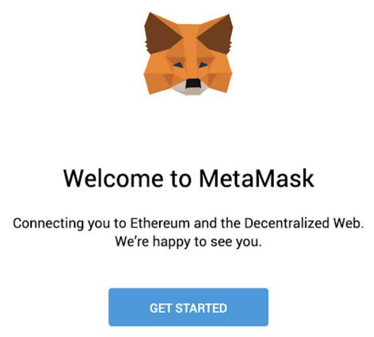
Install MetaMask

Metamask is an Ethereum Wallet that allows you to run Ethereum dApps right in your browser without running a full Ethereum node. To install MetaMask go to https://metamask.io/ and follow the installation instructions.

Create Account

After installing MetaMask a new tab will automatically open. It will guide you to create your first wallet.

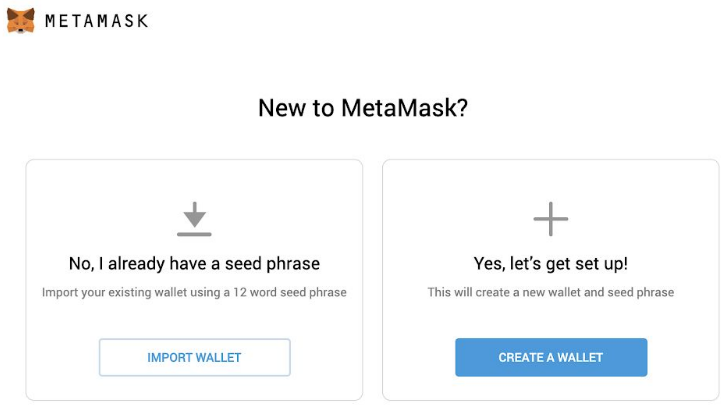
1. Click “Get Started”.



Welcome to MetaMask

Connecting you to Ethereum and the Decentralized Web.
We’re happy to see you.

2. Click “Create wallet”.



METAMASK

New to MetaMask?

No, I already have a seed phrase

Import your existing wallet using a 12 word seed phrase

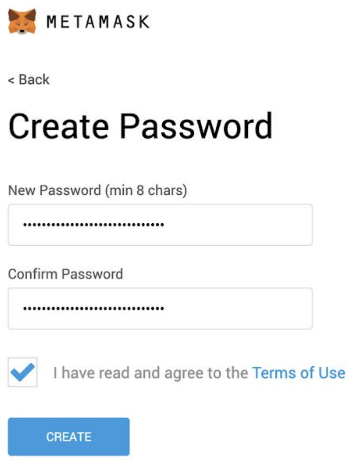
IMPORT WALLET

Yes, let's get set up!

This will create a new wallet and seed phrase

CREATE A WALLET

3. Choose a strong password.



METAMASK

< Back

Create Password

New Password (min 8 chars)

••••••••••••••••••••••••••••••

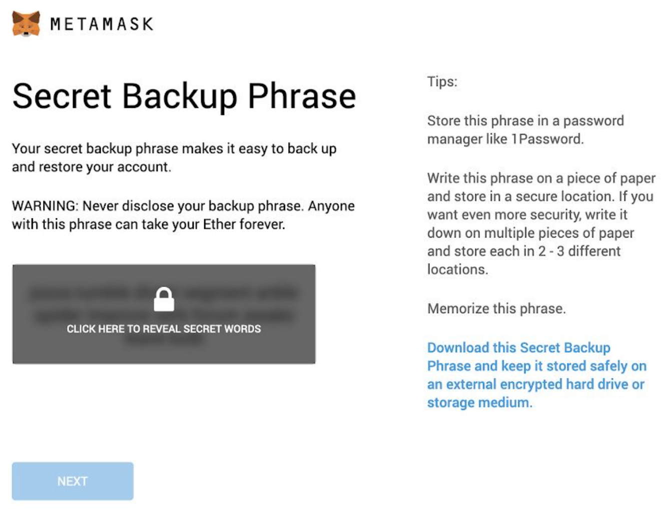
Confirm Password

••••••••••••••••••••••••••••••

✓ I have read and agree to the Terms of Use

CREATE

4. Next your seed phrase will be shown. Store it in a secure location. WARNING: If you lose your seed phrase you will not be able to recover any Ether you may have in this wallet.



METAMASK

Secret Backup Phrase

Your secret backup phrase makes it easy to back up and restore your account.

WARNING: Never disclose your backup phrase. Anyone with this phrase can take your Ether forever.

Tips:

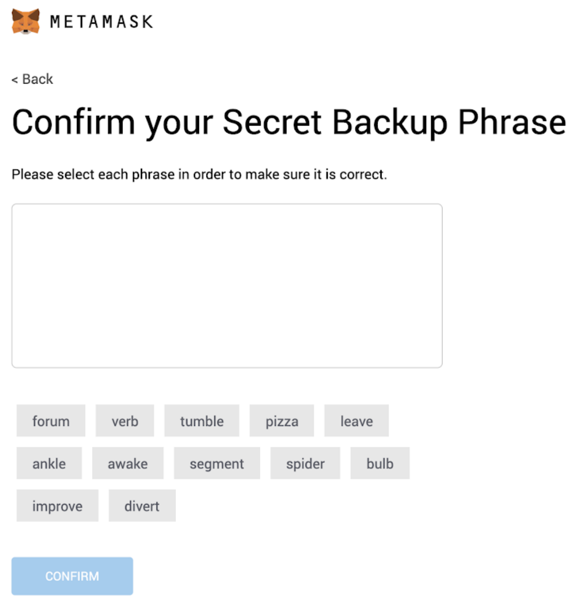
Store this phrase in a password manager like 1Password.

Write this phrase on a piece of paper and store in a secure location. If you want even more security, write it down on multiple pieces of paper and store each in 2 - 3 different locations.

Memorize this phrase.

Download this Secret Backup Phrase and keep it stored safely on an external encrypted hard drive or storage medium.

