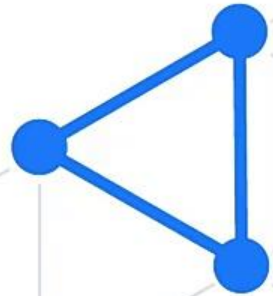
蚂蚁集团 ANT GROUP | 蚂蚁链 ANTCHAIN

Solidity智能合约结构

- 蚂蚁链《区块链系统开发与应用》A认证系列课程

01 Solidity智能合约结构简介



Solidity智能合约结构简介

```
pragma solidity >=0.4.0 <0.6.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

- 从这个简单的例子中，我们可以大致的看到一个 Solidity 智能合约的基本结构：
  - 首先，需要使用 pragma 关键字表明该合约所适用的 solidity 版本；
  - 然后，使用 contract 关键字定义一个合约；
  - 使用大括号 {} 将合约的内容包裹起来；
  - 示例中，合约的内容包含三个部分：
    - uint 类型的状态变量；
    - 带有一个 uint 类型入参的无返回值的 set 函数；
    - 无入参的带有一个 uint 类型返回值的 get 函数；

Solidity智能合约结构简介

其他内容

- 除此之外，Solidity 智能合约还可以包含很多其他类型的声明，我们将其总结如下：
  - 版本标注
  - 状态变量
  - 函数
  - 事件
  - 注释
  - 结构类型
  - 枚举类型
  - 内建合约

Usage

As a node module:

```
const semver = require('semver')

semver.valid('1.2.3') // '1.2.3'
semver.valid('a.b.c') // null
semver.clean('  =v1.2.3   ') // '1.2.3'
semver.satisfies('1.2.3', '1.x || >=2.5.0 || 5.0.0 - 7.2.3') // true
semver.gt('1.2.3', '9.8.7') // false
semver.lt('1.2.3', '9.8.7') // true
semver.minVersion('>=1.0.0') // '1.0.0'
semver.valid(semver.coerce('v2')) // '2.0.0'
semver.valid(semver.coerce('42.6.7.9.3-alpha')) // '42.6.7'
```

As a command-line utility:

```
$ semver -h

A JavaScript implementation of the https://semver.org/ specification
Copyright Isaac Z. Schlueter

Usage: semver [options] <version> [<version> [...]]
Prints valid versions sorted by SemVer precedence

Options:
```


保存

08:16

Solidity智能合约结构简介

版本标注补充说明

- 我们不仅可以使用^符号来规定版本号，还可以使用更为复杂的规则来指定版本号，Solidity 指定版本号的表达式遵循 NPM版本语义；
  - ^
  - <, >, <=, >=, =
  - 多数情况下，我们使用^就可以
- 蚂蚁链 Solidity 目前支持 0.4.24 和0.6.4这两个版本。

BaaS 平台 | BaaS 平台 | semver | npm Docs

openchain.cloud.alipay.com/contract-edit/template/create-contract?projectId=100975&unionId=2647&bizid=a00e36c5&openchain=1

蚂蚁集团 ANT GROUP | 金融科技 | 管理控制台 | 产品与服务

a00e36c5 - 开放联盟链...

编译

Solidity | 0.4.23

- Demo001
  - Demo002.sol
  - Demo003.sol
  - intDemo.sol
  - testStorage.sol

保存

intDemo.sol | Demo003.sol | testStorage.sol

```
1  pragma solidity ^0.4.20;
2
3  contract TestStorage {
4      uint storedData;
5
6      function set(uint x) public {
7          storedData = x;
8      }
9
10     function get() public view returns (uint) {
11         return storedData;
12     }
13 }
```

编译

编译详情 | 合约分析 | 调试详情 | 交易详情

```
        ^----^
intDemo.sol:4:5: The shadowed declaration is here:
    int16 a = 128;
    ^-----------^

intDemo.sol:35:9: Warning: This declaration shadows an existing declaration.
        uint b = 3;
        ^----^
intDemo.sol:6:5: The shadowed declaration is here:
    uint16 b;
    ^------^
```

08:16

Solidity智能合约结构简介

状态变量和函数补充说明

- 状态变量
  - 永久的存储在合约存储中的值
  - 状态变量的可见性默认为 internal
- 函数
  - 合约代码的可执行单元
  - 函数调用可以发生在内部，也可以发生在外部
  - 函数的可见性可以用关键字来控制
    - Public
    - External
    - Internal
    - private

BaaS 平台 | BaaS 平台 | semver | npm Docs

openchain.cloud.alipay.com/contract-edit/template/create-contract?projectId=100975&unionId=2647&bizid=a00e36c5&openchain=1

蚂蚁集团 ANT GROUP | 金融科技 | 管理控制台 | 产品与服务

a00e36c5 - 开放联盟链... | 编译

Solidity | 0.4.23

- Demo001
  - Demo002.sol
  - Demo003.sol
  - intDemo.sol
  - testStorage.sol

保存

intDemo.sol | Demo003.sol | testStorage.sol

```
1  pragma solidity ^0.4.20;
2
3  contract TestStorage {
4      uint storedData;
5      uint public a;
6
7      function set(uint x) public {
8          storedData = x;
9      }
10
11     function get() public view returns (uint) {
12         return storedData;
13     }
14 }
```

编译

编译详情 | 合约分析 | 调试详情 | 交易详情

```
        ^----^
intDemo.sol:4:5: The shadowed declaration is here:
    int16 a = 128;
    ^-----------^

intDemo.sol:35:9: Warning: This declaration shadows an existing declaration.
        uint b = 3;
        ^----^
intDemo.sol:6:5: The shadowed declaration is here:
    uint16 b;
    ^------^
```

08:16

Solidity智能合约结构简介

状态变量和函数补充说明

- 状态变量
  - 永久的存储在合约存储中的值
  - 状态变量的可见性默认为 internal
- 函数
  - 合约代码的可执行单元
  - 函数调用可以发生在内部，也可以发生在外部
  - 函数的可见性可以用关键字来控制
    - Public
    - External
    - Internal
    - private

Solidity智能合约结构简介

事件

- 事件
  - 事件是可以方便的调用蚂蚁链虚拟机日志功能的接口
  - 使用 event 关键字（用法上类似与 function关键字）来定义事件
  - 使用 emit 关键字在函数中触发事件

```
pragma solidity ^0.5.0;

contract Counter {
    uint256 public count = 0;

    event Increment(Identity who);

    function increment() public {
        emit Increment(msg.sender);
        count += 1;
    }
}
```

Solidity智能合约结构简介

```
function getResult() public view returns(uint){

    // 这是单行注释

    /*
     * 这是多行注释
     * 这是多行注释
     * 这是多行注释
     */
}
```

- Solidity 注释
  - 单行注释
    - 使用“//”开头
  - 多行注释
    - 多行注释包裹在“/*”与“*/”中间
  - 编辑器会忽略注释内容
  - 建议在实际开发项目过程中多加注释，方便以后调试

BaaS 平台 | BaaS 平台 | semver | npm Docs

openchain.cloud.alipay.com/contract-edit/template/create-contract?projectId=100975&unionId=2647&bizid=a00e36c5&openchain=1

蚂蚁集团 ANT GROUP | 金融科技 | 管理控制台 | 产品与服务

a00e36c5 - 开放联盟链...

编译

Solidity | 0.4.23

- Demo001
  - Demo002.sol
  - Demo003.sol
  - intDemo.sol
  - testStorage.sol

保存

intDemo.sol | Demo003.sol | testStorage.sol

```
1  pragma solidity ^0.4.20;
2
3  contract TestStorage {
4      uint storedData;
5      uint public a;
6
7      function set(uint x) public {
8          storedData = x;
9      }
10
11     function get() public view returns (uint) {
12         return storedData;
13     }
14 }
```

编译

编译详情 | 合约分析 | 调试详情 | 交易详情

```
        ^----^
intDemo.sol:4:5: The shadowed declaration is here:
    int16 a = 128;
    ^-----------^

intDemo.sol:35:9: Warning: This declaration shadows an existing declaration.
        uint b = 3;
        ^----^
intDemo.sol:6:5: The shadowed declaration is here:
    uint16 b;
    ^------^
```

08:16

BaaS 平台 | BaaS 平台 | semver | npm Docs

openchain.cloud.alipay.com/contract-edit/template/create-contract?projectId=100975&unionId=2647&bizid=a00e36c5&openchain=1

蚂蚁集团 ANT GROUP | 金融科技 | 管理控制台 | 产品与服务

a00e36c5 - 开放联盟链... | 编译

Solidity | 0.4.23

- Demo001
  - Demo002.sol
  - Demo003.sol
  - intDemo.sol
  - testStorage.sol

保存

intDemo.sol | Demo003.sol | testStorage.sol

```
11      这是一个多行注释
12      这是一个多行注释
13      这是一个多行注释
14      这是一个多行注释
15      这是一个多行注释
16      这是一个多行注释
17      */
18
19      function set(uint x) public {
20          storedData = x;
21      }
22
23      function get() public view returns (uint) {
24          return storedData;
25      }
26  }
```

编译

编译详情 | 合约分析 | 调试详情 | 交易详情

