

Ethereum Development Tutorial



Written by: Pham Hoai Luan

Last revision: May, 2018

How to build Ethereum basic: accounts, contracts, nodes and miners

In previous tutorial, we have successfully completed the sending transaction by Geth client on Ethereum network. Now, we continue comprehending deeply Ethereum protocol by building Ethereum basic. So we have a real scenario about money like: "I want to send some money to my friend who is Mr. B. I could send this money by creating a centralized server, but using an Ethereum money contract comes with some free functionalities: for one, it's a decentralized service and money can be still exchanged even if the original service goes down for any reason. This Ethereum protocol eliminates the scenarios where one single server break-in can result in the loss of money".

To resolve this scenario, I will build the Ethereum money contract. Specifically, I have 3 nodes and the account of node 1 will send some money to the account of node 2 through the Ethereum money contract. Besides, the account of node 3 acts as miner to write transactions on ledger as shown in Figure 1.

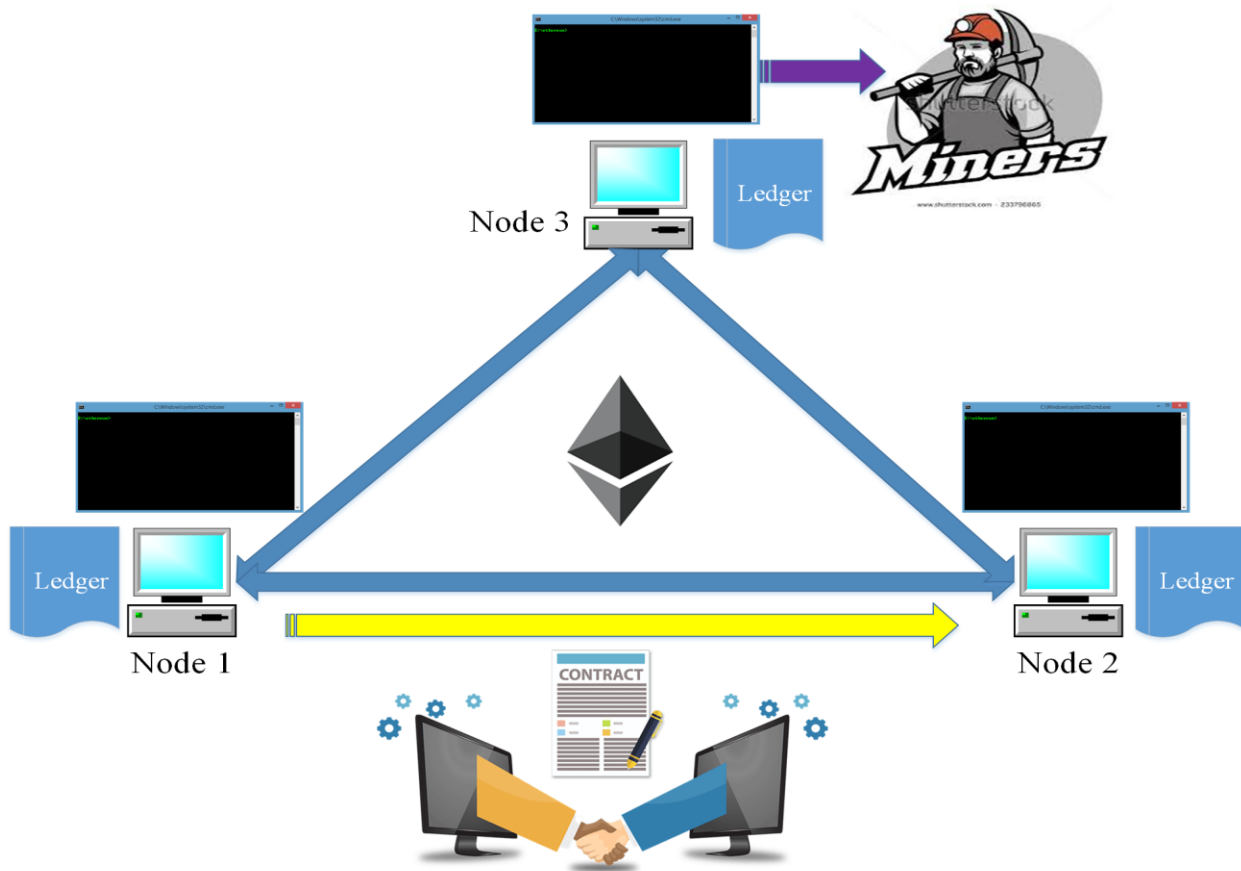


Figure 1. Smart contract Ethereum

Create a network with 3 nodes

I will not explain too much details in this step because it is guided entirely in the previous tutorial. I will go quickly through images.

Starting at node 1

Create a genesis.json file like:

```
{
  "config": {
    "chainId": 1985,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "difficulty": "4000",
  "gasLimit": "2100000",
  "alloc": {}
}
```

And put it in an “ethereum_transaction” folder as shown in Figure 2.

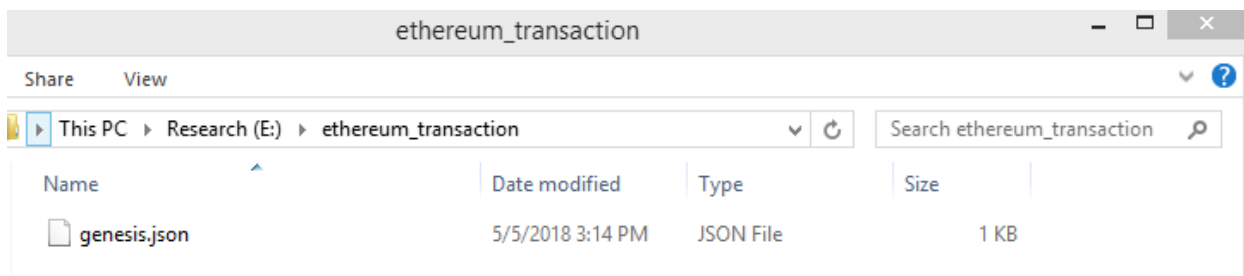


Figure 2. Genesis file

Right-click and choose “Open with Command Prompt as shown in Figure 3.

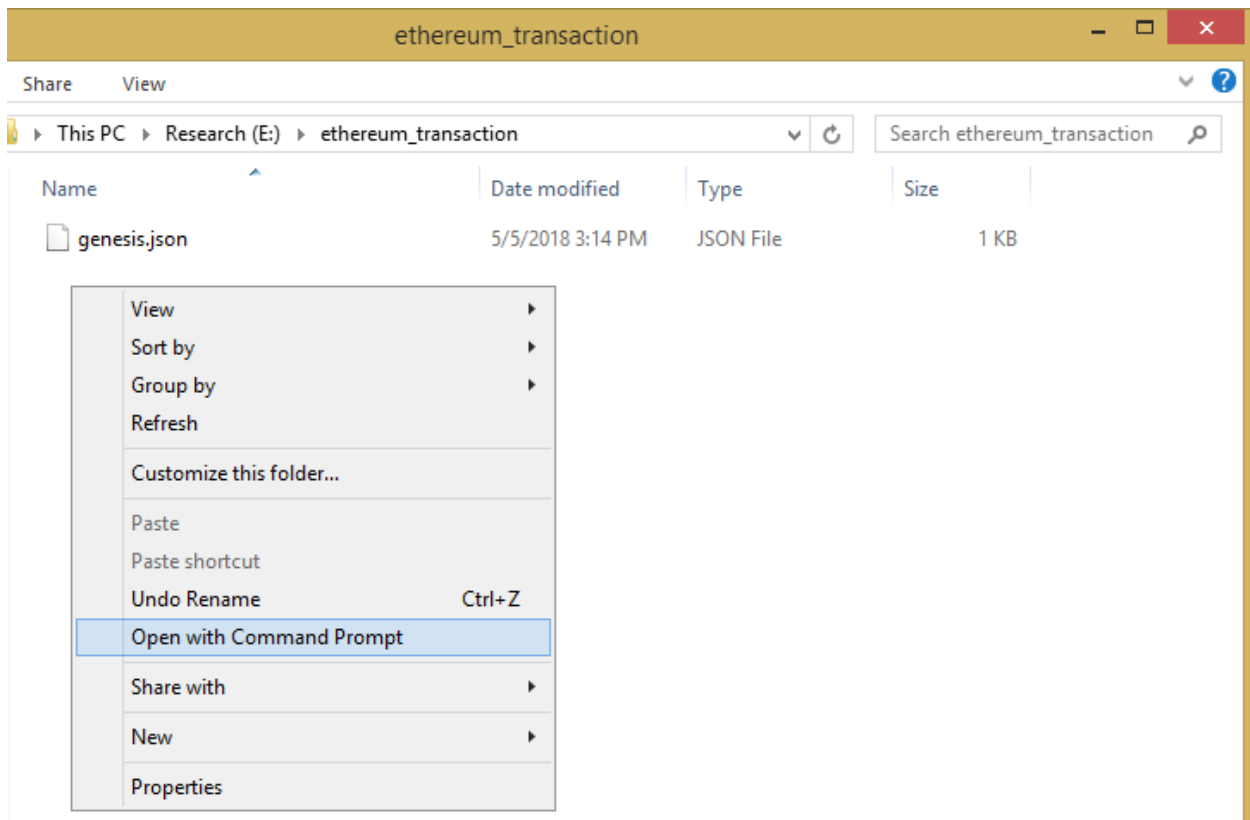


Figure 3. Open with Command Prompt

Then the terminal window will appear as Figure 4.

