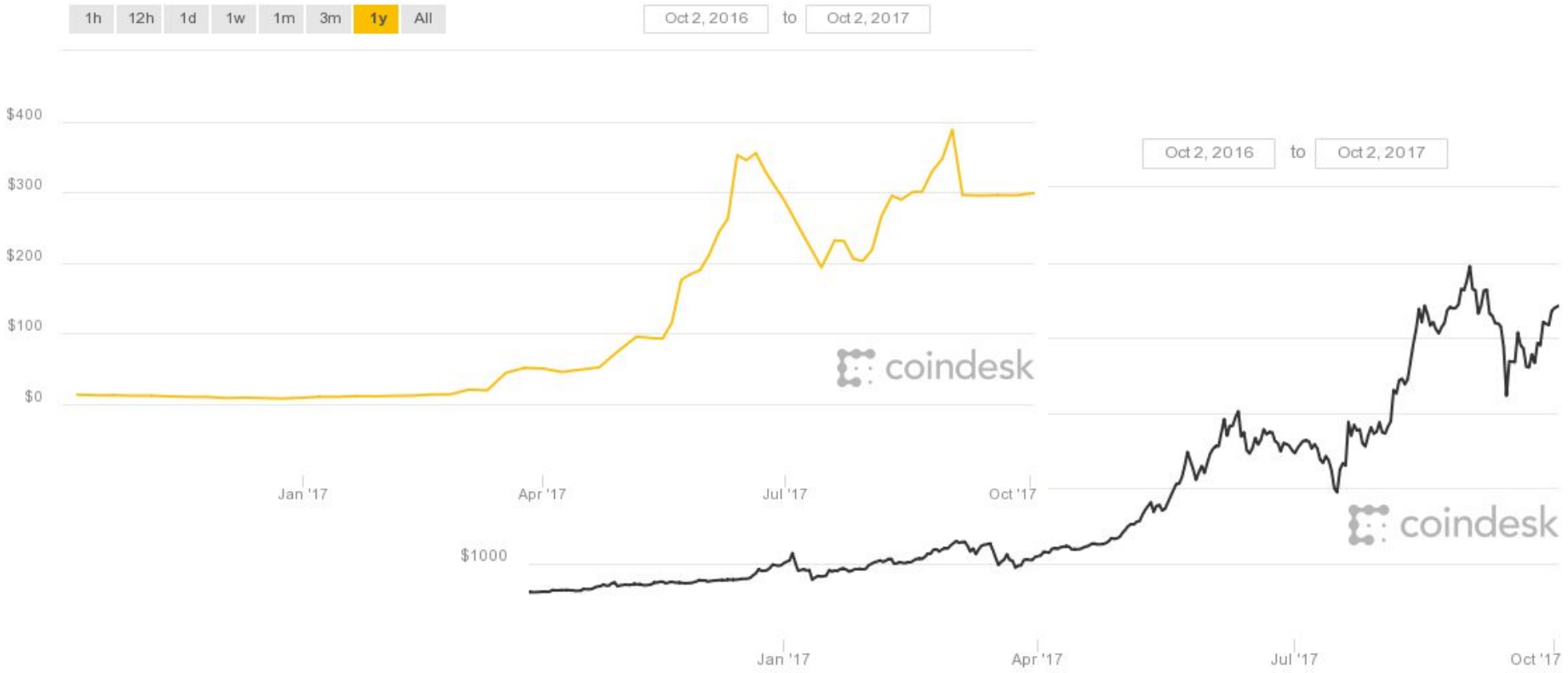
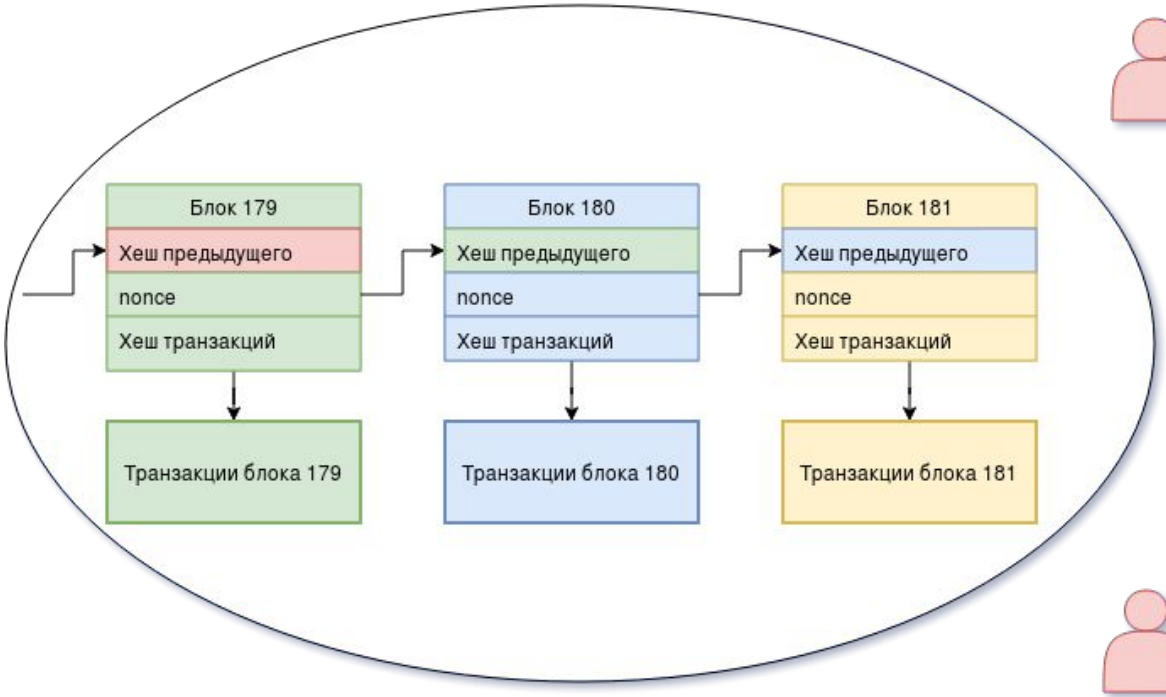
Разработка на Ethereum

