Ambisafe
Software

Blockchain School

2018

Lesson 1. Introduction

We understand what the Ethereum blockchain network is and how transactions are arranged in it, create our own wallets and learn with their help to interact with smart contracts on the network.

Plan

1. Creating a wallet, receiving and sending ether
2. What is a private key?
3. What is a public key and address?
4. Address with check-amount
5. What is a transaction?
6. Transaction Fee. What are Gas and GasPrice?
7. What is a nonce?
8. What is a block?
9. Tasks

Important:

I suggest you watch the video in which my colleague talks about the blockchain: https://www.youtube.com/watch?v=5tY56DkVPsI

Also, read any Ethereum material you like. For example here:https://ethereum.org/ether
You can skip the technical details.

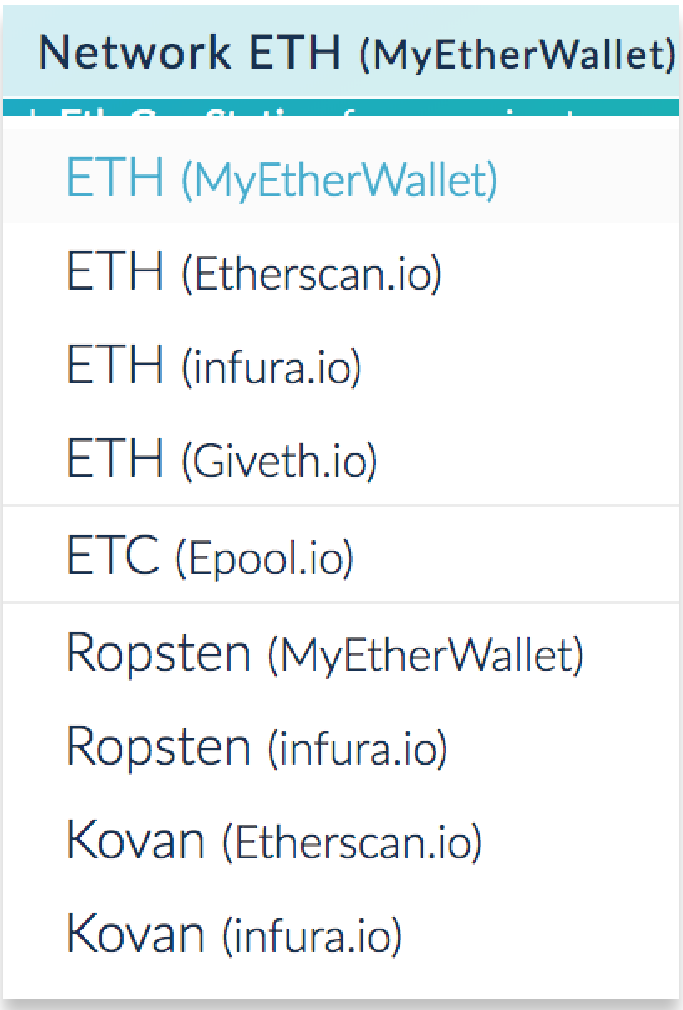
1. Creating a wallet, receiving and sending ether

ViaMyEtherWallet we are creating an Ethereum wallet to store the etheric currency ETH.

MyEtherWallet Is the default ether wallet that runs in the browser.

During the course, we will not be working on the main Ethereum network, but on the Kovan testnet. Transactions on this network are not paid for with regular ether, but with Kovan Ether. - test currency. This way we will not spend real money on deploying smart contracts and will be able to learn as much as necessary.

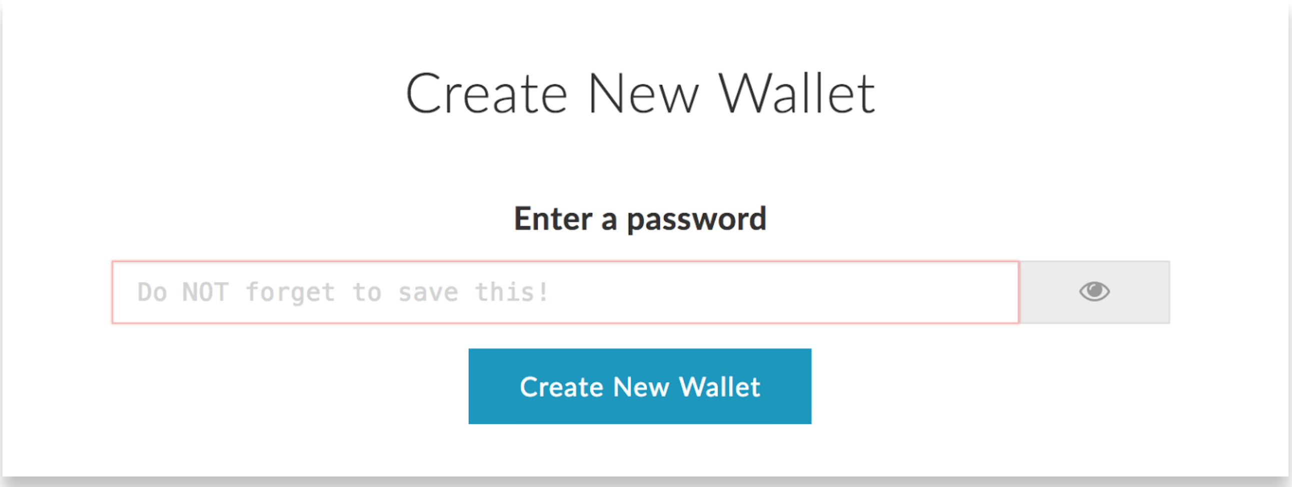
To change the network, on the site MyEtherWallet, in the upper right corner, select Network Kovan (Etherscan.io).



1. Creating a wallet, receiving and sending ether

It's very easy to create a wallet:

1. Enter your password;
2. Save the container with information about your wallet;
3. Save your private key.



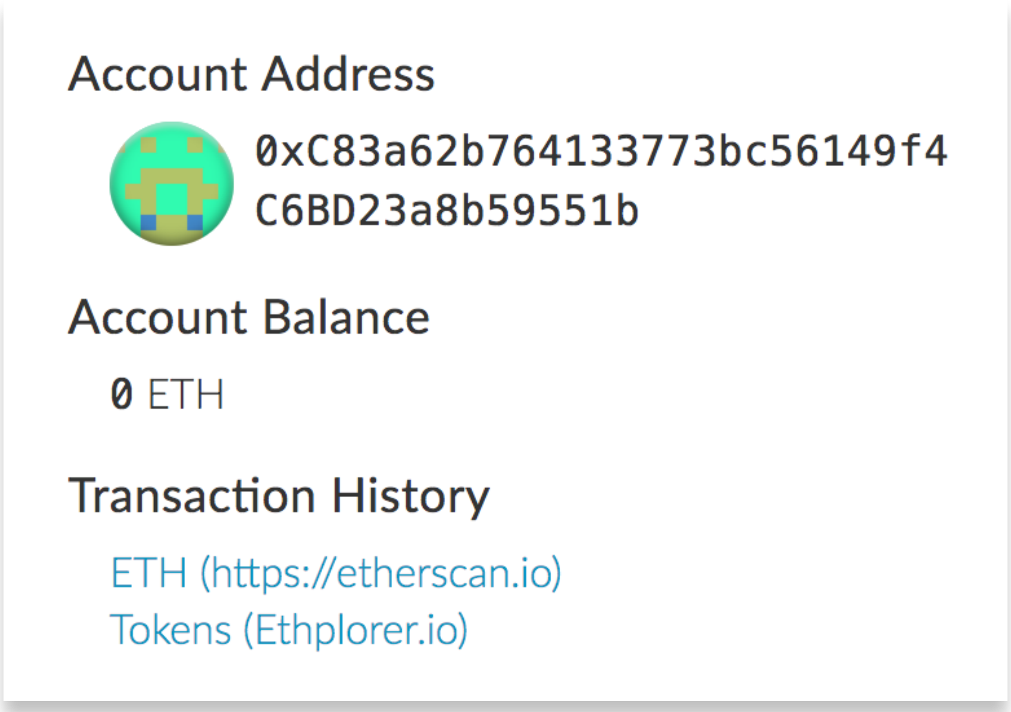
Create New Wallet

Enter a password

Do NOT forget to save this!