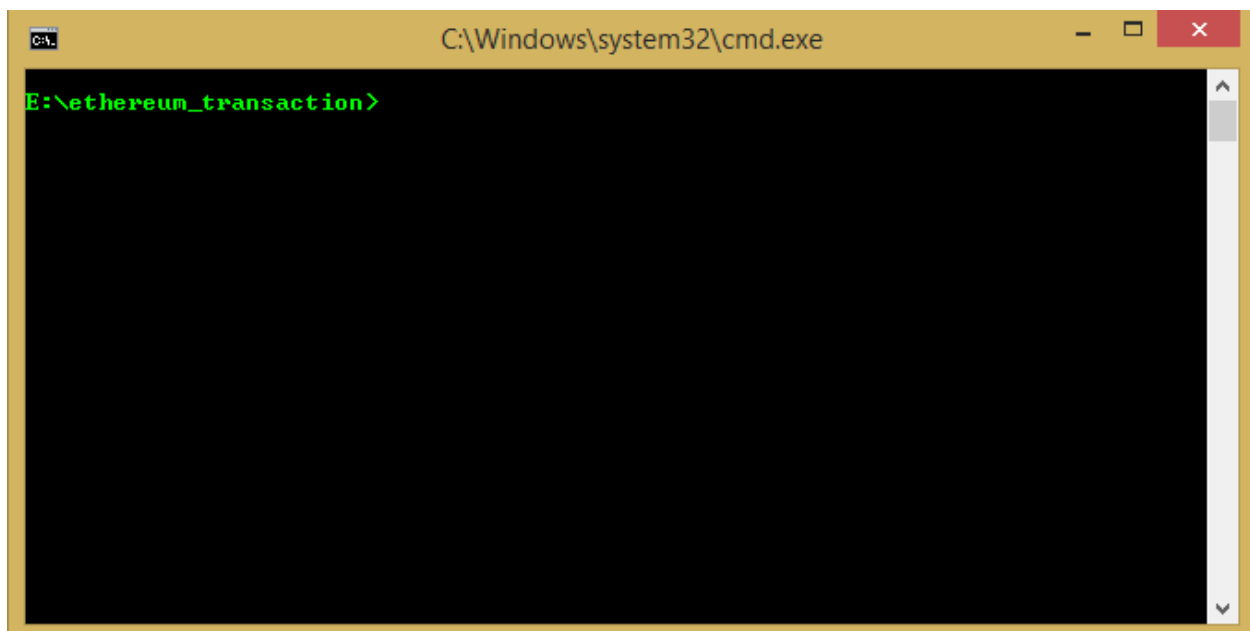
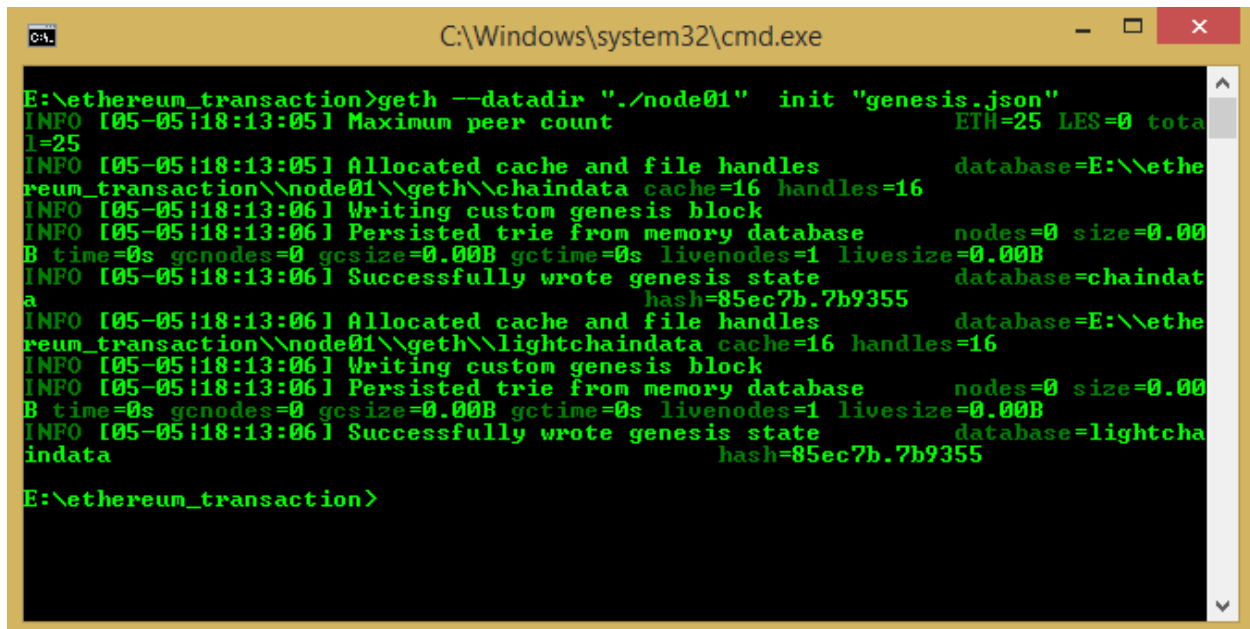


Figure 4. Command Prompt terminal

Type “*geth --datadir "./node01" init "genesis.json"*” in Command Prompt terminal like Figure 5.



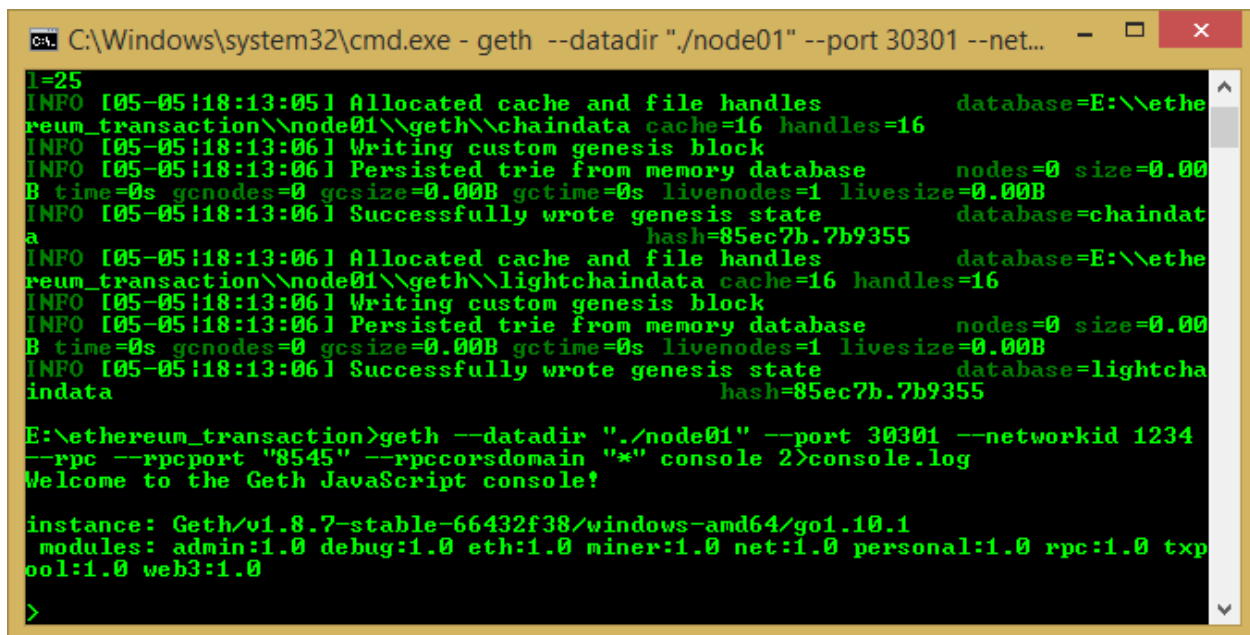
```
C:\Windows\system32\cmd.exe

E:\ethereum_transaction>geth --datadir "./node01" init "genesis.json"
INFO [05-05:18:13:05] Maximum peer count ETH=25 LES=0 total=25
INFO [05-05:18:13:05] Allocated cache and file handles database=E:\ethereum_transaction\node01\geth\chaindata cache=16 handles=16
INFO [05-05:18:13:06] Writing custom genesis block
INFO [05-05:18:13:06] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:13:06] Successfully wrote genesis state database=chaindata hash=85ec7b.7b9355
INFO [05-05:18:13:06] Allocated cache and file handles database=E:\ethereum_transaction\node01\geth\lightchaindata cache=16 handles=16
INFO [05-05:18:13:06] Writing custom genesis block
INFO [05-05:18:13:06] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:13:06] Successfully wrote genesis state database=lightchaindata hash=85ec7b.7b9355

E:\ethereum_transaction>
```

