Module 3

Ethereum and Smart Contracts

Syllabus

3.1 **Ethereum :** History, Components, Architecture of Ethereum, Consensus, Miner and mining node, Ethereum virtual machine, Ether, Gas, Transactions, Accounts, Patricia Merkle Tree, Swarm, Whisper and IPFS, complete transaction working and steps in Ethereum, Case study of Ganache for Ethereum blockchain. Exploring etherscan.io and ether block structure, Comparison between Bitcoin and Ethereum

3.2 **Smart Contracts :** history, characteristics, working of smart contracts, types, Oracles, Structure & Limitations. + **Solidity programming :** set-up tools and installation, Basics, functions, Visibility and Activity Qualifiers, Ethereum networks, solidity compiler, solidity files and structure of contracts, data types, storages, array, functions, Developing and executing smart contracts in Ethereum. Smart Contracts Use cases, Opportunities and Risk.

3.1 Public Blockchain System 3-2
GQ. What is a public blockchain ? State some characteristics of public blockchain. 3-2
3.1.1 Introduction 3-2
3.1.2 Blockchain layers 3-2
GQ. Explain the relationship between different blockchain layers. 3-2
3.1.3 Ethereum Blockchain 3-3
3.1.3.1 Brief History of Ethereum 3-3
GQ. Explain in brief the history of Ethereum. 3-3
3.1.3.2 Architecture of Ethereum 3-4
GQ. Describe the architecture of Ethereum. 3-4
3.1.3.3 Account Types, Gas and Transactions 3-5
3.1.4 Ethereum Smart Contracts 3-5
GQ. What is an Ethereum virtual machine (EVM) ? 3-6
3.2 Metamask 3-9
GQ. Compare Bitcoin and Ethereum. 3-45
3.3 Smart Contracts 3-46
GQ. What is a smart contract ? Explain its characteristics. 3-46
3.3.1 Introduction 3-46
3.3.2 Smart Contract 3-46
GQ. How crowdfunding platforms can be managed using smart contracts ? 3-46
GQ. Elaborate on different types of smart contracts. 3-49
GQ. What is an oracle? State and explain different types of oracles. 3-50
3.4 Smart Contracts in Ethereum 3-51
GQ. Write a short note on : Smart contracts in Ethereum. 3-51
3.5 Smart Contracts in Industry 3-51
GQ. State and explain some use cases of smart contracts in industry. 3-51
3.6 Introduction to Programming : Solidity Programming 3-54
- **Chapter Ends** 3-89

3.1 PUBLIC BLOCKCHAIN SYSTEM

GQ. What is a public blockchain ? State some characteristics of public blockchain.

3.1.1 Introduction

Blockchain is one of the top ten strategic technologies of 2019. As per International Data Corporation (IDC), global expenditure on blockchain solutions is anticipated to reach 9.7 billion dollars in 2021. Blockchain can either be public or private based on whether a network is open or accessible to anyone with an Internet connection. In public blockchain, anyone can join a network, and the users in a private blockchain are unidentified. Blockchain users can download a copy of the ledger, initiate, broadcast, or mine blocks.

Public Blockchain

In a public blockchain, one doesn't need any permission to initiate or read/access a transaction, or participate in a consensus process in order to create a block. There is anonymity maintained in a public blockchain with the help of high cryptographic protocols.

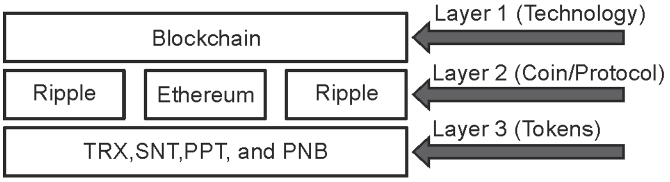
Some of the main characteristics of a public blockchain are as follows :

- **Type of organization :** Public
- **Approach :** Open and visible to all
- **Operations :** Anyone can read, initiate, or receive transactions
- **Immutability :** Secured via hashing
- **Trust :** It is a trust-free system
- **Users :** Anonymous
- **Type of network :** Decentralized
- **Verification :** Any node can participate in the consensus process to validate transactions and create a block
- **Speed of transactions :** Slow
- **Energy consumption :** Very high
- **Scalability :** It has limited scalability, which mainly depends on the growth of the network (i.e., the number of nodes in a network). Other factors could be bandwidth, storage, and an exponential increase in computational power.

3.1.2 Blockchain layers

GQ. Explain the relationship between different blockchain layers.

Relationship between blockchain, cryptocurrency, protocols, and tokens can be comprehended with the help of different layers in a blockchain.



(1B1)**Fig. 3.1.1 : Blockchain layers**

- **Layer 1 :** Blockchain is the technology at the back of a variety of cryptocurrencies such as Bitcoin, Ethereum, Ripple, etc.
- **Layer 2 :** This layer not only defines several coins but also focuses on the protocol of the respective currencies. A protocol is a set of rules that permit people to communicate or transact data with each other to not only reach consensus but also validate transactions within a given network. The protocols used are HTTP, HTTPS, and TCP/IP. Protocols in a blockchain define various consensus and signature mechanisms, use of public keys, and rules related to cryptography.
- **Layer 3 :** It comprises tokens. Tokens serve as the symbol for an underlying contract. For instance, Ethereum has 100s of tokens (i.e., TRX, SNT, PPT, and PNB). Bitcoin and Ripple have no tokens, and hence, they are unable to create smart contracts.

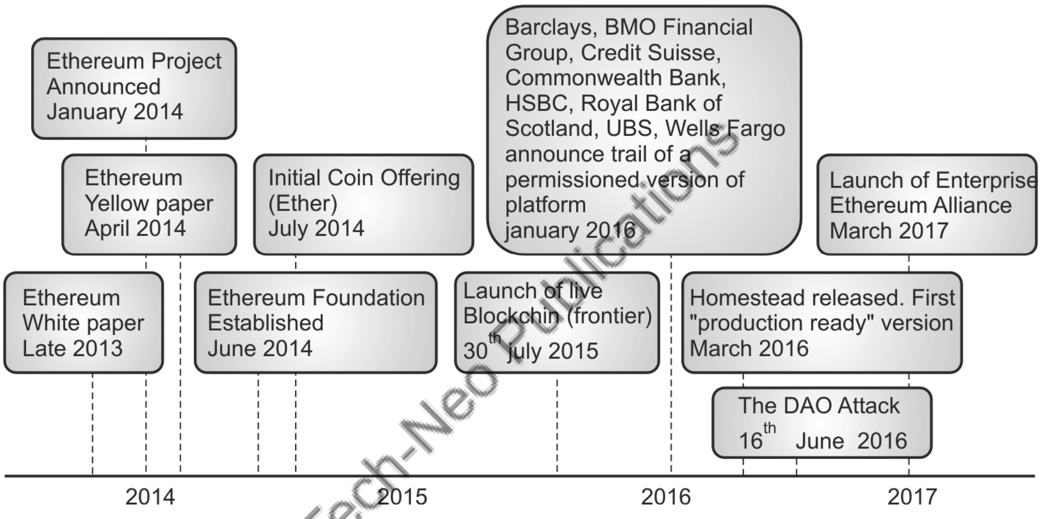
3.1.3 Ethereum Blockchain

3.1.3.1 Brief History of Ethereum

GQ. Explain in brief the history of Ethereum.

Ethereum could be a "do it yourself" platform for most decentralized programs/applications (i.e., DApps). Ethereum platform includes several computers that are completely decentralized. Once a program is setup in an Ethereum system, the computers (nodes) are likely to make sure it implements as composed. Ethereum's programming terminology, i.e., solidity programming, could be used to publish smart contracts, which can be used as a logic to execute DApps.

Ethereum blockchain is a design comprising multiple components that interact and function with each other.


Ethereum Project Announced January 2014
Ethereum Yellow paper April 2014
Initial Coin Offering (Ether) July 2014
Barclays, BMO Financial Group, Credit Suisse, Commonwealth Bank, HSBC, Royal Bank of Scotland, UBS, Wells Fargo announce trail of a permissioned version of platform january 2016
Launch of Enterprise Ethereum Alliance March 2017
Ethereum White paper Late 2013
Ethereum Foundation Established June 2014
Launch of live Blockchin (frontier) 30th July 2015
Homestead released. First "production ready" version March 2016
The DAO Attack 16th June 2016
2014
2015
2016
2017

(1B16)**Fig. 3.1.2 : Brief history of Ethereum**

Miner and mining nodes

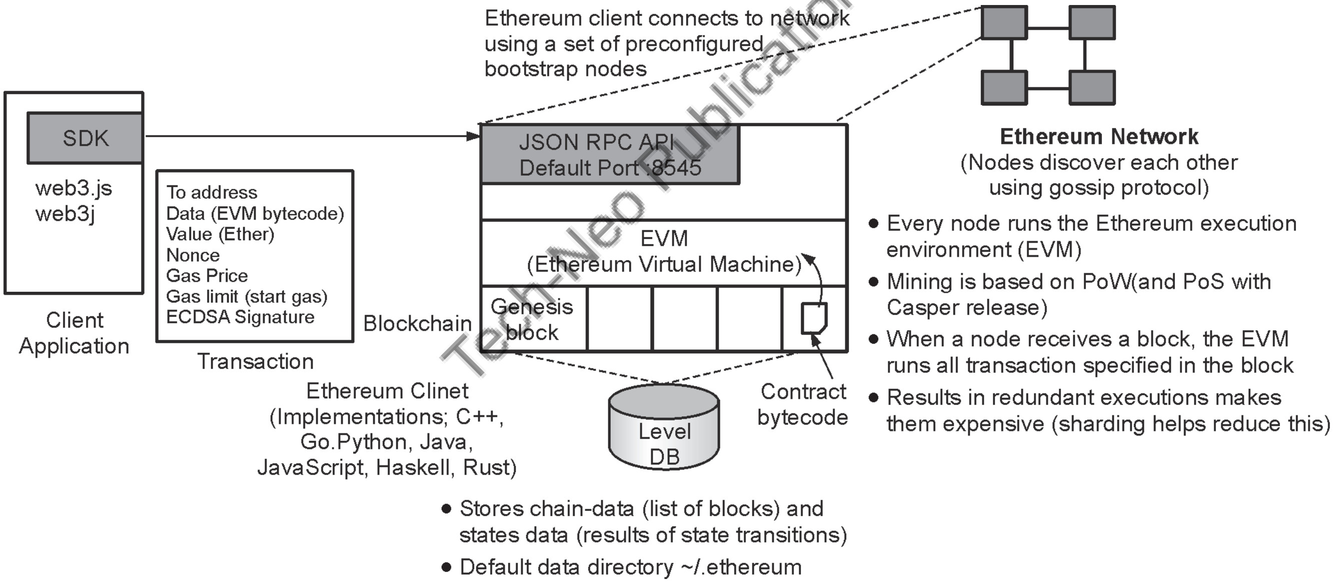
- A miner is responsible for writing a transaction to an Ethereum series. He/she receives a reward when paper transactions are added to a ledger. Here, miners receive two types of rewards, namely benefits for writing a block into a series and accumulative gas price from all transactions in the block.
- Every miner needs to have a backup of the transactions, which includes features like code and other things that exist within the transactions. Data is not stored directly in the blockchain as it is too big. Data is stored in a distributed hash table like InterPlanetary File System (IPFS).
- IPFS provides a hash or content address for the entire content, which helps in retrieving a file from a data storage system. Thus, hash points of the data are stored in the blockchain. There are many miners available in a blockchain network who compete and try to write transactions.
- A single miner can write a block into the ledger, whereas the other miners may not be able to write the block in an existing blockchain, and the conclusion of a miner who will who will eventually compose the block is a challenge. The problem is awarded to each node and every miner attempts to solve the same problem available computing power.

- Lastly, the miner who solves the problem is allowed to write the block comprising transactions to the ledger and receives fiveethers (5 ETH) as a reward. In an Ethereum network, there are two types of nodes:
  - o **Mining nodes :** Nodes that belong to miners.
  - o **Ethereum virtual machines (EVMs) :** They have a unique code attached to it, which is called as a smart contract.

3.1.3.2 Architecture of Ethereum

GQ. Describe the architecture of Ethereum.

- Every Ethereum node runs an EVM and all the smart contracts run within this virtual machine. Client applications can connect to an Ethereum client using a web3j SDK. Transactions have a specific format.
- It has "to" address (i.e., a recipient's address), data pertaining to the smart contracts, Ether value that will be transferred to the recipient, nonce, gas, gas limit, and the signature by a transactor (i.e., ECDSA signature). So the user who is sending the transaction will sign the transaction and send it to blockchain.
- Note that Ethereum will transfer ether as well as execute a smart contract function and that smart contract can hold data on the blockchain.

