智能合约

笔记本:	区块链		
创建时间:	2019/10/15 14:34	**更新时间:**	2019/11/2 16:02
作者:	jyyhermance@163.com		

部署智能合约

- 创建节点

1. 创世区块的初始化文件　genesis.json
2. 创建私有链

geth --datadir ethprivate init genesis.json
指定目录ethprivate，创建子目录geth，keystore

```
D:\BlockChain\Geth>geth --datadir ethprivate init genesis.json
INFO [10-23|17:06:47] Maximum peer count                       ETH=25 LES=0 total=25
INFO [10-23|17:06:48] Allocated cache and file handles         database=D:\\BlockChain\\Geth\\ethprivate\\geth\\chaindata c
ache=16 handles=16
INFO [10-23|17:06:48] Writing custom genesis block
INFO [10-23|17:06:48] Persisted trie from memory database      nodes=0 size=0.00B time=0s gcnodes=0 gcsize=0.00B gctime=0s
livenodes=1 livesize=0.00B
INFO [10-23|17:06:48] Successfully wrote genesis state         database=chaindata                                        h
ash=84e71d…97246e
INFO [10-23|17:06:48] Allocated cache and file handles         database=D:\\BlockChain\\Geth\\ethprivate\\geth\\lightchaind
ata cache=16 handles=16
INFO [10-23|17:06:48] Writing custom genesis block
INFO [10-23|17:06:48] Persisted trie from memory database      nodes=0 size=0.00B time=0s gcnodes=0 gcsize=0.00B gctime=0s
livenodes=1 livesize=0.00B
INFO [10-23|17:06:48] Successfully wrote genesis state         database=lightchaindata
    hash=84e71d…97246e
```

3. 启动私有链

```
D:\BlockChain\Geth>geth --identity "Testnode" --rpc --rpcport "8545" --datadir ethprivate --port "30303" --nodiscover conso
le
INFO [10-23|18:06:22] Maximum peer count                       ETH=25 LES=0 total=25
INFO [10-23|18:06:22] Starting peer-to-peer node               instance=Geth/Testnode/v1.8.3-stable-329ac18e/windows-amd64/
go1.10
INFO [10-23|18:06:22] Allocated cache and file handles         database=D:\\BlockChain\\Geth\\ethprivate\\geth\\chaindata c
ache=768 handles=1024
WARN [10-23|18:06:23] Upgrading database to use lookup entries
INFO [10-23|18:06:23] Database deduplication successful        deduped=0
INFO [10-23|18:06:23] Initialised chain configuration          config="{ChainID: 22 Homestead: 0 DAO: <nil> DAOSupport: fal
se EIP150: <nil> EIP155: 0 EIP158: 0 Byzantium: <nil> Constantinople: <nil> Engine: unknown}"
INFO [10-23|18:06:23] Disk storage enabled for ethash caches   dir=D:\\BlockChain\\Geth\\ethprivate\\geth\\ethash count=3
INFO [10-23|18:06:23] Disk storage enabled for ethash DAGs     dir=C:\\Users\\Hermance\\AppData\\Ethash           count=2
INFO [10-23|18:06:23] Initialising Ethereum protocol           versions="[63 62]" network=1
INFO [10-23|18:06:23] Loaded most recent local header          number=0 hash=84e71d…97246e td=1024
INFO [10-23|18:06:23] Loaded most recent local full block      number=0 hash=84e71d…97246e td=1024
INFO [10-23|18:06:23] Loaded most recent local fast block      number=0 hash=84e71d…97246e td=1024
INFO [10-23|18:06:23] Regenerated local transaction journal    transactions=0 accounts=0
INFO [10-23|18:06:23] Starting P2P networking
INFO [10-23|18:06:23] RLPx listener up                         self="enode://cfccd98cd1e8db787bf349585e8bf90827cd60fad31fda
c449eaa24b7bc6f0c276ddde22ac83ca8ff6c901684471866aa1d19d89a7ddef95ce37d0bfa1cb024b@[::]:30303?discport=0"
INFO [10-23|18:06:23] IPC endpoint opened                      url=\\\\.\\pipe\\geth.ipc
INFO [10-23|18:06:23] HTTP endpoint opened                     url=http://127.0.0.1:8545 cors= vhosts=localhost
Welcome to the Geth JavaScript console!
```

--identity： 指定节点 ID；
--rpc： 表示开启 HTTP-RPC 服务；
--rpcport： 指定 HTTP-RPC 服务监听端口号（默认为 8545）；
--datadir： 指定区块链数据的存储位置；
--port： 指定和其他节点连接所用的端口号（默认为 30303）；
--nodiscover： 关闭节点发现机制，防止加入有同样初始配置的陌生节点；
启动成功后，进入控制台

4. 创建账号

```
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x3bf30751ece35aeff3afc86496f2705852aa256c"
```

输入两次密码确认（此处密码为12345），显示生成的账号

5. 查看账户余额

```
> myAddress = "0x3bf30751ece35aeff3afc86496f2705852aa256c"
"0x3bf30751ece35aeff3afc86496f2705852aa256c"
> eth.getBalance(myAddress)
0
```