Figure 5. Create genesis block at node 1

Then type “*geth --datadir "./node01" --port 30301 --networkid 1234 --rpc --rpcport "8545" --rpccorsdomain "*" console 2>console.log*” in Command Prompt terminal like Figure 6.



```
C:\Windows\system32\cmd.exe - geth --datadir "./node01" --port 30301 --net...

E:\ethereum_transaction>geth --datadir "./node01" --port 30301 --networkid 1234 --rpc --rpcport "8545" --rpccorsdomain "*" console 2>console.log
Welcome to the Geth JavaScript console!

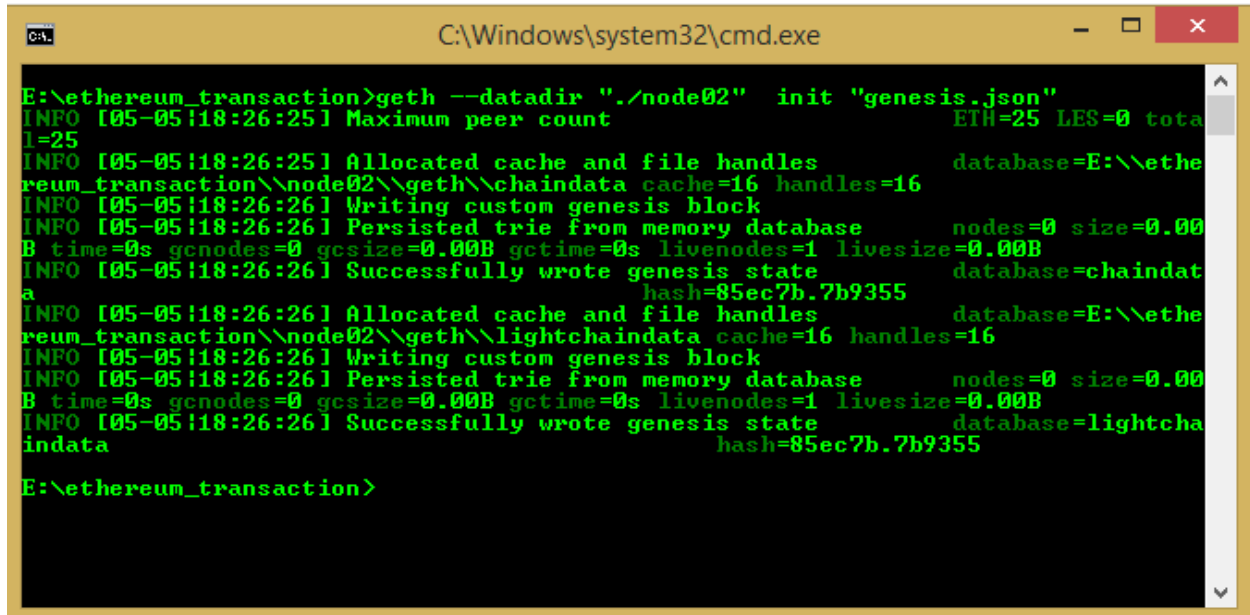
instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

>
```

Figure 6. Open Geth console at node 1

Starting at node 2

Right-click and choose “Open with Command Prompt as shown in Figure 3. Then the terminal window will also appear as Figure 4. Type “*geth --datadir "./node02" init "genesis.json"*” in Command Prompt terminal like Figure 7.



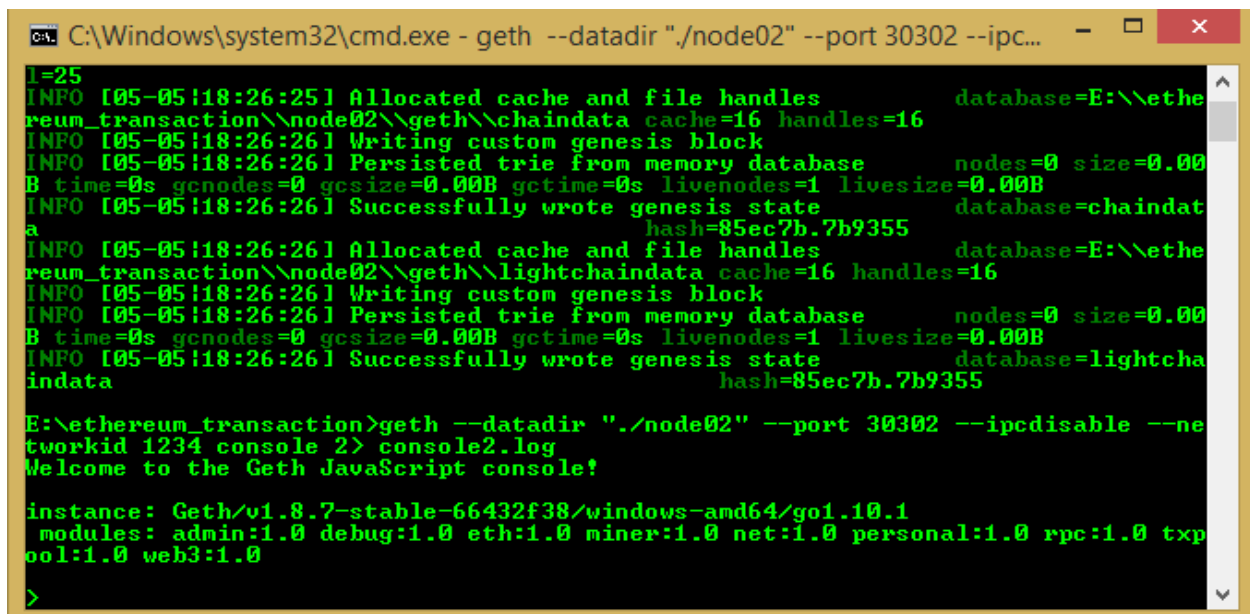
```
C:\Windows\system32\cmd.exe

E:\ethereum_transaction>geth --datadir "./node02" init "genesis.json"
INFO [05-05:18:26:25] Maximum peer count ETH=25 LES=0 total
l=25
INFO [05-05:18:26:25] Allocated cache and file handles database=E:\ethe
reum_transaction\node02\geth\chaindata cache=16 handles=16
INFO [05-05:18:26:26] Writing custom genesis block
INFO [05-05:18:26:26] Persisted trie from memory database nodes=0 size=0.00
B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:26:26] Successfully wrote genesis state database=chaindat
a hash=85ec7b.7b9355
INFO [05-05:18:26:26] Allocated cache and file handles database=E:\ethe
reum_transaction\node02\geth\lightchaindata cache=16 handles=16
INFO [05-05:18:26:26] Writing custom genesis block
INFO [05-05:18:26:26] Persisted trie from memory database nodes=0 size=0.00
B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:26:26] Successfully wrote genesis state database=lightcha
indata hash=85ec7b.7b9355

E:\ethereum_transaction>
```