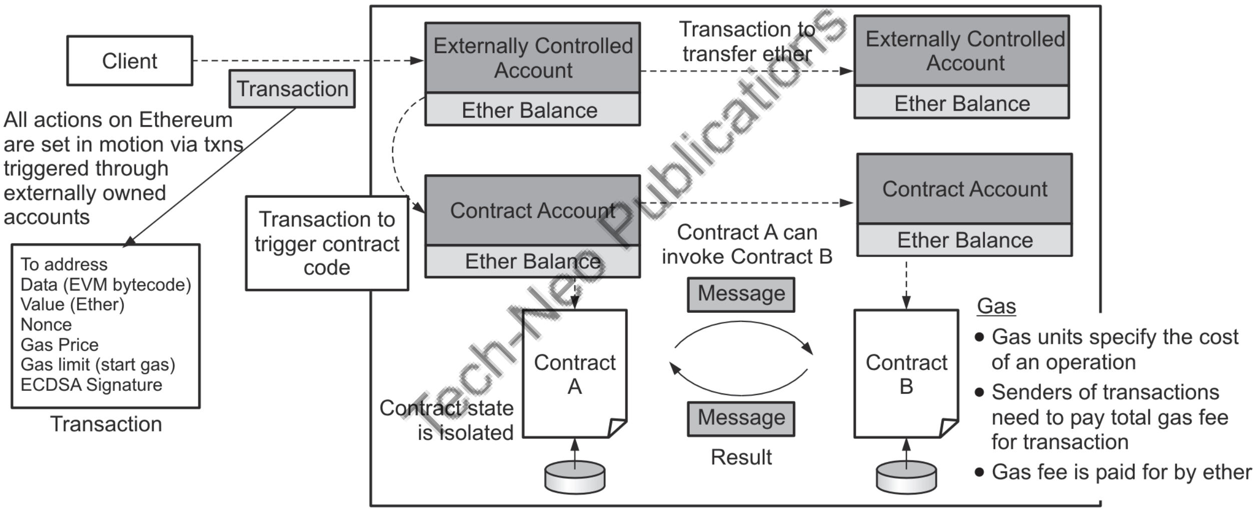

Ethereum client connects to network
using a set of preconfigured
bootstrap nodes
SDK
web3.js
web3j
Client
Application
To address
Data (EVM bytecode)
Value (Ether)
Nonce
Gas Price
Gas limit (start gas)
ECDSA Signature
Transaction
Blockchain
JSON RPC API
Default Port :8545
EVM
(Ethereum Virtual Machine)
Genesis
block
Contract
bytecode
Ethereum Clinet
(Implementations; C++,
Go.Python, Java,
JavaScript, Haskell, Rust)
Level
DB
Ethereum Network
(Nodes discover each other
using gossip protocol)
• Every node runs the Ethereum execution environment (EVM)
• Mining is based on PoW(and PoS with Casper release)
• When a node receives a block, the EVM runs all transaction specified in the block
• Results in redundant executions makes them expensive (sharding helps reduce this)
• Stores chain-data (list of blocks) and states data (results of state transitions)
• Default data directory ~/.ethereum

(1B17)**Fig. 3.1.3 : Ethereum Architecture**

- There exists an Ethereum network of many such nodes that interact with each other and these nodes are the ones that run consensus and determine which transaction blocks should be added next to the blockchain. The consensus is performed through PoW similar to bitcoin.
- When a node receives a block, an Ethereum client will execute all transactions in the block in a sequence and it will then update its ledger once the block is added. It is possible that the client receives the block from multiple people (i.e., redundant executions), but sharding would reduce this problem.
- The database that is used by Ethereum is level DB, which has a key and value store available to smart contracts to store state information. Moreover, the contract bytecode is stored on the blockchain. The state data and list of blocks are stored on the blockchain.

3.1.3.3 Account Types, Gas and Transactions

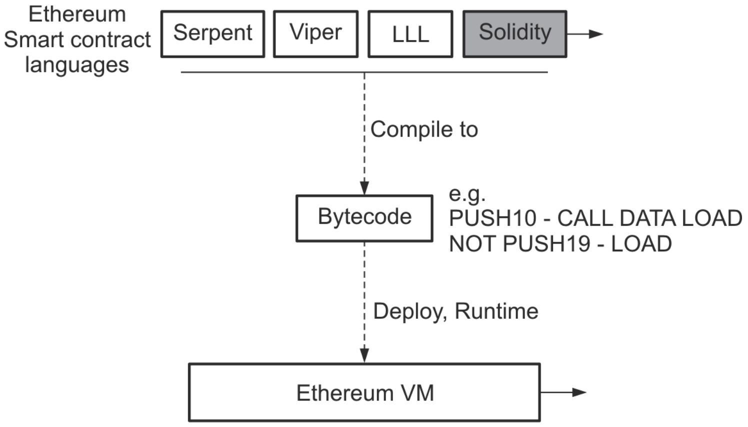
- Ethereum has two types of accounts, i.e., externally controlled account and contract account. Ether balance is stored in the externally controlled account (i.e., a user owns this account). It is possible to transfer ether from one externally controlled account to another externally controlled account.
- Every smart contract will have a contract account and it has an address that is similar to an externally controlled account. Note that a contract account specifically refers to a smart contract and it does not refer to an externally controlled account. For transactions to have an externally controlled account as well as a contract account, ethers can be transferred from one account to the other.
- A specific smart contract can be invoked specified at a particular address and that smart contract will run a particular function defined in the smart contract and that function can update state information on blockchain. Apart from invoking one contract, other contracts can also be invoked. There can be a transfer of balances as well. Invocation of smart contract functions incurs a transaction cost.
- As this is a public permissionless system, there is a need to ensure that people do not spam the network with invalid transactions, so in Ethereum, there is a something called as a gas. So for each transaction, some gas needs to be spent. Until the gas price is paid, a transaction cannot be executed. Every transaction will have a specific gas price. Also, a gas limit can be set so that there is no depletion of gas. The gas fee is paid using ethers. The gas price is borne by the sender of the transaction.


Client
Transaction
All actions on Ethereum are set in motion via txns triggered through externally owned accounts
To address
Data (EVM bytecode)
Value (Ether)
Nonce
Gas Price
Gas limit (start gas)
ECDSA Signature
Transaction
Transaction to trigger contract code
Externally Controlled Account
Ether Balance
Transaction to transfer ether
Externally Controlled Account
Ether Balance
Contract Account
Ether Balance
Contract Account
Ether Balance
Contract A can invoke Contract B
Message
Contract A
Contract state is isolated
Message
Result
Contract B
Gas
Gas units specify the cost of an operation
Senders of transactions need to pay total gas fee for transaction
Gas fee is paid for by ether

(1B18)Fig. 3.1.4 : Account types, gas, and transactions

3.1.4 Ethereum Smart Contracts

- Ethereum allows smart contracts to be written in different languages like Serpent, Viper, LLL, and Solidity. The most popular one is solidity. Solidity is an object-oriented high-level language for writing EVM-based smart contracts. It has a syntax similar to JavaScript. It is executed inside an EVM.
- So, the languages are compiled into a standardized bytecode, and the bytecode runs in an EVM. It offers a support for multiple inheritance. It also supports complex user-defined types through structs.



(1B19)Fig. 3.1.5 : Ethereum smart contracts

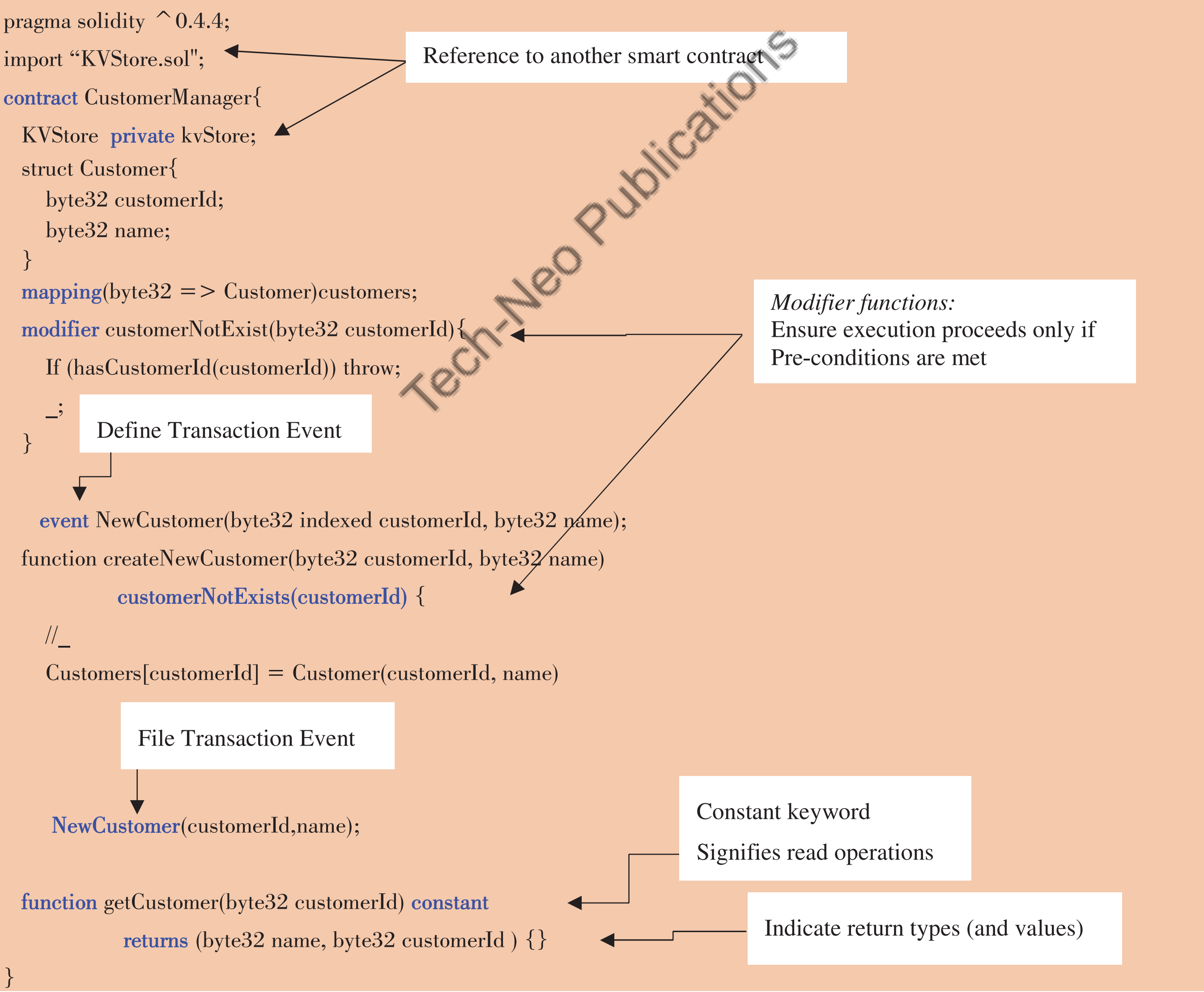
- The programming languages that are defined for Ethereum smart contracts is to reflect the limitations of EVM. For example, basic string manipulation is not supported in a language, but it gets supported through libraries.
- There exits a browser-based IDE with an integrated compiler, and solidity runtime environment can run without any server-side components.
- One of the main aspects in Ethereum blockchain is "*code is law*", which means that there is no auditability or governance or authority who will oversee/administer the code in the blockchain. Whatever the code is, it will be executed in a decentralized manner.

☞ Ethereum Virtual Machine

GQ. What is an Ethereum virtual machine (EVM) ?

- It is a 256 bit virtual machine that is Turing complete, and the computation is intrinsically bounded through gas. It allows execution of EVM bytecode. EVM defines approximately 70 OPCODEs. The set of OPCODEs are interpreted by an interpreter. There exists a JIT interpreter the compiles the bytecode into manageable datatypes and structures.
- Gas is defined for all 70 OPCODEs, and the gas value is set based on the difficulty of the operation (e.g., IF, AND, MUTIPLY, and SEND), which is paid using ether.