5. Confirm your seed phrase.



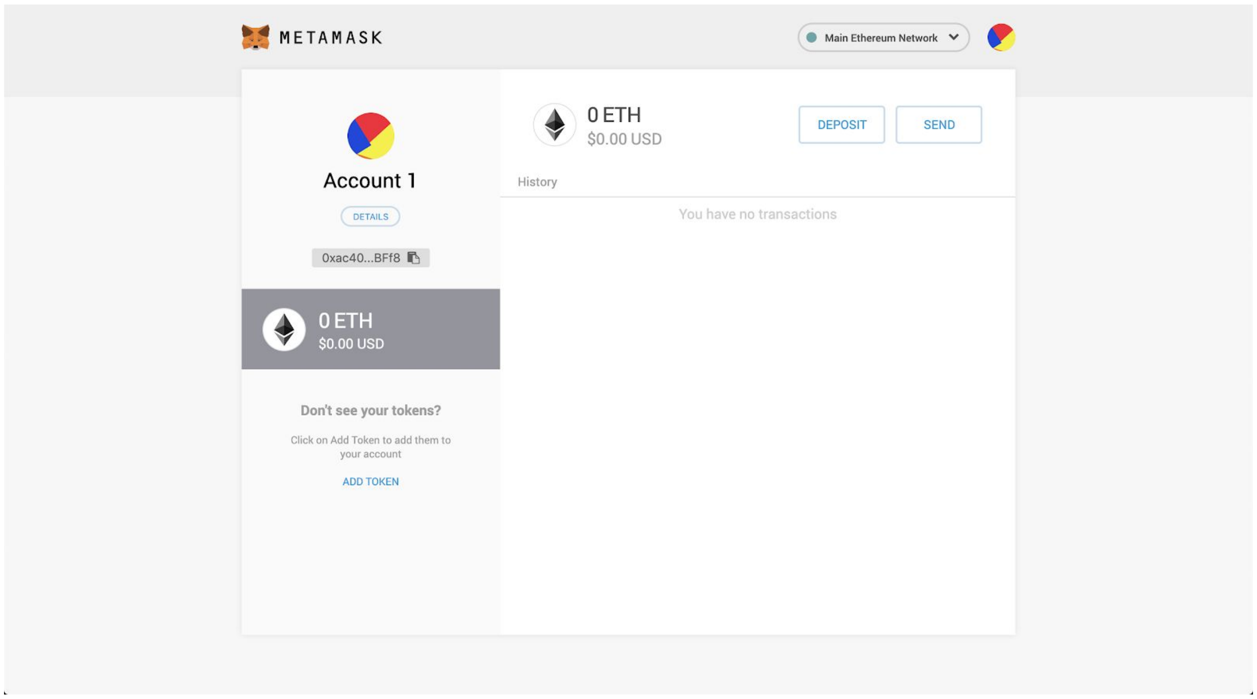
METAMASK

< Back

Confirm your Secret Backup Phrase

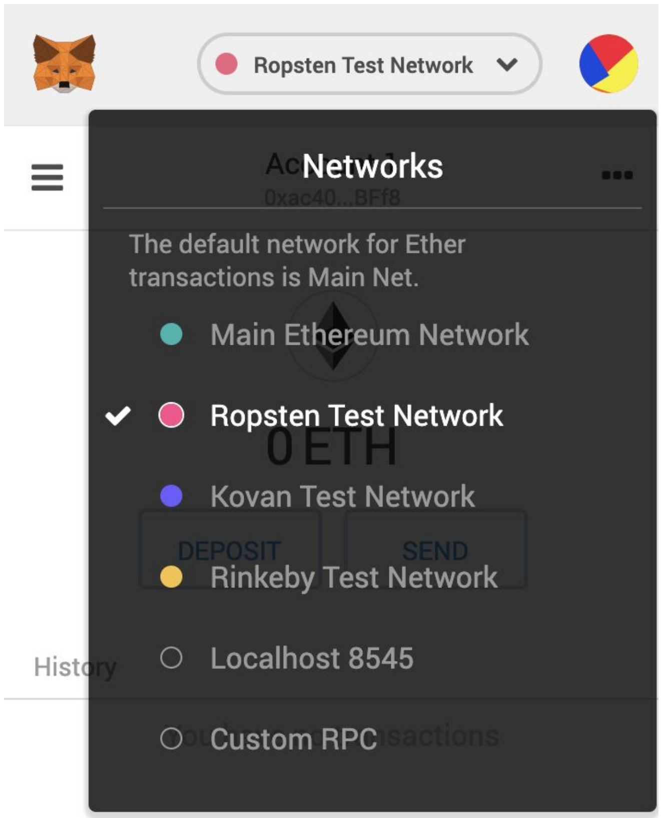
Please select each phrase in order to make sure it is correct.

6. You are ready to use your account to send and receive Ether!

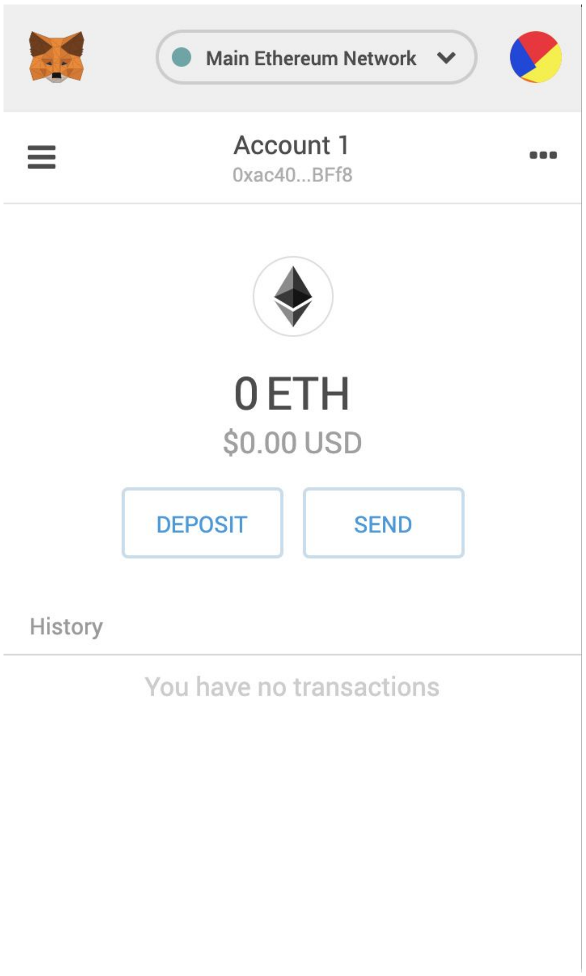

METAMASK
Main Ethereum Network
Account 1
DETAILS
0xac40...BFf8
0 ETH
$0.00 USD
Don't see your tokens?
Click on Add Token to add them to your account
ADD TOKEN
0 ETH
$0.00 USD
DEPOSIT
SEND
History
You have no transactions

Get Ether from a faucet for Ropsten testnet

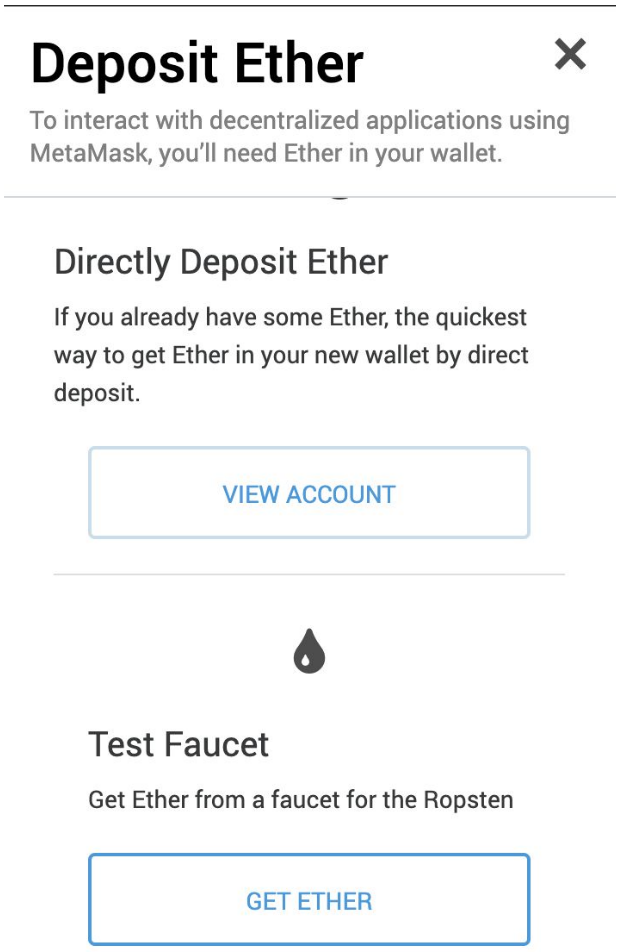
1. Login to your wallet and select the Ropsten testnet network



2. Press “Deposit”



3. Press “Get Ether”.



Deposit Ether

To interact with decentralized applications using MetaMask, you'll need Ether in your wallet.

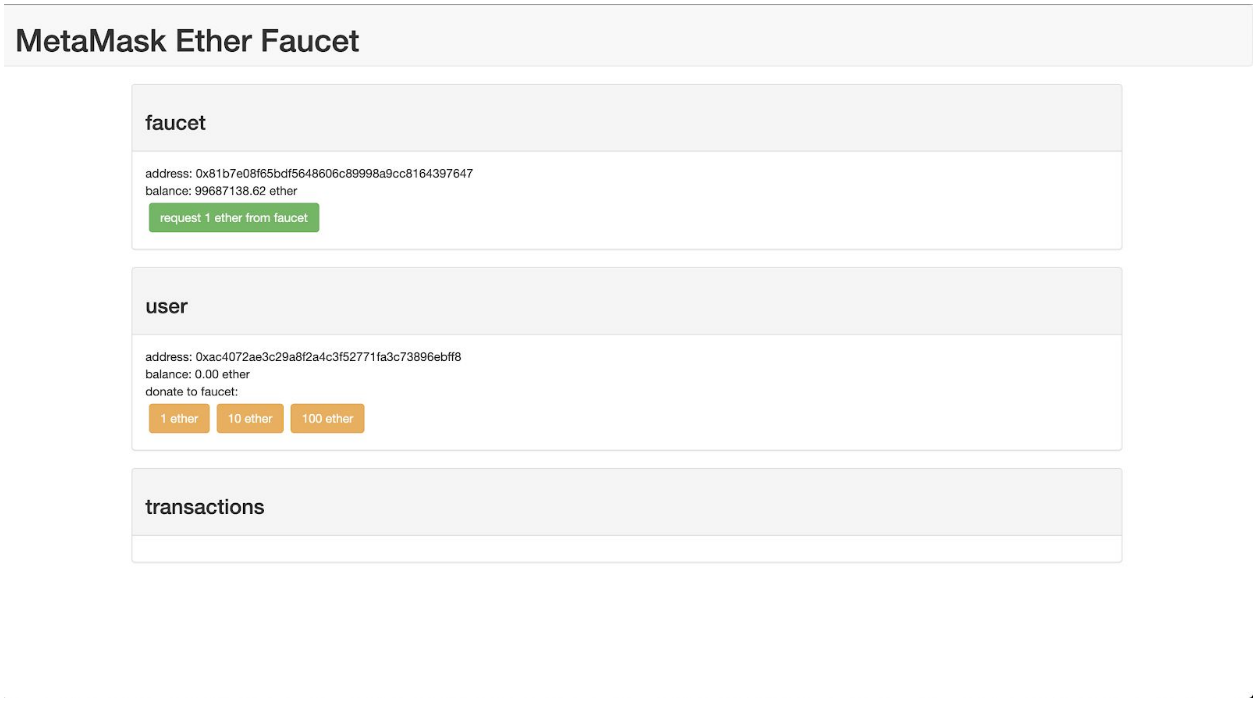
Directly Deposit Ether

If you already have some Ether, the quickest way to get Ether in your new wallet by direct deposit.

Test Faucet

Get Ether from a faucet for the Ropsten

4. On the page that opened press “request 1 ether from faucet”



MetaMask Ether Faucet

faucet

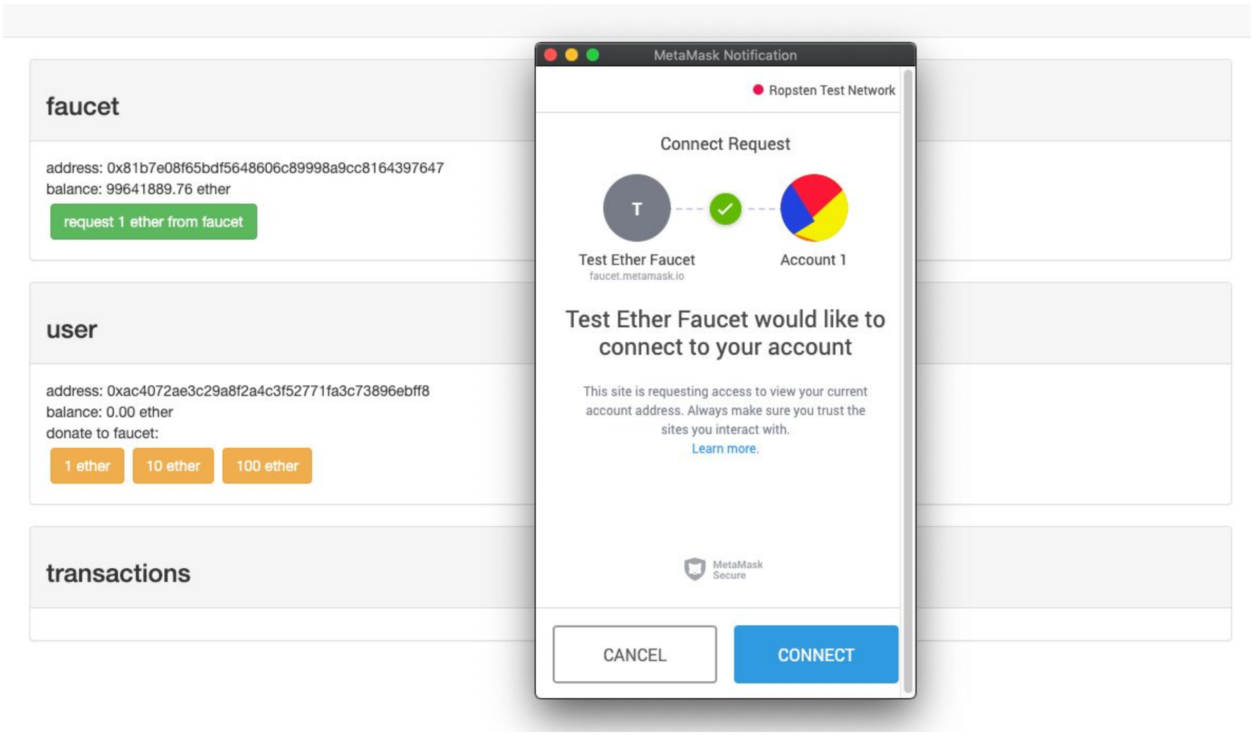
address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99687138.62 ether

user

address: 0xac4072ae3c29a8f2a4c3f52771fa3c73896ebff8
balance: 0.00 ether
donate to faucet:

transactions

5. Press “Connect”



faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99641889.76 ether

user

address: 0xac4072ae3c29a8f2a4c3f52771fa3c73896ebff8
balance: 0.00 ether
donate to faucet:

transactions

MetaMask Notification

Ropsten Test Network

Connect Request

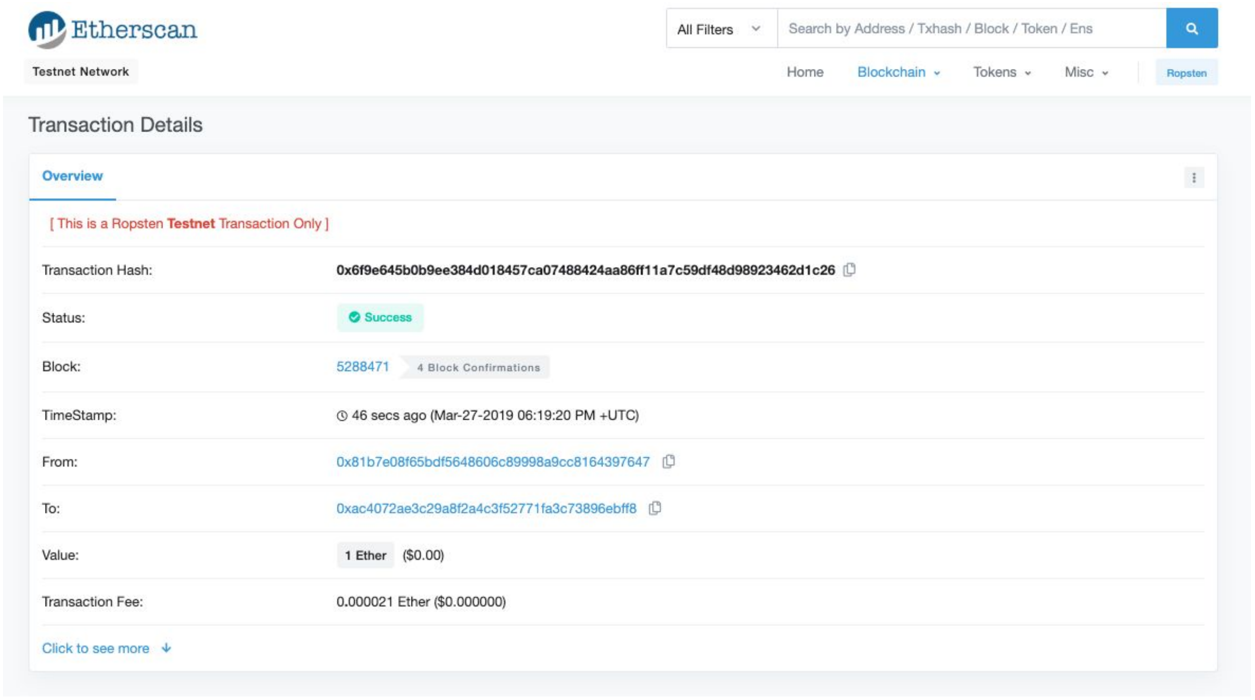
Test Ether Faucet
faucet.metamask.io

Account 1

Test Ether Faucet would like to connect to your account

This site is requesting access to view your current account address. Always make sure you trust the sites you interact with.
Learn more.

6. Wait for the transaction to be mined. You can see the transaction details at https://ropsten.etherscan.io/



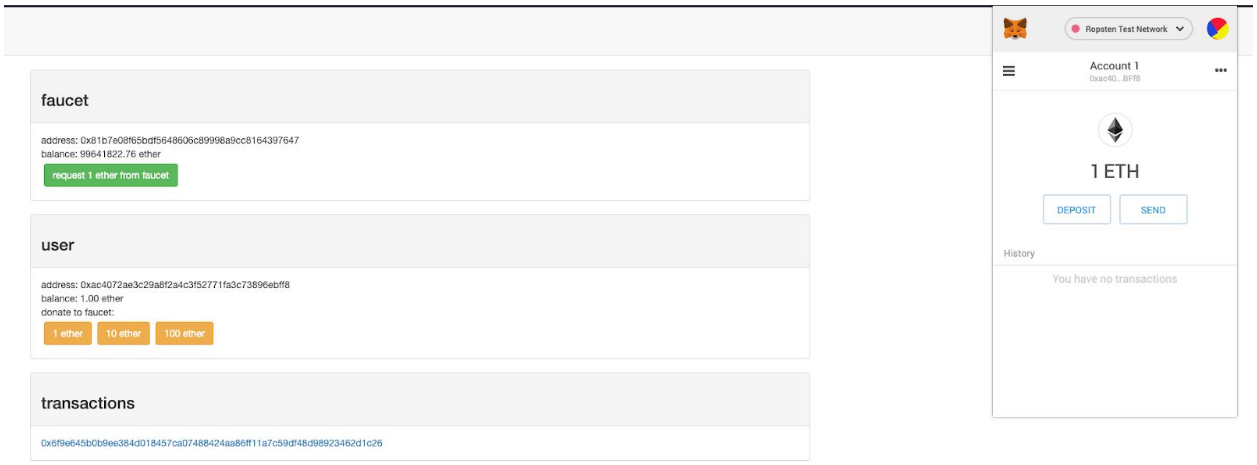
Transaction Details

Overview

[This is a Ropsten **Testnet** Transaction Only]