Figure 7. Create genesis block at node 2

Then type “*geth --datadir "./node02" --port 30302 --ipcdisable --networkid 1234 console 2> console2.log*” in Command Prompt terminal like Figure 8.



```
C:\Windows\system32\cmd.exe - geth --datadir "./node02" --port 30302 --ipc...

E:\ethereum_transaction>geth --datadir "./node02" --port 30302 --ipcdisable --ne
tworkid 1234 console 2> console2.log
Welcome to the Geth JavaScript console!

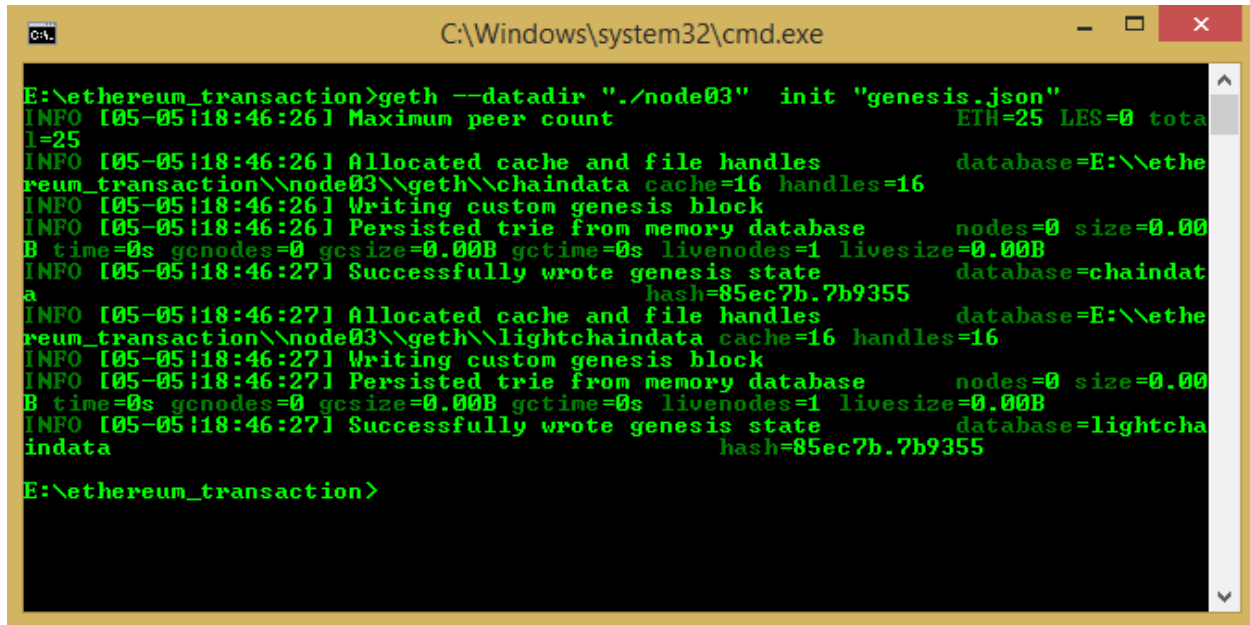
instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp
ool:1.0 web3:1.0

>
```

Figure 8. Open Geth console at node 2

Starting at node 3

Right-click and choose “Open with Command Prompt as shown in Figure 3. Then the terminal window will also appear as Figure 4. Type “*geth --datadir "/node03" init "genesis.json"*” in Command Prompt terminal like Figure 9.



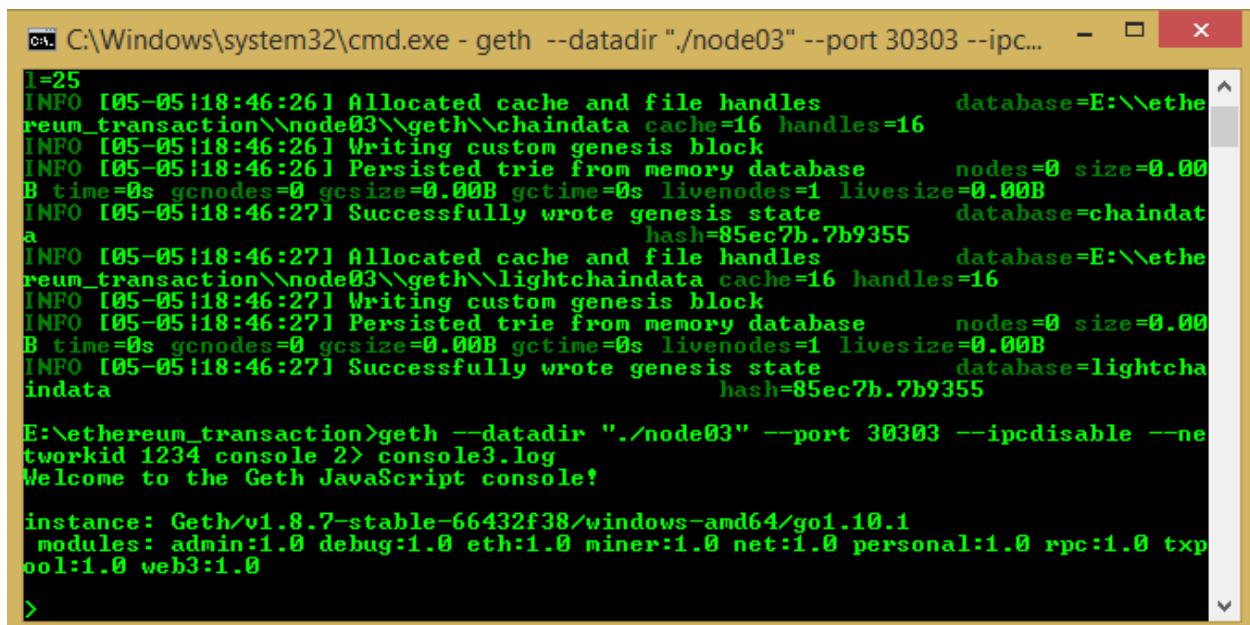
```
C:\Windows\system32\cmd.exe

E:\ethereum_transaction>geth --datadir "/node03" init "genesis.json"
INFO [05-05:18:46:26] Maximum peer count ETH=25 LES=0 total=25
INFO [05-05:18:46:26] Allocated cache and file handles database=E:\ethereum_transaction\node03\geth\chaindata cache=16 handles=16
INFO [05-05:18:46:26] Writing custom genesis block
INFO [05-05:18:46:26] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcspace=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:46:27] Successfully wrote genesis state database=chaindata hash=85ec7b.7b9355
INFO [05-05:18:46:27] Allocated cache and file handles database=E:\ethereum_transaction\node03\geth\lightchaindata cache=16 handles=16
INFO [05-05:18:46:27] Writing custom genesis block
INFO [05-05:18:46:27] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcspace=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:46:27] Successfully wrote genesis state database=lightchaindata hash=85ec7b.7b9355

E:\ethereum_transaction>
```

Figure 9. Create genesis block at node 3

Then type “*geth --datadir "/node03" --port 30303 --ipcdisable --networkid 1234 console 2> console3.log*” in Command Prompt terminal like Figure 10.



```
C:\Windows\system32\cmd.exe - geth --datadir "/node03" --port 30303 --ipc...

1=25
INFO [05-05:18:46:26] Allocated cache and file handles database=E:\ethereum_transaction\node03\geth\chaindata cache=16 handles=16
INFO [05-05:18:46:26] Writing custom genesis block
INFO [05-05:18:46:26] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcspace=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:46:27] Successfully wrote genesis state database=chaindata hash=85ec7b.7b9355
INFO [05-05:18:46:27] Allocated cache and file handles database=E:\ethereum_transaction\node03\geth\lightchaindata cache=16 handles=16
INFO [05-05:18:46:27] Writing custom genesis block
INFO [05-05:18:46:27] Persisted trie from memory database nodes=0 size=0.00B time=0s gcnodes=0 gcspace=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-05:18:46:27] Successfully wrote genesis state database=lightchaindata hash=85ec7b.7b9355

E:\ethereum_transaction>geth --datadir "/node03" --port 30303 --ipcdisable --networkid 1234 console 2> console3.log
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

>
```

Figure 10. Open Geth console at node 3

Linking 3 nodes

At **node 2**, we type “admin.nodeInfo.enode” like Figure 11.

```
C:\Windows\system32\cmd.exe - geth --datadir "./node02" --port 30302 --ipc... - □ ×

INFO [05-06:18:44:44] Writing custom genesis block
INFO [05-06:18:44:44] Persisted trie from memory database      nodes=0 size=0.00B
B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-06:18:44:44] Successfully wrote genesis state      database=chaindata
                        hash=85ec7b.7b9355
INFO [05-06:18:44:44] Allocated cache and file handles      database=E:\ether
reum_2\node02\geth\lightchaindata cache=16 handles=16
INFO [05-06:18:44:44] Writing custom genesis block
INFO [05-06:18:44:44] Persisted trie from memory database      nodes=0 size=0.00B
B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-06:18:44:44] Successfully wrote genesis state      database=lightchaindata
                        hash=85ec7b.7b9355

E:\ethereum_2>geth --datadir "./node02" --port 30302 --ipcdisable --networkid 1234 console 2> console2.log
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> admin.nodeInfo.enode
'enode://201d687c297e87321aca03235dd362e88f613473a645a74d410eb3bc61fc164ed5d7a3f0197bfc74fa6d0343c211fa7480276d393c3f75528e8a504616466ee@1:30302'
>
```

