

---

# ERC20 Deployment Guide

Friday, 06.15.2018

**Keyur Patel**

## Pre- Requisites

1. Laptop or Computer
2. Code - Editor
3. Internet Access (via Chrome Web Browser)

## Basics

### Ethereum based ERC20 Tokens

In Ethereum tokens represent any tradable goods such as coins, loyalty points etc. You can create your own crypto-currencies based on Ethereum. Additionally the benefit of following ERC20 standard is that your tokens will be compatible with any other client or wallets that use the same standards.

### Smart Contracts

Smart Contracts are self executing code blocks deployed on the Ethereum blockchain. They contain data & code functions. Contracts make decisions, interact with other contracts, store data and transfer Ether (the unit of crypto-currency in the Ethereum blockchain) among users.

### Solidity

A language for writing smart contracts.

### MetaMask Wallet

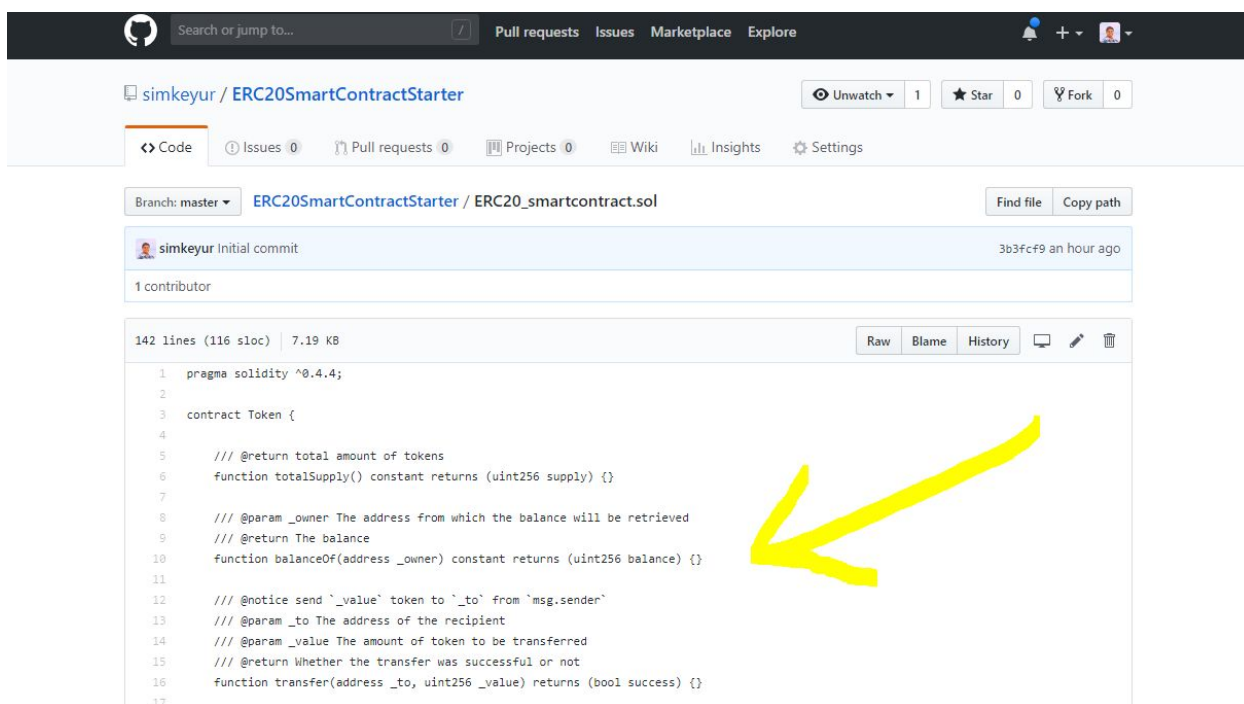
A digital facility that holds your Ether and other Ethereum based tokens.