```
        ^----^
intDemo.sol:4:5: The shadowed declaration is here:
    int16 a = 128;
    ^-----------^

intDemo.sol:35:9: Warning: This declaration shadows an existing declaration.
        uint b = 3;
        ^----^
intDemo.sol:6:5: The shadowed declaration is here:
    uint16 b;
    ^------^
```

08:16

Solidity智能合约结构简介

结构类型

- 结构类型
  - Solidity支持通过结构体来定义新的类型
  - 使用struct关键字来定义结构体
  - 左侧的小例子就展示了一个新的结构体，名称为“Voter”，包含三个结构：
    - “uint”类型的 weight
    - “bool”类型的 voted
    - “uint” 类型的vote

```
function structDefinedFunction() public{
    // 定义一个结构体
    struct Voter {
        uint weight;
        bool voted;
        uint vote;
    }
}
```

BaaS 平台 | BaaS 平台 | semver | npm Docs

openchain.cloud.alipay.com/contract-edit/template/create-contract?projectId=100975&unionId=2647&bizid=a00e36c5&openchain=1

蚂蚁集团 ANT GROUP | 金融科技 | 管理控制台 | 产品与服务

a00e36c5 - 开放联盟链... | 编译

Solidity | 0.4.23

- Demo001
  - Demo002.sol
  - Demo003.sol
  - intDemo.sol
  - testStorage.sol

保存

intDemo.sol | Demo003.sol | testStorage.sol

```
1  pragma solidity ^0.4.20;
2
3  contract TestStorage {
4      uint storedData;
5      uint public a;
6
7      // 这是一个单行注释
8
9      /*
10     这是一个多行注释
11     这是一个多行注释
12     这是一个多行注释
13     这是一个多行注释
14     这是一个多行注释
15     这是一个多行注释
16     这是一个多行注释
17     */
18
```

编译

编译详情 | 合约分析 | 调试详情 | 交易详情

```
        ^----^
intDemo.sol:4:5: The shadowed declaration is here:
    int16 a = 128;
    ^-----------^

intDemo.sol:35:9: Warning: This declaration shadows an existing declaration.
        uint b = 3;
        ^----^
intDemo.sol:6:5: The shadowed declaration is here:
    uint16 b;
    ^------^
```

08:16

Solidity智能合约结构简介

```
pragma solidity ^0.5.0;

contract test {

   enum FreshJuiceSize{ SMALL, MEDIUM, LARGE }
   FreshJuiceSize choice;

   FreshJuiceSize constant defaultChoice = FreshJuiceSize.MEDIUM;

   function setLarge() public {
      choice = FreshJuiceSize.LARGE;
   }

   function getChoice() public view returns (FreshJuiceSize) {
      return choice;
   }
}
```

- 枚举类型
  - 枚举将一个变量的取值限制为几个预先定义的值中的一个
  - 使用 enum 关键字来定义枚举类型
  - 如左侧例子中，我们定义了一个枚举类型“FreshJuiceSize”，它的取值只有三个
    - SMALL
    - MEDIUM
    - LARGE

Solidity智能合约结构简介

内建合约

- 在合约内部，我们可以使用 new 关键字来创建另一个合约，如右侧例子所示；
- 在一个智能合约文件内部，我们声明了两个智能合约：Test1 和 Test2；
- 在 Test2 合约内部，我们使用 new 关键字创建了 Test1 合约；
- 注意：蚂蚁链 Solidity 语言不支持使用这种方式来内建合约，这是和原生 Solidity 语言的不同点之一；

```
pragma solidity ^0.4.24;

contract Test1 {
   uint test;
   bytes32 name;
}

contract Test2 {
   function createTest1(bytes32 name)
      public
      returns (Test1 test1)
   {
      // 创建一个新的 Test1 合约
      return new Test1(name);
   }
}
```

Solidity智能合约结构简介

总结

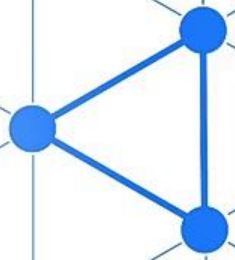
- 在 Solidity 中，合约类似于面向对象中的类（如 Java 语言中的 Class）；
- Solidity 智能合约中可以包含诸如：版本声明、状态变量、函数 、注释、事件、结构类型和枚举类型等的声明；
- 除了上面所讲的几个内容之外，Solidity 还具有继承、库、接口等高级语言特性；
- 原生 Solidity 语言支持在合约内创建合约，类似于面向对象语言中的内部类，但是蚂蚁链 Solidity 语言不支持内建合约；
- 初级课程不会涉及到事件、继承、库、接口等高级内容，其他内容会在后续教程中详细说明；

随堂练习

练习题

- 使用 Cloud IDE 编译和部署课件中的小例子，并给每行代码做注释；
- 尝试在上述小例子中声明枚举类型和结构体类型；
- 理解 Solidity 智能合约的基本结构。

谢 谢



蚂蚁集团 ANT GROUP | 蚂蚁链 ANTCHAIN