Figure 11. Get enode address at node 2

At **node 1**, we type “admin.addPeer(enode address **node 2**)” like Figure 12.

```
C:\Windows\system32\cmd.exe - geth --datadir "./node01" --port 30301 --net... - □ ×

INFO [05-06:18:44:07] Persisted trie from memory database      nodes=0 size=0.00B
B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-06:18:44:07] Successfully wrote genesis state      database=chaindata
                        hash=85ec7b.7b9355
INFO [05-06:18:44:07] Allocated cache and file handles      database=E:\ether
reum_2\node01\geth\lightchaindata cache=16 handles=16
INFO [05-06:18:44:07] Writing custom genesis block
INFO [05-06:18:44:07] Persisted trie from memory database      nodes=0 size=0.00B
B time=0s gcnodes=0 gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-06:18:44:07] Successfully wrote genesis state      database=lightchaindata
                        hash=85ec7b.7b9355

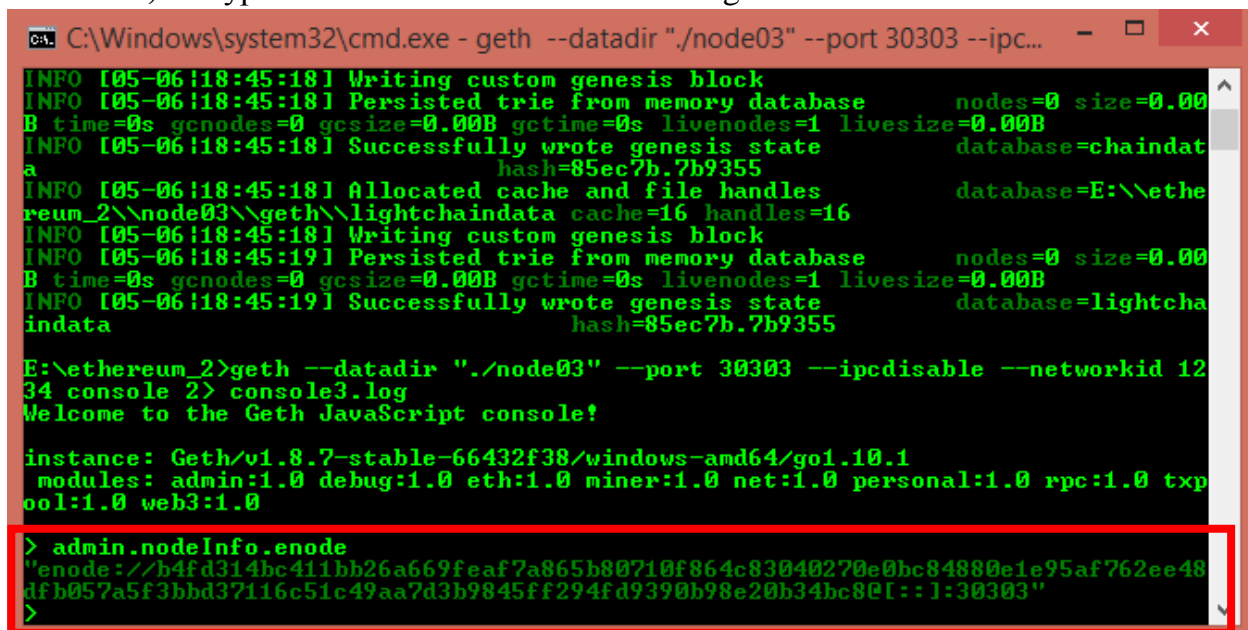
E:\ethereum_2>geth --datadir "./node01" --port 30301 --networkid 1234 --rpc --rpcport "8545" --rpccorsdomain "*" console 2>console.log
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> admin.addPeer("enode://201d687c297e87321aca03235dd362e88f613473a645a74d410eb3bc61fc164ed5d7a3f0197bfc74fa6d0343c211fa7480276d393c3f75528e8a504616466ee@1:30302")
true
>
```

Figure 12. Add node 2 at node 1

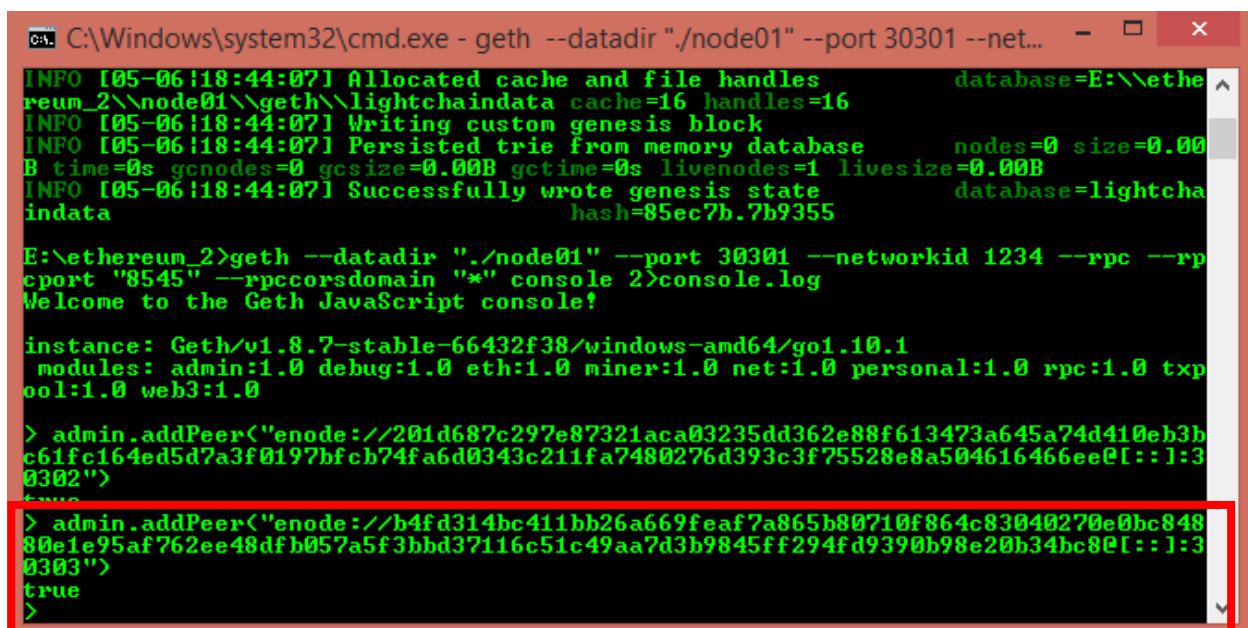
At **node 3**, we type “admin.nodeInfo.enode” like Figure 13.



```
C:\Windows\system32\cmd.exe - geth --datadir "./node03" --port 30303 --ipc...  
INFO [05-06:18:45:18] Writing custom genesis block  
INFO [05-06:18:45:18] Persisted trie from memory database      nodes=0 size=0.00B  
B time=0s gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B  
INFO [05-06:18:45:18] Successfully wrote genesis state      database=chaindata  
hash=85ec7b.7b9355  
INFO [05-06:18:45:18] Allocated cache and file handles      database=E:\ethe  
reum_2\node03\geth\lightchaindata cache=16 handles=16  
INFO [05-06:18:45:18] Writing custom genesis block  
INFO [05-06:18:45:19] Persisted trie from memory database      nodes=0 size=0.00B  
B time=0s gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B  
INFO [05-06:18:45:19] Successfully wrote genesis state      database=lightcha  
indata hash=85ec7b.7b9355  
  
E:\ethereum_2>geth --datadir "./node03" --port 30303 --ipcdisable --networkid 12  
34 console 2> console3.log  
Welcome to the Geth JavaScript console!  
  
instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1  
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp  
ool:1.0 web3:1.0  
  
> admin.nodeInfo.enode  
"enode://b4fd314bc411bb26a669feaf7a865b80710f864c83040270e0bc84880e1e95af762ee48  
dfb057a5f3bbd37116c51c49aa7d3b9845ff294fd9390b98e20b34bc80[:]:1:30303"  
>
```

