

智能合约

什么是智能合约？

- 智能合约是运行在区块链上的一段代码，代码的逻辑定义了合约的内容
- 智能合约的帐户保存了合约当前的运行状态
 - balance：当前余额
 - nonce：交易次数
 - code：合约代码
 - storage：存储，数据结构是一棵MPT
- Solidity是智能合约最常用的语言，语法上与JavaScript很接近

使用语言：solidity

```
pragma solidity ^0.4.21;

contract SimpleAuction {
    ...address public beneficiary; ... // 拍卖受益人
    ...uint public auctionEnd; ... // 结束时间
    ...address public highestBidder; ... // 当前的最高出价人
    ...mapping(address => uint) bids; ... // 所有竞拍者的出价
    ...address[] bidders; ... // 所有竞拍者

    ...// 需要记录的事件
    ...event HighestBidIncreased(address bidder, uint amount);
    ...event Pay2Beneficiary(address winner, uint amount);

    ...// 以受益者地址`_beneficiary`的名义,
    ...// 创建一个简单的拍卖, 拍卖时间为`_biddingTime`秒。
    ...constructor(uint _biddingTime, address _beneficiary) public {
    ...    beneficiary = _beneficiary;
    ...    auctionEnd = now + _biddingTime;
    ...}

    ...// 对拍卖进行出价, 随交易一起发送的ether与之前已经发送的
    ...// ether的和为本次出价。
    ...function bid() public payable { ...
    ...}

    ...// 使用withdraw模式
    ...// 由投标者自己取回出价, 返回是否成功
    ...function withdraw() public returns (bool) { ...
    ...}

    ...// 结束拍卖, 把最高的出价发送给受益人
    ...function pay2Beneficiary() public returns (bool) { ...
    ...}
}
```

声明使用solidity的版本

状态变量

log记录

构造函数, 仅在合约创建时调用一次

成员函数, 可以被一个外部账户或合约账户调用

本实例改编自Solidity文档: 简单的公开拍

外部账户如何调用智能合约?

创建一个交易, 接收地址为要调用的那个智能合约的地址, data域填写要调用的函数及其参数的编码值。

BACK

TX 0x73275297b391f3e08b1cc7144d7ab5fcf77fecee92b46ca9ec2946f56ebf8ea2

SENDER ADDRESS 0x903db0EbD4206669Ab50BCF93c550df9b5Da178c		TO CONTRACT ADDRESS 0x5E31d519A6F34d224C25B706687EE2AbF170B888		CONTRACT CALL
VALUE 0.00 ETH	GAS USED 21657	GAS PRICE 1000000000	GAS LIMIT 6000000	MINED IN BLOCK 3
TX DATA 0x2a24f46c				

一个合约如何调用另一个合约:

- 直接调用, 如果失败, 全部回滚
- 使用地址类型的call()函数, 只回滚被调用的函数
- 代理调用 delegate call

payable: 标志此函数接受转入帐

fallback函数 (仅合约账户有):

fallback()函数

```
function() public [payable]{
```

```
.....
```

```
}
```

- 匿名函数，没有参数也没有返回值。
- 在两种情况下会被调用：
 - 直接向一个合约地址转账而不加任何data
 - 被调用的函数不存在
- 如果转账金额不是0，同样需要声明payable，否则会抛出异常。

合约的创建与运行

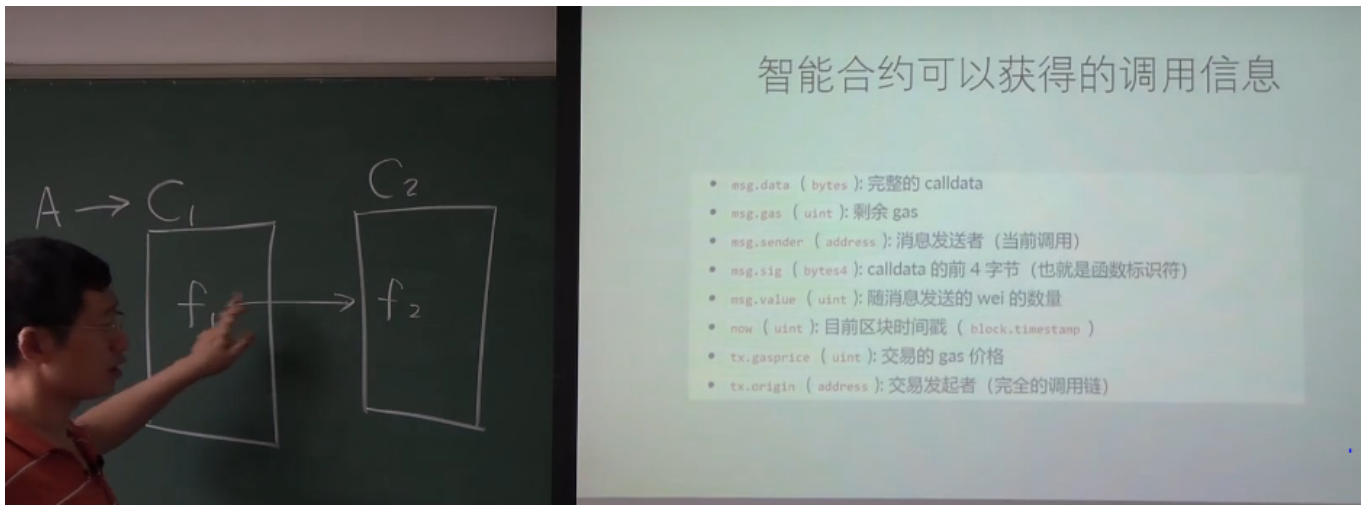
智能合约的创建和运行

- 智能合约的代码写完后，要编译成bytecode
- 创建合约：外部帐户发起一个转账交易到0x0的地址
 - 转账的金额是0，但是要支付汽油费
 - 合约的代码放在data域里
- 智能合约运行在EVM（Ethereum Virtual Machine）上
- 以太坊是一个交易驱动的状态机
 - 调用智能合约的交易发布到区块链上后，每个矿工都会执行这个交易，从当前状态确定性地转移到下一个状态

EVM的好处：寻址空间256位，普通计算机只有64位

所有的全节点都是每次先执行将要打包的交易（包括合约操作），然后执行后修改本地维护的状态树。全部执行完后，算出state tree root放在区块头里，再计算nonce值挖矿。一旦一个全节点算出了符合target的值（即挖到矿），就将自己这一区块所打包的交易广播出去，所有全节点听到后都独立将所有的交易在本地执行一遍，算出三棵树的根hash值，并验证挖到矿的节点nonce值是否正确。若正确，则维护更新本地的状态树。

由于出于验证的角度，所有节点在这个状态机下运行完后都要处于同一个状态，所以智能合约是不支持多线程的，同时也没有真的随机函数及获取节点计算机系统参数的函数。



msg.sender不同于tx.origin，如上图，对于合约C2，它的f2函数的msg.sender是f1,tx.origin是A。

三种转账方式：transfer，send，call.value

区别：transfer出错不支持回滚，另外两个支持。call.value被调用的同时，移交所有剩余的汽油费，另外两个只转移2300，大概仅支持写一条log的汽油费。