Creating a Crypto Token Using a Smart Contract

Campbell R. Harvey

Duke University and NBER

Developed in collaboration with Hyoung-Yoon Kim, Anibal Granada Gonzales, Manmit Singh and Yash Patil

Things you'll need

- A basic understanding of the Solidity language https://cryptozombies.io/en/course/
- Metamask extension: chrome web store
- Remix IDE: https://remix.ethereum.org/
- At least 1 full testnet ether: https://faucet.ropsten.be/

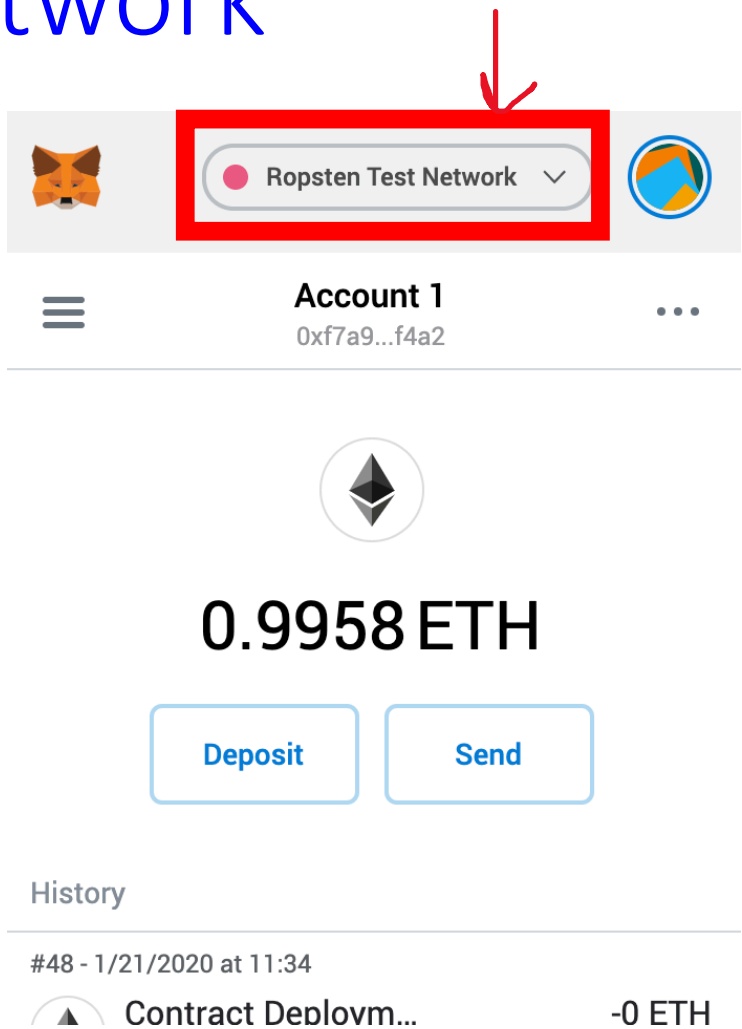
Parameters for the token

- Name (e.g., FQ1)
- Symbol (e.g., FQ1, default is SYM)
- Decimal places (e.g., 2 decimal places))
- Number of units in circulation (e.g., 100 billion)

- **The** number of whole tokens will be 1 billion in the above example, i.e.,

  100 billion/100 (10^2 from 2 decimal places)