---

## Step 1: Smart Contract Code

1. Very first step is to code your smart contract. Don't worry, I won't make you write whole smart contract.
2. Open any text editor and paste code from this link:

<https://github.com/simkeyur/ERC20SmartContractStarter>



The screenshot shows the GitHub interface for the repository 'simkeyur / ERC20SmartContractStarter'. The 'Code' tab is selected, displaying the Solidity code for 'ERC20\_smartcontract.sol'. A yellow arrow points to the 'function balanceOf' line in the code.

```

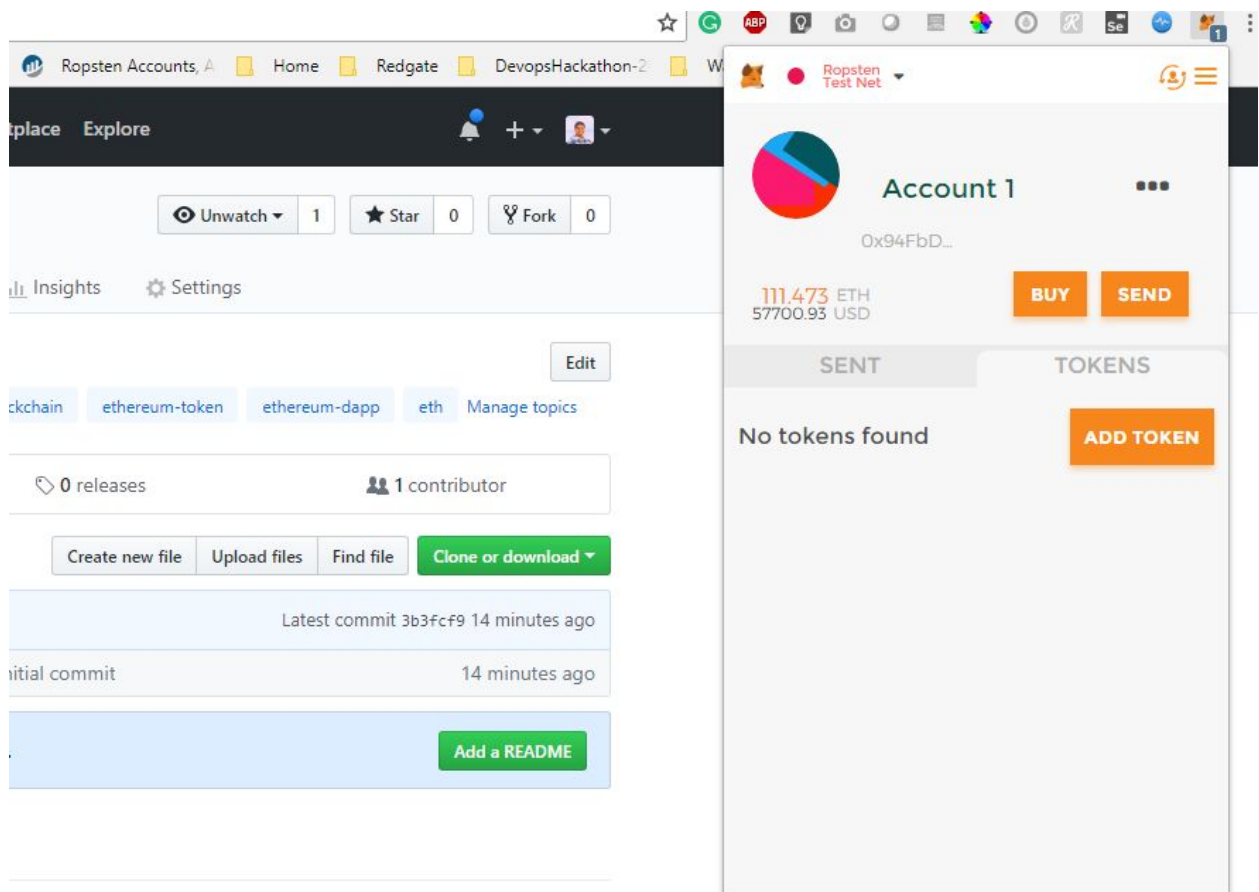
1  pragma solidity ^0.4.4;
2
3  contract Token {
4
5      /// @return total amount of tokens
6      function totalSupply() constant returns (uint256 supply) {}
7
8      /// @param _owner The address from which the balance will be retrieved
9      /// @return The balance
10     function balanceOf(address _owner) constant returns (uint256 balance) {}
11
12     /// @notice send `_value` token to `_to` from `msg.sender`
13     /// @param _to The address of the recipient
14     /// @param _value The amount of token to be transferred
15     /// @return Whether the transfer was successful or not
16     function transfer(address _to, uint256 _value) returns (bool success) {}
17

```

3. Now make these changes to create personalized tokens:
  - a. Smart Contract Name & Function name)
  - b. Balances[msg.sender]
  - c. totalSupply
  - d. Name
  - e. Symbol (No more than 5 characters)
  - f. UnitsOneEthCanBuy (optional)
4. The above code uses Solidity language to build a simple ERC20 token.
5. We also set our ICO pricing in this as 1 ETH = **UnitsOneEthCanBuy** ERC20 Tokens., this means if someone sends 1 ETH to this smart contract, they will get **UnitsOneEthCanBuy** ERC20 units of this.