[Example of a solidity code]



```
pragma solidity ^0.4.4;
import "KVStore.sol";
contract CustomerManager{
  KVStore  private kvStore;
  struct Customer{
     byte32 customerId;
     byte32 name;
  }
  mapping(byte32 => Customer)customers;
  modifier customerNotExist(byte32 customerId){
     If (hasCustomerId(customerId)) throw;
     _;
  }

   event NewCustomer(byte32 indexed customerId, byte32 name);
  function createNewCustomer(byte32 customerId, byte32 name)
             customerNotExists(customerId) {
     //_
     Customers[customerId] = Customer(customerId, name)

       NewCustomer(customerId,name);

  function getCustomer(byte32 customerId) constant
             returns (byte32 name, byte32 customerId ) {}
}
```

Reference to another smart contract
Modifier functions:
Ensure execution proceeds only if
Pre-conditions are met
Define Transaction Event
File Transaction Event
Constant keyword
Signifies read operations
Indicate return types (and values)

☞ Merkle Patricia Trie

It combines the concepts of Merkle Tree and Patricia Trie, thereby creating an efficient and secure way to organize and verify the state of accounts in a blockchain.

The components of Merkle Patricia Trie are as follows:

- **Trie :** It is a tree-like data structure where each node represents a part of a key or identifier, typically a hash of the data being stored. In a Merkle Patricia Trie, the keys are typically the hexadecimal representations of the Ethereum addresses (account addresses) or the storage keys (in case of smart contracts).
- **Patricia Trie :** It is also known as the radix tree or crit-bit tree, which is a type of trie that optimizes storage by sharing common prefixes among keys. It efficiently stores key-value pairs where keys are usually strings. The "Patricia" part comes from the term "Practical Algorithm to Retrieve Information Coded in Alphanumeric."
- **Merkle Tree :** It is a cryptographic data structure used to verify the integrity of data. In the context of Ethereum, it is used to represent and verify the content of blocks and transactions.
- **Merkle Patricia Trie :** In Ethereum, the state of blockchain, including account balances and contract storage, is represented using a Merkle Patricia Trie. This data structure allows efficient verification of account states and facilitates quick retrieval and updates of account information.

By using a Merkle Patricia Trie, Ethereum can easily verify the state of an account or a contract without needing to store the entire blockchain. This data structure is crucial for the security and efficiency of the Ethereum blockchain.

☞ Swarm

It is a decentralized and distributed storage platform and content distribution service that is part of the Ethereum ecosystem that aims to provide a decentralized alternative to traditional hosting and content delivery services by allowing users to store and access data in a secure and censorship-resistant manner.

The key features of Swarm are as follows:

- **Decentralization :** Swarm is built on the principles of decentralization, where data is distributed across multiple nodes (computers) in the network. This eliminates the need for centralized servers and ensures that data is not controlled by any single entity.
- **Incentive mechanism :** Swarm uses an incentive mechanism to encourage participants to store and provide access to content. Users who contribute resources (such as disk space and bandwidth) to the network are rewarded with cryptocurrency tokens, providing an economic incentive for participation.
- **Content addressing :** Similar to how content is addressed in the InterPlanetary File System (IPFS), Swarm uses content-based addressing. Content is identified by its cryptographic hash, which allows for efficient retrieval and verification.
- **Redundancy and availability :** Swarm ensures data redundancy by storing multiple copies of content on different nodes. This redundancy enhances data availability and fault tolerance, reducing the risk of data loss.
- **Data integrity :** Swarm utilizes Merkle Trees and cryptographic hashing to ensure data integrity. Merkle Trees enable efficient verification of the stored data, and any tampering or corruption can be easily detected.
- **Privacy and security :** Swarm employs encryption and privacy measures to protect data and user privacy. Data is encrypted before being stored on the network, and access control mechanisms are implemented to ensure only authorized users can access specific content.
- **Scalability :** Swarm is designed to be scalable, capable of handling large amounts of data and a high number of users.

Swarm is closely associated with the Ethereum blockchain, and it is intended to complement Ethereum's capabilities by providing a decentralized storage layer for decentralized applications (dApps) and smart contracts. With Swarm, developers can host and serve data required by their applications in a distributed and censorship-resistant manner.

☞ Whisper

It is a peer-to-peer communication protocol that is part of the Ethereum ecosystem. It is designed to provide secure and decentralized messaging between nodes within the Ethereum network. Whisper is primarily used for off-chain communication, allowing Ethereum nodes to exchange messages and data without directly involving the Ethereum blockchain.

The key features of Whisper are as follows :

- **Privacy and encryption :** Whisper messages are end-to-end encrypted, ensuring that only the intended recipients can read the messages. This provides a level of privacy and security for communication within the network.
- **Decentralization :** Like other components of Ethereum, Whisper operates in a decentralized manner, with messages being relayed through a distributed network of nodes. There is no central authority controlling the messaging process.
- **Asynchronous communication :** Whisper enables asynchronous communication, meaning that nodes can send and receive messages even if the recipient is currently offline. Messages are stored and delivered to recipients when they come online, ensuring message delivery even in non-real-time scenarios.
- **Topic-based communication :** Whisper messages are organized into topics, similar to message channels or topics in pub/sub systems. Nodes can subscribe to specific topics to receive messages related to their interests.
- **Message filtering :** Nodes can filter incoming messages based on the topics they have subscribed to, reducing unnecessary message traffic and improving network efficiency.
- **Message TTL (time-to-live) :** Each Whisper message has a TTL, which determines how long the message will be stored and delivered. This helps prevent messages from being stored indefinitely and ensures network resources are used efficiently.

Whisper is especially useful for decentralized applications (dApps) and smart contracts that require off-chain communication or want to implement messaging functionality without relying on centralized solutions. It can be used for various purposes, such as sending notifications, exchanging data, and facilitating communication between different components of a decentralized application.

☞ InterPlanetary File System (IPFS)

It is a peer-to-peer protocol and network designed to create a decentralized and distributed method of storing and sharing files. It provides a more efficient and permanent way to store and retrieve data compared to traditional centralized web hosting and content delivery systems.

The key features of IPFS are as follows :

- **Content-addressable storage :** IPFS uses content-based addressing, meaning that files are identified and accessed using their unique cryptographic hash. This ensures data integrity and allows for efficient caching and sharing of content across the network.
- **Decentralization :** IPFS operates in a decentralized manner, where files are distributed and stored across multiple nodes in the network. This eliminates the need for centralized servers, making the system more resilient and censorship-resistant.

- **Versioning and history:** IPFS retains multiple versions of files, allowing users to access previous versions and maintain a historical record of changes. This is particularly useful for versioning applications and ensuring data permanence.
- **Distributed web:** IPFS can be used as a distributed alternative to the traditional Hypertext Transfer Protocol (HTTP). IPFS provides a peer-to-peer distributed web, where content is served from multiple nodes instead of a single central server.
- **Efficient data transfer:** IPFS utilizes a BitSwap protocol, inspired by BitTorrent, to efficiently exchange data between peers. This peer-to-peer data transfer mechanism optimizes bandwidth usage and reduces duplication of content across the network.
- **Permanent and immutable data:** Content stored on IPFS is immutable, meaning that once it is added to the network, it cannot be changed or tampered with. This property is valuable for applications that require verifiable and auditable data.

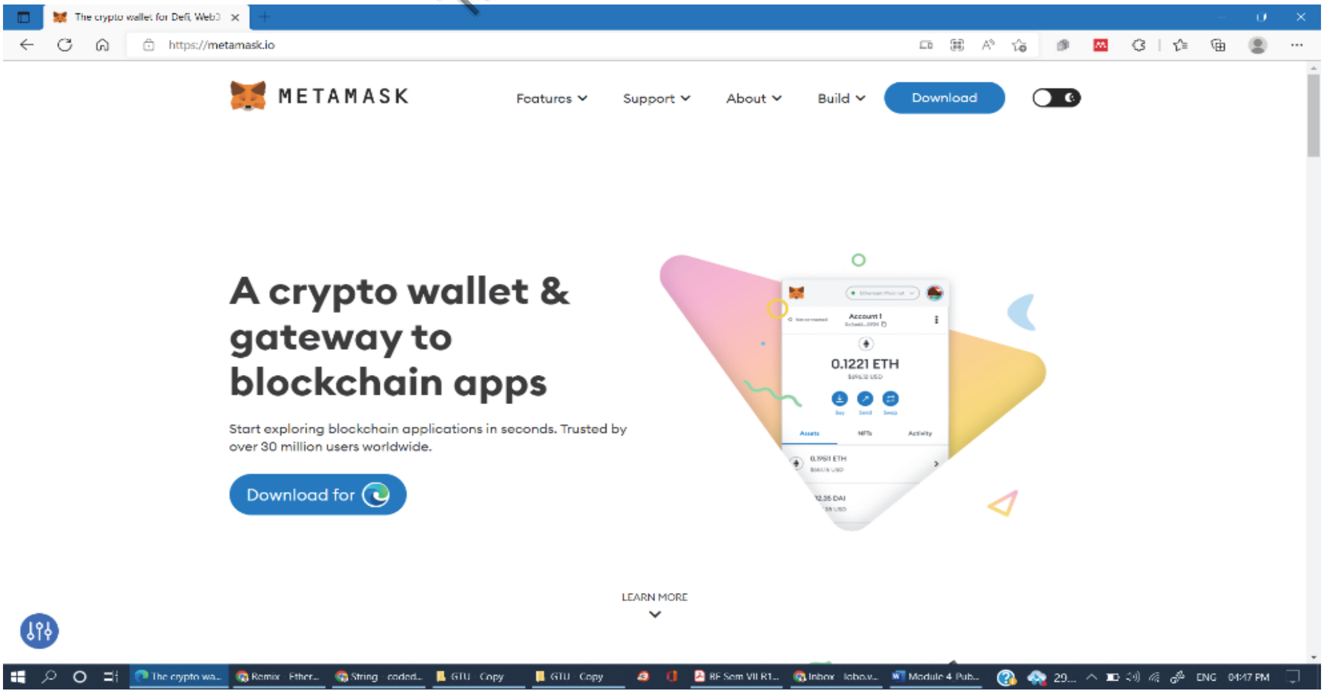
IPFS has gained popularity for various use cases, including decentralized applications (dApps), distributed content delivery, and sharing large files. It can also be used as a backend storage solution for blockchain systems, such as Ethereum, to store off-chain data securely and efficiently.

3.2 METAMASK

Metamask is a browser plugin that hold an Ethereum wallet and connects the computer with the Ethereum network. Metamask can connect and network with other providers who offer free ethers (i.e., test ethers) to accounts created in wallets of Metamask. For instance, an account created in Metamask can be connected with an external network named Ropsten Test Network, which can then inject free ethers (ETH) to a corresponding account.

Setting up Metamask account

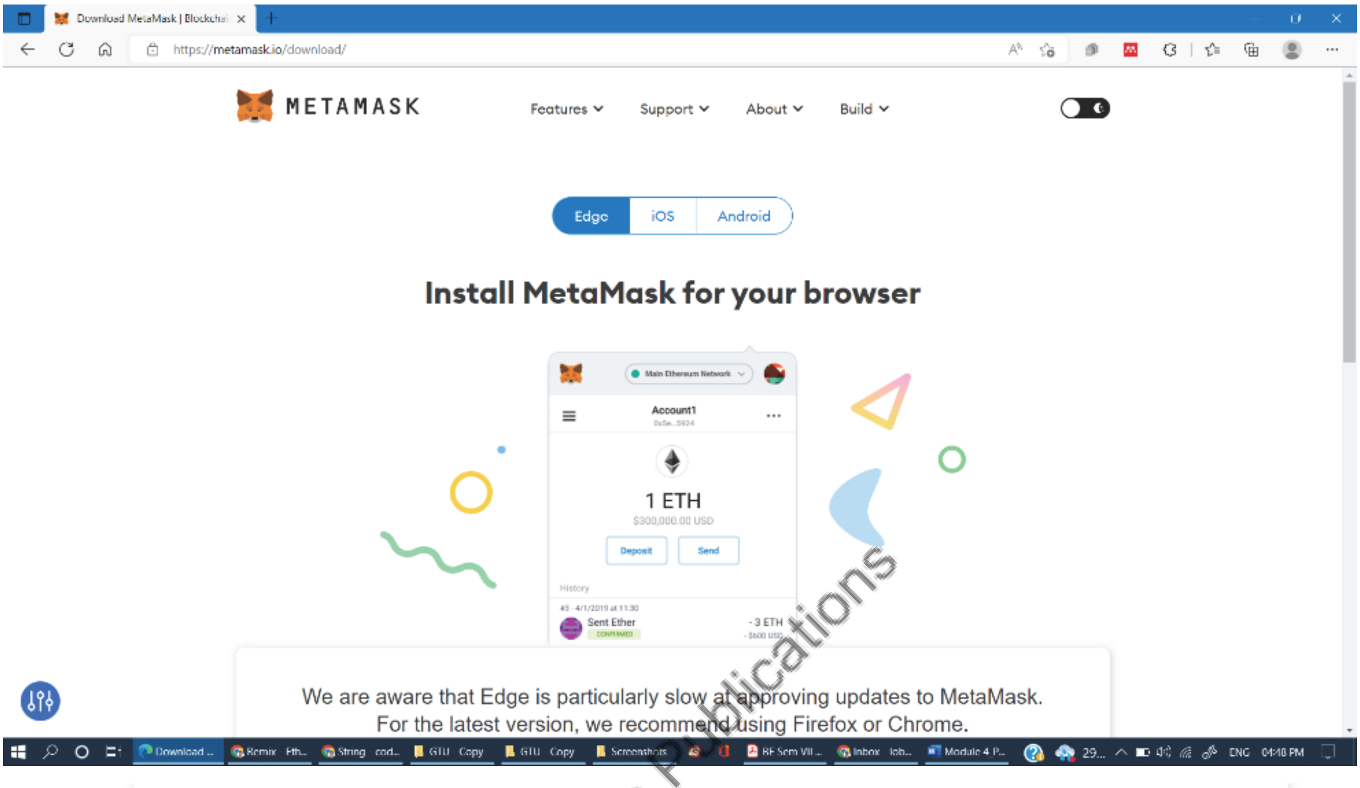
Step 1 : Go to https://metamask.io/ and download MetaMask, which is a crypto wallet and gateway to blockchain applications



A crypto wallet & gateway to blockchain apps

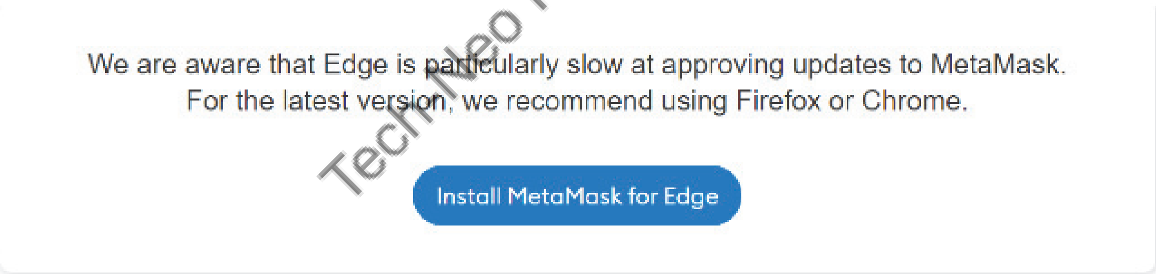
Start exploring blockchain applications in seconds. Trusted by over 30 million users worldwide.

- **Step 2 :** Install Meta Mask for your browser.

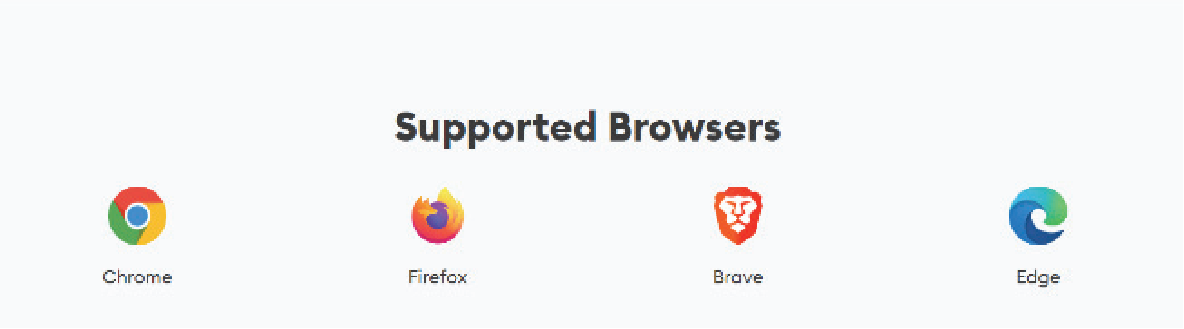


Install MetaMask for your browser

We are aware that Edge is particularly slow at approving updates to MetaMask. For the latest version, we recommend using Firefox or Chrome.

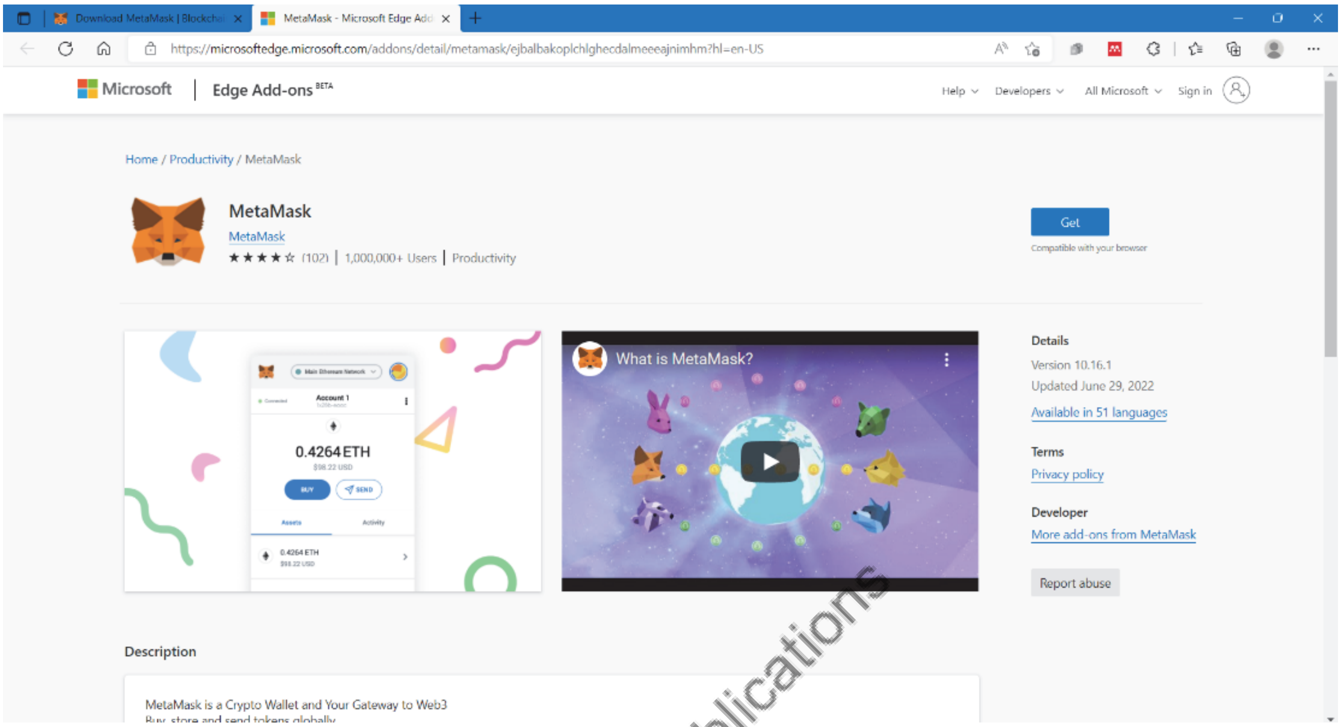


We are aware that Edge is particularly slow at approving updates to MetaMask. For the latest version, we recommend using Firefox or Chrome.



Supported Browsers

- **Step 3 :** You can refer to the version number and other details while installing MetaMask.


Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
https://microsoftedge.microsoft.com/addons/detail/metamask/ejbalbakoplchlghecdalmeeeajnimhm?hl=en-US
Microsoft
Edge Add-ons BETA
Help
Developers
All Microsoft
Sign in
Home / Productivity / MetaMask
Get
0.4264 ETH
BUY
SEND
What is MetaMask?

MetaMask

MetaMask

★★★★☆ (102) | 1,000,000+ Users | Productivity

Compatible with your browser

Details

Version 10.16.1

Updated June 29, 2022

Available in 51 languages

Terms

Privacy policy

Developer

More add-ons from MetaMask

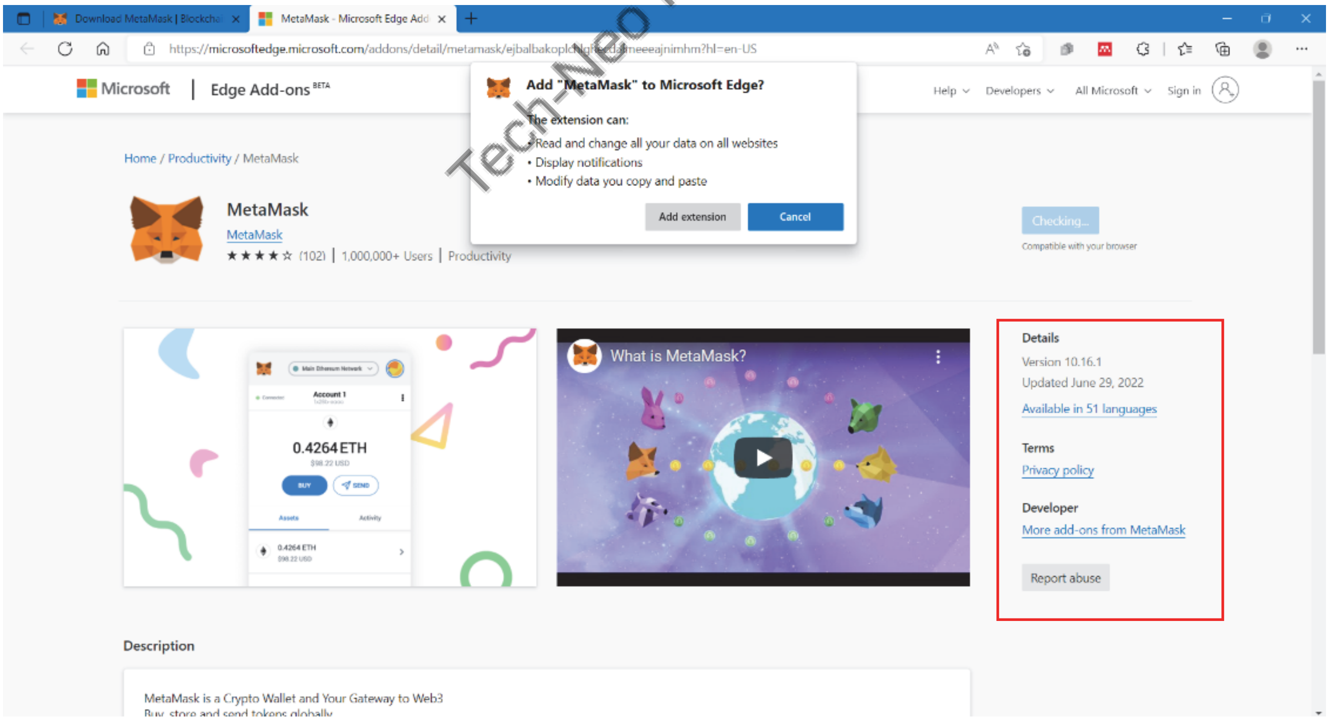
Report abuse

Description

MetaMask is a Crypto Wallet and Your Gateway to Web3

Buy, store and send tokens globally

- **Step 4 :** It is a browser plug-in extension.


Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
https://microsoftedge.microsoft.com/addons/detail/metamask/ejbalbakoplchlghecdalmeeeajnimhm?hl=en-US
Microsoft
Edge Add-ons BETA
Help
Developers
All Microsoft
Sign in
Home / Productivity / MetaMask
Checking...
0.4264 ETH
BUY
SEND
What is MetaMask?

Add "MetaMask" to Microsoft Edge?

The extension can:

- Read and change all your data on all websites
- Display notifications
- Modify data you copy and paste

Add extension Cancel

MetaMask

MetaMask

★★★★☆ (102) | 1,000,000+ Users | Productivity

Compatible with your browser

Details

Version 10.16.1

Updated June 29, 2022

Available in 51 languages

Terms

Privacy policy

Developer

More add-ons from MetaMask

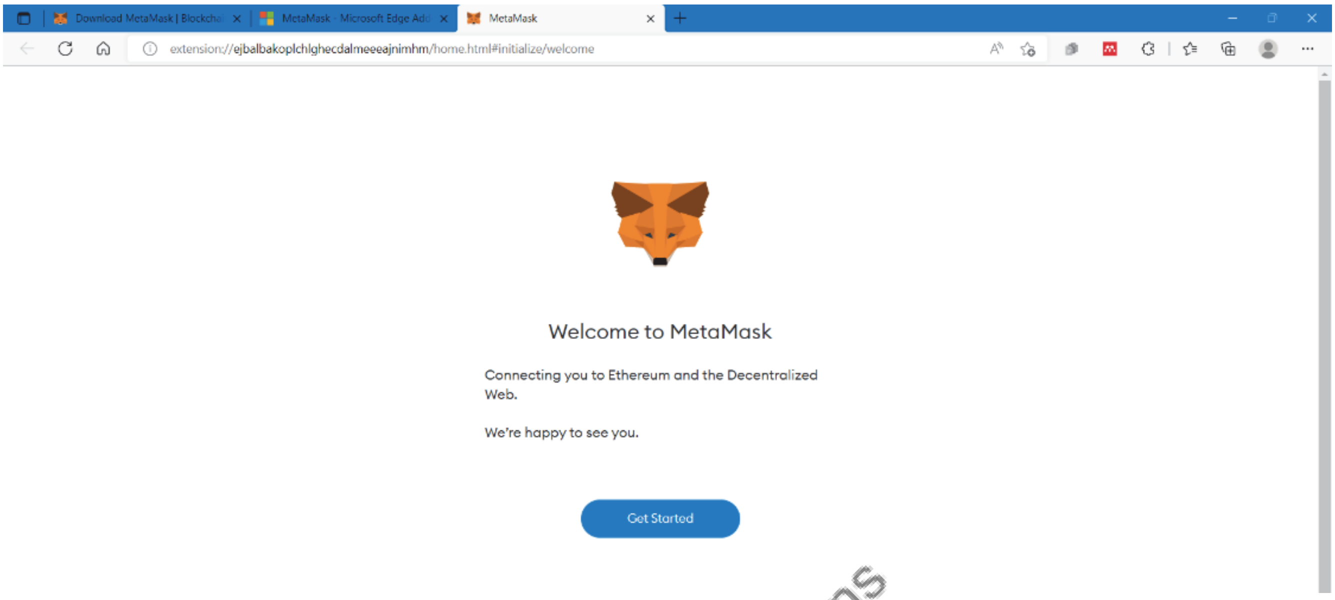
Report abuse

Description

MetaMask is a Crypto Wallet and Your Gateway to Web3

Buy, store and send tokens globally

- **Step 5 :** MetaMask has been successfully installed, which connects us to Ethereum and the decentralized web.

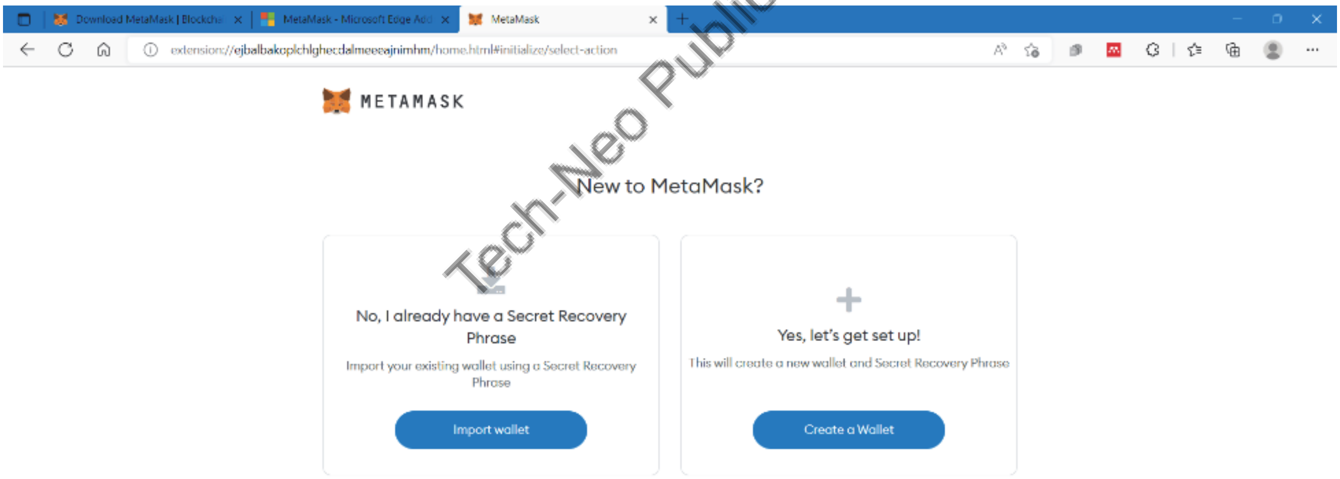

Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/welcome
Get Started

Welcome to MetaMask

Connecting you to Ethereum and the Decentralized Web.

We're happy to see you.

- **Step 6 :** If you do not have a wallet, then you need to create one. If you have a wallet, then it can be imported via a *Secret Recovery Phase*.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/select-action
METAMASK
Import wallet
Create a Wallet

New to MetaMask?

No, I already have a Secret Recovery Phrase

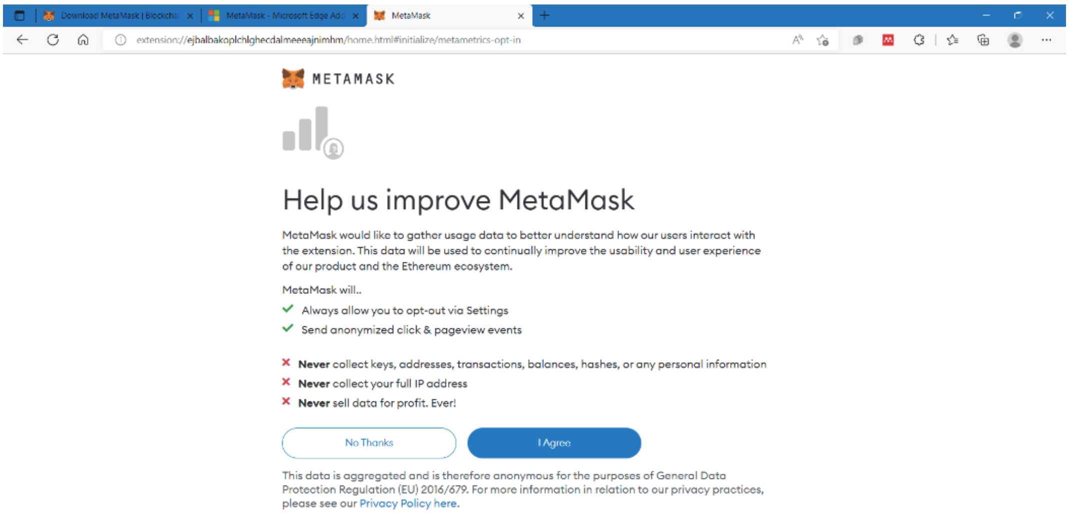
Import your existing wallet using a Secret Recovery Phrase

Yes, let's get set up!

This will create a new wallet and Secret Recovery Phrase

NOTES

- **Step 7 :** Once the wallet is created, then you need to agree to the T&C of MetaMask.



extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/metametrics-opt-in

METAMASK

Help us improve MetaMask

MetaMask would like to gather usage data to better understand how our users interact with the extension. This data will be used to continually improve the usability and user experience of our product and the Ethereum ecosystem.

MetaMask will..

✓ Always allow you to opt-out via Settings

✓ Send anonymized click & pageview events

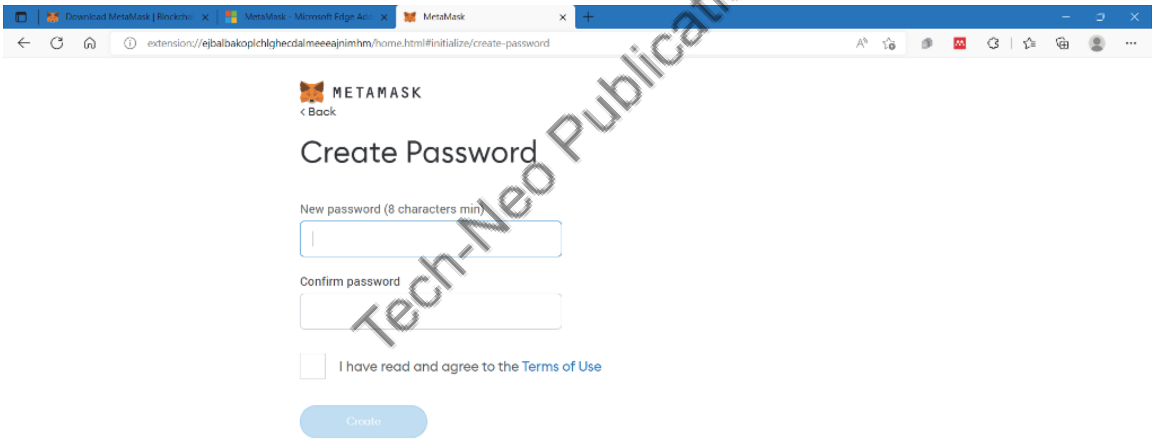
✗ **Never** collect keys, addresses, transactions, balances, hashes, or any personal information

✗ **Never** collect your full IP address

✗ **Never** sell data for profit. Ever!

This data is aggregated and is therefore anonymous for the purposes of General Data Protection Regulation (EU) 2016/679. For more information in relation to our privacy practices, please see our Privacy Policy here.

- **Step 8 :** A password needs to be created for your wallet.



extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/create-password

METAMASK

< Back

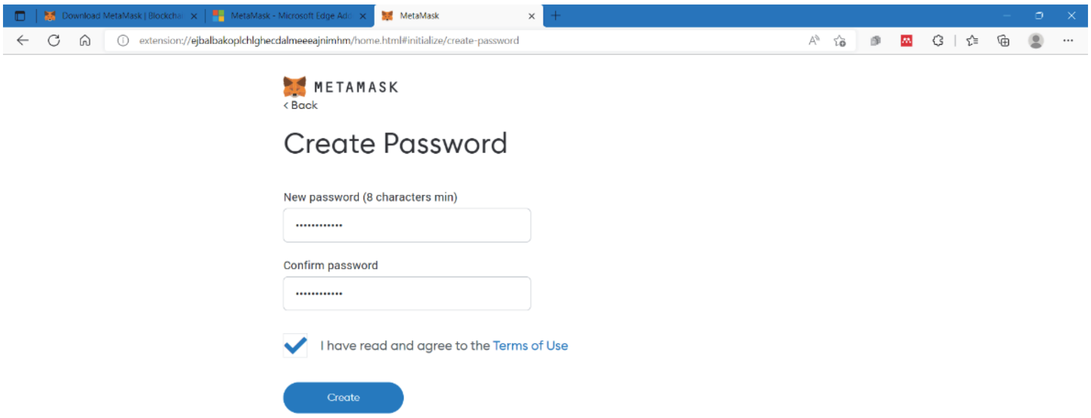
Create Password

New password (8 characters min)

Confirm password

□ I have read and agree to the Terms of Use

- **Step 9 :** Conformation of the password and agreement to the ‘Terms of Use’.



extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/create-password

METAMASK

< Back

Create Password

New password (8 characters min)

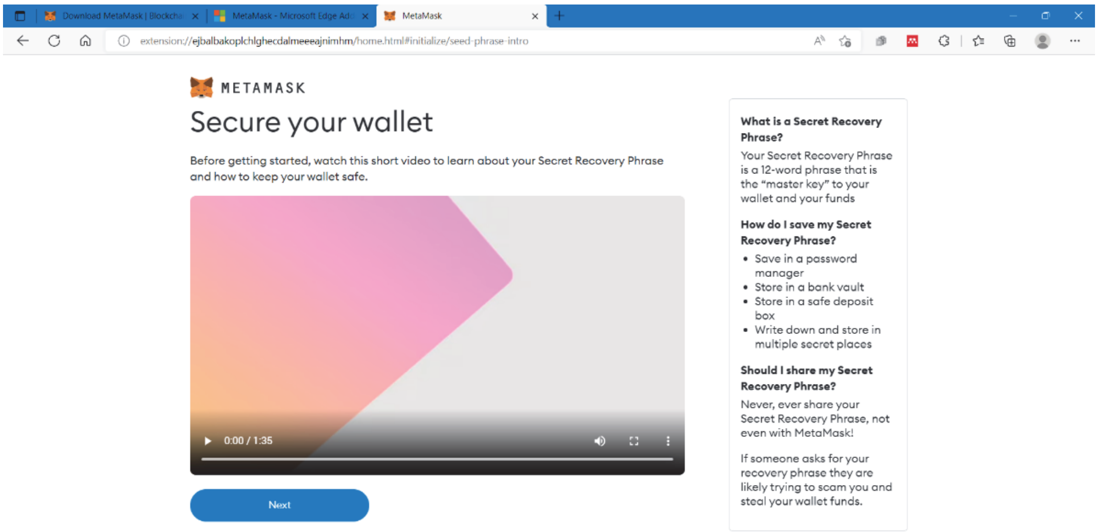
••••••••••••

Confirm password

••••••••••••

☑ I have read and agree to the Terms of Use

- **Step 10 :** Details about secret recovery phase can be seen in the video that helps in securing your wallet.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Ad
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/seed-phrase-intro
0:00 / 1:35
Next

METAMASK

Secure your wallet

Before getting started, watch this short video to learn about your Secret Recovery Phrase and how to keep your wallet safe.