First off, make sure your MetaMask is on Ropsten Test Network

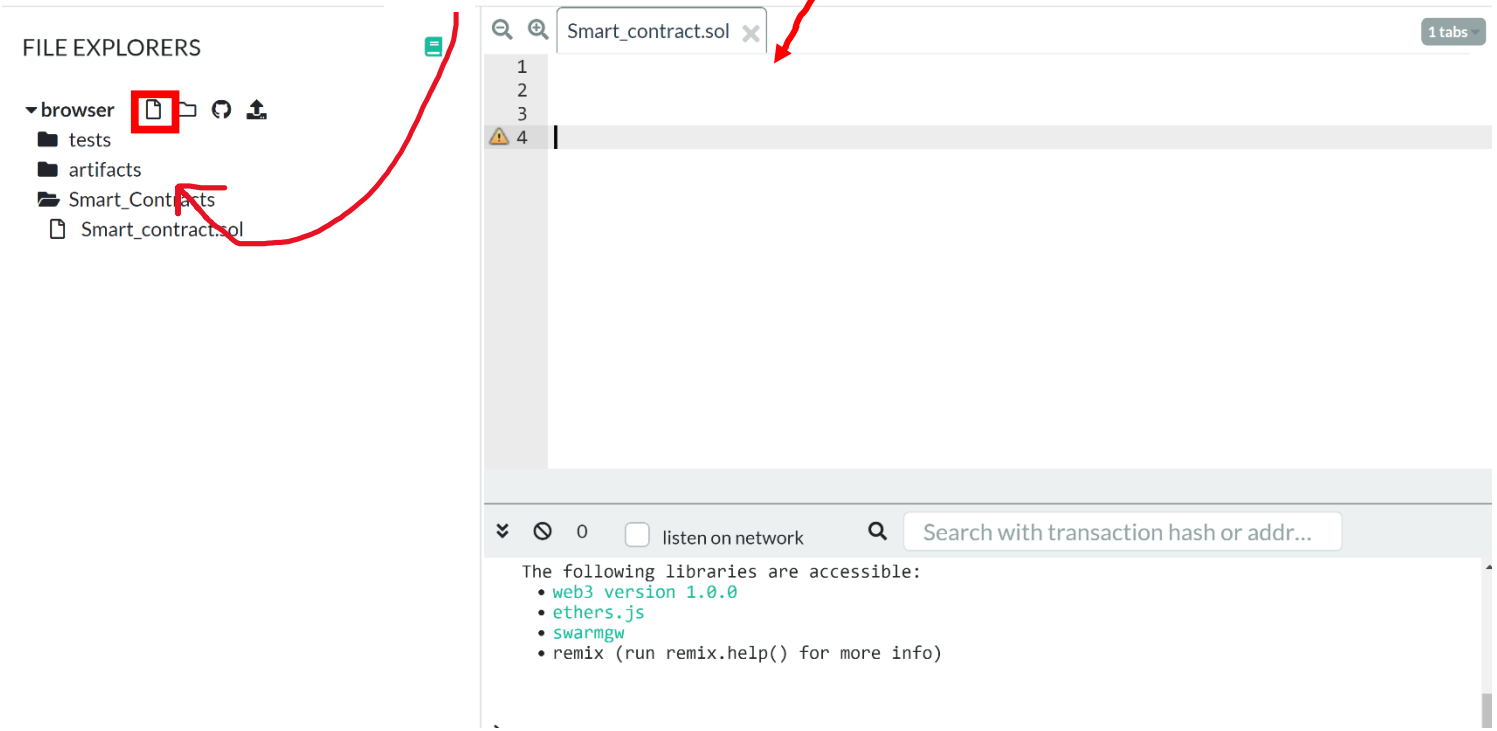


Step 1. Acquire test ethereum for the gas fee

Go to https://faucet.metamask.io/ and request at least 1 Ethereum from the faucet

Step 2. Write a contract

1. Go to the IDE (Integrated Development Environment) https://remix.ethereum.org
2. Select the Solidity Environment
3. Create new file by clicking on



Step 2. Write a contract

3. Name the file “Token.sol”
4. Copy and paste the contract code*
   - Note: Be careful here because some copy and pastes might change some characters like ‘ or -.
   - Depending on your browser, the code will be highlighted; noticed that all the code preceded by a slash (‘/’) takes the same color. **Solidity** understands text after / as comments and will not attempt to run it as code (similar to ‘#’ in **R** and **Python**).
   - Notice that functions, object names and parameters take distinctive colors.

*File available at https://faculty.Fuqua.duke.edu/~charvey/Teaching/697_2022/Public_Presentations_697/fq1.sol

Step 2. Write a contract

5. Modify the code –

At the name of the contract

The parameters of the contract constructor

Notes:

- **Use ctrl + f to find "FQ1" and replace it with the name you give to your token elsewhere in the code**
- **Do not use thousands separator**

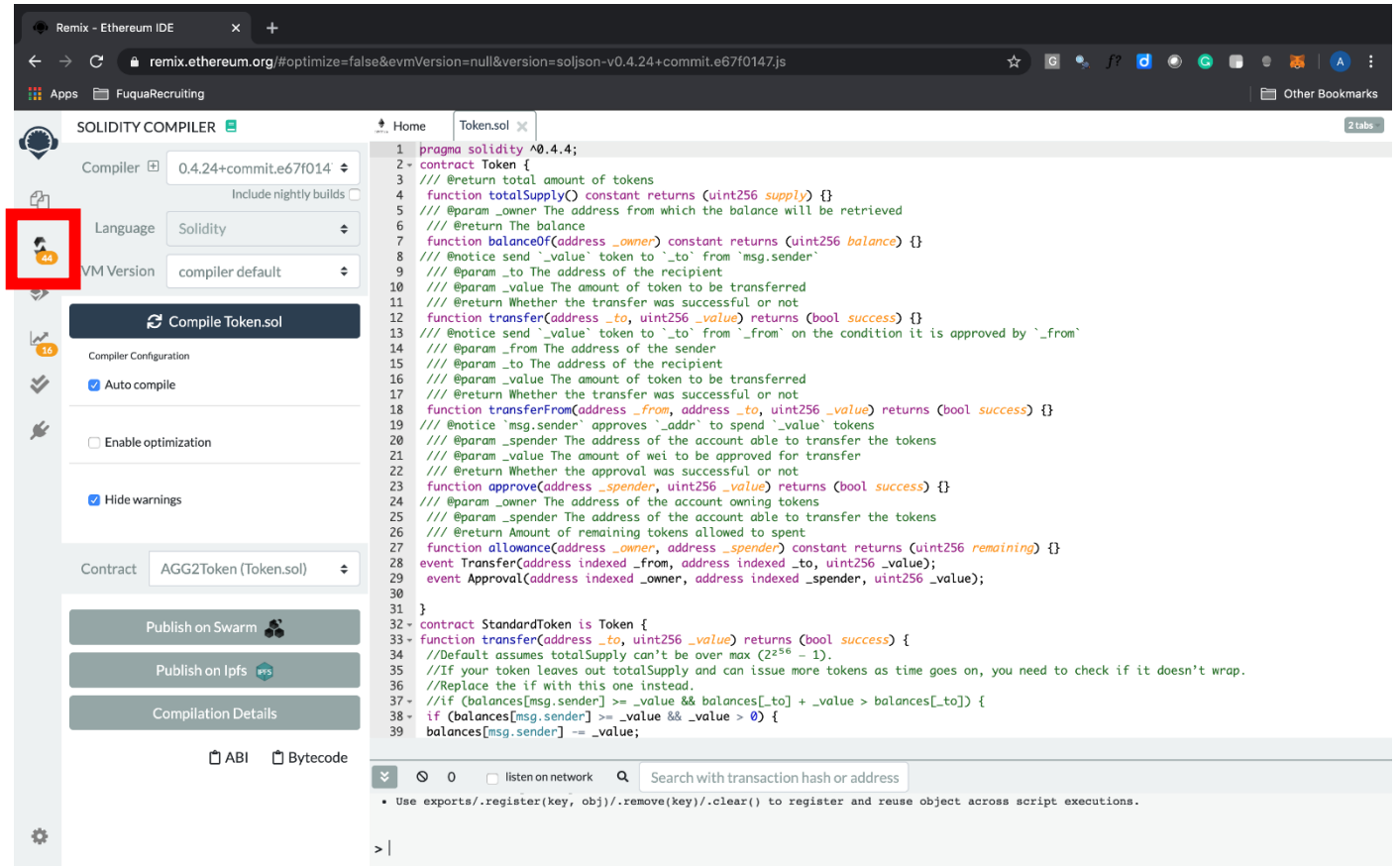
```
contract DevilToken is ERC20Interface, SafeMath {
    string public name;
    string public symbol;
    uint8 public decimals; // 18 decimals is the st
```

```
constructor() public {
    name = "ManmitToken";
    symbol = "MST";
    decimals = 2;
    _totalSupply = 1000000000000000000000000000;

    balances[msg.sender] = _totalSupply;
    emit Transfer(address(0), msg.sender, _totalSupply);
}
```

