STREAKK

# Streakk Chain:

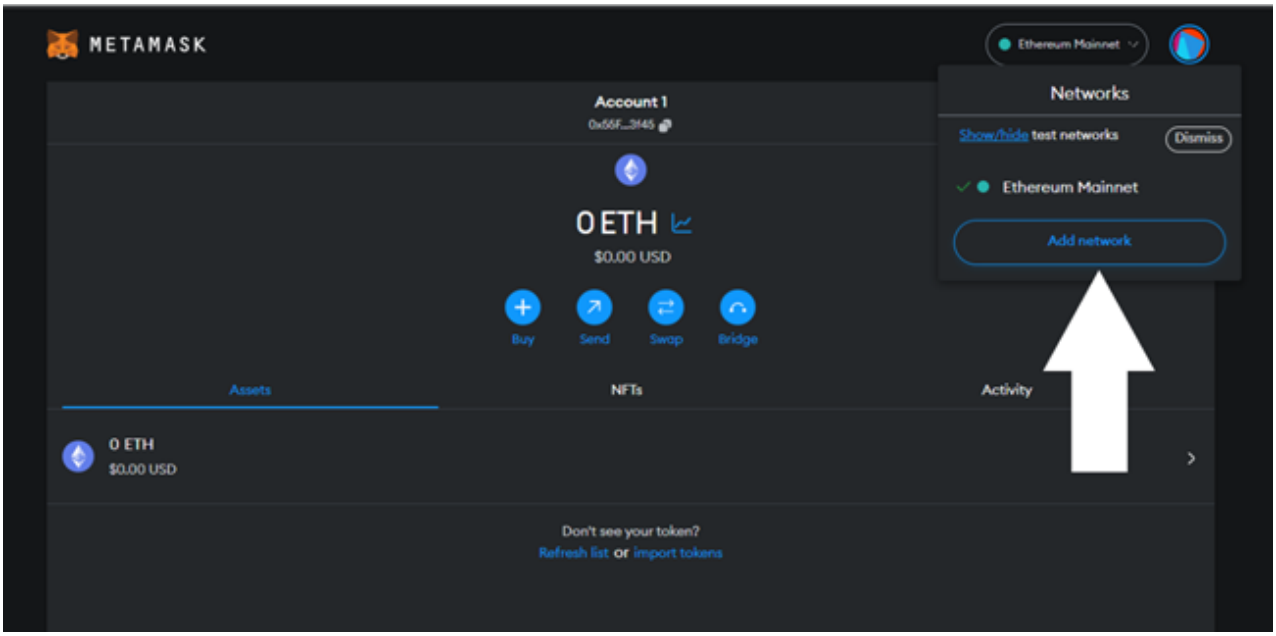
## Build on the Streakk Chain

# Setting up Streakk Chain on Metamask

To perform transactions on the Streakk Chain, you would require Streakk coins (STKC) to pay for the transaction processing and contract deployment fee. Follow the instructions given in this guide to setup a Streakk wallet and deploy a smart contract on Streakk Chain.

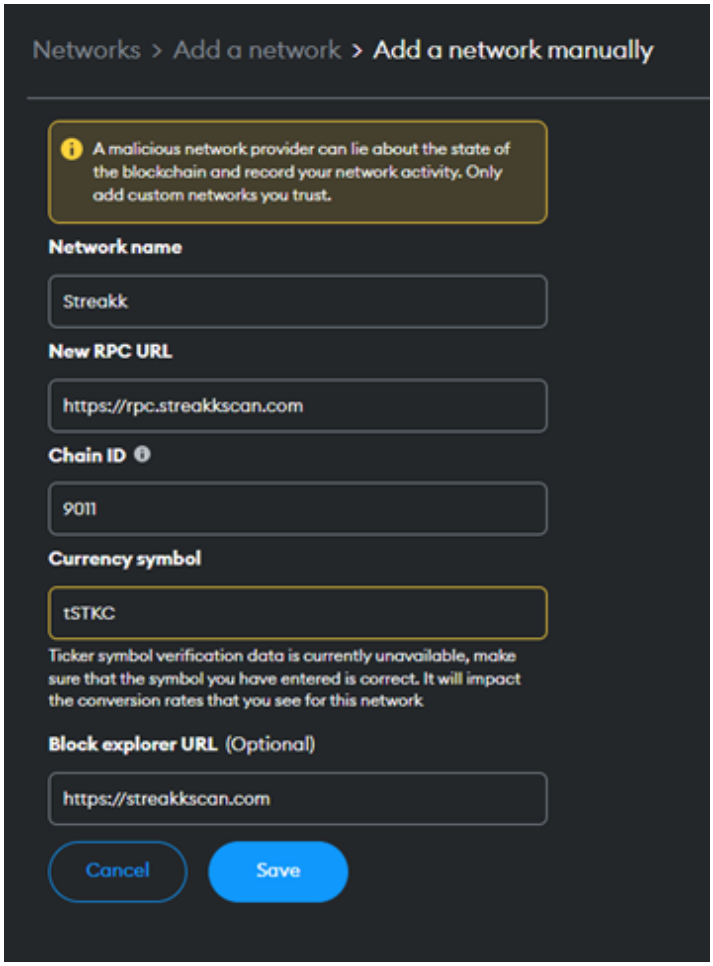
**Method:** Add the Streakk network manually
**Note:** *Please make sure you have already installed Metamask!*

**Step 1:** click on the Network selection dropdown and then click on “Add network” and then “Add a network manually”



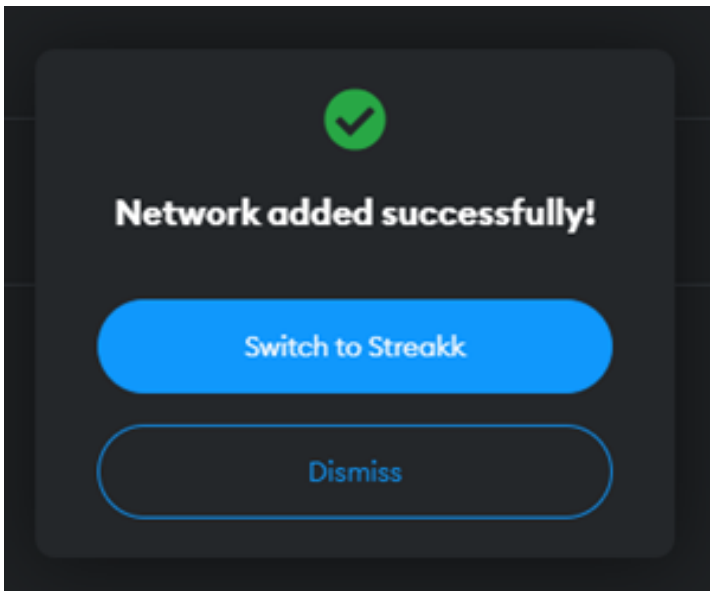

**Step 2:** Enter the following details and save

| PROPERTIES | NETWORK DETAILS |
|---|---|
| Network Name | Streakk |
| New RPC Url | https://rpc.streakkscan.com |
| Chain ID | 9011 |
| Currency Symbol | tSTKC |
| Explorer URL | https://streakkscan.com |




Step 3: Network added successfully

Once added, click on the “switch to streakk” button.

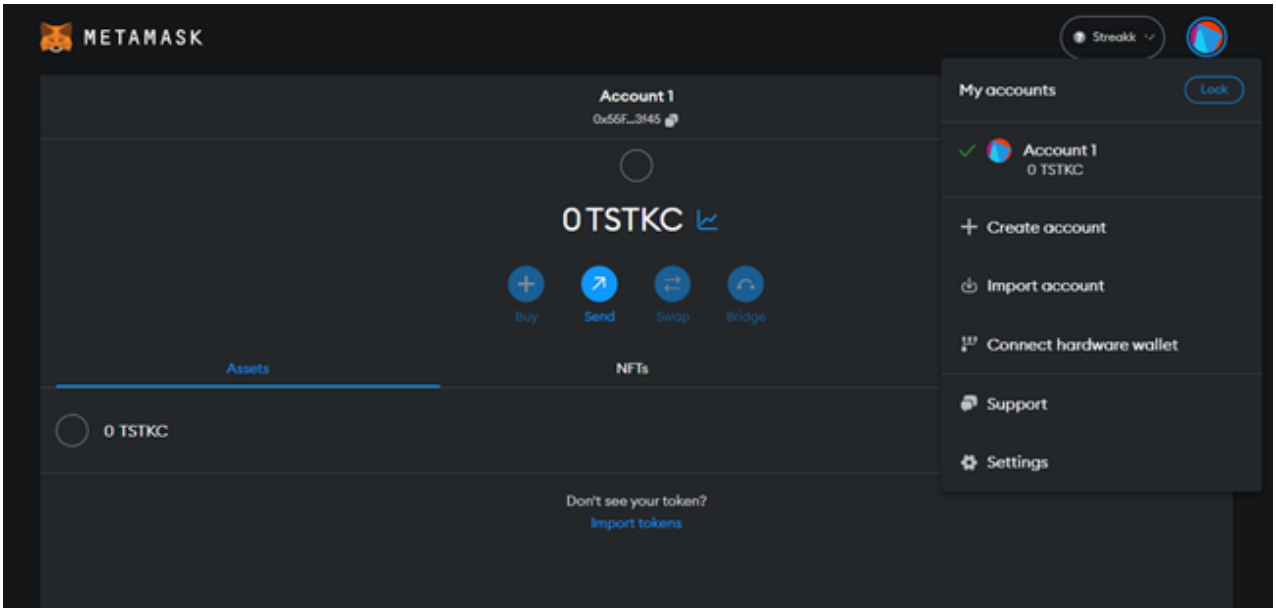


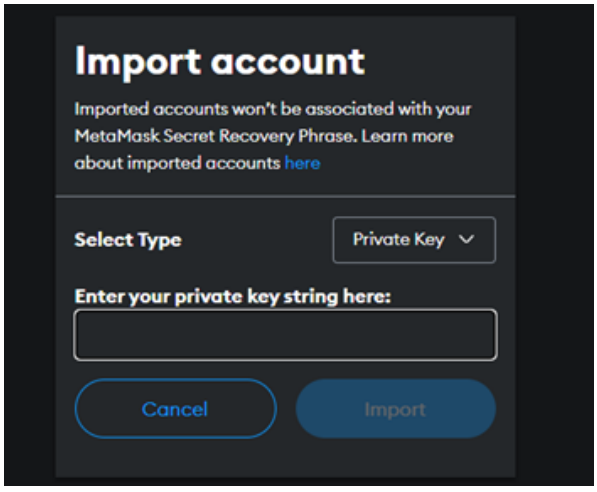
# Claiming Testnet Streakk tokens(tSTKC) from Faucet

### Step 1: Import Wallet

- You will be provided with a default wallet at the time of wallet creation and if you don't know what to use, you can import an existing wallet. You can do so by selecting the “user image” icon and clicking on “Import Account”.

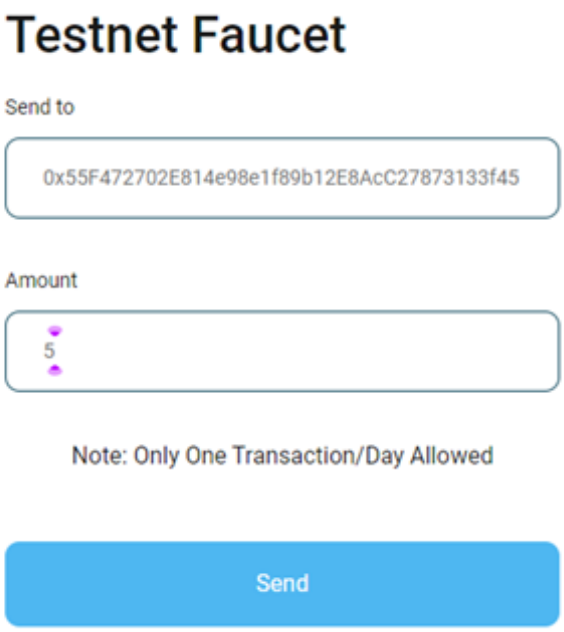
- Import your wallet by using the private key.



### Step 2: Claming Testnet Streakk Coins from Faucet

- Copy the public address of the account in which you want to claim tSTKC tokens.
- Go to Streakk Faucet: https://streakkscan.com/faucet
- Enter your wallet's public address and enter the amount of Streakk coins you require.
- Clicking on send you will receive the testnet tokens to play around the Streakk ecosystem.

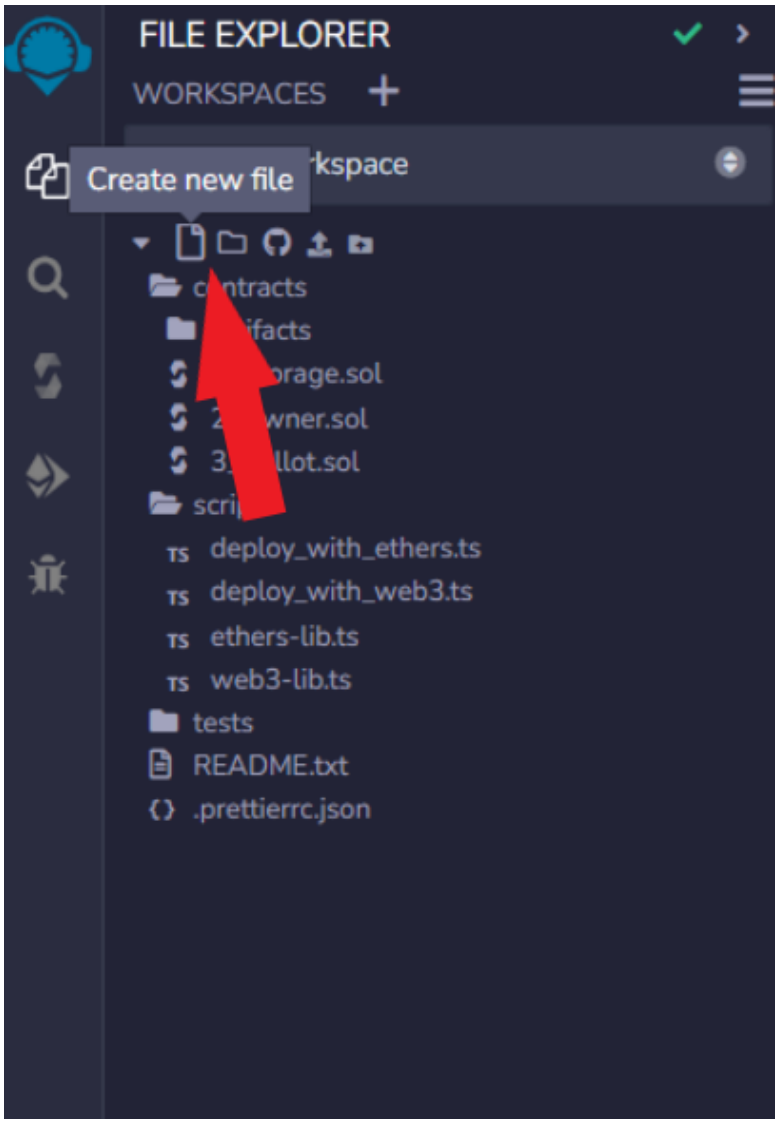


# Deploying a Contract on Streakk Blockchain using Remix IDE

**Step 1: Access the Remix IDE Platform**

The user can access the Remix IDE Platform by visiting this link: https://remix.ethereum.org

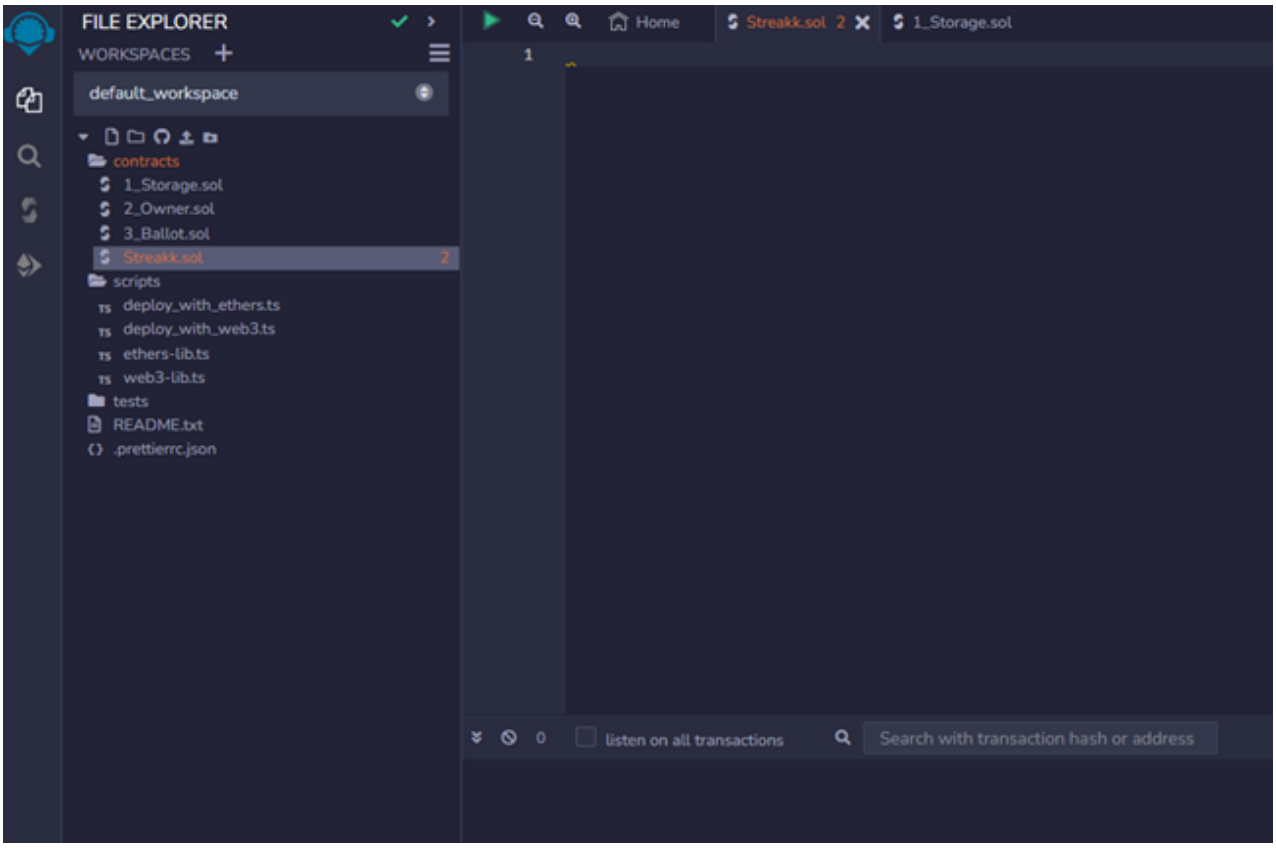
**Step 2: Go to Workspace from the left-panel menu and choose “Create New File” icon.**



### Step 3: Create a "New File" under contracts folder with the name “streakk.sol”

- The new file will be created and the main panel will show blank.