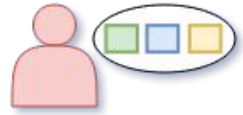
Евгений Пашенцев

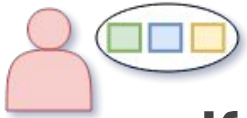
Блокчейн на волне



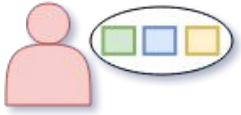
Блокчейн: что такое в принципе

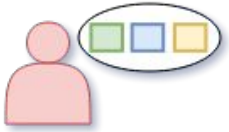




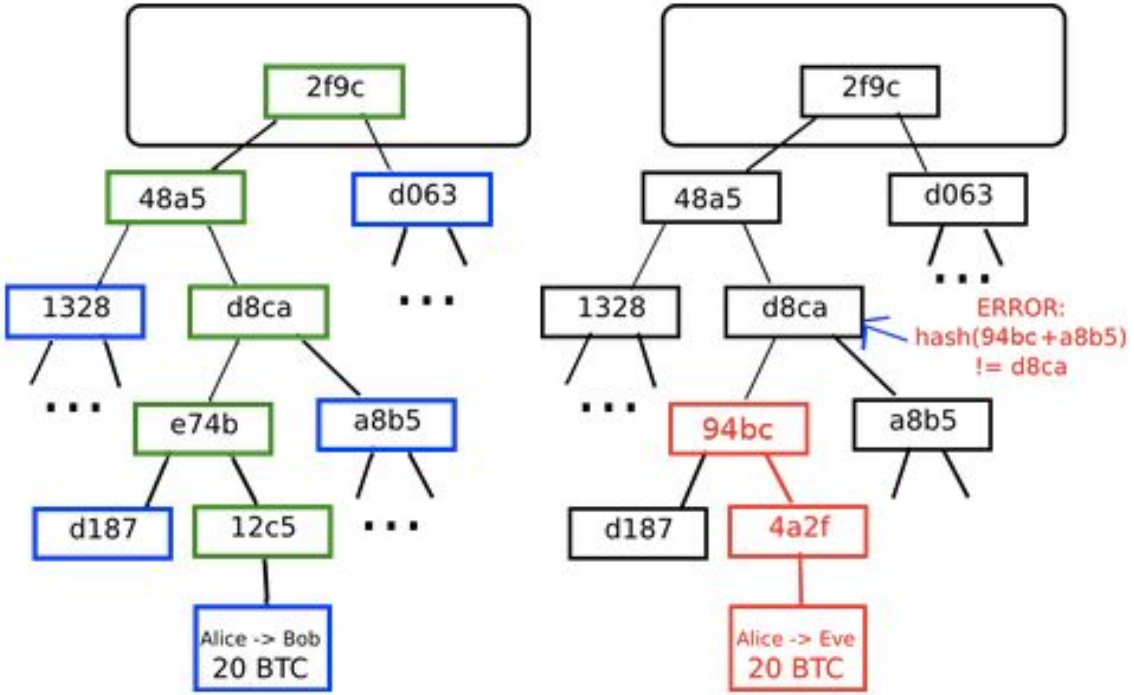


Копии у всех!





Транзакции: подтверждение валидности



Валидация конкретной транзакции без обхода всего дерева!

Ethereum

С чем нужно разобраться

- На чем программировать?
- Как выводить код в продакшн?
- Как обращаться к внешним источникам?
- Где хранить данные?
- Как делать фронтенд?