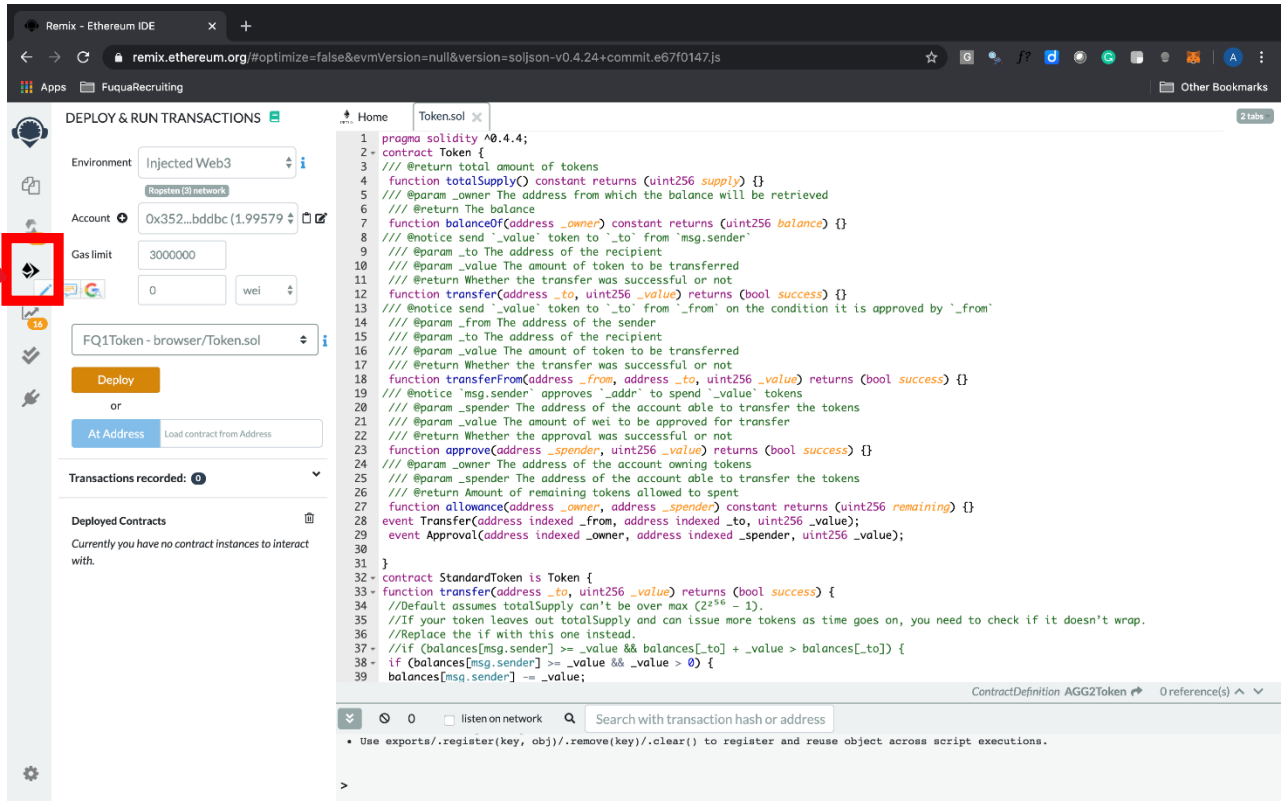
Step 3. Compile the contract

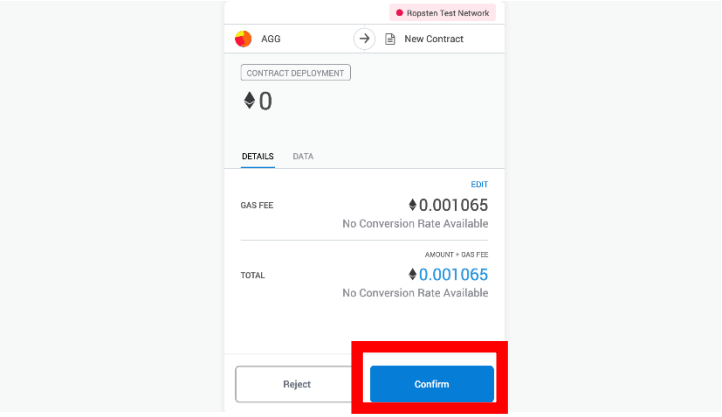
- 1. Switch to Compiler tab
- 2. Choose 0.4.24 commit version compiler
- 3. ‘Auto compile’ ON, ‘Enable optimization’ OFF, ‘Hide warnings’ ON
- 4. Compile the contract