**Note:** *.sol is the extension of solidity files.*



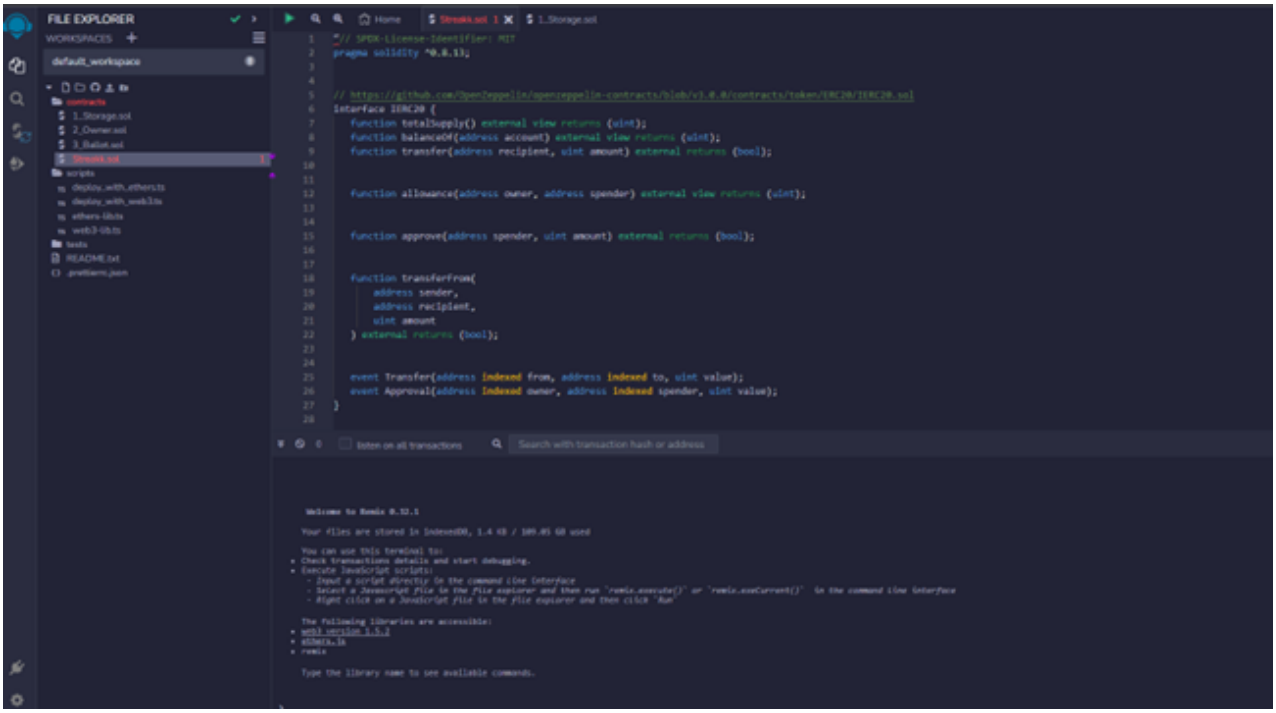

### Step 4: Code on Main Panel

- You can either paste your existing code in the main panel of streakk.sol file or upload your contrat file in the Workspace
- You could also use the below sample ERC-20 code

Code snippet:

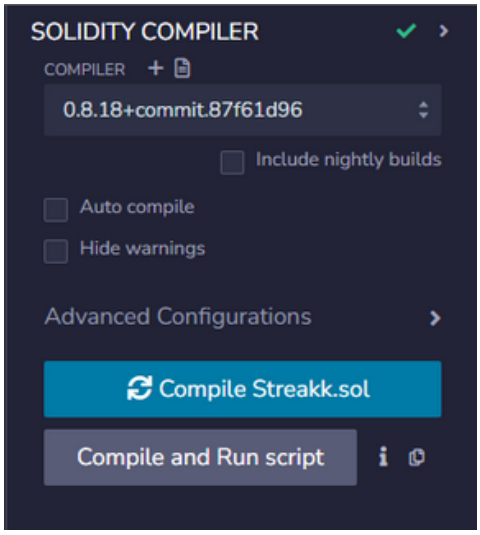
```
“// SPDX-License-Identifier: MIT
pragma solidity ^0.8.13;

// https://github.com/OpenZeppelin/openzeppelin-
contracts/blob/v3.0.0/contracts/token/ERC20/IERC20.sol
interface IERC20 {

 function totalSupply() external view returns (uint);
  function balanceOf(address account) external view returns (uint);
  function transfer(address recipient, uint amount) external returns (bool);

  function allowance(address owner, address spender) external view returns
(uint);

  function approve(address spender, uint amount) external returns (bool);

  function transferFrom(
      address sender,
      address recipient,
      uint amount
  ) external returns (bool);

  event Transfer(address indexed from, address indexed to, uint value);
  event Approval(address indexed owner, address indexed spender, uint
value);
}

contract ERC20 is IERC20 {
  uint public totalSupply;
  mapping(address => uint) public balanceOf;
  mapping(address => mapping(address => uint)) public allowance;
  string public name = "Solidity by Example";
  string public symbol = "SOLBYEX";
  uint8 public decimals = 18;
```

```
  function transfer(address recipient, uint amount) external returns (bool) {
      balanceOf[msg.sender] -= amount;
      balanceOf[recipient] += amount;
      emit Transfer(msg.sender, recipient, amount);
      return true;
  }

  function approve(address spender, uint amount) external returns (bool) {
      allowance[msg.sender][spender] = amount;
      emit Approval(msg.sender, spender, amount);
      return true;
  }


  function transferFrom(
      address sender,
      address recipient,
      uint amount
  ) external returns (bool) {
      allowance[sender][msg.sender] -= amount;
      balanceOf[sender] -= amount;
      balanceOf[recipient] += amount;
      emit Transfer(sender, recipient, amount);
      return true;
  }


  function mint(uint amount) external {
      balanceOf[msg.sender] += amount;
      totalSupply += amount;
      emit Transfer(address(0), msg.sender, amount);
  }


  function burn(uint amount) external {
      balanceOf[msg.sender] -= amount;
      totalSupply -= amount;
      emit Transfer(msg.sender, address(0), amount);
  }
}
```

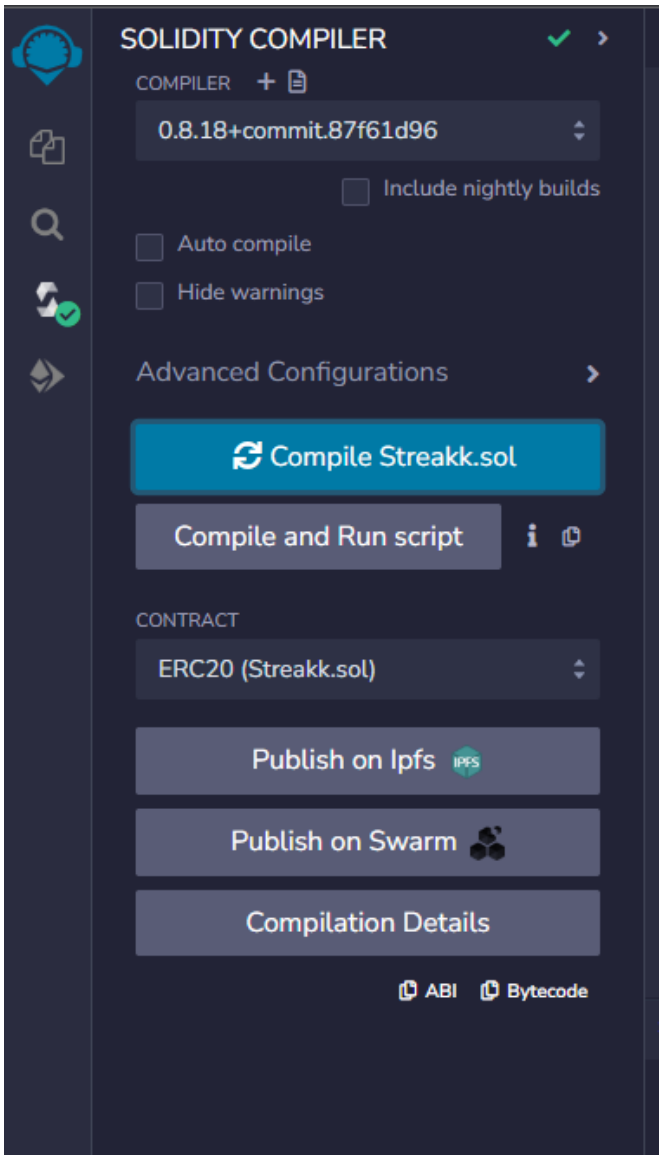


## Step 5: Compilation

- Go to the Solidity Compiler from the left-panel menu
- To select a contract to compile, choose a file in the File Explorer. Or if there are several files open, make sure the one you want to compile is the active file in the Editor.
- If there is an active file chosen in the file explorer, then the solidity computer will look like this:

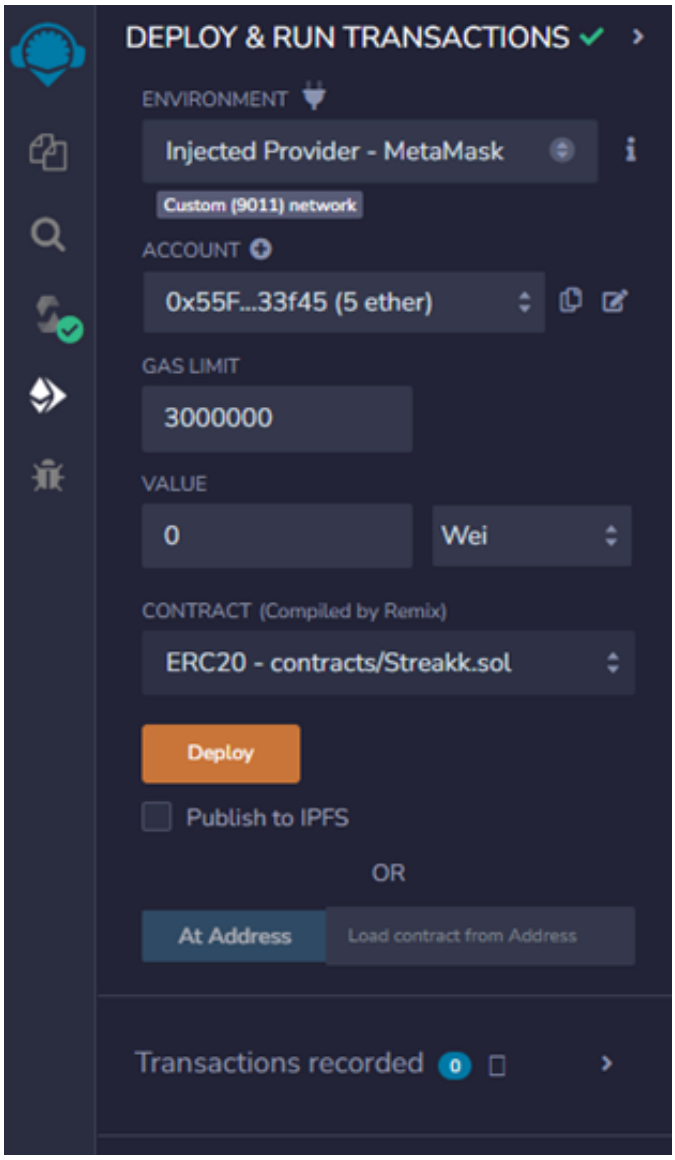



- Click on “Compile streakk.sol” to compile the contract file.
- After succcssfull compilation, the section will look like this




### Step 6: Deployment

- Go to deploy & run transaction sidebar from the left-panel menu.
- To use this module, you need to have a contract compiled.
- So, if there is a contract name in the CONTRACT select box (the select box is under the VALUE input field), you can use this module.
- Select environment: Injected Provider - Metamask. We will be authorizing transaction fee using metamask wallet.
- Injected Provider: For connecting Remix IDE to an injected web3 provider. The most common injected provider is Metamask.
- Hardhat Provider: For connecting Remix IDE to a local Hardhat test chain.
- Ganache Provider: For connecting Remix IDE to a local Truffle Ganache test chain.




- Click on “Deploy”
- Confirm the transaction on metamask access request window:

- Once the transaction is confirmed, the deployment details - contract address and transaction will be visible in the terminal:


STREAKK