Ethereum: смарт-контракты на Solidity

```
contract SimpleStorage {
    uint storedData;

    function set(uint x) {
        storedData = x;
    }

    function get() constant returns (uint retVal) {
        return storedData;
    }
}
```

Смарт-контракт внутри блокчейна

ABI: Application Binary Interface


```
"abi": [
  {
    "constant": false,
    "inputs": [
      {
        "name": "addr",
        "type": "address"
      }
    ],
    "name": "getBalanceInEth",
    "outputs": [
      {
        "name": "",
        "type": "uint256"
      }
    ],
    "payable": false,
    "type": "function"
  },
  {
    "constant": false,
    "inputs": [
      {
        "name": "receiver",
        "type": "address"
      },
      {
        "name": "amount",
        "type": "uint256"
```

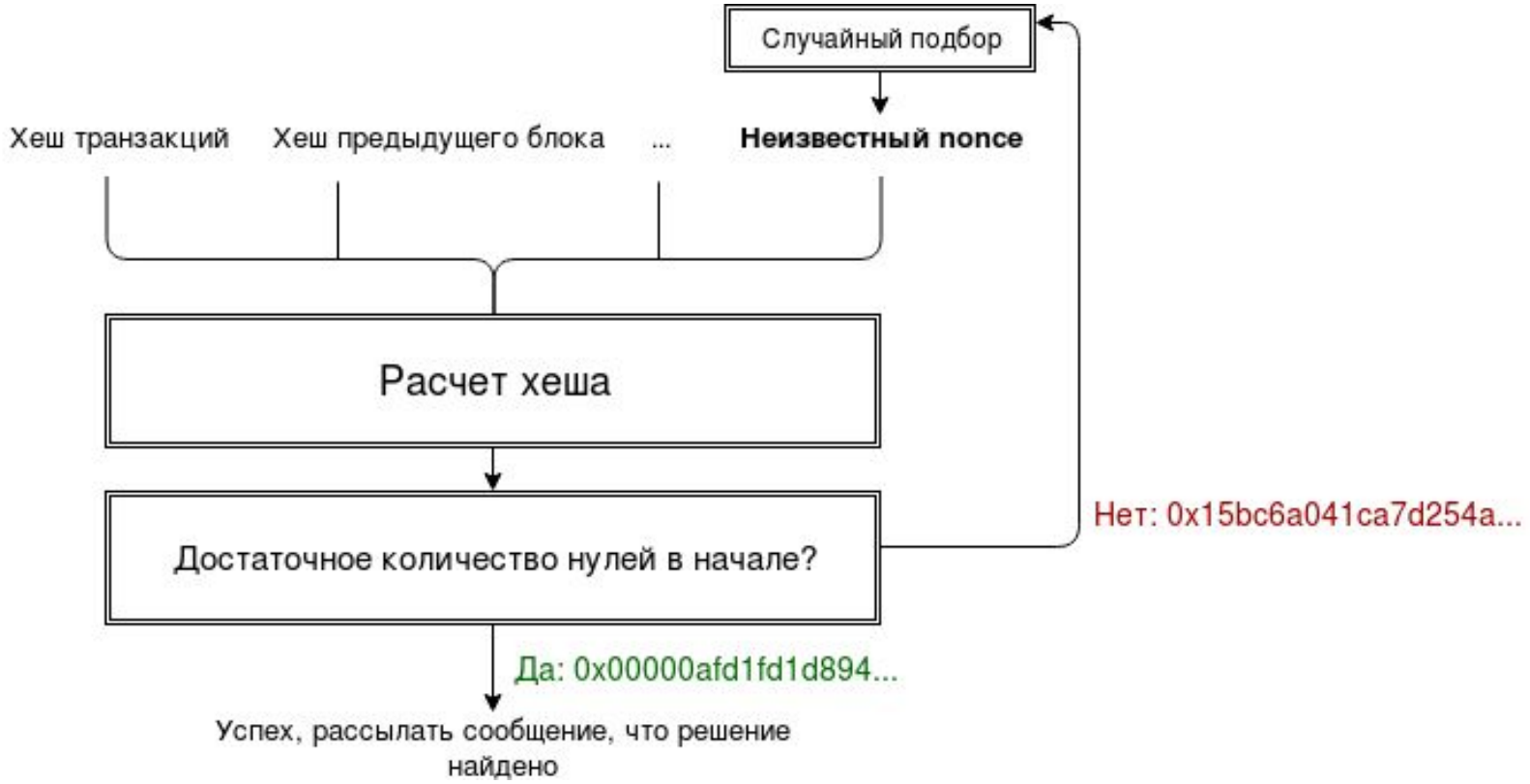

Бинарный код


```
"0x6003805460a060020a60ff02191690556 0a0604052600c60608190527f4a696e636f7220546f6b656e0000000000000000000000
0556200011391168264010000000006200011c810262000632170456 5b505b50620002db565b600354600090336001 60a060020a03
cde1e73087d944c0ea20544137d412139688592918290030190a25060015b5b5b92915050565b6000828201838110156200022657