Transaction Hash:	**0x6f9e645b0b9ee384d018457ca07488424aa86ff11a7c59df48d98923462d1c26**
Status:	Success
Block:	5288471 4 Block Confirmations
TimeStamp:	46 secs ago (Mar-27-2019 06:19:20 PM +UTC)
From:	0x81b7e08f65bdf5648606c89998a9cc8164397647
To:	0xac4072ae3c29a8f2a4c3f52771fa3c73896ebff8
Value:	1 Ether ($0.00)
Transaction Fee:	0.000021 Ether ($0.000000)

Click to see more



faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99641822.76 ether

user

address: 0xac4072ae3c29a8f2a4c3f52771fa3c73896ebff8
balance: 1.00 ether
donate to faucet:

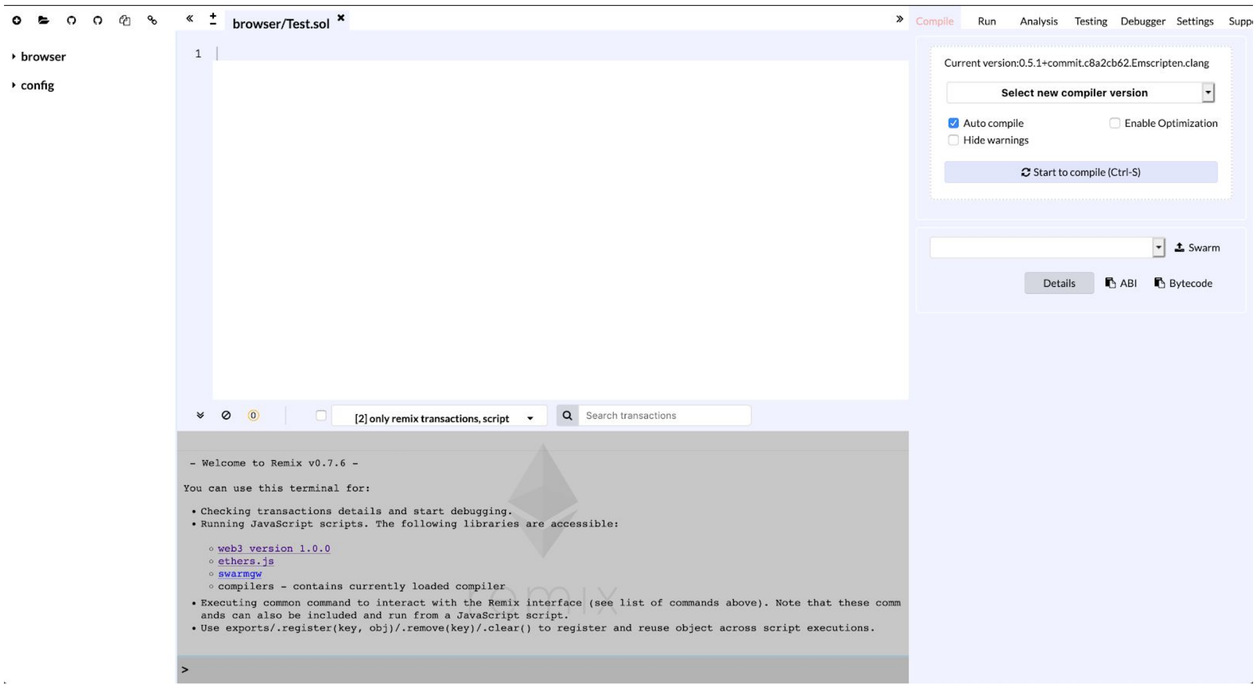
transactions

0x6f9e645b0b9ee384d018457ca07488424aa86ff11a7c59df48d98923462d1c26

Interacting with Remix

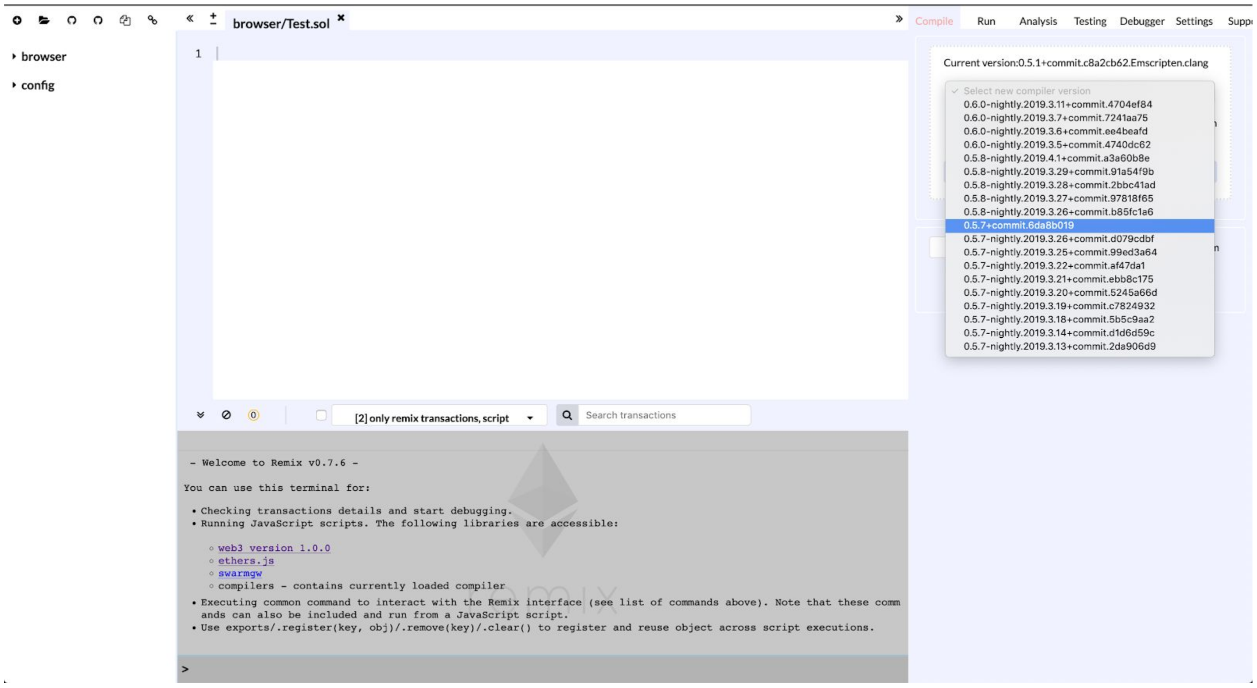
Remix is a browser-based compiler and IDE that enables users to build Ethereum contracts with Solidity language and to debug transactions.