将产生的地址存入myAddress，查看目前的余额为0

6. 挖矿和停止挖矿　　miner.start()　　miner.stop()

- 编写智能合约

安装solidity编辑器

```
D:\BlockChain\Solc>solc --version
solc, the solidity compiler commandline interface
Version: 0.5.12+commit.7709ece9.Windows.msvc
```

通过文本编辑器编写.sol文件，用solc编译

```
D:\BlockChain\Solc>solc testCon.sol
Compiler run successful, no output requested.

D:\BlockChain\Solc>solc --bin testCon.sol

======= testCon.sol:testContract =======
Binary:
608060405234801560f57600080fd5b5060ae8061001e6000396000f3fe6080604052348015600f57600080fd5b506004361060285760003560e01c
8063320ef95414602d575b600080fd5b6056600480360360208110156041576000080fd5b810190808035906020019092919050505050606c565b604051
8082815260200191505060405180910390f35b60006007820290509190505056fea265627a7a72315820f6e178f6588430bd053126c29572e11aaa6601
1b4a425b3eab233e55cdb6bde864736f6c634300050c0032

D:\BlockChain\Solc>solc --abi testCon.sol

======= testCon.sol:testContract =======
Contract JSON ABI
[{"constant":true,"inputs":[{"internalType":"uint256","name":"a","type":"uint256"}],"name":"multiply","outputs":[{"inter
nalType":"uint256","name":"d","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"}]
```

--bin获得EVM二进制码
--abi获得合约的JSON abi

- 部署智能合约

```
> code = "0x608060405234801560f57600080fd5b5060ae8061001e6000396000f3fe6080604052348015600f57600080fd5b506004361060285760003560e01c8063320ef95414602d575b600080fd5b6056600480360360208110156041576000080fd5b810190808035906020019092919050505060606c565b6040518082815260200191505060405180910390f35b60006007820290509190505fea265627a7a72315820f6e178f6588430bd053126c29572e11aaa66011b4a425b3eab233e55cdb6bde864736f6c634300050c0032"
"0x608060405234801560f57600080fd5b5060ae8061001e6000396000f3fe6080604052348015600f57600080fd5b506004361060285760003560e01c8063320ef95414602d575b600080fd5b6056600480360360208110156041576000080fd5b810190808035906020019092919050505060606c565b6040518082815260200191505060405180910390f35b60006007820290509190505fea265627a7a72315820f6e178f6588430bd053126c29572e11aaa66011b4a425b3eab233e55cdb6bde864736f6c634300050c0032"
> abi = [{"constant":true,"inputs":[{"internalType":"uint256","name":"a","type":"uint256"}],"name":"multiply","outputs":[{"internalType":"uint256","name":"d","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"}]
[{
    constant: true,
    inputs: [{
        internalType: "uint256",
        name: "a",
        type: "uint256"
    }],
    name: "multiply",
    outputs: [{
        internalType: "uint256",
        name: "d",
        type: "uint256"
    }],
    payable: false,
    stateMutability: "pure",
    type: "function"
}]
```

将上面获得的二进制码和abi存入变量

解锁当前账户

```
> personal.unlockAccount(myAddress)
Unlock account 0x3bf30751ece35aeff3afc86496f2705852aa256c
Passphrase:
true
> myContract = eth.contract(abi)
```

发送部署合约:

```
> myContract = eth.contract(abi)
> contract = myContract.new({from:myAddress,data:code,gas:1000000})
```

查看目前待确认的交易:

```
> txpool.status
{
  pending: 1,
  queued: 0
}
> eth.getBlock("pending",true).transactions
[{
    blockHash: "0xbee2ea2f984c72475a56837bfacc45688faaefc5fbcdbbebc4be05cb726b476d",
    blockNumber: 66,
    from: "0x3bf30751ece35aeff3afc86496f2705852aa256c",
    gas: 1000000,
    gasPrice: 18000000000,
    hash: "0xd4acae28b748b8eec791f166f8b759ae0a4a51ab21c685e92652036324e2f0af",
    input: "0x608060405234801560f57600080fd5b5060ae8061001e6000396000f3fe6080604052348015600f57600080fd5b506004361060285760003560e01c8063320ef95414602d575b600080fd5b6056600480360360208110156041576000080fd5b810190808035906020019092919050505060606c565b6040518082815260200191505060405180910390f35b60006007820290509190505fea265627a7a72315820f6ea66011b4a425b3eab233e55cdb6bde864736f6c634300050c0032",
    nonce: 0,
    r: "0x4184df78a503c5d7efce44b672a9c3e60ece3a9a38997d6fec1b97e52199661d",
    s: "0x241d3e940fea204500dc3c2da86339e64046aa166f00bcb8a2239383dd0433b",
    to: null,
    transactionIndex: 0,
    v: "0x4f",
    value: 0
}]
```

miner.start()开始挖矿，一段时间后交易被确认

调用智能合约

交易记录到区块链中

```
> contract.multiply.sendTransaction(10, {from:myAddress})
```

本地运行查看结果

```
> contract.multiply.call(10)
```