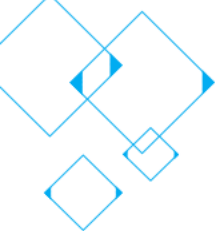


# 第9章 智能合约抽奖



重航区块链

CONTENTS

# 目录

第四章 游戏代币-合约设计

第五章 游戏代币-合约编写

第六章 游戏代币-合约测试





重航区块链

CONTENTS

# 章节

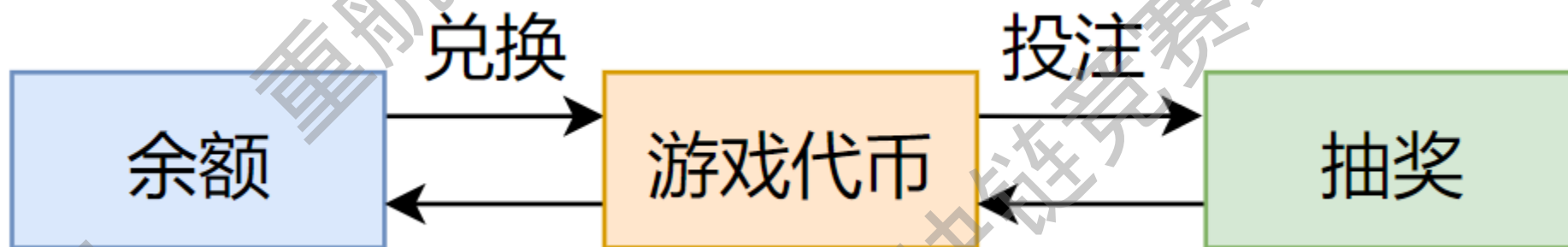
## 游戏代币-合约设计

### 4.1 合约设计



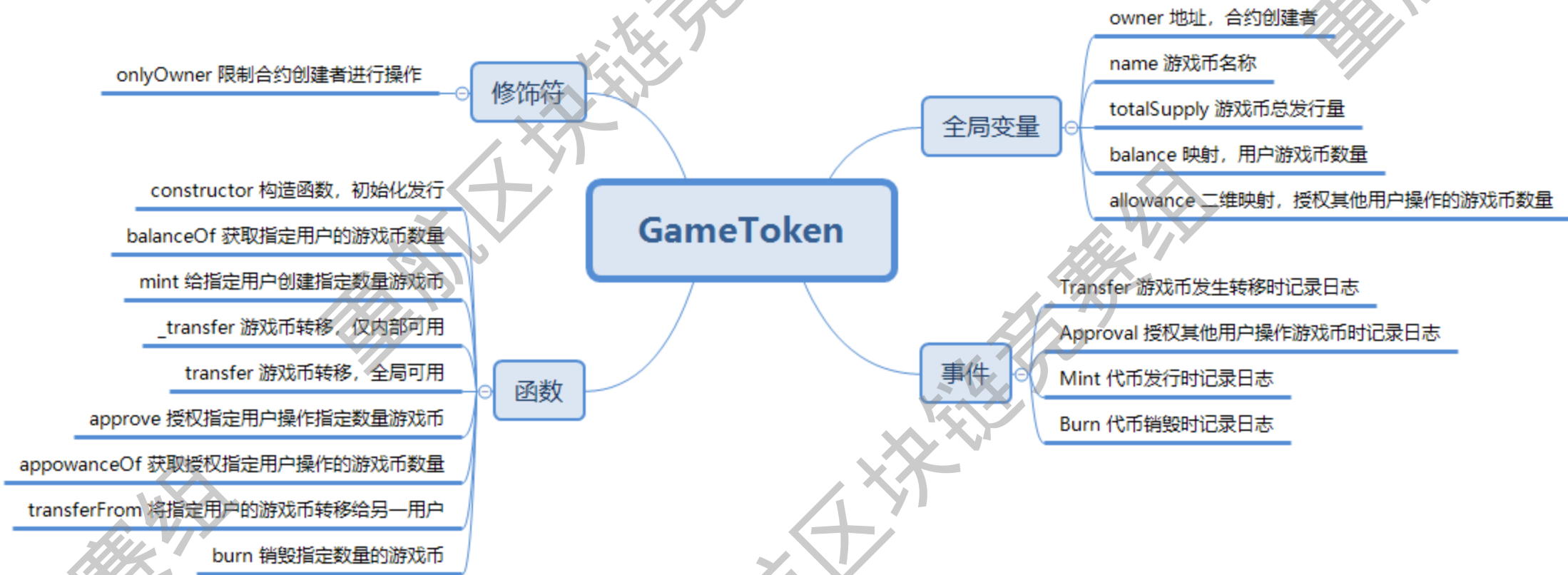
# 合约设计

- 在“简单抽奖”合约以及“拍卖竞价”合约中，都是使用**balance**映射类型、**deposit**和**withdraw**函数来模拟用户的余额、存款和取款操作，本章节就开发一个游戏代币合约，在下一章节中使用游戏代币来进抽奖操作。



# 合约设计

- 游戏代币合约中包括以下全局变量，事件和函数





重航区块链

CONTENTS

# 章节

游戏代币-合约编写

5.1 合约编写



# 合约编写

- 初始化游戏代币合约，使用solidity 0.6.10版本，合约名为GameToken

```
// 声明solidity版本为0.6.10
pragma solidity 0.6.10;

// 定义GameToken合约
contract GameToken {

}
```

# 合约编写

- 定义全局变量，包括合约创建者，游戏币名称，总发行量，用户余额映射，授权金额二维映射

```
// 合约创建者
address owner;

// 游戏币名称
string public name;

// 总发行量
uint public totalSupply;

// 用户余额
mapping (address => uint) balance;

// 授权金额
mapping (address => mapping (address => uint256)) allowance;
```



# 合约编写

- 定义事件，在代币转移，代币授权，代币发行，代币销毁时记录日志

```
// 事件日志, 代币转移
event Transfer(address from, address to, uint value);

// 事件日志, 授权
event Approval(address _owner, address _spender, uint256 _value);

// 事件日志, 代币发行
event Mint(address from, uint value);

// 事件日志, 代币销毁
event Burn(address from, uint value);
```

# 合约编写

- 定义修饰符，限制只有合约创建者进行操作

```
modifier onlyOwner() {  
    require(msg.sender == owner);  
    _;  
}
```

# 合约编写

- 定义构造函数，功能为初始化初始发行量、游戏币名称、合约创建者

```
// 构造函数, 初始化初始发行量、游戏币名称、合约创建者
constructor(uint initialSupply, string memory tokenName) public {
    // 初始化发行量
    totalSupply = initialSupply;
    // 将初始发行的余额转移给合约创建者
    balance[msg.sender] = totalSupply;
    // 初始化合约名
    name = tokenName;
    // 初始化合约创建者
    owner = msg.sender;
}
```

## 合约编写

- 定义mint函数，功能为给指定用户创建指定数量的游戏代币，增加指定地址余额，总发行量，并记录事件日志

```
// 给指定用户创建指定金额的游戏币
function mint(address who, uint _value) public onlyOwner returns(bool success) {
    // 给指定用户增加余额
    balance[who] += _value;
    // 增加总发行量
    totalSupply += _value;
    // 记录事件日志
    emit Mint(who, _value);
    return true;
}
```

## 合约编写

- 定义 `_transfer` 函数，仅在内部可用，功能为游戏代币转移，减少转出地址的余额，增加转入地址的余额，并记录事件日志

```
// 游戏币转移，仅在内部使用
function _transfer(address _from, address _to, uint _value) private returns(bool
success) {
    // 余额充足才可操作
    require(balance[_from] >= _value, "余额不足");
    // 减少转出地址的余额
    balance[_from] -= _value;
    // 增加转入地址的余额
    balance[_to] += _value;
    // 记录事件日志
    emit Transfer(_from, _to, _value);
    return true;
}
```

# 合约编写

- 定义transfer函数，全局可用，在函数中调用\_transfer内部函数

```
// 将调用者的游戏币转移给_to用户  
function transfer(address _to, uint _value) public returns (bool success) {  
    return _transfer(msg.sender, _to, _value);  
}
```

- 定义approve函数，功能为授权一个用户操作指定数量游戏币

```
// 授权_spender用户操作调用者用户_value数量余额  
function approve(address _spender, uint256 _value) public returns (bool success) {  
    // 授权用户操作  
    allowance[msg.sender][_spender] = _value;  
    // 记录事件日志  
    emit Approval(msg.sender, _spender, _value);  
    return true;  
}
```

# 合约编写

- 定义balanceOf函数，功能为获取指定地址的游戏币数量

```
// 获取指定用户的游戏币余额  
function balanceOf(address who) public view returns(uint) {  
    return balance[who];  
}
```

- 定义allowanceOf函数，功能为获取授权指定地址操作的游戏币数量

```
// 获取owner用户允许spender用户操作的金额  
function allowanceOf(address owner, address spender) public view  
returns(uint) {  
    return allowance[owner][spender];  
}
```

## 合约编写

- 定义transferFrom函数，功能将用户授权的游戏币数量转移给指定地址

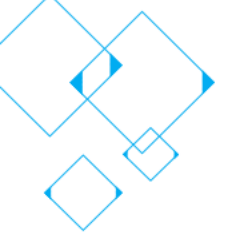
```
// 调用者将_from用户的_value数量余额转移给_to用户
function transferFrom(address _from, address _to, uint256 _value)
public returns (bool success) {
    // 授权余额必须充足
    require(_value <= allowance[_from][msg.sender], "授权金额不足");
    // 减少授权金额数量
    allowance[_from][msg.sender] -= _value;
    // 执行_transfer内部函数
    _transfer(_from, _to, _value);
    return true;
}
```



# 合约编写

- 定义burn函数，功能为销毁指定用户指定数量的游戏币

```
// 销毁指定用户_value数量余额
function burn(address who, uint _value) public onlyOwner returns
(bool success) {
    // 限制销毁数量不能大于余额
    require(balance[who] >= _value);
    // 减少指定地址游戏币数量
    balance[who] -= _value;
    // 减少总发行量
    totalSupply -= _value;
    // 记录事件日志
    emit Burn(who, _value);
    return true;
}
```



重航区块链

CONTENTS

# 章节

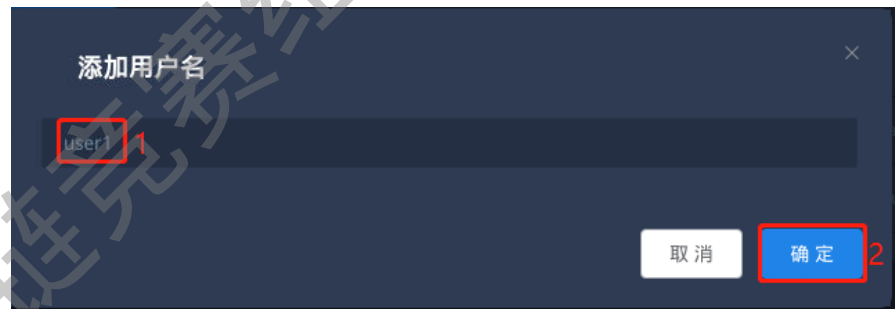
游戏代币-合约测试

6.1 合约测试



# 合约测试

- 合约部署前，创建三个测试账户，分别为admin, user1, user2





# 合约测试

- user1、user2、user3三个用户对应的地址



The screenshot shows a user management interface with a table containing three rows of user data. At the top, there are buttons for '新增用户' (Add User) and '导入私钥' (Import Private Key), along with an information icon. The table has four columns: '地址' (Address), '公钥' (Public Key), '用户' (User), and '操作' (Action). Each row contains a copy icon, a truncated address, a truncated public key, a user name, and '导出' (Export) and '删除' (Delete) buttons.

地址	公钥	用户	操作
 0x5d50170faf334f6647cc16b26cb0c6668b3a2a8e	 044df903568735cb972bb207157860578d78ec...	admin	导出 删除
 0xf3100fe2d12028e3f1bc23d332fbf5eccc396520	 04a685f367536c1c8e2fbe56f0e5ded903914cac...	user1	导出 删除
 0xfc9faefe9edf2cf3c7d332a8ec2d66b7725045f3	 049a9cfc405fdb46fe1d4be5b26765adccdbb7...	user2	导出 删除

# 合约测试

- 合约编译完成后，使用管理员admin账户部署，初始化发行100游戏代币，将游戏代币命名为ZHCoin

选择用户地址

用户: 0x5d50170faf334f6647cc16b26cb0c6668b (admin)

CNS:

参数:

initialSupply	100
tokenName	ZHCoin

❗ 如果参数类型是数组，请按照以下格式输入，以逗号分隔，非数值和布尔值须使用双引号，例如：["aaa","bbb"]和[100,101]；如果数组参数包含双引号，需转义，例如：["aaa\"bbb","cc c"]。

取消 确认

# 合约测试

- 调用transfer函数，将admin账户的100个游戏币转移给user1地址

合约调用

合约名称: GameToken

CNS:

合约地址: 0x0a950dc353f8dbbe3e4e96fa6030f3030b546e86

方法: transfer 1

用户: 0x5d50170faf334f6647cc16b26cb0c6668b3a2a8e (admin)

参数:

_to	0xf3100fe2d12028e3f1bc23d332fbf5eetc3
_value	100

取消 确认 3

# 合约测试

- 调用transfer函数，将0xf31...的50个游戏币转移给user2地址

合约调用

合约名称: GameToken

CNS:

合约地址: 0x0a950dc353f8dbbe3e4e96fa6030f3030b546e86

方法: transfer 1

用户: 0xf3100fe2d12028e3f1bc23d332fbf5eccc396520 2 (user1)

参数:

_to	0xfc9faefe9edf2cf3c7d332a8ec2d66b7725f
_value	50

取消 确认 4

# 合约测试

- 调用balanceOf函数，获取user1地址的余额，在交易回执中观察到余额为50

### 合约调用

合约名称: GameToken

CNS:

合约地址: 0x0a950dc353f8dbbe3e4e96fa6030f3030b546e86

方法: balanceOf

参数: who 0xf3100fe2d12028e3f1bc23d332fbf5eccc3

如果参数类型是数组，请按照以下格式输入，以逗号分隔，非数值和布尔值须使用双引号，例如: ["aaa","bbb"]和[100,101]; 如果数组参数包含双引号，需转义，例如: ["aaa\"bbb","ccc"]。

取消 确认

### 交易回执

```
[  
  "50"  
]
```



# 合约测试

- 调用balanceOf函数，获取user2地址的余额，在交易回执中观察到余额也为50

合约调用

合约名称: GameToken

CNS:

合约地址: 0x0a950dc353f8dbbe3e4e96fa6030f3030b546e86

方法: balanceOf 1

参数: who 0xfc9faefe9edf2cf3c7d332a8ec2d66b77250 2

① 如果参数类型是数组，请按照以下格式输入，以逗号分隔，非数值和布尔值须使用双引号，例如: ["aaa","bbb"]和[100,101]; 如果数组参数包含双引号，需转义，例如: ["aaa\"bbb","ccc"]。

取消 确认 3

交易回执

```
[  
  "50"  
]
```



# 合约测试

- 调用transferFrom函数，user2用户操作user1账户向admin账户转账10个游戏代币，执行成功后生成了transferFrom事件日志

合约调用

合约地址: 0x0a950dc353f8dbbe3e4e96fa6030f3030b546e86

方法: transferFrom 1

用户: 0xfc9faefe9edf2cf3c7d332a8ec2d66b7725045f3 2 (user2)

参数:

_from	0xf3100fe2d12028e3f1bc23d332fbf5eccc3
_to	0x5d50170faf334f6647cc16b26cb0c6668b:
_value	10

取消 确认 4

# 合约测试

- 再次调用balanceOf函数，获取user2地址的余额，在交易回执中观察到余额变为了40

合约调用

合约名称: GameToken

CNS:

合约地址: 0x0a950dc353f8dbbe3e4e96fa6030f3030b546e86

方法: balanceOf 1

参数: who 0xf3100fe2d12028e3f1bc23d332fbf5eccc3 2

❗ 如果参数类型是数组，请按照以下格式输入，以逗号分隔，非数值和布尔值须使用双引号，例如: ["aaa","bbb"]和[100,101]; 如果数组参数包含双引号，需转义，例如: ["aaa\"bbb\",\"ccc"]。

取消 确认 3

交易回执

"40"

# 总结

- 本节课我们学习了
  - 游戏代币合约的设计
  - 游戏代币合约的编写
  - 游戏代币合约的测试

谢谢