

# Основы работы со смарт-контрактом

## Взаимодействие со смарт-контрактом

Взаимодействие со смарт-контрактом происходит посредством вызова функций контракта с указанием параметров. Для взаимодействия необходимо знать адрес смарт-контракта. А на основе адреса нужно получить ABI-интерфейс. По ABI интерфейсу можно вызвать целевую функцию.

## Получение ABI интерфейса

Для получения интерфейса вам необходимо:

1. открыть [etherscan.io](https://etherscan.io)
2. В правом верхнем углу в окне с надписью «search by address» ввести адрес целевого смарт-контракта и нажать «go»
3. Затем нажать на вкладку «contract source»:

Sponsored Link: [Buy your 1st Ethereum in 5 minutes.](#) Trusted by more than 100k buyers.

**Contract Overview**



ETH Balance: 0 Ether

ETH USD Value: \$0

No Of Transactions: 0 txn

**Misc**

More Options 

Address Watch [Add To Watch List](#)

Contract Creator [0x766a0518ce0d02...](#) at txn [0xf25f6da028e386ff...](#)

**Transactions**

Internal Transactions

Contract Source <sup>Yes</sup>

Read Smart Contract

Comments

TxHash	Age	From	To	Value	[TxFee]
--------	-----	------	----	-------	---------

There are no matching entries



4. Найти поле с надписью «Contract ABI» и нажать на кнопку «сору» справа от надписи

Интерфейс ABI контракта теперь скопирован в буффер обмена.

Transactions
Internal Transactions
**Contract Source Yes**
Read Smart Contract
Comments

**Contract Source Code Verified**

Contract Name:	ICO
Compiler Version:	v0.4.18   commit.9cf6c910
Optimization Enabled:	Yes
Runs (Optimiser):	200

Contract Source Code </>

CopyFind Similiar Contracts

```

9 contract Ownable {
10     address public owner;
11
12
13     event OwnershipTransferred(address indexed previousOwner, address indexed newOwner);
14
15
16     /**
17      * @dev The Ownable constructor sets the original `owner` of the contract to the sender
18      * account.
19      */
20     function Ownable() public {
21         owner = msg.sender;
22     }
23
24
25     /**
26      * @dev Throws if called by any account other than the owner.
27      */
28     modifier onlyOwner() {
29         require(msg.sender == owner);

```

Contract ABI

CopyExport ABI

```

[{"constant":false,"inputs":[{"name":"newToken","type":"address"}],"name":"setToken","outputs":[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":[],"name":"devTokensWallet","outputs":[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":false,"inputs":[{"name":"newAdvisorsTokensWallet","type":"address"}],"name":"setAdvisorsT


```

## Вызов функции

1. Откройте кошелек [myetherwallet.com](https://myetherwallet.com)
2. Выберите вкладку «контракты» и заполните поля **адрес** и **ABI** (мы его получали в предыдущей части инструкции) и нажмите на «подключиться»

DON'T GET PHISHED, please! 🙏 Thank you! 😊

1. BOOKMARK [MYETHERWALLET.COM](https://myetherwallet.com) 2. INSTALL [EAL](#) or [MetaMask](#) or [Cryptonite](#)


 **MyEtherWallet** 3.10.10.3 English ▾ Gas Price: 1 Gwei ▾ Network ETH (MyEtherWallet) ▾

The network is really full right now. Check [Eth Gas Station](#) for gas price to use.

[New Wallet](#) [Send Ether & Tokens](#) [Swap](#) [Send Offline](#) [Contracts](#) [ENS](#) [DomainSale](#) [Check TX Status](#) ▾ >>

## Interact with Contract or Deploy Contract

**Contract Address**

 **Select Existing Contract**  

Select a contract... ▾

**ABI / JSON Interface**  

```
[{"type": "constructor", "inputs": [{"name": "param1", "type": "uint256", "indexed": true}], "name": "Event"}, {"type": "function", "inputs": [{"name": "a", "type": "uint256"}], "name": "foo", "outputs": []}]
```

**Access**

3. После этого в нас станет доступен список функций контракта. Выбираем целевую функцию и заполняем параметры

## Interact with Contract or Deploy Contract

Contract Address

0x6E08A8B4Da2A13C655AF2Dd74f7Bf



Select Existing Contract

Select a contract...