1. Go to https://remix.ethereum.org. You will see a screen with an editor in the middle and two sidebars, one on the left and one on the right. In the editor you can write Ethereum smart contracts with the use of the Solidity programming language.



```
 - Welcome to Remix v0.7.6 -

You can use this terminal for:

 • Checking transactions details and start debugging.
 • Running JavaScript scripts. The following libraries are accessible:

    ○ web3 version 1.0.0
    ○ ethers.js
    ○ swarmgw
    ○ compilers - contains currently loaded compiler
 • Executing common command to interact with the Remix interface (see list of commands above). Note that these comm
   ands can also be included and run from a JavaScript script.
 • Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.

>
```

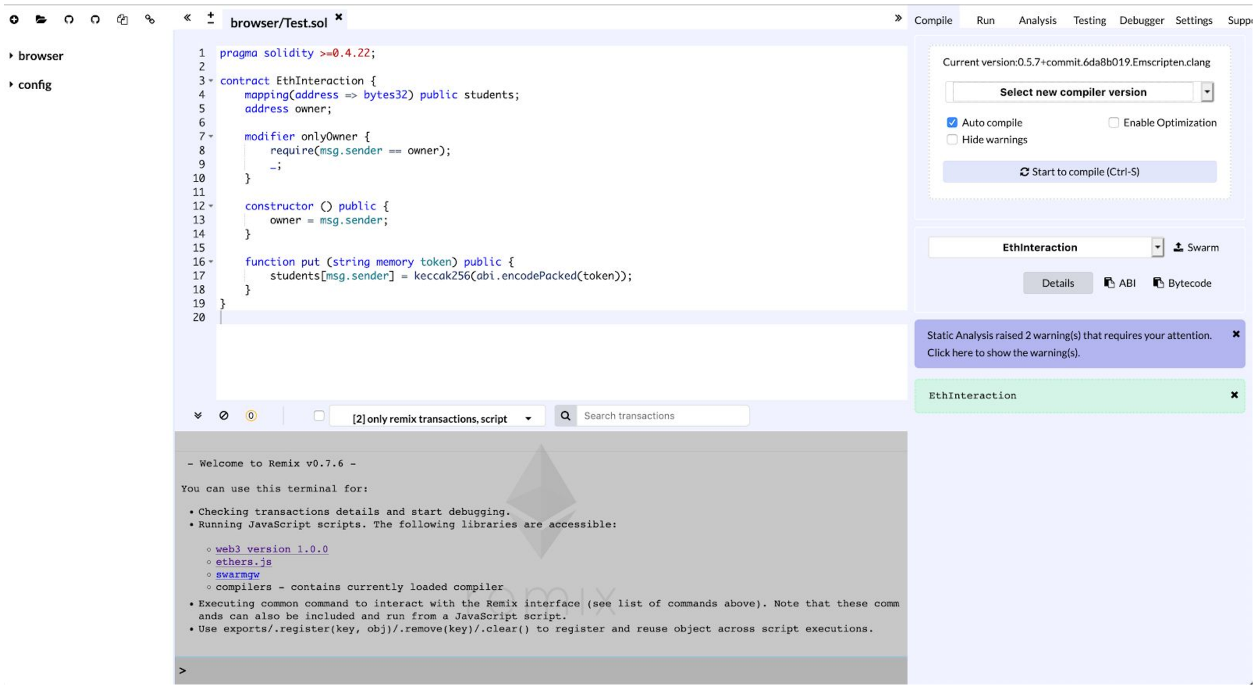
2. Write your contract you want to test and debug in the editor, select the desired compiler version (on the right) and press “Start to compile” (if auto compile is not enabled).



Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

```
Select new compiler version
0.6.0-nightly.2019.3.11+commit.4704ef84
0.6.0-nightly.2019.3.7+commit.7241aa75
0.6.0-nightly.2019.3.6+commit.ee4beafd
0.6.0-nightly.2019.3.5+commit.4740dc62
0.5.8-nightly.2019.4.1+commit.a3a60b8e
0.5.8-nightly.2019.3.29+commit.91a54f9b
0.5.8-nightly.2019.3.28+commit.2bbc41ad
0.5.8-nightly.2019.3.27+commit.97818f65
0.5.8-nightly.2019.3.26+commit.b85fc1a6
0.5.7+commit.6da8b019
0.5.7-nightly.2019.3.26+commit.d079cdbf
0.5.7-nightly.2019.3.25+commit.99ed3a64
0.5.7-nightly.2019.3.22+commit.af47da1
0.5.7-nightly.2019.3.21+commit.ebb8c175
0.5.7-nightly.2019.3.20+commit.5245a66d
0.5.7-nightly.2019.3.19+commit.c7824932
0.5.7-nightly.2019.3.18+commit.5b5c9aa2
0.5.7-nightly.2019.3.14+commit.d1d6d59c
0.5.7-nightly.2019.3.13+commit.2da906d9
```

```
- Welcome to Remix v0.7.6 -

You can use this terminal for:

 • Checking transactions details and start debugging.
 • Running JavaScript scripts. The following libraries are accessible:

    ○ web3 version 1.0.0
    ○ ethers.js
    ○ swarmgw
    ○ compilers - contains currently loaded compiler
 • Executing common command to interact with the Remix interface (see list of commands above). Note that these comm
   ands can also be included and run from a JavaScript script.
 • Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.

>
```



```
pragma solidity >=0.4.22;

contract EthInteraction {
    mapping(address => bytes32) public students;
    address owner;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    constructor () public {
        owner = msg.sender;
    }

    function put (string memory token) public {
        students[msg.sender] = keccak256(abi.encodePacked(token));
    }
}
```

Current version:0.5.7+commit.6da8b019.Emscripten.clang

Select new compiler version

☑ Auto compile ☐ Enable Optimization
☐ Hide warnings

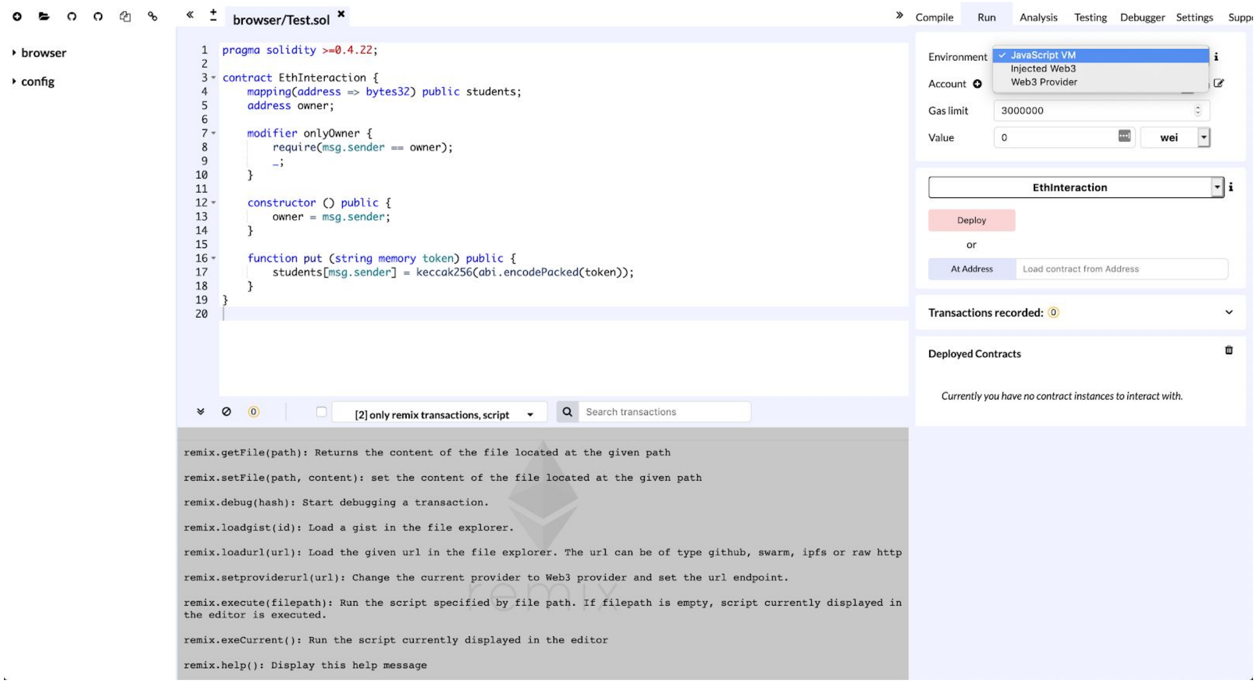
EthInteraction

Static Analysis raised 2 warning(s) that requires your attention. Click here to show the warning(s).

EthInteraction

```
- Welcome to Remix v0.7.6 -

You can use this terminal for:

 • Checking transactions details and start debugging.
 • Running JavaScript scripts. The following libraries are accessible:

    ○ web3 version 1.0.0
    ○ ethers.js
    ○ swarmgw
    ○ compilers - contains currently loaded compiler
 • Executing common command to interact with the Remix interface (see list of commands above). Note that these comm
   ands can also be included and run from a JavaScript script.
 • Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.

>
```

3. On the right sidebar click “Run” and then select “Environment -> JavaScript VM”. If you want to interact with a testnet network you must install MetaMask, select your testnet, and then select “Environment -> Injected Web3” from Remix.

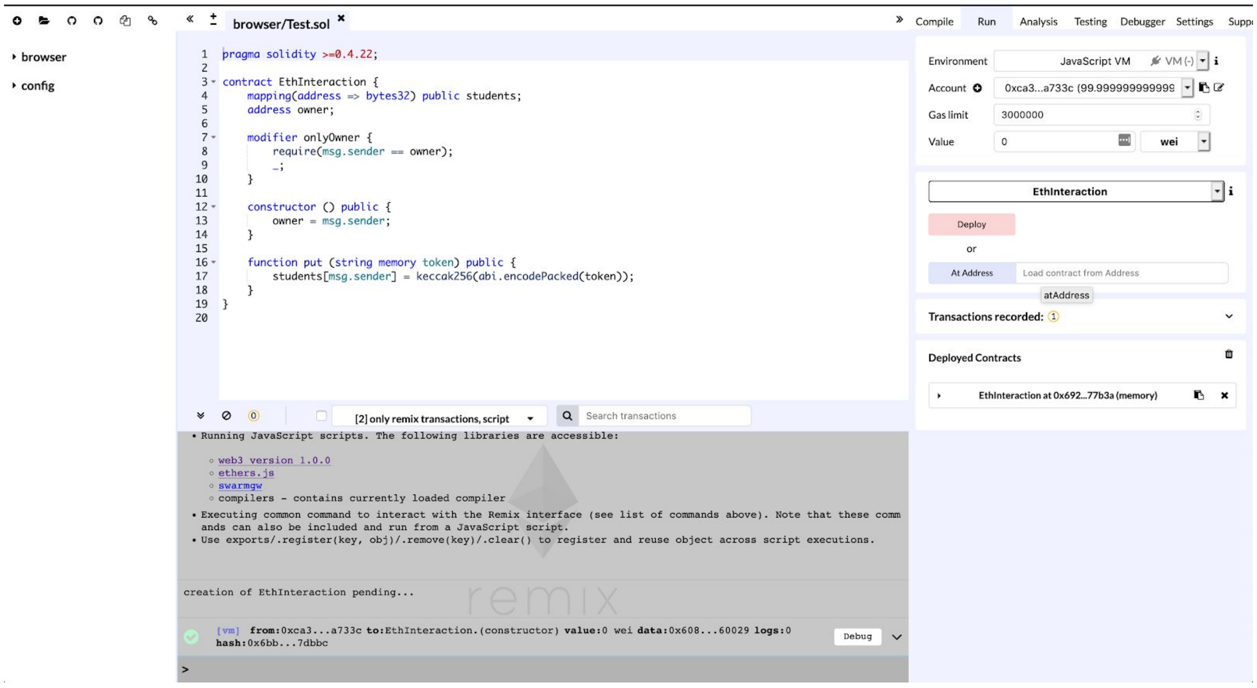


```
pragma solidity >=0.4.22;

contract EthInteraction {
    mapping(address => bytes32) public students;
    address owner;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    constructor () public {
        owner = msg.sender;
    }

    function put (string memory token) public {
        students[msg.sender] = keccak256(abi.encodePacked(token));
    }
}
```

Environment: ✓ JavaScript VM | Injected Web3 | Web3 Provider
Gas limit: 3000000
Value: 0 wei
EthInteraction
Transactions recorded: 0
Deployed Contracts
Currently you have no contract instances to interact with.

```
remix.getFile(path): Returns the content of the file located at the given path

remix.setFile(path, content): set the content of the file located at the given path

remix.debug(hash): Start debugging a transaction.

remix.loadgist(id): Load a gist in the file explorer.

remix.loadurl(url): Load the given url in the file explorer. The url can be of type github, swarm, ipfs or raw http

remix.setproviderurl(url): Change the current provider to Web3 provider and set the url endpoint.

remix.execute(filepath): Run the script specified by file path. If filepath is empty, script currently displayed in
the editor is executed.

remix.exeCurrent(): Run the script currently displayed in the editor

remix.help(): Display this help message
```

4. Click “Deploy”. In the output console, above the editor, you should see the transaction details.

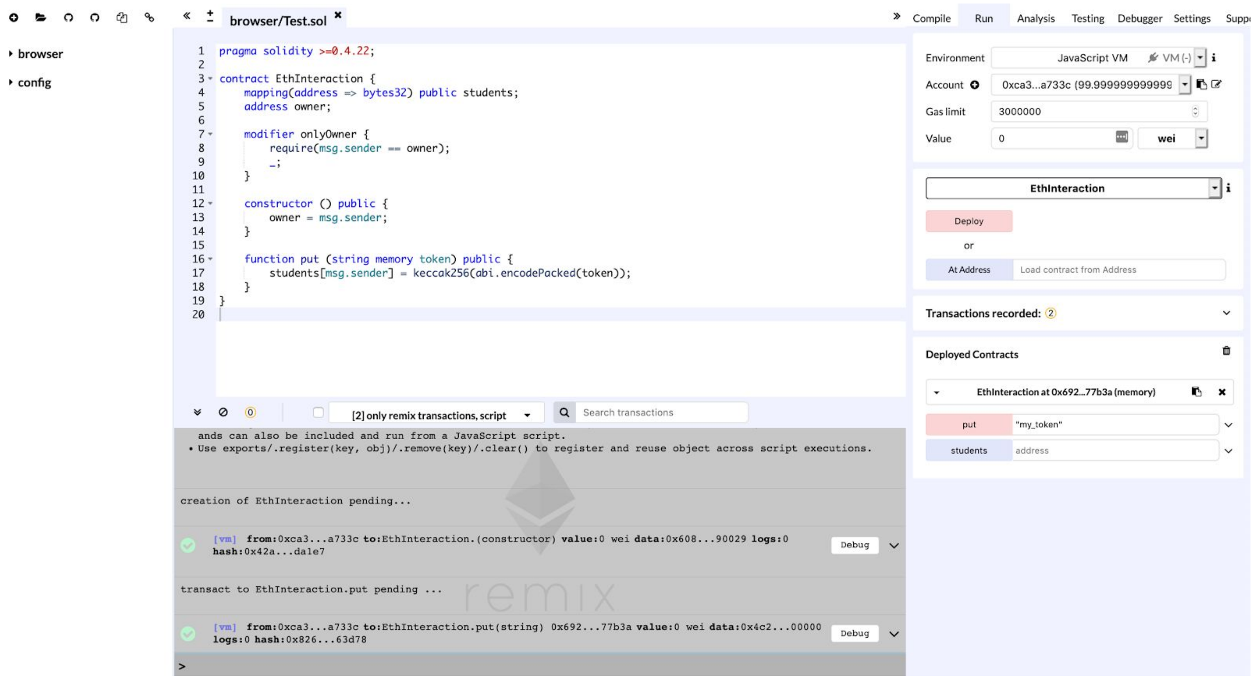


```
pragma solidity >=0.4.22;

contract EthInteraction {
    mapping(address => bytes32) public students;
    address owner;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    constructor () public {
        owner = msg.sender;
    }

    function put (string memory token) public {
        students[msg.sender] = keccak256(abi.encodePacked(token));
    }
}
```

Environment: JavaScript VM
Account: 0xca3...a733c (99.999999999999
Gas limit: 3000000
Value: 0 wei
EthInteraction
Transactions recorded: 1
Deployed Contracts
EthInteraction at 0x692...77b3a (memory)

```
• Running JavaScript scripts. The following libraries are accessible:
    ○ web3 version 1.0.0
    ○ ethers.js
    ○ swarmgw
    ○ compilers - contains currently loaded compiler
• Executing common command to interact with the Remix interface (see list of commands above). Note that these comm
  ands can also be included and run from a JavaScript script.
• Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.

creation of EthInteraction pending...

[vm]  from:0xca3...a733c to:EthInteraction.(constructor) value:0 wei data:0x608...60029 logs:0
hash:0x6bb...7dbbc
>
```

5. You are ready to interact with your contract. At the right sidebar, at deployd contracts section, you can see all the functions of the contract. You can call each function, pass parameters and get outputs.



```
pragma solidity >=0.4.22;

contract EthInteraction {
    mapping(address => bytes32) public students;
    address owner;

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    constructor () public {
        owner = msg.sender;
    }

    function put (string memory token) public {
        students[msg.sender] = keccak256(abi.encodePacked(token));
    }
}
```

Environment: JavaScript VM — VM (-)
Account: 0xca3...a733c (99.999999999999
Gas limit: 3000000
Value: 0 wei
EthInteraction
EthInteraction at 0x692...77b3a (memory)
put: "my_token"
students: address

```
  ands can also be included and run from a JavaScript script.
 • Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.

creation of EthInteraction pending...

[vm] from:0xca3...a733c to:EthInteraction.(constructor) value:0 wei data:0x608...90029 logs:0
hash:0x42a...da1e7

transact to EthInteraction.put pending ...

[vm] from:0xca3...a733c to:EthInteraction.put(string) 0x692...77b3a value:0 wei data:0x4c2...00000
logs:0 hash:0x826...63d78

>
```