3006060604052361561 00b45763ffffffff60e060020a60003504166305d2035b81146100b657806306fdde03146100da57806309
3015283519192839290830191850190808383821561013057 5b8051825260208311156101305760 1f1990920191602091820191 01
190036020019 0f35b341561020057fe5b6101ad61062c565b604080519182525190819003602001 90f35b341561022257fe5b6100
1528351919283929083019185019080838382156101305 75b8051825260208311156101305760 1f1990920191602091820191 0161
35460a060020a900460ff1681565b60048054604080516020600260001851615610100026000190190941693909304601f81018490
6808452948252918290208690558151868152915 17f8c5be1e5ebec7d5bd14f71427d1e84f3dd0314c0f7b2291e5b200ac8c7c3
20a038087166000818152600260209081526040808320338616845282529182902094909455805187815290519288169391926 0
05191927f0f6798a560793a54c3bcfe86a93cde1e73087d944c0ea20544137d412139688592918290030190a25060015b5b5b9291
190925281815292918301828280156104765780601f1061044b576101008083540402835291602001916104765 65b820191906000
98151915292918290030190a35060015b92915050565b600160a060020a038083166000908152600260209081526040808320938 5
879710e4e54f038b4abb53556eacdb4410029",
```

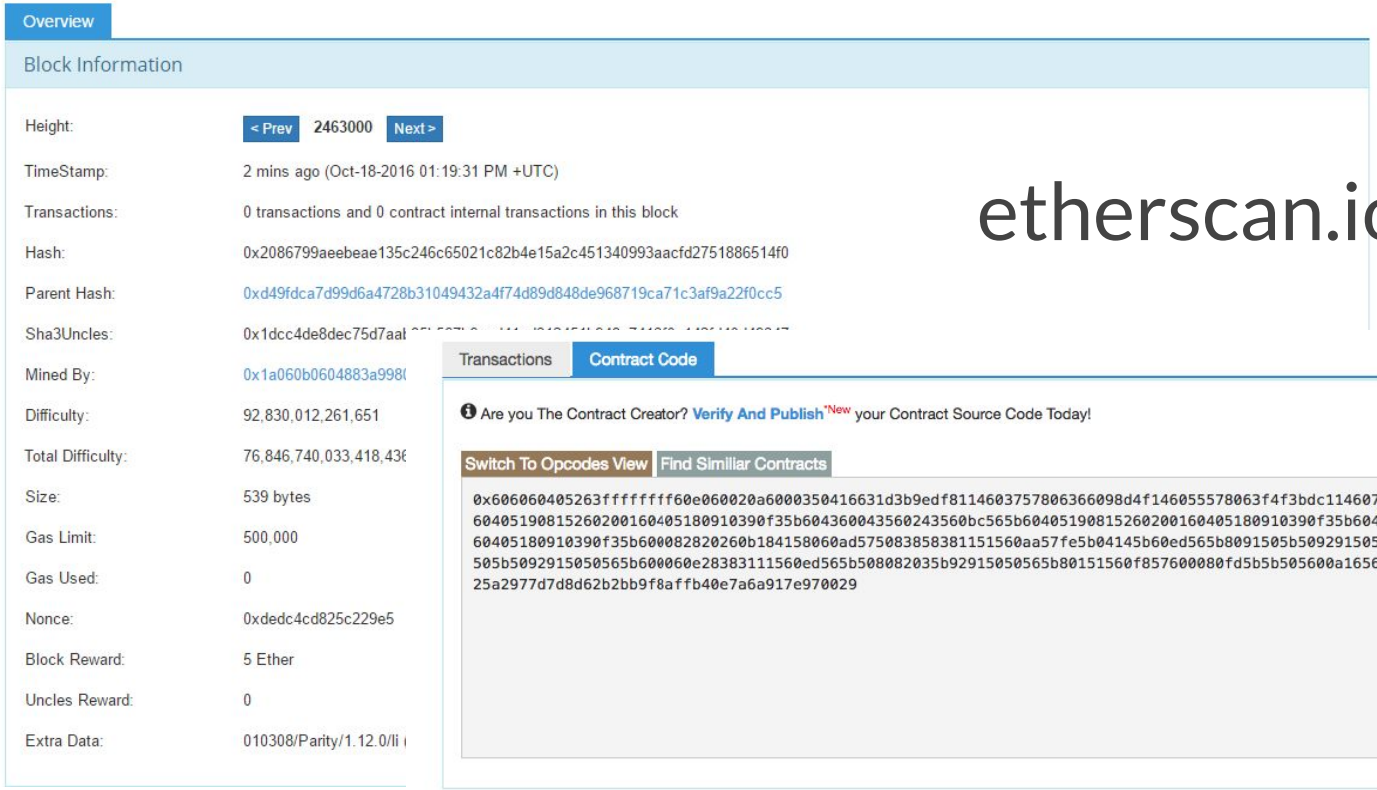

Адрес

"0xd4ec763c2f4a0b2403296e183eae2ecd6b8eb74c"

Кто такие майнеры

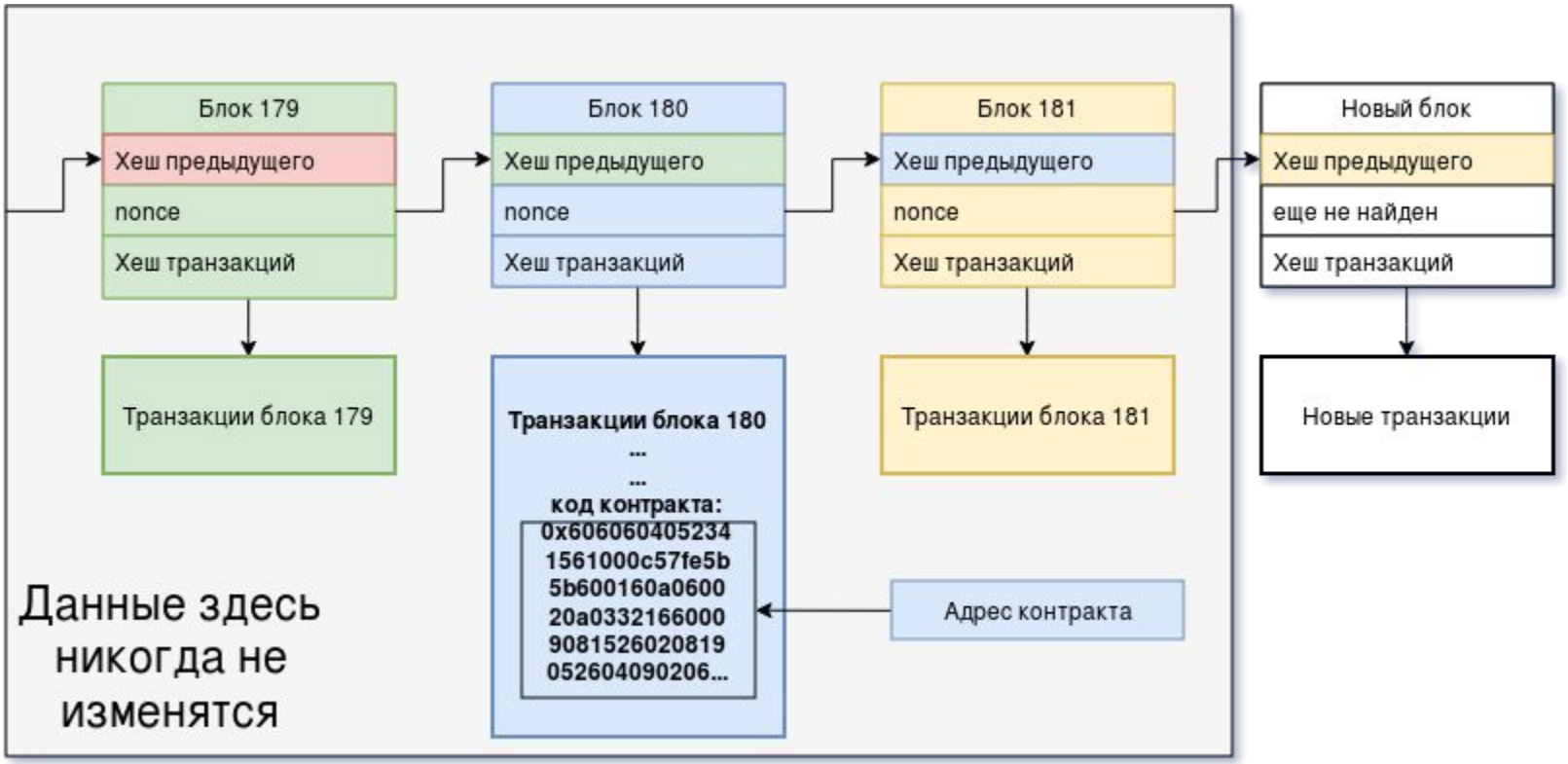


Гарантии смарт-контрактов



etherscan.io

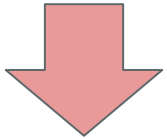
Деплой - это навсегда



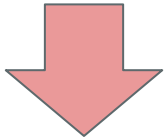
Деплой - это навсегда?

```
contract InternalLib
{
  function importantWorkFromLib();
}

contract TestTest
{
  address libAddress;

  function switchLibrary( address _address ) {
    libAddress = _address;
  }

  function importantWork() {
    InternalLib lib = InternalLib( libAddress );
    lib.importantWorkFromLib();
  }
}
```

Деплой новой библиотеки с новым адресом 0xabcd



TestTest.switchLibrary(0xabcd)



TestTest.importantWork() теперь работает по-другому!

Разработка

Разработка: основные моменты

- Медленный и перманентный деплой
- Контракты связаны друг с другом
- Последствия ошибок страшны
- Выполнение кода платное

testrpc

Одна команда: testrpc и все работает!

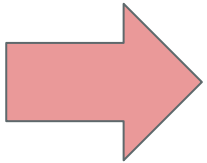
```
EthereumJS TestRPC v4.0.1 (ganache-core: 1.0.1)

Available Accounts
==================
(0) 0x7a2326344dde4f910429da5f396c849d795cc2d6
(1) 0x9cbe45006b30641083b9ca40341d760e1b1863f9
(2) 0x0eeeed5a820a31c3c385079bb5c4905bec9654d7
(3) 0xd46b70c2496d0149b3bccaa1086ba26f4c5515a7
(4) 0x9c706a33cae7bda432d329a987c4d52de159cc7c
(5) 0x6475386221e8f072f3273d83f849e582fa016098
(6) 0xdc42710a8326adcaf7443e25e4915caf7bb43871
(7) 0x58cb90622dbc1bddcca529a28d035eca53681309
(8) 0xbf8af0874afab9f7c0af4bff009133c4805b1c4e
(9) 0xf50356c42bf628ae995e7f47b8d7af19b8b7fcbe

Private Keys
==================
(0) 801ab74ef24c74b644c8afb53ccc20751a3b2c10db5
(1) 0b1adac68b0f4750189b6303884bafbc2553bfb05f0
(2) d09db75ef1f276f965d80ce849e8911c0d3e815a5e4
(3) 2e846d37bb9a4e34a9febd0c6d0de8615c2f530ea32
(4) c740afb72583f1f278184e1deb9c5afe7f44a433b46
(5) 515fb8bfce5cdaf8f3f1e6232428bab6bfe0faf08dc
(6) 109e097e7d37323e2cf504e1a640ef8c237f196fce5e119786959cb8698ea1b4
(7) 246b2c3ce44757bf472a9ccf144c0161ba3ceeb2b663ac4d9b08f496e9519d5e
(8) d970c44fabcd0225c1bc8966158074a91c39871122657bdd6bb76f346cad4ced
(9) 555c67356aae02d85e5de373f9d8d756468bd360ef9abea658d4cf58ce5a47ff

HD Wallet
==================
Mnemonic:      mom dragon neutral social bulb cement knife collect hundred rain
Base HD Path:  m/44'/60'/0'/0/{account_index}

Listening on localhost:8545
net_version
eth_accounts
eth_accounts
net_version
net_version
eth_sendTransaction

  Transaction: 0x5455806c20c182ab8040cf049c27d9d1dde67ec5b0a5448ccf029e6514882bc
  Contract created: 0xc66cad6ec3f027619bd668993900ec4973e425d3
  Gas usage: 186708
  Block Number: 1
  Block Time: Thu Oct 05 2017 14:00:49 GMT+0300 (MSK)

eth_newBlockFilter
eth_getFilterChanges
```

Truffle: компиляция

$ truffle init

$ truffle compile

```
Compiling ./contracts/ConvertLib.sol...
Compiling ./contracts/MetaCoin.sol...
Compiling ./contracts/Migrations.sol...
Writing artifacts to ./build/contracts
```



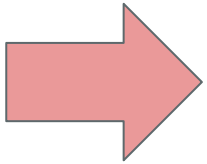
JSON файлы


```
{
  "contract_name": "MetaCoin",
  "abi": [
    {
      "constant": false,
      "inputs": [
        {
          "name": "addr",
          "type": "address"
      "type": "constructor"
    }
  ],
  "unlinked_binary":
  "0x6060604052341561000c57fe5b5b60008054600160a060020a03191633
  152565b005b6000805433600160a060020a039081169116141561013757815
  "networks": {
    "3": {
      "events": {},
      "links": {},
      "address": "0xd4ec763c2f4a0b2403296e183eae2ecd6b8eb74c",
      "updated_at": 1506621175686
    }
  },
  "schema_version": "0.0.5",
  "updated_at": 1506621175686
}
```

Truffle: миграция

Файл миграции

```
var ConvertLib = artifacts.require("./ConvertLib.sol");
var MetaCoin = artifacts.require("./MetaCoin.sol");

module.exports = function(deployer) {
  deployer.deploy(ConvertLib);
  deployer.link(ConvertLib, MetaCoin);
  deployer.deploy(MetaCoin);
};
```



$ truffle migrate

```
Using network 'development'.

Running migration: 1_initial_migration.js
  Deploying Migrations...
  Migrations: 0xc66cad6ec3f027619bd668993900ec4973e425d3
Saving successful migration to network...
Saving artifacts...
Running migration: 2_deploy_contracts.js
  Deploying ConvertLib...
  ConvertLib: 0x965f37509c0166192f3a6db4d0b5aedb48f4b9de
  Linking ConvertLib to MetaCoin
  Deploying MetaCoin...
  MetaCoin: 0x09fbb9ab6f8fca8450a781808dcb56d5399c53bb
Saving successful migration to network...
Saving artifacts...
```

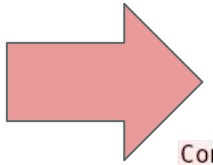
Truffle: тестирование

Solidity

```
import "truffle/Assert.sol";
import "truffle/DeployedAddresses.sol";
import "../contracts/MetaCoin.sol";

contract TestMetacoin {

  function testInitialBalanceUsingDeployedContract() {
    MetaCoin meta = MetaCoin(DeployedAddresses.MetaCoin());

    uint expected = 10000;

    Assert.equal(meta.getBalance(tx.origin), expected,
                 "Owner should have 10000 MetaCoin initially");
  }
```

JS

```
var MetaCoin = artifacts.require("./MetaCoin.sol");

contract('MetaCoin', function(accounts) {
  it("should put 10000 MetaCoin in the first account", function() {
    return MetaCoin.deployed().then(function(instance) {
      return instance.getBalance.call(accounts[0]);
    }).then(function(balance) {
      assert.equal(balance.valueOf(), 10000, "10000 wasn't in the first account");
    });
  });
```



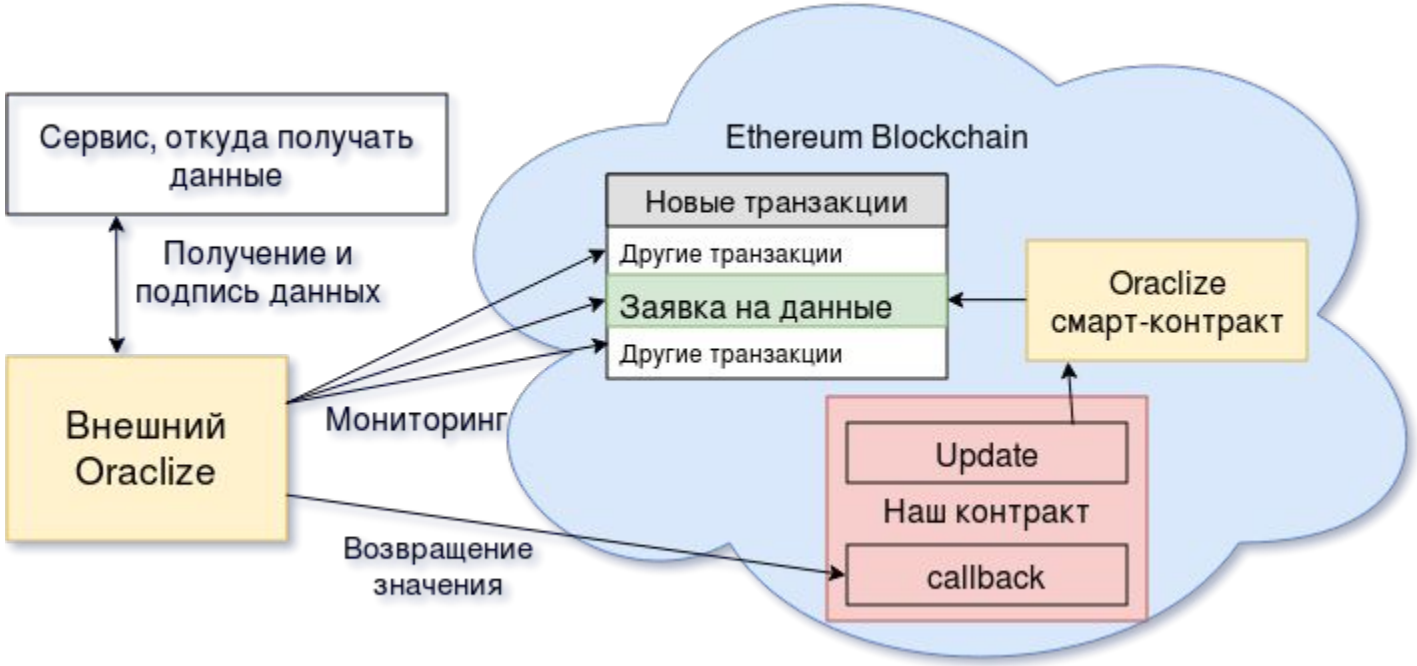
$ truffle test

```
Compiling truffle/Assert.sol...
Compiling truffle/DeployedAddresses.sol...


  TestMetacoin
    ✓ testInitialBalanceUsingDeployedContract (78ms)
    ✓ testInitialBalanceWithNewMetaCoin (53ms)

  Contract: MetaCoin
    ✓ should put 10000 MetaCoin in the first account (40ms)
    ✓ should call a function that depends on a linked library (73ms)
    ✓ should send coin correctly (125ms)


  5 passing (724ms)
```

Oraclize

```
import "github.com/oraclize/ethereum-api/oraclizeAPI.sol";

contract ExampleContract is usingOraclize {
    string public EURGBP;

    function ExampleContract() payable {
        updatePrice();
    }

    function __callback(bytes32 myid, string result) {
        if (msg.sender != oraclize_cbAddress()) throw;
        EURGBP = result;
    }

    function updatePrice() payable {
        if (oraclize_getPrice("URL") < this.balance) {
            oraclize_query("URL", "json(http://api.fixer.io/latest?symbols=USD,GBP).rates.GBP");
        }
    }
```

Oraclize: как это работает



IPFS

$ ipfs daemon &

$ echo “very important” > HelloWorld.txt

$ ipfs add HelloWorld.txt

```
added QmfYv4GXMgmQEEtrwiyQiWcut8agXiJmvaWkCu1fpyFAHz HelloWorld.txt
```

hash(“very important”) == QmfYv4GXMgmQEEtrwiyQiWcut8agXiJmvaWkCu1fpyFAHz

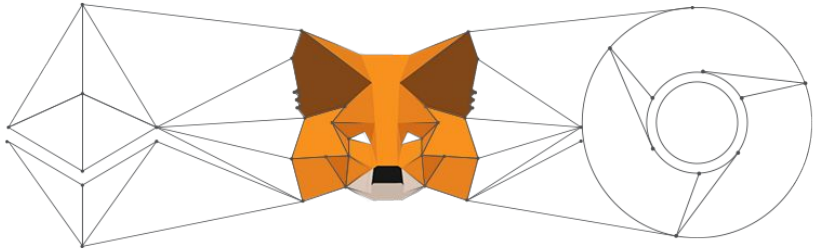
Получение из Solidity:

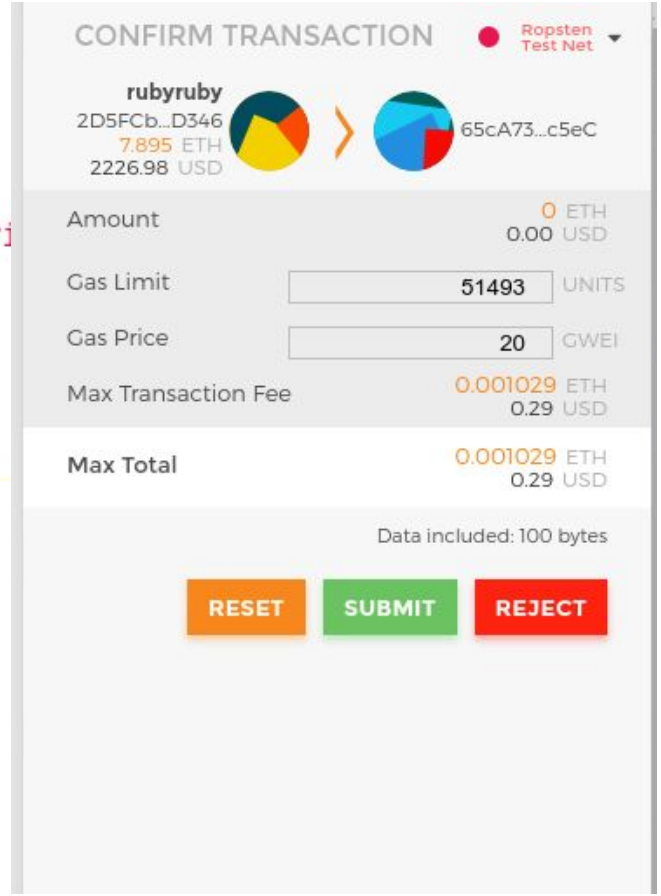
```
oraclize_query("IPFS", "QmfYv4GXMgmQEEtrwiyQiWcut8agXiJmvaWkCu1fpyFAHz");
```

Web3.js

```
var address = "0x65cA73D13a2cc1dB6B92fd04eb4EBE4cEB70c5eC";
var abi = [ { "constant": false, "inputs": [ { "name": "newStri
var contract = web3.eth.contract(abi);
var stringHolder = contract.at(address);

stringHolder.getString(console.log);
stringHolder.setString("Hello World!", console.log);
```

MetaMask





Что в итоге

- Смарт-контракты - **Solidity**
- Деплой и тестирование - **Truffle** и **testrpc**
- Общение с внешним миром - **Oraclize**
- Хранение ресурсов - **IPFS**
- Приложения - **Web3.js** + **MetaMask**

Спасибо за внимание!

Туториалы на Хабре	https://**habrahabr.ru**/users/**rubyruby**
Презентация	https://**github.com**/**rubyruby**/**rif2017**
Подписывайтесь	https://**facebook.com**/**rubyruby.digital**
Пишите	info@**rubyruby.ru**