Create New Wallet

Done. Now you can see your wallet address, balance and other details.



Account Address

0xC83a62b764133773bc56149f4C6BD23a8b59551b

Account Balance

0 ETH

Transaction History

ETH (https://etherscan.io)

Tokens (Ethplorer.io)

2. What is a private key?

The private key gives you access to a wallet - a specific money account on the Ethereum network. In the case of ether,the private key is 32 bytes of information.

MyEtherWallet allows you to securely generate private keys. The key is created using a random number generator, immediately encrypted and placed in a special container - a text document in which the data about the wallet is written. Only those who know the password that you enter when creating the wallet can get the key from the container. If you have a strong password, no one will be able to get your key from the container, even if it has been compromised.

3. What is a public key and address?

Each private key has a corresponding public key. It is generated by hashing the private key. The public key is the wallet address, the account identifier in the Ethereum blockchain. You share the public key with other network participants so that they can interact with you, for example, send you money or include you in the list of trusted addresses for a contract.

With this public key, you can access the corresponding wallet address from any device.

More about the public key: https://en.wikipedia.org/wiki/Public-key_cryptography

Elliptic curve cryptography is used for encryption. You can read more about it here: https://habrahabr.ru/post/188958

4. Address with check-amount

To add a check-sum to the address, you need to specify some alphabetic characters of the address in upper case (upper case). By these symbols, you can understand if there is a typo in the address.

More details: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-55.md

Or on the wiki: https://en.wikipedia.org/wiki/Checksum

5. What is a transaction?

A transaction is a message sent by one of the accounts (wallets) and signed with a private key that corresponds to this account (wallet). If you signed a message, no one can fake your signature.

and all recipients of the message can be sure that it was you who signed it.

Creating wallets, exchanging currency between them are the simplest transactions that can be performed on the network. The Ethereum blockchain also allows for more complex transactions related to smart contracts - you can deploy contracts, interact with them. For example, send money to contracts, view public data of a contract, change data values in this contract, use its functions in your contracts.

Each transaction has many characteristics, the details of which can be viewed at etherscan.io:

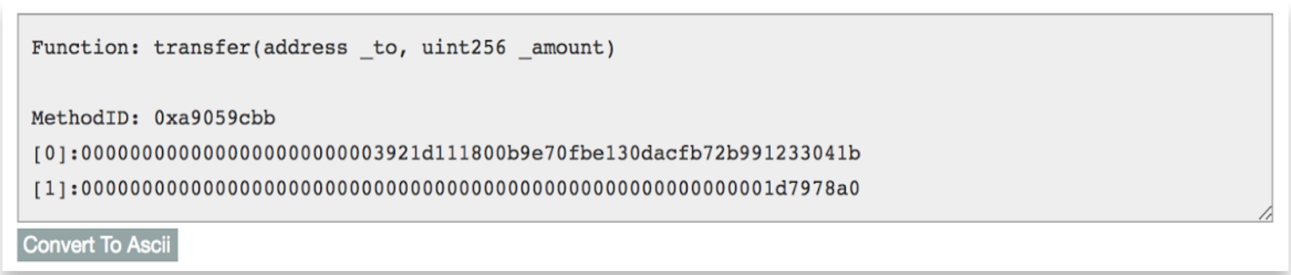
TxHash: A transaction hashed using the keccak256 algorithm.
TxReceipt Status: Transaction status - whether it was confirmed by the network or not.
Block Height: The number of blocks that confirmed the transaction.
TimeStamp: Time when the transaction was mined = Block TimeStamp, the time when the block containing the transaction was found.
From: Sender's address.
To: The recipient's address (in this case, the contract).
Value: The amount of money (ETH) sent with the transaction. This value can be zero.

Gas Limit: Gas quantity[one], which the sender was willing to spend.
Gas Used By Txn: The amount of gas that was spent per transaction. In this case, this number is less than the Gas Limit. This means that the sender, just in case, “poured” gas with a margin. The remains were returned to him.
Gas Price: The price in ETH that the sender is willing to pay for a unit of gas. If we have a gas limit of 250,000, and the price of one unit of gas is 4 Gwei, then the sender was ready to pay 1 million Gwei or 0.001 ETH for the transaction.[2...]
Actual Tx Cost / Fee: The actual price of the transaction, taking into account the amount of gas used and the price set for it
Actual Tx Cost = Gas Used By Txn * Gas Price
Cumulative Gas Used: The amount of gas that was spent on all previous transactions in the same block.
Nonce: Transaction sequence number[3...]

one - More in paragraph # 5
2 - 1 Gwei = 0.000000001 ETH
3 - More in paragraph 6

5. What is a transaction?

Input Data: When we send money from one wallet to another, this field remains blank. When we interact with the contract and pass some values (money, name, numbers, etc.) into it, then we can see their hexadecimal equivalents in this field.



```
Function: transfer(address _to, uint256 _amount)

MethodID: 0xa9059cbb
[0]:0000000000000000000000003921d111800b9e70fbe130dacfb72b991233041b
[1]:000000000000000000000000000000000000000000000000000000001d7978a0
```

Etherscan does not show that there is still a field **chainID** - blockchain network identifier. For example, 42 is the Kovan network ID. If we try to take a transaction with this ID and execute it on another network, then nothing will come of it, because the chainID for the Ethereum network is a unit.
The ChainID can be seen when you sign the transaction.

6. Transaction Fee. What is Gas and what is GasPrice?

The main purpose of the Ethereum network is transaction processing. Each of them requires a certain amount of resources, such as CPU time and storage space, so you need to pay a commission and use it to reward the executors.

To understand what Gas is, consider a simple example.

Imagine that there is a market that combines money delivery services (ether). In this market, you can order the transfer of money (ether), but you have to pay for it with the same money. You publicly announce the task: to deliver 1000 units of money (ether) to such and such an address. The distance between you and the specified address is 10 km. You say that you are ready to pay 2 units of money for every kilometer. It turns out that in this case Gas = 10 (you need to drive 10 km), GasPrice = 2 (for each kilometer you pay 2 units of money). In total, the shipping fee will be 20 units of money. To order the completion of this task, you need to have 1020 money in your account (send 1000 units and pay 20 units for delivery). Further, the performers (services) decide for themselves whether they want to take on such a task for such payment.

Suppose some performer agreed, but during the execution it turned out that the distance was only 9 km. He will return the payment for the 10th kilometer, which did not need to be driven, to the customer.

Or vice versa, if the distance turned out to be 15 km, and not 10 km, then the contractor takes the payment for the delivery (20 units of money), and the customer returns the amount that he could not deliver (1000 units).

The sender pays for the execution of the transaction. When he creates a transaction, he determines how many resources are needed to execute it (Gas) and how much ether he is willing to pay for a unit of gas (gasPrice). For example, the sender knows that it will take 30,000 gas to complete his transaction, and he is willing to pay 0.00000002 ETH (gasPrice) for each unit of gas.

6. Transaction Fee. What is Gas and what is GasPrice?

Transaction price

gas * gasPrice = 30000 * 0.00000002 ETH = 0.0006 ETH.

If as a result of the transaction, only 21000 gas was used, then the sender will receive back 9000 * 0.00000002 ETH.

Thus, the sender can 'order' the execution of the transaction for a specified price. Potential executors of the transaction (miners) see its price and decide for themselves whether or not to agree to its execution. If they are ready to sell their resources, the computing power of their computers, for such a price, then after the transaction is completed, they will receive the specified payment.

It is important to set the correct gas price for the transaction to proceed quickly.

Information on the state of the Ethereum network is available on the Etherscan website. You can view the transactions that are in the queue or the completed transactions.

etherscan.io => Blockchain => View Pending Transactions

You can sort Pending Transactions by gas price and simply put a slightly higher price in your transaction, then it will be the first in line.

If you want your transaction to complete faster, it is better to increase the gas price rather than the amount of gas. Each block on the Ethereum network has a limited size. All transactions included in the block must not exceed the total gas limit. The gas limit in the block is constantly different. Now, it is about 8 million of gas, but over time, this amount is growing slightly.

If your transaction contains a very large amount of gas, then it will not be immediately taken to the block.

7. What is a nonce?

Nonce was already discussed when we looked at the details of the transactions. Now let's figure out why it is needed at all.

All transactions are performed in a strict order. Nobody will execute (mine) your transaction with serial number 2 earlier than transaction with number 1.

Why is it important?

Let's say in the first transaction you register, and in the second you send money. These transactions can only go in this order, otherwise there will be no logic in them.

That it was impossible to use the same transaction several times. For example, you have given permission to one address to withdraw 100 tokens from you. This should be a one-time transaction, because you do not want 100 tokens to be withdrawn from you several times.

Every time we change any data in a transaction, its signature (hash) changes. If we send the same amount of money to the same address for the second time, the electronic signature must be different. This is helped by the nonce - the sequence number of the transaction for a specific sender.

The Ethereum network does not accept repeated transactions with the same signature. Nonce changes every time, respectively, the signature changes every time. Thus, we control everything that happens with our wallet.

If you send a transaction with nonce 5, when there are still no transactions with nonce 3 and 4, then it will hang in the queue and will not be executed until transactions with nonce 3 and 4 are created and executed.

If we send two different transactions with the same nonce, but with different data, then we cannot be sure which one will be executed (smile) first. But after one of them smiles, the second will not work, because it has the same nonce that is already inscribed in the blockchain. Such transactions are deleted automatically.

8. What is a block?

A block is a unit of change in the state of a blockchain. The state of the network is changed by blocks of transactions.

Let's imagine that you have a new network in which so far there are no wallets and no transactions. Let's say your network holds 8 million gas in a block. This means that you can include such a number of transactions in a block so that their total cost in gas does not exceed 8 million.

Let's say in your network there are 2 addresses with money in your account: 'A' and 'B'. The block contains the transaction of transferring all the money from address 'A' to 'B'. Now the state of your network is two addresses: 'A' without money, and 'B' with money.

9. Task number 1

We are starting to learn to communicate with the smart contract on the Kovan on-air testnet. To do this, you need:

1. Create yourself an Ethereum wallet usingMyEtherWallet
2. Tell me the address of your wallet in Telegram. My username is @lastperson
3. I will send some ether from the Kovan testnet (KEther, kether, KETH) to your wallet address
4. You will need to return 0.005 ether to the same address from which it will come.

I wrote and deployed a small contract into the Kovan test blockchain (you can see the address, source code, and ABIfour): https://kovan.etherscan.io/address/0xc6d68c96964839215694d85ff63e05690e02fc64#code

Attention!

Kovan is written in the domain before Etherscan, which means that Ethereum Block Explorer will take information from this testnet. We will use the Kovan testnet because the air is free there.
You can read information from the contract directly from the explorer: https: // kovan. etherscan.io/address/0xc6d68c96964839215694d85ff63e05690e02fc64#readContract

It allows you to call read-only functions from a contract in real time. In order to interact with the contract (and not just read it), we need to send transactions to it, which will call functions that change its state.

There is only one such function in this contract - **function vote (string _answer).**

Instructions for calling this function using MEW:

1. First you need to switch to the Kovan network (in the upper right corner);
2. Select the Contracts tab;
3. Enter the address of the contract https://kovan.etherscan.io/address/0xc6d68c96964839215694d85ff63e05690e02f-c64 # code
4. Copy the ABI of the contract from Etherscan and paste it into the ABI field;
5. Press Access - and forward. You can read read-only functions and send transactions to call write-functions.

four - ABI is a description of the interface to the contract. It contains the names of the functions, the parameters they accept (read or write), information about what they return, and information about whether they can accept ether as input.

9. Task number 2

You need to vote on the poll. You can read about what kind of survey this is and what answer options are available in the source code of the contract, or using read-only functions.

I sent 0.05 KETH (Kovan ETH) to your addresses to have something to pay for gas (put gasPrice = 1 gwei).

Ask questions in our Telegram chat!