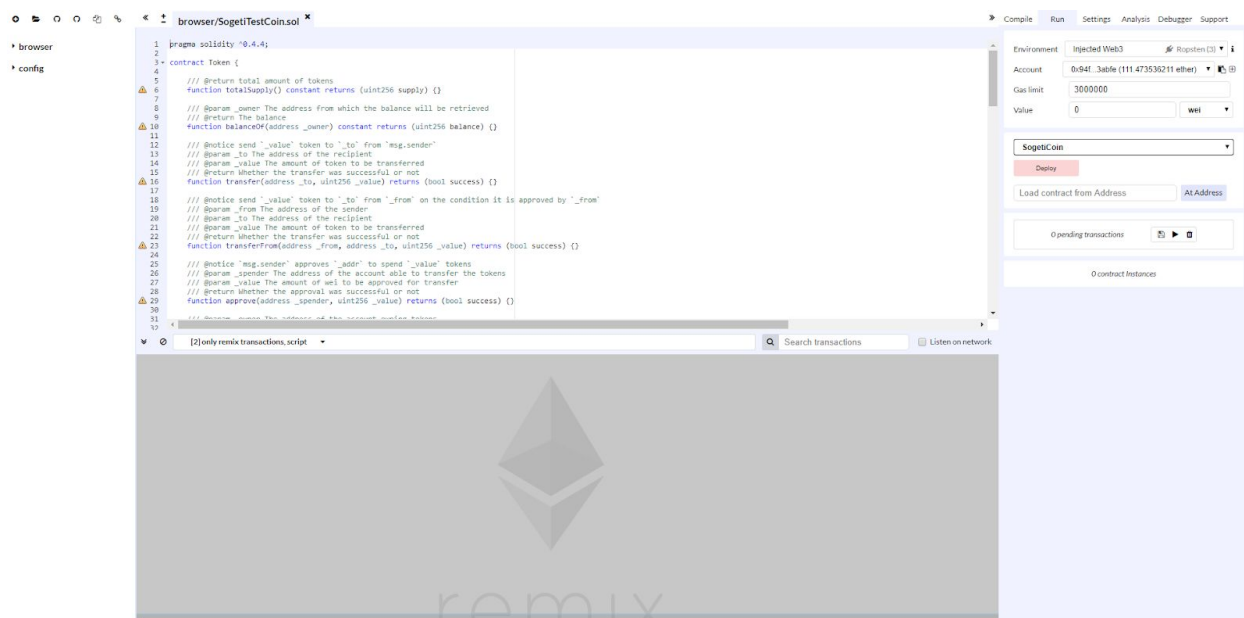
## Step 2: Get a ETH Wallet (Metamask.io)

1. Download [MetaMask](#) chrome extension to generate a wallet. This is going to be the owner of the smart contract.
2. Once you download the extension, create a password protected account (if possible save the phrase in secured text file).
3. Then choose "Ropsten TestNet" from top left corner. Click on Buy button and it will redirect you to faucet, which will give u some test Ethers. We will need it deploy our token.
4. If everything is good, it should look like this:

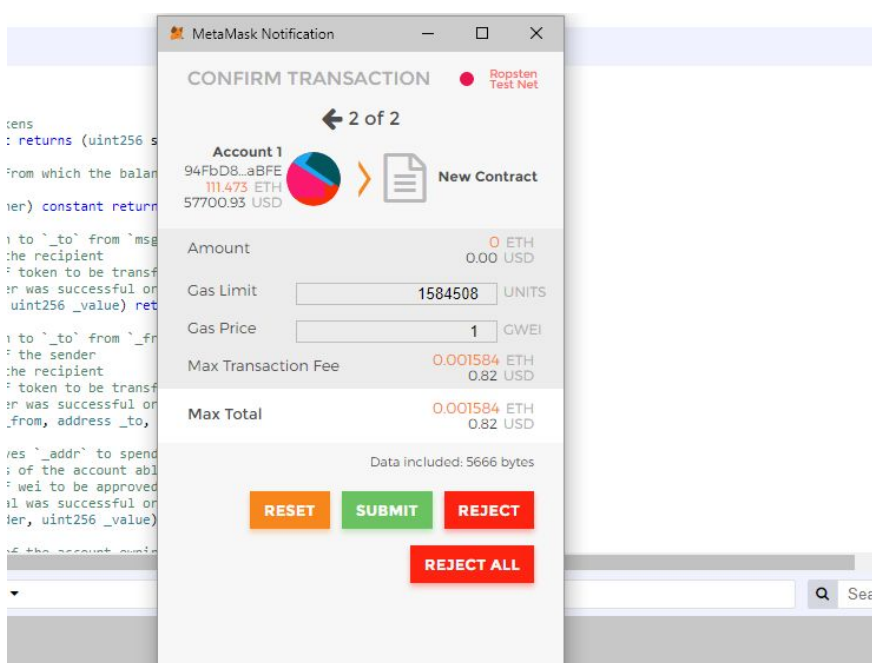


## Step 3: Create Smart Contract (On Remix IDE)

1. Go to Remix IDE website: <http://remix.ethereum.org/>
2. Paste the code from Step 1 into the browser:



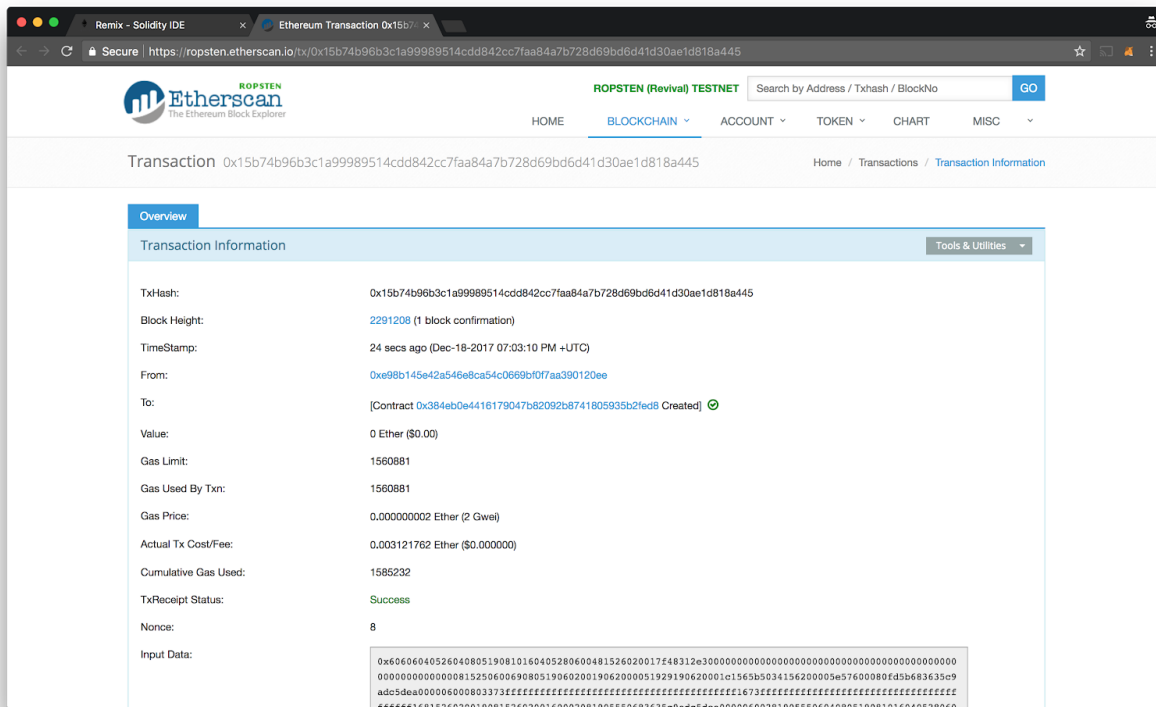
3. Go to Run Tab and select your contract name on click on 'Deploy' button. If your wallet is still open you should see this:



4. Click on Submit and wait for 30 seconds for transaction to get mined.

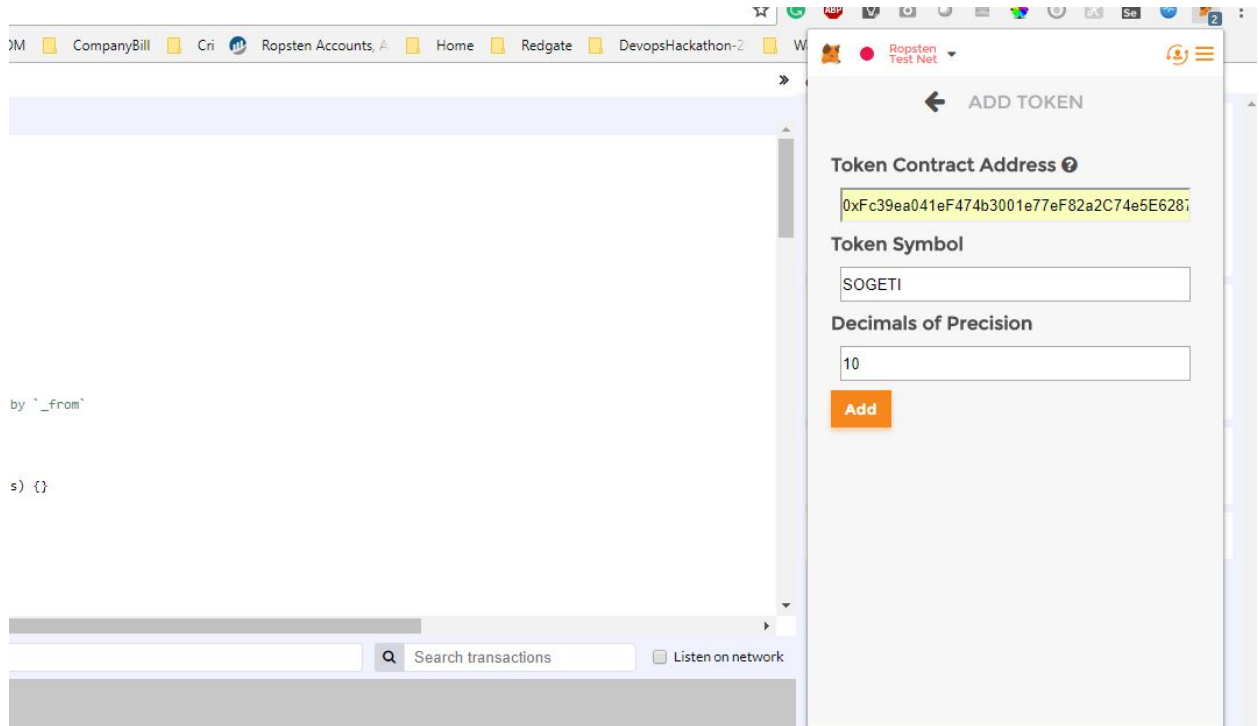
## Step 4: Verify Contract Deployment (On Etherscan.io)

1. Open Metamask Wallet and click on the transaction, it should navigate to ropsten contract deployment, if it is successful, it should look like this:

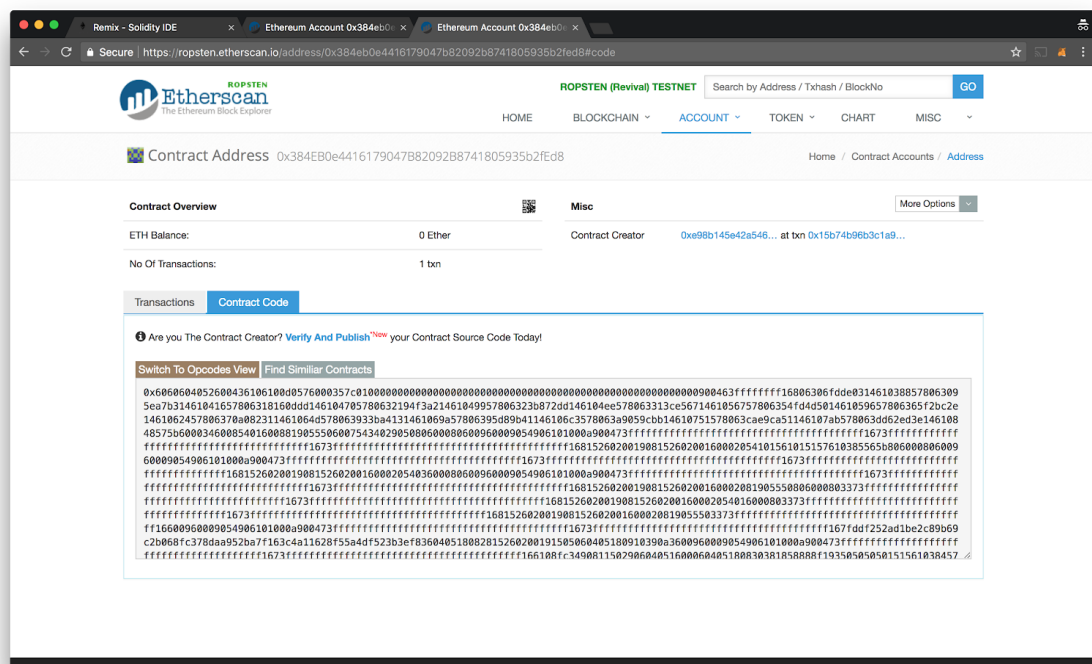


**Congrats! You just deployed a smart contract correctly!!! But hold on, your job isn't done yet. We need to do some verification process.**

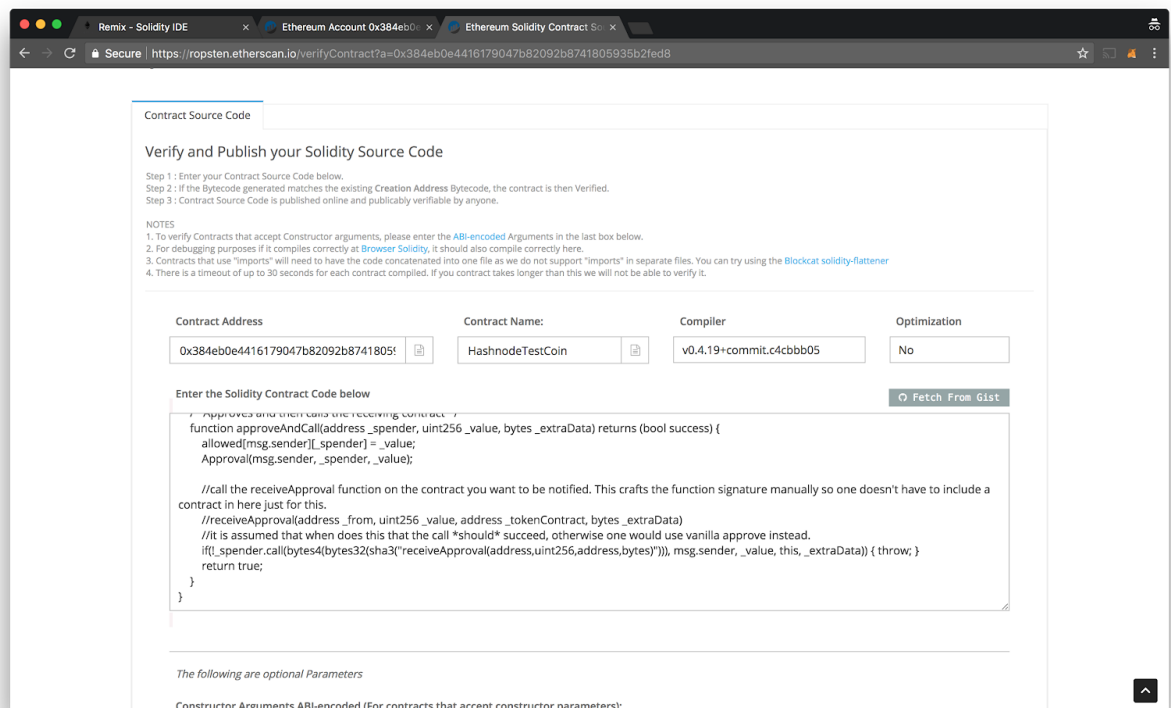
- Copy that smart contract address and open Metamask wallet again. Go to Metamask -> Add Token and paste the address. It looks like this:



- Hit Add and you should now see all the initial supply for the Coin/Token you created.
- Go to contract address on etherscan website and click on contract tab:



## 5. Click on verify and publish button:



## 6. Fill up the details:

- Compiler version used in your Remix Browser. If not sure, go to remix browser and check.
- Contract Name (Again should be in your contract code)
- Optimization as No

## 7. Last Step: Click on verify and Publish. If successful, it will generate bytecode and ABI as following:

