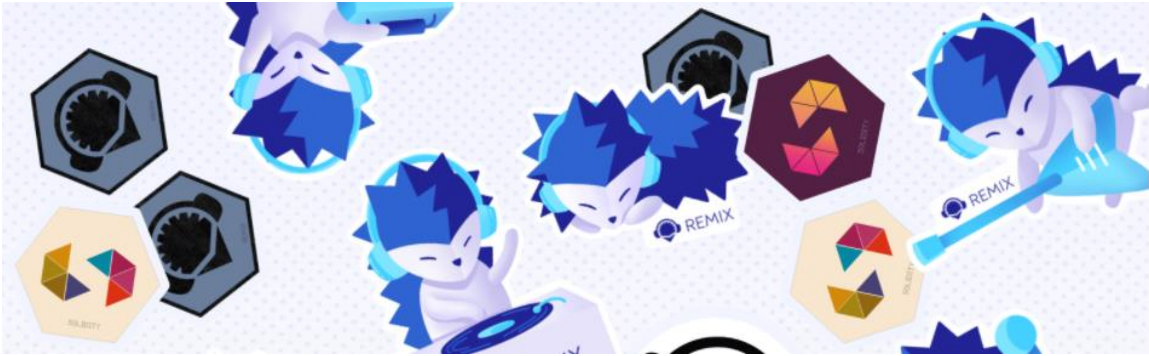


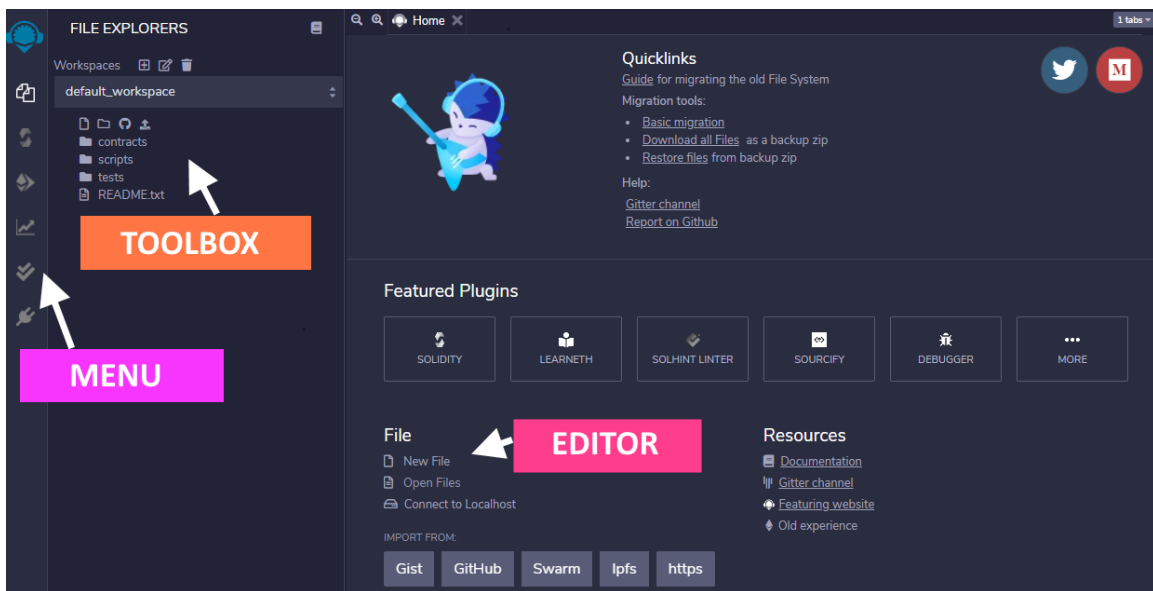
Deploying Smart Contracts



Ganache and Metamask is required to run this activity.

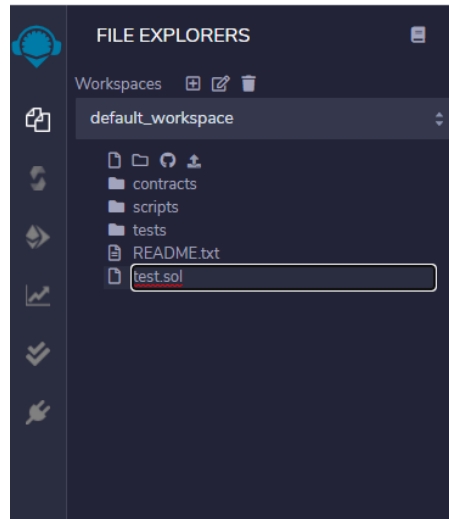
You write code at the Remix editor hosted at: <https://remix.ethereum.org/>

Full documentation: <https://remix-ide.readthedocs.io/en/latest/index.html>

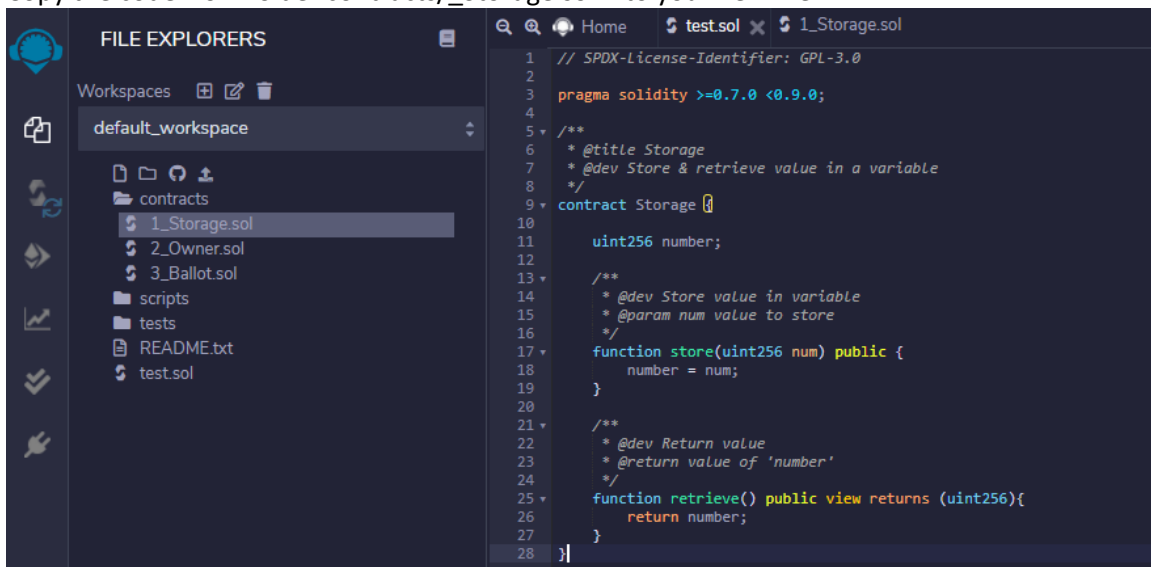


Click on New File to open the File Editor, an empty file will appear in the default_workspace.

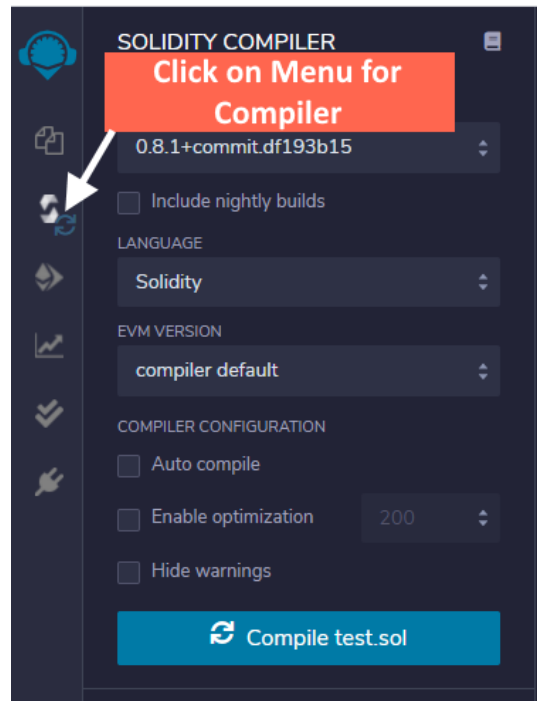
Key in a contract file name.



Copy the code from folder contracts/_Storage.sol into your new file.



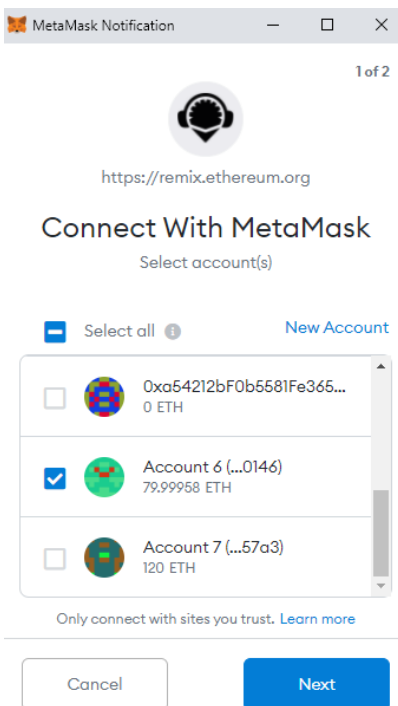
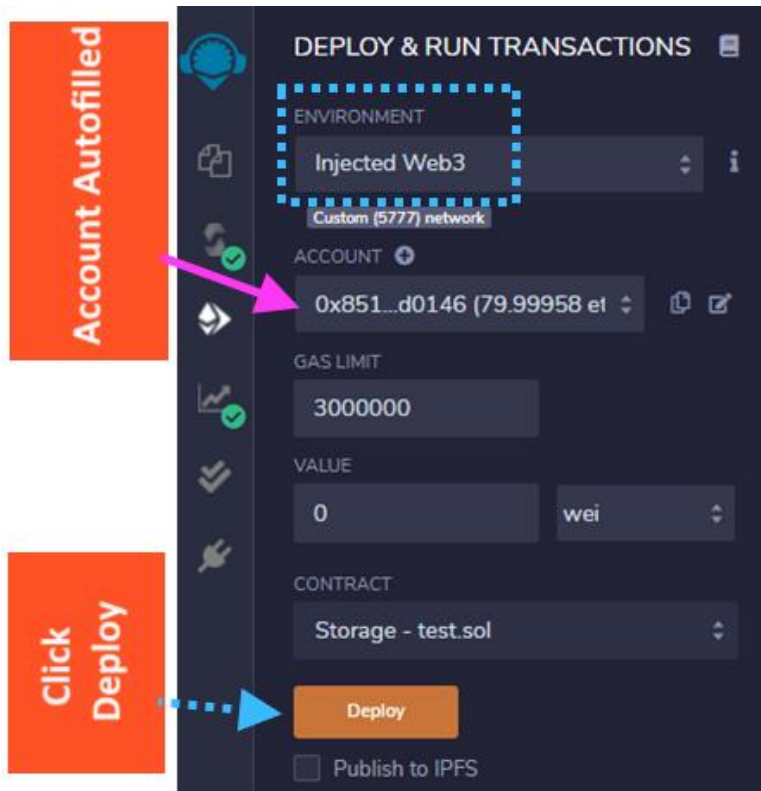
Click on the Compiler and compile the code.



After compiling successfully, click on the next menu item: **Deploy and Run**.
Select:

Injected Web3 for Environment.

You will be asked to connect to Metamask. Select the account that is available at Ganache. You will have the account number filled automatically in Remix.



< Back

2 of 2



https://remix.ethereum.org

Connect to Account 6 (...0146)

Allow this site to:



View the addresses of your permitted accounts (required)

Only connect with sites you trust. [Learn more](#)

Cancel

Connect

After connecting to Metamask, click at Remix:

Deploy

The Metamask window will appear asking to confirm the deployment of contract.

MetaMask Notification

Localhost 7545

Account 6 → New Contract

https://remix.ethereum.org

CONTRACT DEPLOYMENT

0

DETAILS DATA

GAS FEE

0.002376

No Conversion Rate Available

Gas Price (GWEI)

20

Gas Limit

118819

AMOUNT + GAS FEE

TOTAL

0.002376

No Conversion Rate Available

Reject Confirm

Go back to Ganache, at the Transactions will you will see **contract creation** at the transaction.

The screenshot shows the Ganache interface with the 'TRANSACTIONS' tab selected. The transaction details are as follows:

TX HASH	FROM ADDRESS	CREATED CONTRACT ADDRESS	GAS USED	VALUE
0x3d543661c306479d5e08419204e597bdbdfbfaa8c77c9c44af89809f297e4d1	0x8512dfBfa600d0C72A92659A7c8019ECBcd0146	0x998713BFa908b0c4Eb20a09Fe7F99385B0B46B02	118819	0

A red button labeled 'CONTRACT CREATION' is visible next to the transaction hash.

Back at Remix, you can see the address that the Contract (named Storage) is deployed.

The screenshot shows the Remix IDE interface with the 'CONTRACT' tab selected. The contract name is 'Storage - test.sol'. A red box labeled 'Contract Address' with a pink arrow points to the 'Deployed Contracts' section, which shows the address '0x998...46B02 (BLOCKCH)'. The 'Deploy' button is highlighted in orange.

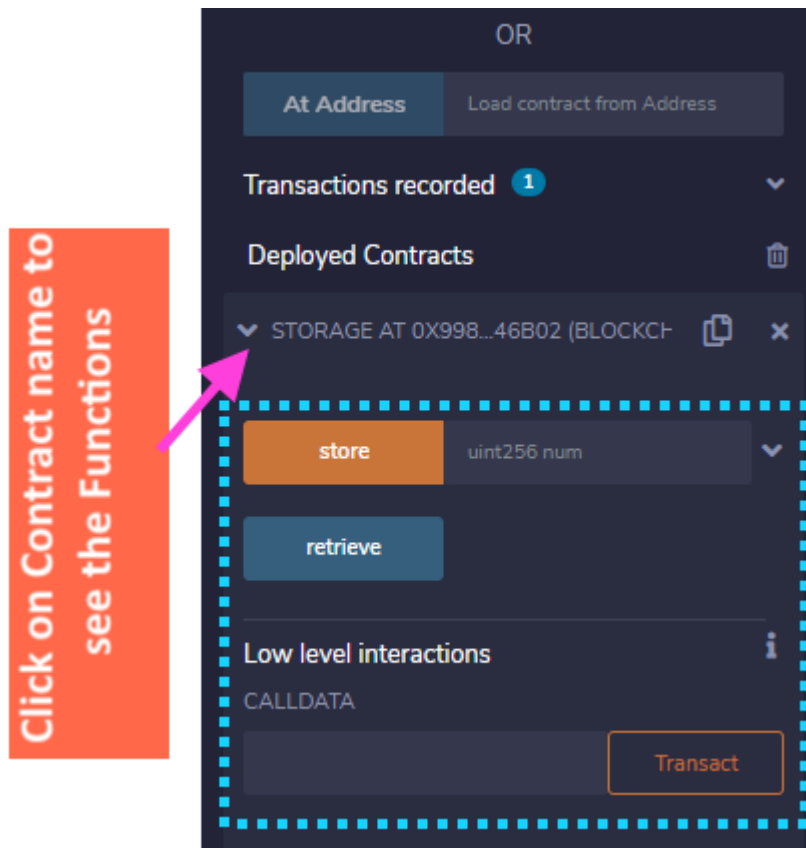
You can also confirm the address is actually created at Ganache.

The screenshot shows the Ganache interface with the 'TRANSACTIONS' tab selected. The transaction details are as follows:

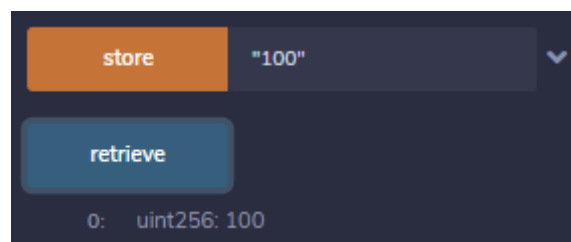
TX HASH	SENDER ADDRESS	CREATED CONTRACT ADDRESS	VALUE	GAS USED	GAS PRICE	GAS LIMIT	MINED IN BLOCK
0x3d543661c306479d5e08419204e597bdbdfbfaa8c77c9c44af89809f297e4d1	0x8512dfBfa600d0C72A92659A7c8019ECBcd0146	0x998713BFa908b0c4Eb20a09Fe7F99385B0B46B02	0.00 ETH	118819	20000000000	118819	2

A red button labeled 'CONTRACT CREATION' is visible next to the transaction hash.

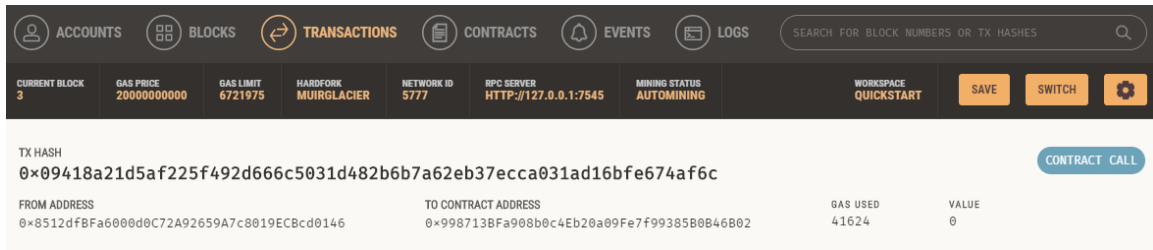
Go back to Remix to call the contract from an account. Click on the Contract name to expand to its functions (methods).



Key in a number at **store**: 100. Metamask will ask for confirmation. Click **retrieve** to see the value that was previously stored.



Transactions at Ganache registered a contract call.



What the Code does

The code simply stores a number for the associated account. Here are some questions that you might ask yourself:

1. What is the sender's address?
2. What is the contract address?
3. Can the contract address be changed?
4. Who is the contract owner?
5. Where is the number stored?
6. Can the contract store a number for another account?
7. Does the store method create a transaction in a block?
8. What happens when the store is called more than once?
9. Does the retrieve method create a transaction block?
10. Is creating a contract easy?

Exercise

Modify the code so that you have an **add** method that adds to the existing store value instead of overwriting it.

Compile it and check that it is running correctly.

Conclusion

You have successfully executed a smart contract that stored a number on a private Ethereum blockchain. This number could well be a token or a crypto currency. This is how tokens are created.

Make sure you understand all the steps, as we will be using them a fair bit. We will discuss further details on Contract writing in the following weeks.

Answer: 3:N;6:N;7:Y;9:N;10:Y