Figure 13. Get enode address at node 2

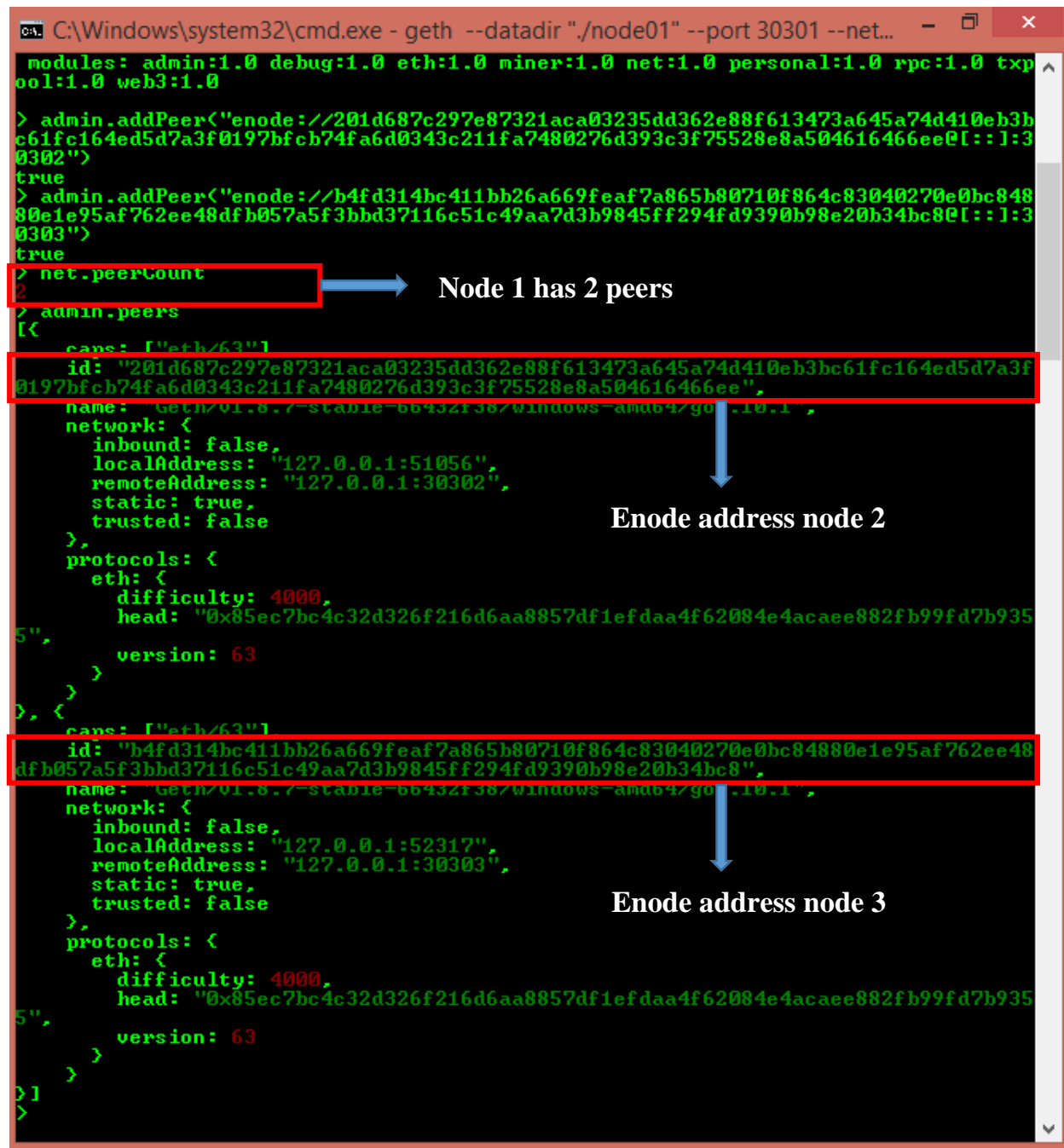
At **node 1**, we type “admin.addPeer(enode address **node 3**)” like Figure 14.



```
C:\Windows\system32\cmd.exe - geth --datadir "./node01" --port 30301 --net...  
INFO [05-06:18:44:07] Allocated cache and file handles      database=E:\ethe  
reum_2\node01\geth\lightchaindata cache=16 handles=16  
INFO [05-06:18:44:07] Writing custom genesis block  
INFO [05-06:18:44:07] Persisted trie from memory database      nodes=0 size=0.00B  
B time=0s gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B  
INFO [05-06:18:44:07] Successfully wrote genesis state      database=lightcha  
indata hash=85ec7b.7b9355  
  
E:\ethereum_2>geth --datadir "./node01" --port 30301 --networkid 1234 --rpc --rp  
cport "8545" --rpccorsdomain "*" console 2>console.log  
Welcome to the Geth JavaScript console!  
  
instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1  
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp  
ool:1.0 web3:1.0  
  
> admin.addPeer("enode://201d687c297e87321aca03235dd362e88f613473a645a74d410eb3b  
c61fc164ed5d7a3f0197bfc74fa6d0343c211fa7480276d393c3f75528e8a504616466ee[:]:1:3  
0302")  
true  
  
> admin.addPeer("enode://b4fd314bc411bb26a669feaf7a865b80710f864c83040270e0bc848  
80e1e95af762ee48dfb057a5f3bbd37116c51c49aa7d3b9845ff294fd9390b98e20b34bc80[:]:1:3  
0303")  
true  
>
```

Figure 14. Add node 3 at node 1

Now, you can check that **node 1** sees **node 2** and **node 3** as its **peers** by typing “net.peerCount” and “admin.peers” as shown in Figure 15.



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - geth --datadir "/>

```

C:\Windows\system32\cmd.exe - geth --datadir "/.node01" --port 30301 --net...
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp
ool:1.0 web3:1.0

> admin.addPeer("enode://201d687c297e87321aca03235dd362e88f613473a645a74d410eb3b
c61fc164ed5d7a3f0197bfc74fa6d0343c211fa7480276d393c3f75528e8a504616466ee@::1:3
0302")
true
> admin.addPeer("enode://b4fd314bc411bb26a669feaf7a865b80710f864c83040270e0bc848
80e1e95af762ee48dfb057a5f3bbd37116c51c49aa7d3b9845ff294fd9390b98e20b34bc8@::1:3
0303")
true
> net.peerCount
2
> admin.peers
[{"
  caps: ["eth/63"],
  id: "201d687c297e87321aca03235dd362e88f613473a645a74d410eb3bc61fc164ed5d7a3f
0197bfc74fa6d0343c211fa7480276d393c3f75528e8a504616466ee",
  name: "Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:51056",
    remoteAddress: "127.0.0.1:30302",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 4000,
      head: "0x85ec7bc4c32d326f216d6aa8857df1efdaa4f62084e4acaae882fb99fd7b935
5",
      version: 63
    }
  }
}, {
  caps: ["eth/63"],
  id: "b4fd314bc411bb26a669feaf7a865b80710f864c83040270e0bc84880e1e95af762ee48
dfb057a5f3bbd37116c51c49aa7d3b9845ff294fd9390b98e20b34bc8",
  name: "Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:52317",
    remoteAddress: "127.0.0.1:30303",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 4000,
      head: "0x85ec7bc4c32d326f216d6aa8857df1efdaa4f62084e4acaae882fb99fd7b935
5",
      version: 63
    }
  }
}]
>

```

Annotations in the image:

- A red box highlights the command `> net.peerCount` and its output `2`. A blue arrow points from this box to the text "Node 1 has 2 peers".
- A red box highlights the first peer's ID in the `admin.peers` output. A blue arrow points from this box to the text "Enode address node 2".
- A red box highlights the second peer's ID in the `admin.peers` output. A blue arrow points from this box to the text "Enode address node 3".

Figure 15. Check peer-to-peer network

Smart contract

Create account

To be able to send transactions as well as mine, we must create an account at each node. We type "*personal.newAccount()*" at each node to create the account like Figure 16.

A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe - geth --datadir './node01' --port 30301 --net...". The window displays the JSON configuration for a Geth node. At the bottom, the command `> personal.newAccount()` has been entered, and the prompt has changed to `Passphrase:`. The user has entered a passphrase, and the prompt has changed to `Repeat passphrase:`. The user has entered the same passphrase again, and the prompt has changed to `>`. The new account address is displayed as `"0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d"`.

```
C:\Windows\system32\cmd.exe - geth --datadir './node01' --port 30301 --net...
dfb057a5f3bbd37116c51c49aa7d3b9845ff294fd9390b98e20b34bc8",
  name: "Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:52317",
    remoteAddress: "127.0.0.1:30303",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 4000,
      head: "0x85ec7bc4c32d326f216d6aa8857df1efdaa4f62084e4acae882fb99fd7b9355",
      version: 63
    }
  }
}
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d"
>
```

Figure 16. Create account at node 1

Then we have to unlock account to transmit transaction or mine by typing "*personal.unlockAccount('account',')*" as shown in Figure 17, just like login your account into facebook to post status or chat.