ABI / JSON Interface

```
{
  "payable": true,
  "stateMutability": "payable",
  "type": "fallback",
  {
    "anonymous": false,
    "inputs": [
      {
        "indexed": true,
        "name": "previousOwner",
        "type": "address"
      },
      {
        "indexed": true,
        "name": "newOwner",
        "type": "address"
      }
    ],
    "name": "OwnershipTransferred",
    "type": "event"
  }
}
```

Access

Read / Write Contract

0x6E08A8B4Da2A13C655AF2Dd74f7BDb979efE141b

Select a function

setToken  
devTokensWallet  
setAdvisorsTokensWallet  
PERCENT\_RATE  
setFoundersTokensWallet



4. После выбора функции и заполнения параметров получите доступ к кошельку-владельцу смарт-контракта удобным вам способом и нажмите на кнопку «записать» и «Да, я уверен, выполнить транзакцию».

**Внимание!** Перед тем как нажать «записать» внимательно проверьте введенные данные. Часто формат данных отличается от привычного пользователю вида.

### Read / Write Contract

0x6E08A8B4Da2A13C655AF2Dd74f7BDdb979efE141b

setDirectMintAgent ▾

newDirectMintAgent address

0x314156...



#### How would you like to access your wallet?

- ☐ Metamask / Mist ?
- ☐ Ledger Wallet ?
- ☐ TREZOR ?
- ☐ Digital Bitbox ?
- ☒ Keystore / JSON File ?
- ☐ Mnemonic Phrase ?
- ☐ Private Key ?
- Parity Phrase ?

#### Select Your Wallet File

SELECT WALLET FILE...

WRITE

5. После этого внизу появится зеленая панель со ссылкой «Verify transaction». Нажмите на эту ссылку и откроется Etherscan. Ждите пока транзакция не выполнится.

## Как посмотреть интерфейс смарт-контракта

1. открыть [etherscan.io](https://etherscan.io)
2. В правом верхнем углу в окне с надписью «search by address» ввести адрес целевого смарт-контракта и нажать «go»
3. Появится окно в котором надо нажать на вкладку «Read smart contract»

Sponsored Link: **SHIFT.cash** - 1st ever car title loan platform on blockchain. [Join NOW](#).

#### Contract Overview



ETH Balance: 0 Ether

ETH USD Value: \$0

No Of Transactions: 0 txn

#### Misc

More Options ▾

Address Watch

[Add To Watch List](#)

Contract Creator

[0x766a0518ce0d02...](#) at txn [0xf25f6da028e386ff...](#)

#### Transactions

Internal Transactions

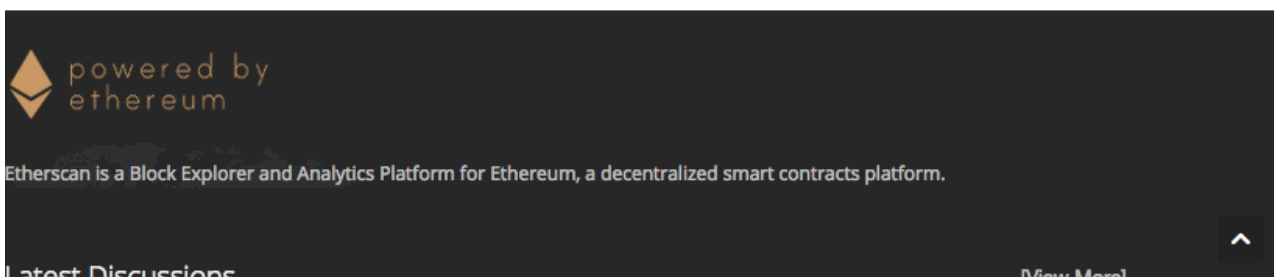
Contract Source <sup>Yes</sup>

[Read Smart Contract](#)

Comments

TxHash	Age	From	To	Value	[TxFee]
--------	-----	------	----	-------	---------

There are no matching entries



## Подготовка к работе — только если контракт заливали самостоятельно!

Адрес смарт-контракта конфигуратора:

**0x766a0518ce0d022247d55d068d0b23db0e5ca7e5**

Для начала работы со смарт-контрактами их необходимо проинициализировать. Для этого нужно у контракта-конфигуратора вызывать функцию `deploy`. Во время ее вызова MyEtherWallet может не определить количество газа, необходимое для вызова функции. И поэтому поставит «-1».

## Warning!



You are about to **execute a function on contract**.

It will be deployed on the following network: **ETH** (MyEtherWallet).

**Amount to Send** *In most cases you should leave this as 0.*

**Gas Limit**

**Generate Transaction**

Исправьте число на 4600000 и нажмите «Записать транзакцию» а затем «Да, я уверен! Выполнить транзакцию».

## Warning!



You are about to **execute a function on contract**.

It will be deployed on the following network: **ETH** (MyEtherWallet).

**Amount to Send** *In most cases you should leave this as 0.*

**Gas Limit**

**Generate Transaction**

**Raw Transaction**

```
{"nonce":"0x03","gasPrice":"0x3b9aca00","gasLimit":"0x4630c0","to":"0x766A0518ce0D02247d55D068D0B2"}
```

**Signed Transaction**


```
0xf86803843b9aca00834630c094766a0518ce0d02247d55d068d0b23db0e5ca7e58084775c300c25a066c0abe5ae93857
```

No, get me out of here!

Yes, I am sure! Make transaction.

После выполнения транзакции (вам нужно дождаться ее выполнения). Вы можете открыть интерфейс смарт-контракта и вы увидите 4 поля с ненулевыми адресами:



Transactions	Internal Transactions	Contract Source <small>Yes</small>	Read Smart Contract	Comments
<div>  Read Contract Information         <span>[Reset]</span> </div>				
1. > ico → 0x6e08a8b4da2a13c655af2dd74f7bdb979efe141b address				
2. > owner → 0x69f5c3850d1f1d5baeae71e947e915a539088bb0 address				
3. > token → 0x5160680fb520e41f49a8411dbd0592b730f5480e address				
4. > presale → 0x720608228bda773fe22e816ee72463040f11c19e address				

Важные для нас адреса смарт-контрактов:

- **Presale** — адрес Presale контракта. Именно на него инвесторы должны закидывать эфир во время Presale-распродажи. Все даты распродажи, а также цена и функция завершения распродажи вызываются по этому адресу.
- **ICO** - адрес ICO контракта. На него инвесторы должны закидывать эфир во время ICO-распродажи. Все даты распродажи, а также цена и функция завершения распродажи вызываются по этому адресу.
- **Token** — адрес смарт-контракта токена. Этот адрес пользователи будут добавлять себе в кошелек для отображения токена.

## Конфигурация

### Адреса кошельков:

#### Presale

ETH Wallet	0x72EcAEB966176c50CfFc0Db53E4A2D3DbC0d538B
Founders tokens wallet	0x4227859C5A9Bb4391Cc4735Aa655e980a3DD4380
Bounty tokens wallet	0x7E513B54e3a45B60d6f92c6CECE10C68977EEA8c
Advisors tokens wallet	0x6e740ef8618A7d822238F867c622373Df8B54a22
Developers tokens wallet	0xCaDca9387E12F55997F46870DA28F0af1626A6d4
Contract manager	0x69F5C3850D1f1d5BAeAe71E947e915A539088Bb0

#### ICO

ETH Wallet	0x72EcAEB966176c50CfFc0Db53E4A2D3DbC0d538B
Founders tokens wallet	0x4227859C5A9Bb4391Cc4735Aa655e980a3DD4380
Bounty tokens wallet	0x7E513B54e3a45B60d6f92c6CECE10C68977EEA8c
Advisors tokens wallet	0x6e740ef8618A7d822238F867c622373Df8B54a22
Developers tokens wallet	0xCaDca9387E12F55997F46870DA28F0af1626A6d4
Contract manager	0x69F5C3850D1f1d5BAeAe71E947e915A539088Bb0

## Адреса контрактов:

Токен [0x5160680fb520e41f49a8411dbd0592b730f5480e](#)

Presale [0x720608228bda773fe22e816ee72463040f11c19e](#)

ICO [0x6e08a8b4da2a13c655af2dd74f7bdb979efe141b](#)

## Рекомендованные значения для отправки транзакции инвестора

Обычно кошельки сами способны определить количество газа и цену, но не всегда. Поэтому вводятся рекомендованные значения:

1. Меньше чем **0.1 Eth** контракт не принимает.
2. Количество газа - **250 000**

## Памятка владельцу контрактов

Владелец контрактов должен после каждой распродажи вызвать функцию **finishMinting()** у соответствующего контракта распродажи. А именно:

1. По окончании распродажи у контракта Presale вызвать **finishMinting()**  
Адрес контракта: **0x720608228bda773fe22e816ee72463040f11c19e**
2. По окончании распродажи у контракта ICO вызвать **finishMinting()**  
Адрес контракта: **0x766a0518ce0d022247d55d068d0b23db0e5ca7e5**
3. Для вывода средств с контракта **Presale** нужно вызвать функцию **withdraw()**.  
Во время **ICO** средства выводятся автоматически.

Перед вызовом убедитесь, что адрес контракта соответствует текущей распродаже!

## Расположение адресов на странице контракта в etherscan.io

### Presale

Transactions	Internal Transactions	Contract Source <small>Yes</small>	Read Smart Contract	Comments
<div>Read Contract Information <span>[Reset]</span></div> <div><div>1. &gt; devTokensWallet → <a href="#">0xcadca9387e12f55997f46870da28f0af1626a6d4</a> address</div><div>2. &gt; balances <input type="text" value="&lt;input&gt; (address)"/> <input type="button" value="Query"/></div><div>uint256</div><div>3. &gt; PERCENT_RATE → 100 uint256</div></div>				

4. > devTokensPercent	→ 4	uint256
5. > foundersTokensWallet	→ 0x4227859c5a9bb4391cc4735aa655e980a3dd4380	address
6. > minted	→ 0	uint256
7. > wallet	→ 0x72ecaeb966176c50cfc0db53e4a2d3dbc0d538b	address
8. > refundOn	→ False	bool
9. > advisorsTokensPercent	→ 1	uint256
10. > directMintAgent	→ 0x00	address
11. > softcapAchieved	→ False	bool
12. > foundersTokensPercent	→ 10	uint256
13. > getBonus	→ 42	uint256
14. > owner	→ 0x69f5c3850d1f1d5baeae71e947e915a539088bb0	address
15. > devWallet	→ 0xea15adb66dc92a4bbccc8bf32fd25e2e86a2a770	address
16. > devLimit	→ 7000	uint256
17. > changesLocked	→ True	bool
18. > bountyTokensWallet	→ 0x7e513b54e3a45b60d6f92c6cece10c68977eea8c	address
19. > bountyTokensPercent	→ 3	uint256
20. > price	→ 100	uint256
21. > devWithdrawn	→ False	bool
22. > bonuses	<input type="text" value="&lt;input&gt; (uint256)"/>	<input type="button" value="Query"/>
... periodInDays uint256, bonus uint256		
23. > hardcap	→ 6300	uint256
24. > start	→ 1513774800	uint256
25. > invested	→ 0	uint256
26. > minInvestedLimit	→ 100	uint256
27. > nextSaleAgent	→ 0x6e08a8b4da2a13c655af2dd74f7bdb979efe141b	address
28. > bonusesCount	→ 3	uint256
29. > advisorsTokensWallet	→ 0x6e740ef8618a7d822238f867c622373df8b54a22	address
30. > end	→ 1516885200	uint256
31. > softcap	→ 200	uint256
32. > token	→ 0x5160680fb520e41f49a8411dbd0592b730f5480e	address