherectangle.

///@paramhHeightoftherectangle.
```

///@returnsThecalculatedsurface.

///@returnpThecalculatedperimeter.

functionrectangle(uintw,uinth)publicpurereturns(uints,uintp){

s=w*h;

p=2*(w+h);

}

}

合约的结构

状态变量

状态变量是**地存储在合约存储中的值，它具有数据的类型，也有可见性的属性。在函数外的都是store状态变量。

pragmasolidity>=0.4.0<0.9.0;

contractTinyStorage{

uintstoredXlbData;//状态变量

//...

}

函数

函数是代码的可执行单元。函数通常在合约内部定义，但也可以在合约外定义。

//SPDX-License-Identifier:GPL-3.0

pragmasolidity>0.7.0<0.9.0;

contractTinyAuction{

functionMybid()publicpayable{//定义函数

//...

}

```
}

//Helperfunctiondefinedoutsideofacontract
functionhelper(uintx)purereturns(uint){
returnx*2;
}
```

函数调用可发生在合约内部或外部，且函数对其他合约有不同程度的可见性（可见性和getter函数）。

函数可以接受参数和返回值。

函数修饰

函数修饰符用来修饰函数，比如添加函数执行前必须的先决条件，

```
contractOwner{
modifieronlyOwner{
require(msg.sender==owner);
_;
}
modifiercosts(uintprice){
if(msg.value>=price){
_;
}
}
}
```

函数体会插入在修饰函数的下划线_的位置。所以只有当修饰条件满足之后才能执行这个函数，否则报错。

注意下面的用法。实际上常常会被继承，作为模块化处理。

```
//SPDX-License-Identifier:MIT
pragmasolidity^0.8;
```

```
contractTest{
uintpublica;
uintpublicb;
functionset(uint_a,uint_b)public{
a=_a;
b=_b;
}
modifierFunc(uint_a)
{
require(a>_a,"error:aissosmall.");
_;
}
functionf(uint_a)publicviewFunc(_a)returns(uint){
return_a;
}
}
```

事件

事件是能方便地调用以太坊虚拟机日志功能的接口，分为设置事件和触发事件

```
pragmasolidity>=0.4.21<0.9.0;
contractTinyAuction{
eventHighestBidIncreased(addressbidder,uintamount);//事件

functionbid()publicpayable{
//...
emitHighestBidIncreased(msg.sender,msg.value);//触发事件
```

```
}
}
```

结构体

```
pragmasolidity>=0.4.0<0.9.0;

contractTinyBallot{
structVoter{//结构体
uintweight;
boolvoted;
addressdelegate;
uintvote;
}
}
```

枚举类型

```
pragmasolidity>=0.4.0<0.9.0;

contractUpchain{
enumState{Created,Locked,InValid}//枚举
```