A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe - geth --datadir './node01' --port 30301 --net...". The window displays the JSON configuration for a Geth node. At the bottom, the command `> personal.newAccount()` has been entered, and the prompt has changed to `Passphrase:`. The user has entered a passphrase, and the prompt has changed to `Repeat passphrase:`. The user has entered the same passphrase again, and the prompt has changed to `>`. The new account address is displayed as `"0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d"`. Then, the command `> personal.unlockAccount('0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d', '')` has been entered, and the prompt has changed to `true`.

```
C:\Windows\system32\cmd.exe - geth --datadir './node01' --port 30301 --net...
dfb057a5f3bbd37116c51c49aa7d3b9845ff294fd9390b98e20b34bc8",
  name: "Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:52317",
    remoteAddress: "127.0.0.1:30303",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 4000,
      head: "0x85ec7bc4c32d326f216d6aa8857df1efdaa4f62084e4acae882fb99fd7b9355",
      version: 63
    }
  }
}
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d"
> personal.unlockAccount('0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d', '')
true
>
```

Figure 17. Unlock account at node 1

Create smart contract

We open a web browser (Google Chrome, Internet Explorer,...) and type address <https://remix.ethereum.org> like Figure 18.

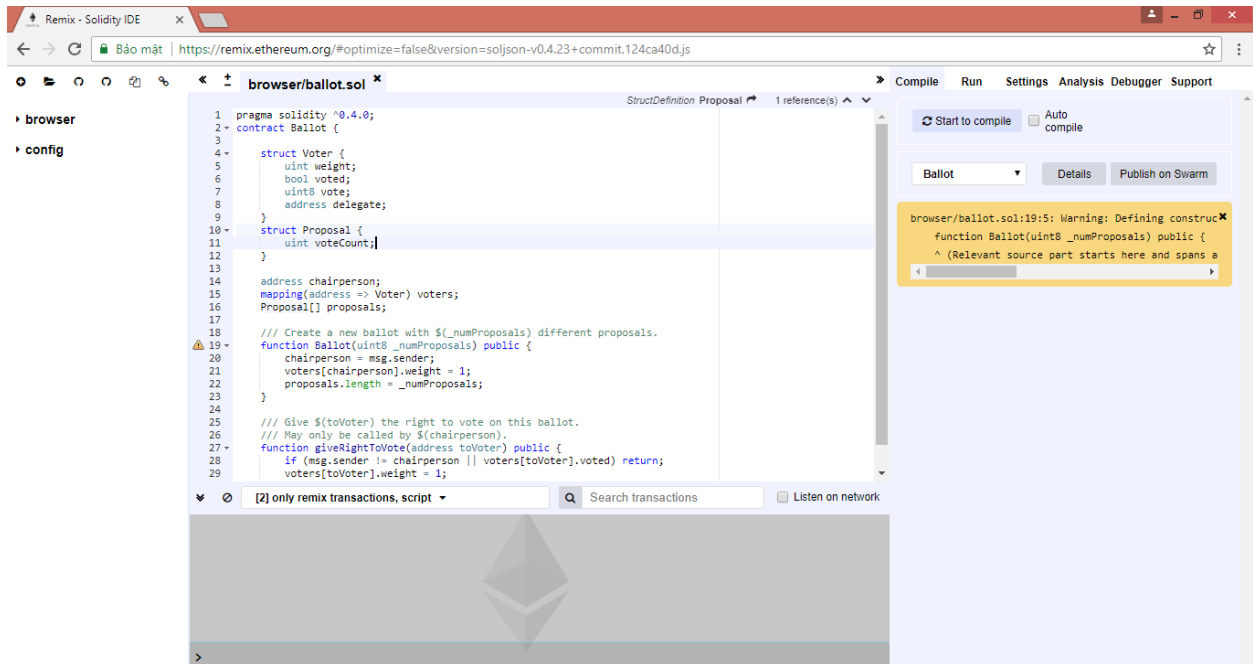


Figure 18. Solidity IDE web

Then we choose “Create New File in the Browser Storage Explore” and rename “mycontract” as shown in Figure 19.

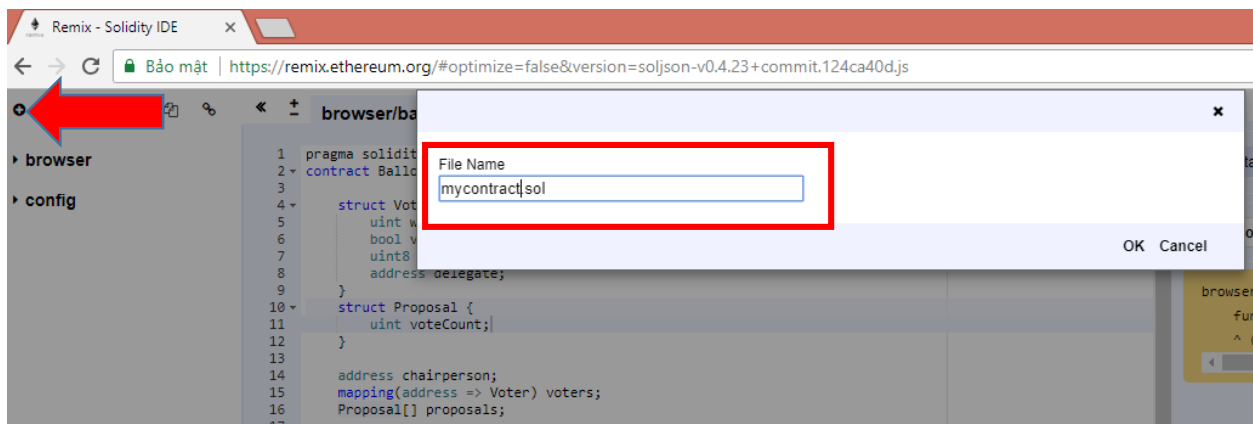


Figure 19. Create a new contract file

Then we write this contract with contents like:

```
pragma solidity ^0.4.16;  
contract Money {  
    mapping (address => uint) public MoneyBalanceOf;  
    event MoneyTransfer(address sender, address receiver, uint amount);  
  
    /* Initializes contract with initial supply money to the creator of the contract */  
    function Money(uint supply) {  
        MoneyBalanceOf[msg.sender] = supply;  
    }  
  
    /* Very simple trade function */  
    function sendMoney(address receiver, uint amount) returns(bool sufficient) {  
        if (MoneyBalanceOf[msg.sender] < amount) return false;  
        MoneyBalanceOf[msg.sender] -= amount;  
        MoneyBalanceOf[receiver] += amount;  
        MoneyTransfer(msg.sender, receiver, amount);  
        return true;  
    }  
}
```

Then we select "Start to compile" like Figure 20.

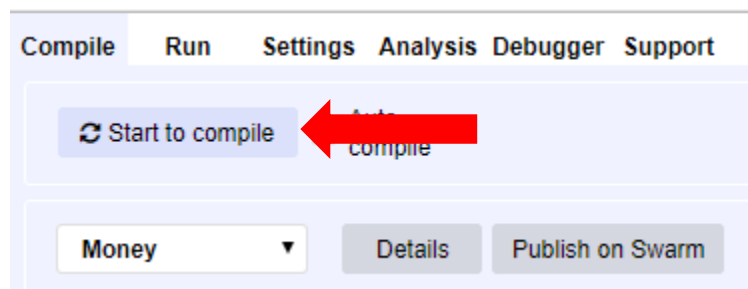


Figure 20. Start to compile

Afterwards, we choose “Run” -> “Environment” -> “Web3 Provider” like Figure 21.

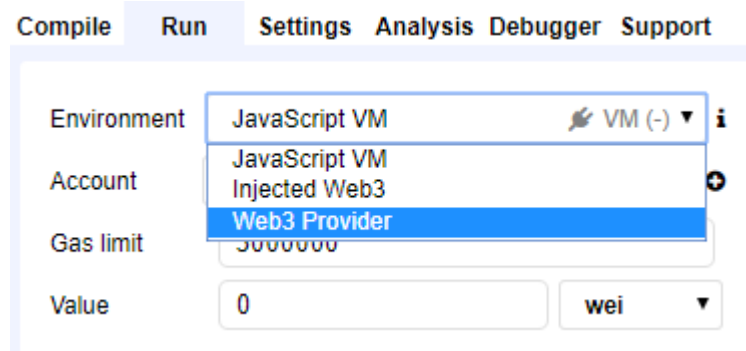


Figure 21s. Choose a Web3 Provider Environment

Then the message will appear like Figure 22, we click OK.

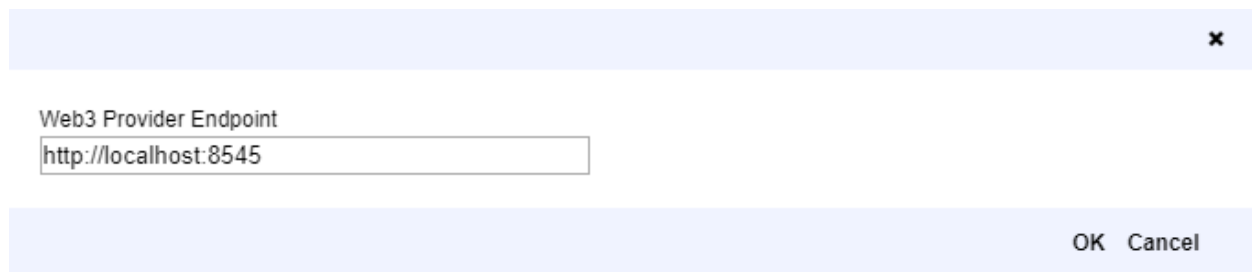


Figure 22. Select local host

Finally, the Web3 Provider will add our Ethereum network with 3 nodes like Figure 23.

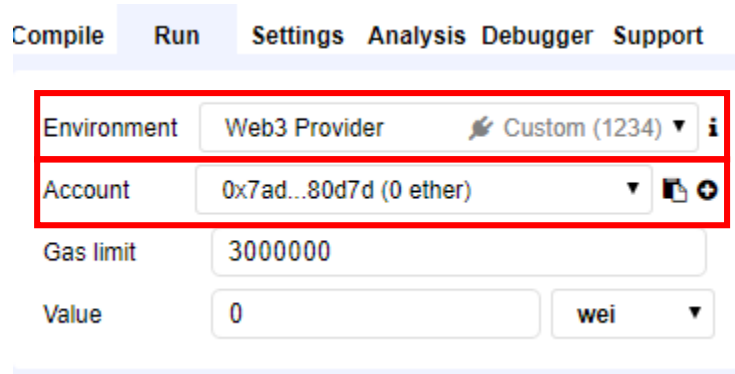


Figure 23. Complete connecting with Ethereum network

Accounts created in the previous steps do not have any ether, meaning that the balance is 0. How do we know that? The account of node 1 have 0 ether in Figure 23. If your account doesn't have any ethers, you can't totally send the transaction. So I have a little trick to get some ethers as follow: You type "*miner.start (1)*" in few minutes as shown in Figure 24. Then type "*miner.stop()*".

```

C:\Windows\system32\cmd.exe - geth --datadir ".\node01" --port 30301 --net...
{
  localAddress: "127.0.0.1:52317",
  remoteAddress: "127.0.0.1:30303",
  static: true,
  trusted: false
},
protocols: {
  eth: {
    difficulty: 4000,
    head: "0x85ec7bc4c32d326f216d6aa8857df1efd4a4f62084e4acaee882fb99fd7b9355",
    version: 63
  }
}
>1
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d"
> personal.unlockAccount('0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d', '')
true
> personal.unlockAccount('0x7ad8e450b7288ceaae51c7c5d97db9ae7ae80d7d', '')
true
> miner.start(1)
null
>

```

Figure 24. Get ether

Now we can see that the account of node 1 has some ethers like Figure 25.

Compile	Run	Settings	Analysis	Debugger	Support
Environment		Web3 Provider Custom (1234) ⓘ			
Account		0x7ad...80d7d (155 ether) ⓘ ⓘ			
Gas limit		3000000			
Value		0		wei ▼	

Figure 25. Check ethers at the account of node 1

Now, let's say I have \$ 1,000 in the account of node 1 by typing 1000 in this empty boxes as shown in Figure 26. After that we click **Create**.

The screenshot shows the 'Run' tab of an Ethereum client interface. The 'Environment' is set to 'Web3 Provider' with a custom URL 'Custom (1234)'. The 'Account' is '0x7ad...80d7d (155 ether)'. The 'Gas limit' is '3000000' and the 'Value' is '0 wei'. Below these, a dropdown menu is set to 'Money', and a text input field contains '1000'. A red rectangle highlights the '1000' input and the 'Create' button. Below the input field are buttons for 'Load contract from Address' and 'At Address'. At the bottom, it shows '0 pending transactions' and '0 contract instances'.

Figure 25. Recharge for the account of node 1

At **node 3**, we type “miner.start(1)” to mine the transaction of recharging for the account of node 1 as shown in Figure 26.

The screenshot shows a terminal window with the following commands and output:

```

C:\Windows\system32\cmd.exe - geth --datadir "./node03" --port 30303 --ipc...
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x87e4aa851e596e55bf2c3ed586e402cbc0e44ecc"
> personal.unlockAccount('0x87e4aa851e596e55bf2c3ed586e402cbc0e44ecc', '')
true
> personal.unlockAccount('0x87e4aa851e596e55bf2c3ed586e402cbc0e44ecc', '')
true
> miner.start(1)
null

```

A red rectangle highlights the last three lines of the terminal output, from the second 'personal.unlockAccount' command to the 'miner.start(1)' command and its 'null' response.

Figure 26. Mine at the account of node 3

[illegible]

After finishing mine of the recharge for the account of node 1, we can check by typing the account of node 1 into “MoneyBalanceOf” box like Figure 28.

Figure 28. Check money at the account of node 1

Now the account of node 1 wants to send money to the account of node 2, but we check firstly the account of node 2 by typing "eth.accounts" like Figure 29.

```
C:\Windows\system32\cmd.exe - geth --datadir "./node02" --port 30302 --ipc...
INFO [05-06:18:44] Persisted trie from memory database      nodes=0 size=0.00
B time=0s gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [05-06:18:44] Successfully wrote genesis state      database=lightcha
indata hash=85ec7b.7b9355

E:\ethereum_2>geth --datadir "./node02" --port 30302 --ipcdisable --networkid 12
34 console 2> console2.log
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.7-stable-66432f38/windows-amd64/go1.10.1
modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp
ool:1.0 web3:1.0

> admin.nodeInfo.enode
"enode://201d687c297e87321aca03235dd362e88f613473a645a74d410eb3bc61fc164ed5d7a3f
0197bfc74fa6d0343c211fa7480276d393c3f75528e8a504616466ee0[:]:1:30302"
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x8bd1b5500b7b9e316a1656e2a72eb8c718d42336"
> personal.unlockAccount('0x8bd1b5500b7b9e316a1656e2a72eb8c718d42336', '')
true
> eth.accounts
["0x8bd1b5500b7b9e316a1656e2a72eb8c718d42336"]
>
```

Figure 29. Check the account of node 2

Then the account of node 1 sends \$ 400 to the account of node 2 by typing "account of nod 2, 400" into "sendMoney" box like Figure 30.

Money at 0xbc1...f1068 (blockchain)

sendMoney 0x8bd1b5500b7b9e316a16

MoneyBalanceOf address 0: uint256: 0

Figure 29. Send Money

At **node 3**, we type "miner.start(1)" to mine the transaction of sending money from the account of node 1 to the account of node 2 like Figure 26.

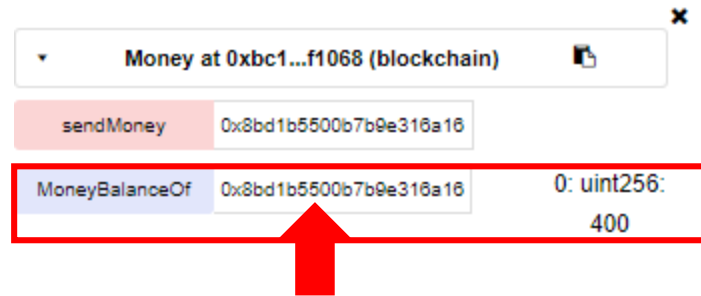
Now we check the account of node 1 and node 2 as shown in Figure 30.

Money at 0xbc1...f1068 (blockchain)

sendMoney 0x8bd1b5500b7b9e316a16

MoneyBalanceOf 0x7ad8e450b7288ceaae51 0: uint256: 600

The account of node 1



The account of node 2

Figure 29. Check Money

Conclusion

In this tutorial, we have totally comprehended insight into Ethereum basic.

Reference

https://ethereum.gitbooks.io/frontier-guide/content/example_script.html