Step 4. Deploy the contract

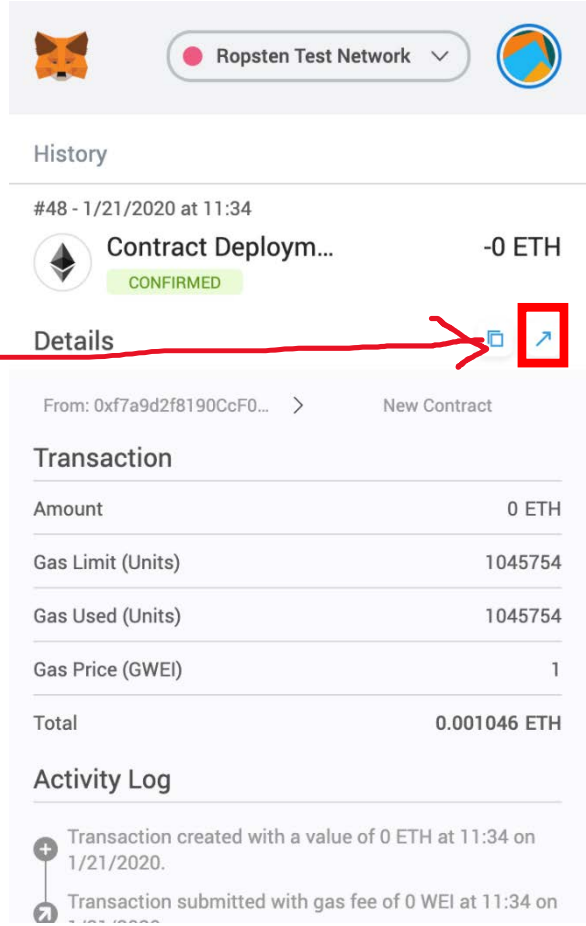
- 1. Switch to Deploy & Run tab
- 2. Choose Injected Web3 Environment
- 3. Deploy your token using your account address!
  - You should be logged in with your MetaMask; a new screen will ask you to confirm the contract





Step 4. Deploy the contract

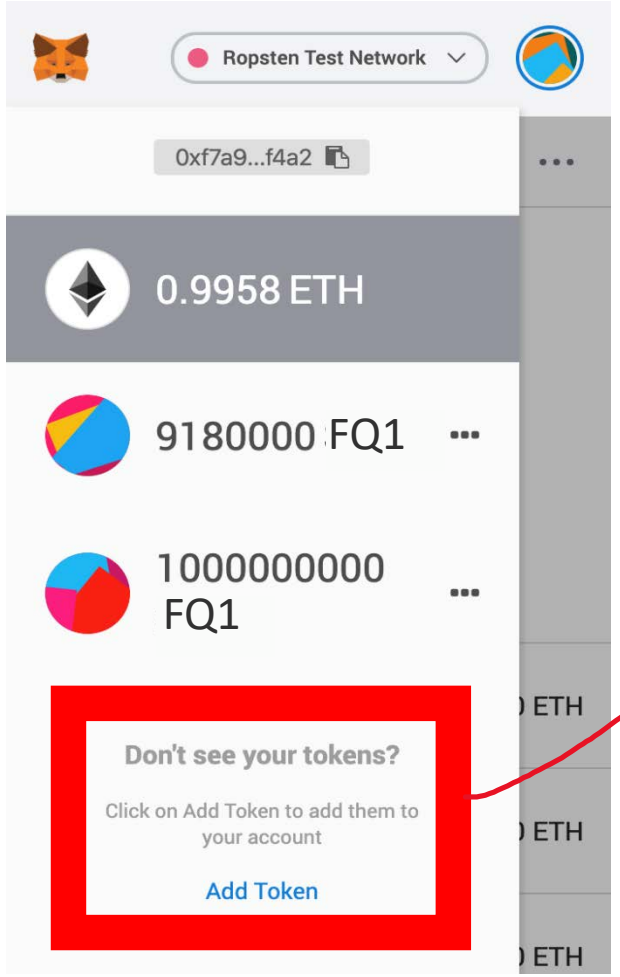
- 4. The deployed contract should show up on your MetaWallet. View it on Etherscan and obtain the contract ID.

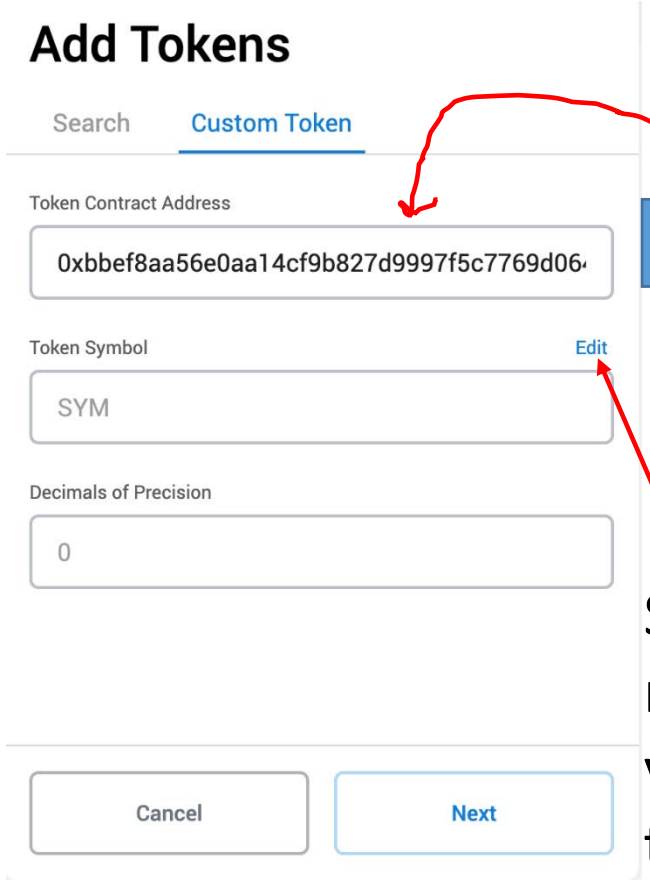


Look carefully:

<u>- The contract ID **is not** your public address</u>

Step 5. Add tokens





The contract ID

(It **is not** your public address)

Symbol should match the one you establish in the code

Step 6. Using the tokens

You can send tokens to anyone with a MetaMask address

- They will have to follow Step 5 to add the custom token to their wallet.

Further information

The Solidity code is from:

- https://github.com/CodeWithJoe2020/ERC20Token/blob/main/ERC20.sol