What is a Secret Recovery Phrase?

Your Secret Recovery Phrase is a 12-word phrase that is the "master key" to your wallet and your funds

How do I save my Secret Recovery Phrase?

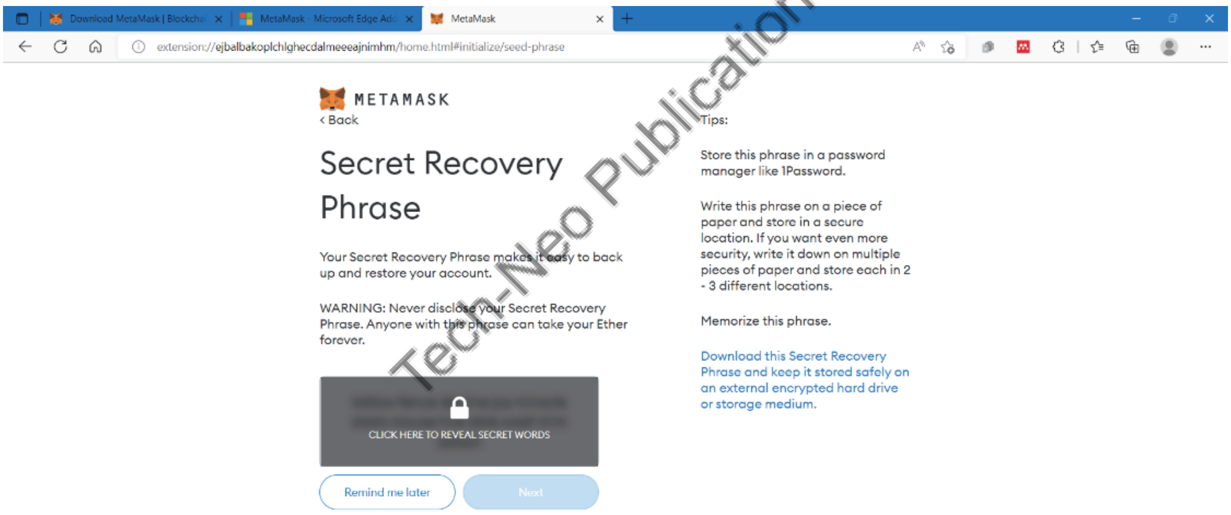
- Save in a password manager
- Store in a bank vault
- Store in a safe deposit box
- Write down and store in multiple secret places

Should I share my Secret Recovery Phrase?

Never, ever share your Secret Recovery Phrase, not even with MetaMask!

If someone asks for your recovery phrase they are likely trying to scam you and steal your wallet funds.

- **Step 11 :** Secret recovery phase helps is creating a backup for your account, which can be restored at a later stage.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Ad
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/seed-phrase
CLICK HERE TO REVEAL SECRET WORDS
Remind me later
Next

METAMASK

‹ Back

Secret Recovery Phrase

Your Secret Recovery Phrase makes it easy to back up and restore your account.

WARNING: Never disclose your Secret Recovery Phrase. Anyone with this phrase can take your Ether forever.

Tips:

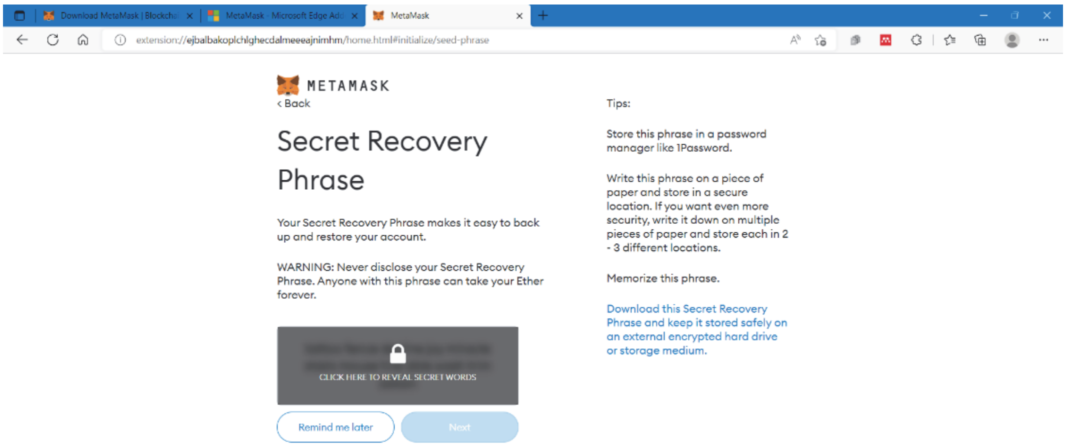
Store this phrase in a password manager like 1Password.

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 Recovery Phrase and keep it stored safely on an external encrypted hard drive or storage medium.

- **Step 12 :** It is generally recommended not to share/reveal the secret recovery phase.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Ad
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/seed-phrase
CLICK HERE TO REVEAL SECRET WORDS
Remind me later
Next

METAMASK

‹ Back

Secret Recovery Phrase

Your Secret Recovery Phrase makes it easy to back up and restore your account.

WARNING: Never disclose your Secret Recovery Phrase. Anyone with this phrase can take your Ether forever.

Tips:

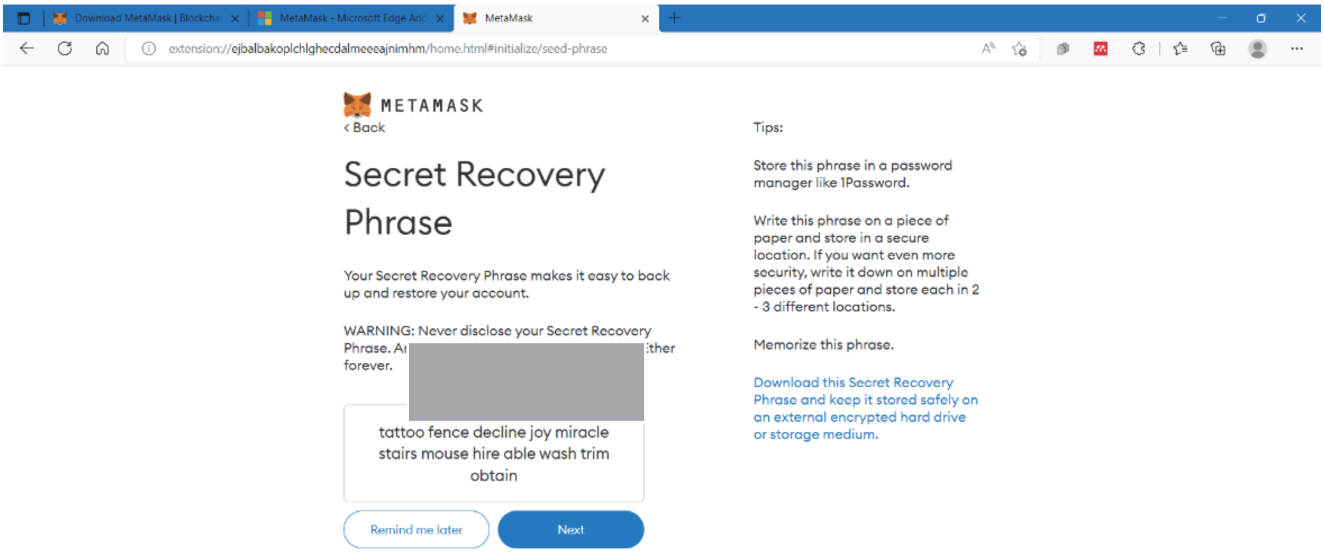
Store this phrase in a password manager like 1Password.

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 Recovery Phrase and keep it stored safely on an external encrypted hard drive or storage medium.

- **Step 13 :** You will see 12 words, which you need to remember in the same order as provided.



Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/seed-phrase
METAMASK
‹ Back

Secret Recovery Phrase

Your Secret Recovery Phrase makes it easy to back up and restore your account.

WARNING: Never disclose your Secret Recovery Phrase. A[redacted]ther forever.

tattoo fence decline joy miracle stairs mouse hire able wash trim obtain

Tips:

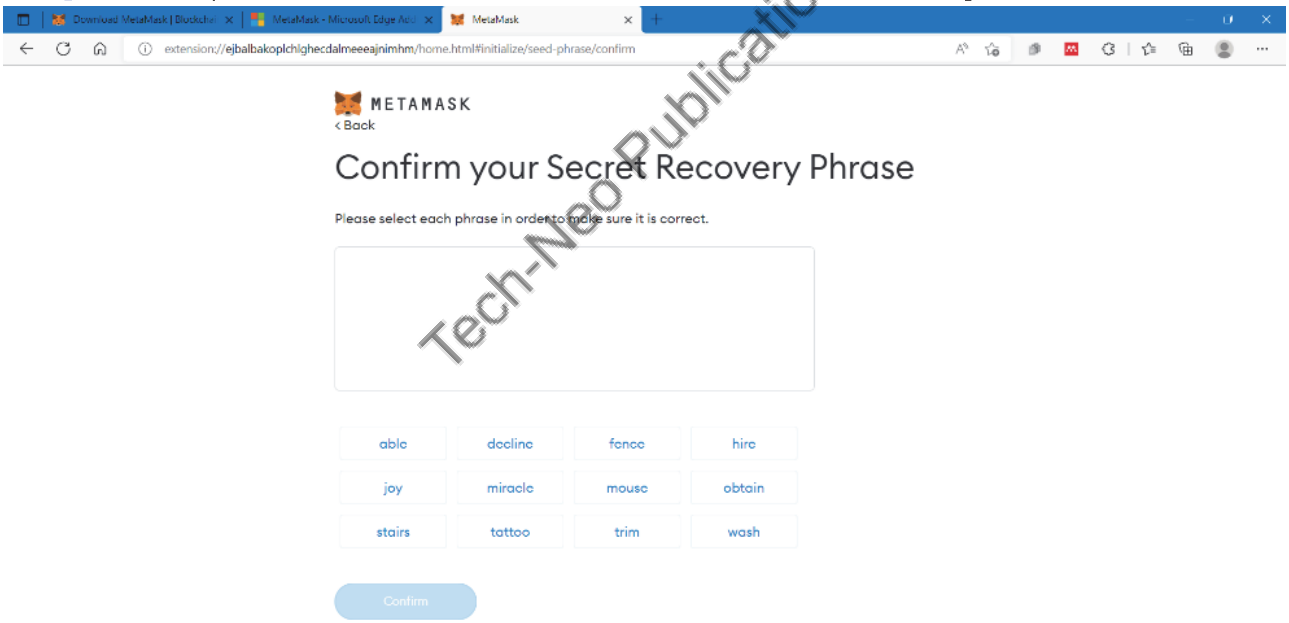
Store this phrase in a password manager like 1Password.

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 Recovery Phrase and keep it stored safely on an external encrypted hard drive or storage medium.

- **Step 14 :** Next, you need to select/click on these 12 words in the same order that were provided.



Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/seed-phrase/confirm
METAMASK
‹ Back

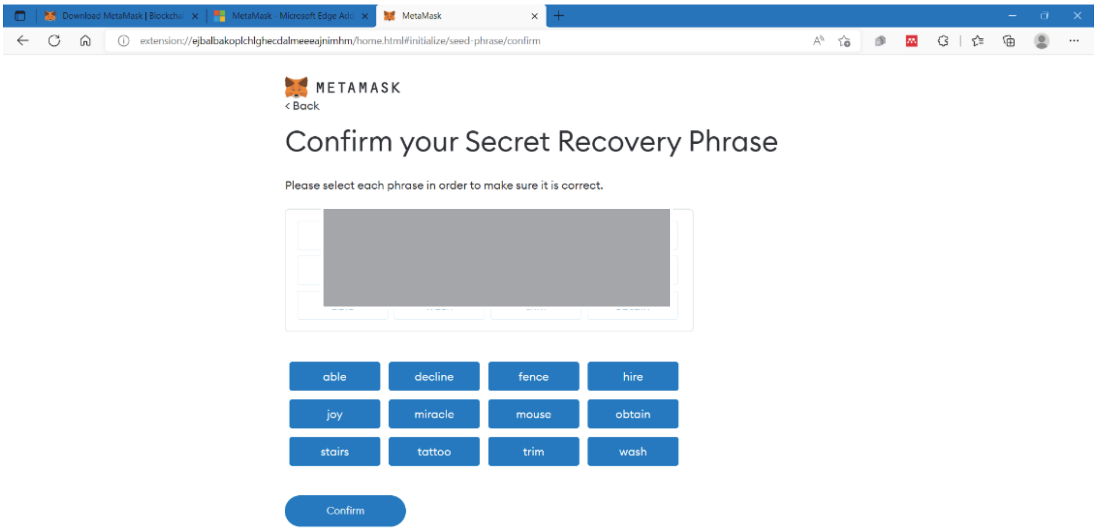
Confirm your Secret Recovery Phrase

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

able	decline	fence	hire
joy	miracle	mouse	obtain
stairs	tattoo	trim	wash

NOTES

- **Step 15 :** Once you have selected/clicked on the same order, then you need to click on the conform button.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/seed-phrase/confirm
METAMASK
‹ Back

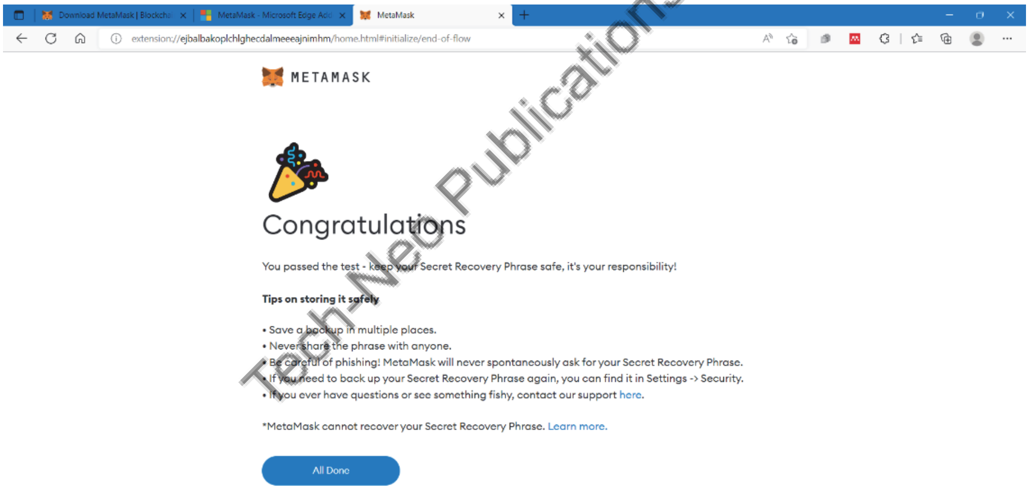
Confirm your Secret Recovery Phrase

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

[redacted]

able
decline
fence
hire
joy
miracle
mouse
obtain
stairs
tattoo
trim
wash
Confirm

- **Step 16 :** If found correct, you can proceed further.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#initialize/end-of-flow
METAMASK

Congratulations

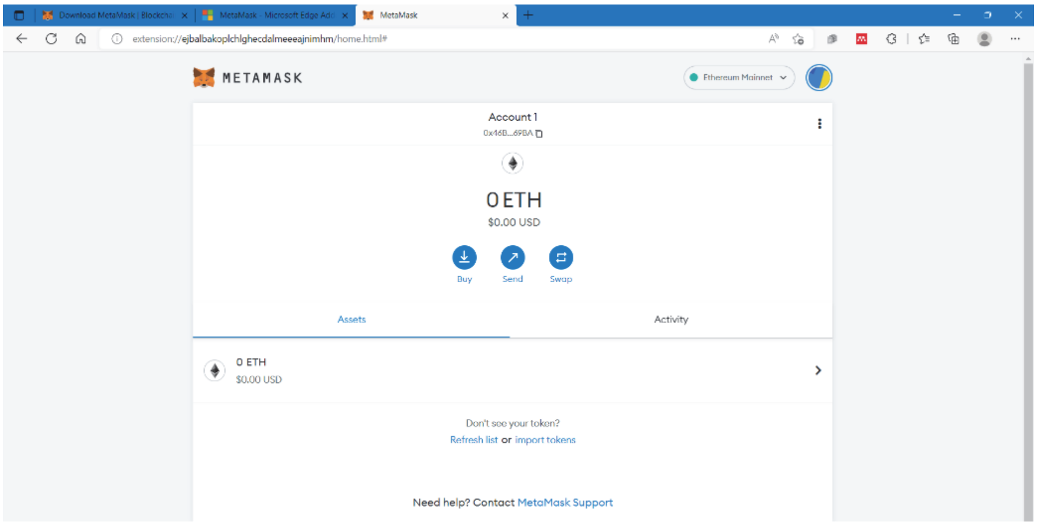
You passed the test - keep your Secret Recovery Phrase safe, it's your responsibility!

Tips on storing it safely

- Save a backup in multiple places.
- Never share the phrase with anyone.
- Be careful of phishing! MetaMask will never spontaneously ask for your Secret Recovery Phrase.
- If you need to back up your Secret Recovery Phrase again, you can find it in Settings -> Security.
- If you ever have questions or see something fishy, contact our support here.

*MetaMask cannot recover your Secret Recovery Phrase. Learn more.

- **Step 17 :** On successfully completing the above steps, you are then directed to your digital crypto wallet.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#
METAMASK
Ethereum Mainnet

Account 1
0x46B...69BA

0 ETH
$0.00 USD

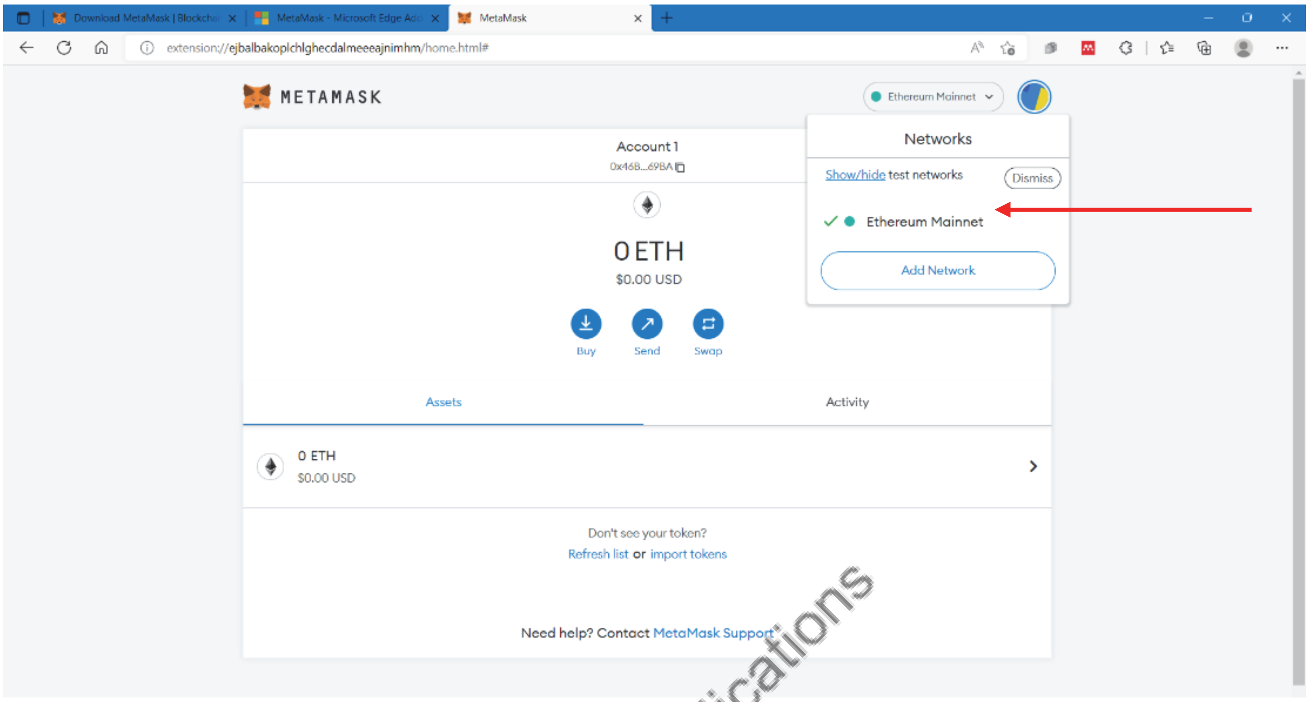
Buy
Send
Swap
Assets
Activity

0 ETH
$0.00 USD

Don't see your token?
Refresh list or import tokens

Need help? Contact MetaMask Support

- **Step 18 :** As you can see, we are on the Ethereum Mainnet. We need to check for test networks.



Download MetaMask | Blockcha
MetaMask - Microsoft Edge Ad
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#
METAMASK
Ethereum Mainnet
Account 1
0x468...698A
Buy
Send
Swap
Assets
Activity

0 ETH
$0.00 USD

Networks

Show/hide test networks (Dismiss)

✓ ● Ethereum Mainnet

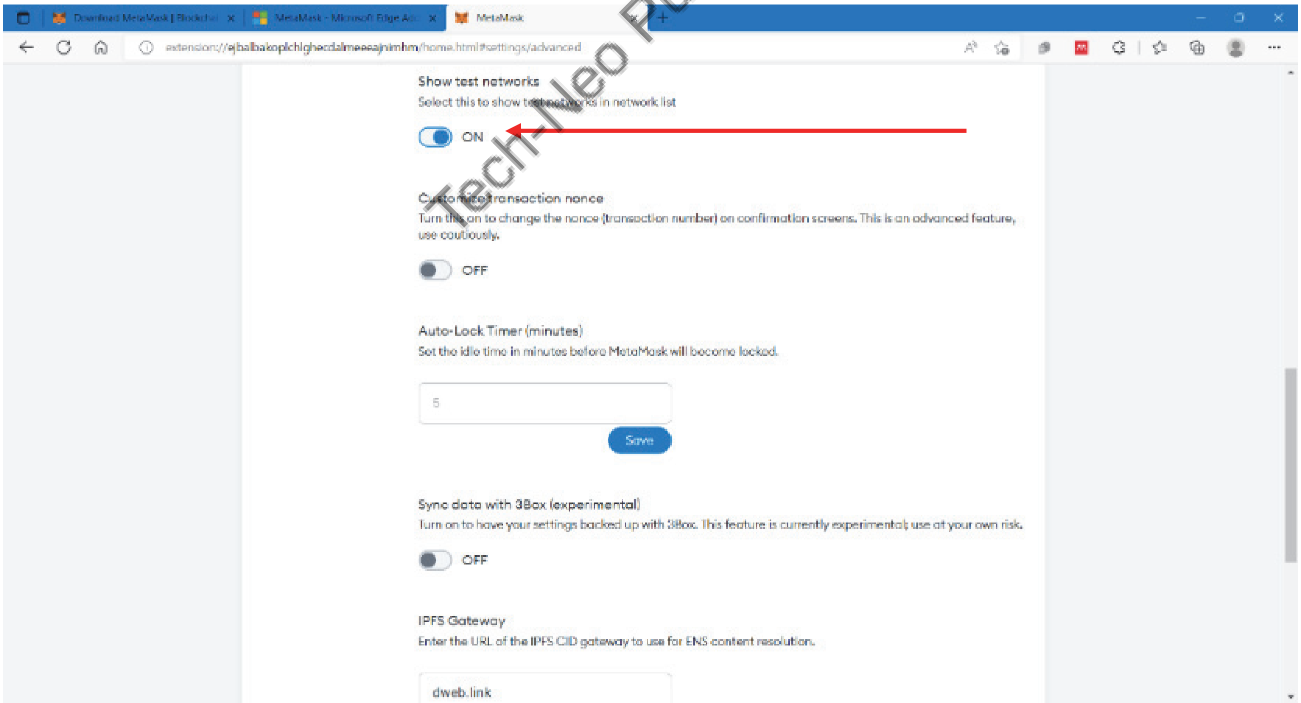
Add Network

0 ETH
$0.00 USD

Don't see your token?
Refresh list or import tokens

Need help? Contact MetaMask Support

- **Step 19 :** Under settings, you need to "Turn ON" show test networks.



Download MetaMask | Blockcha
MetaMask - Microsoft Edge Ad
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#settings/advanced

Show test networks
Select this to show test networks in network list
ON

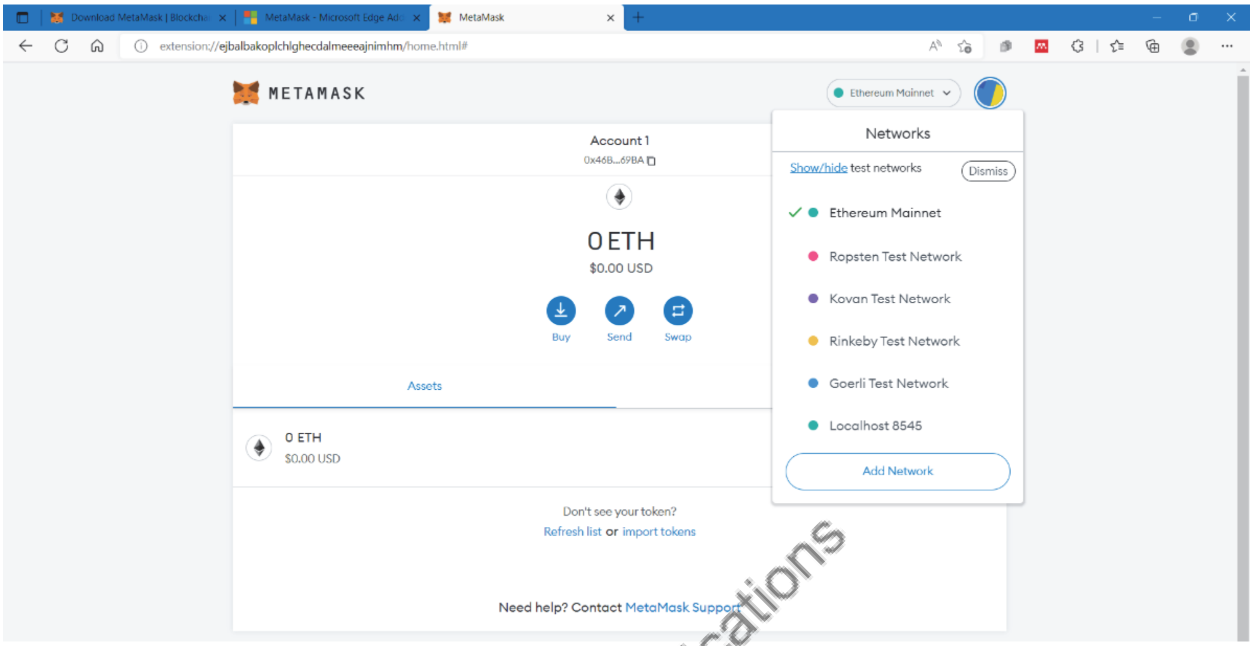
Customize transaction nonce
Turn this on to change the nonce (transaction number) on confirmation screens. This is an advanced feature, use cautiously.
OFF

Auto-Lock Timer (minutes)
Set the idle time in minutes before MetaMask will become locked.
5
Save

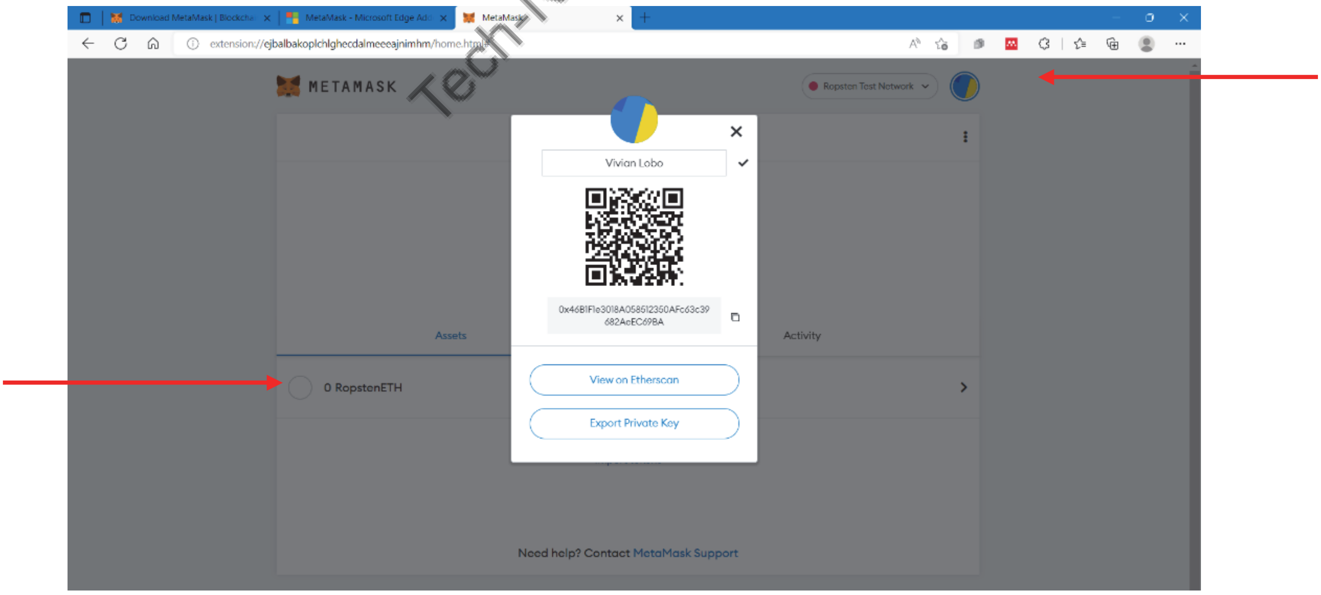
Sync data with 3Box (experimental)
Turn on to have your settings backed up with 3Box. This feature is currently experimental; use at your own risk.
OFF

IPFS Gateway
Enter the URL of the IPFS CID gateway to use for ENS content resolution.
dweb.link

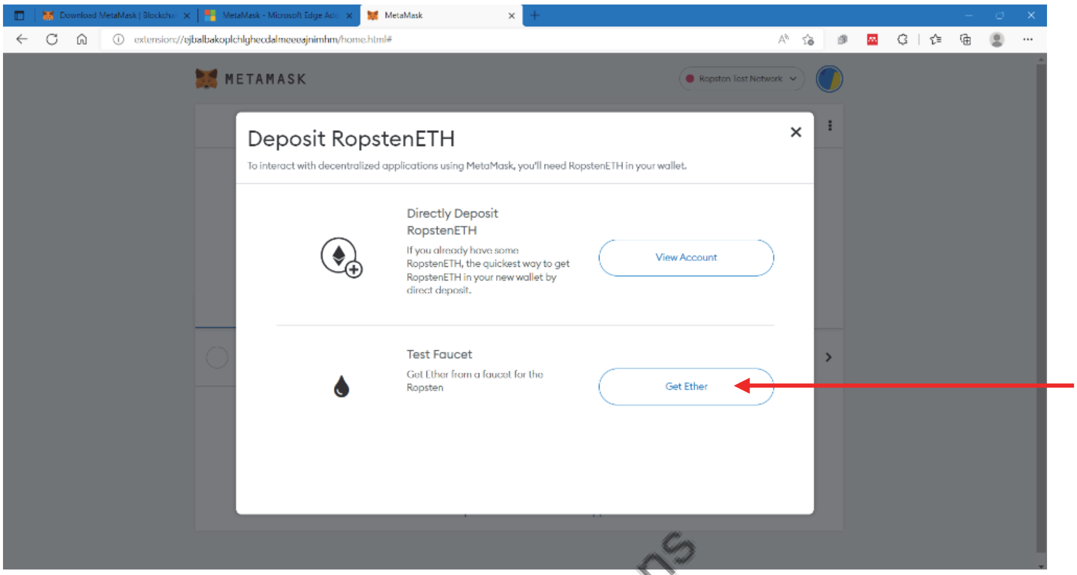
- **Step 20 :** Once the test networks have been 'Turned ON', then you will be able to see all test networks such as Ropsten, Kovan, Rinkeby, and Goerli.



- **Step 21 :** On clicking on the icon, you can edit on the account name as well as you can see the account number (i.e., the public address). Note that I have selected Ropsten as the test network. You can select any test network of your choice. At present, there are 0 RopstenETH. We need to hunt for some ethers.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#
METAMASK
Ropsten Test Network
Vivian Lobo
0x46B1F1e3018A058612350AFc63c39682AcEC69BA
View on Etherscan
Export Private Key
Assets
Activity
0 RopstenETH
Need help? Contact MetaMask Support

- **Step 22 :** On clicking on 0 RopstenETH, a window opens, that offers **Deposit RopstenETH** or **Test Faucet**(i.e., get Ether [test ethers]).


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#
METAMASK
Ropsten Test Network

Deposit RopstenETH

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

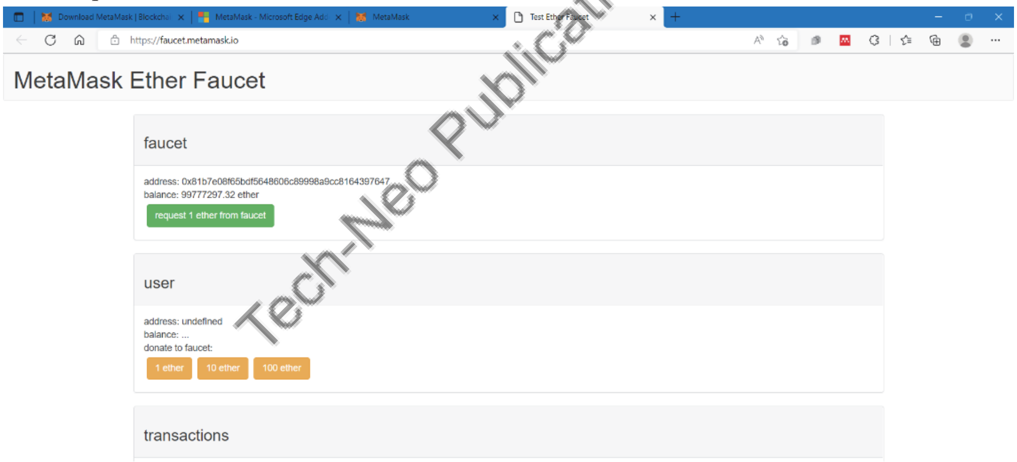
Directly Deposit RopstenETH

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

Test Faucet

Get Ether from a faucet for the Ropsten

- **Step 23 :** Next, we request for minimum 1 ETH from Test Faucet.


Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
Test Ether Faucet
https://faucet.metamask.io

MetaMask Ether Faucet

faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99777297.32 ether

request 1 ether from faucet

user

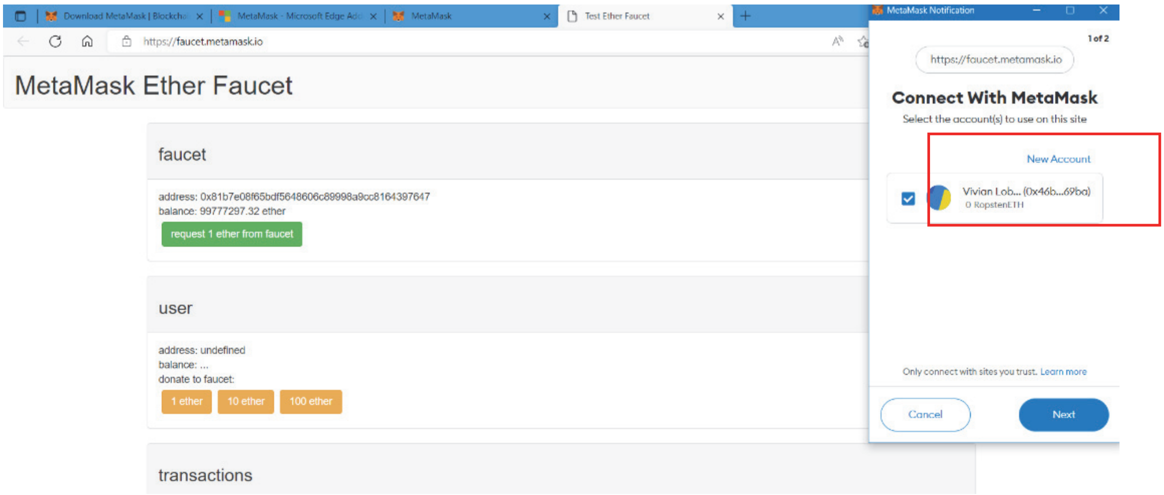
address: undefined
balance: ...
donate to faucet:

1 ether
10 ether
100 ether

transactions

NOTES

- **Step 24 :** On requesting for 1 ether from faucet, it tries to connect with MetaMask wallet.



https://faucet.metamask.io

MetaMask Ether Faucet

faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99777297.32 ether

user

address: undefined
balance: ...
donate to faucet:

transactions

MetaMask Notification

1 of 2

https://faucet.metamask.io

Connect With MetaMask

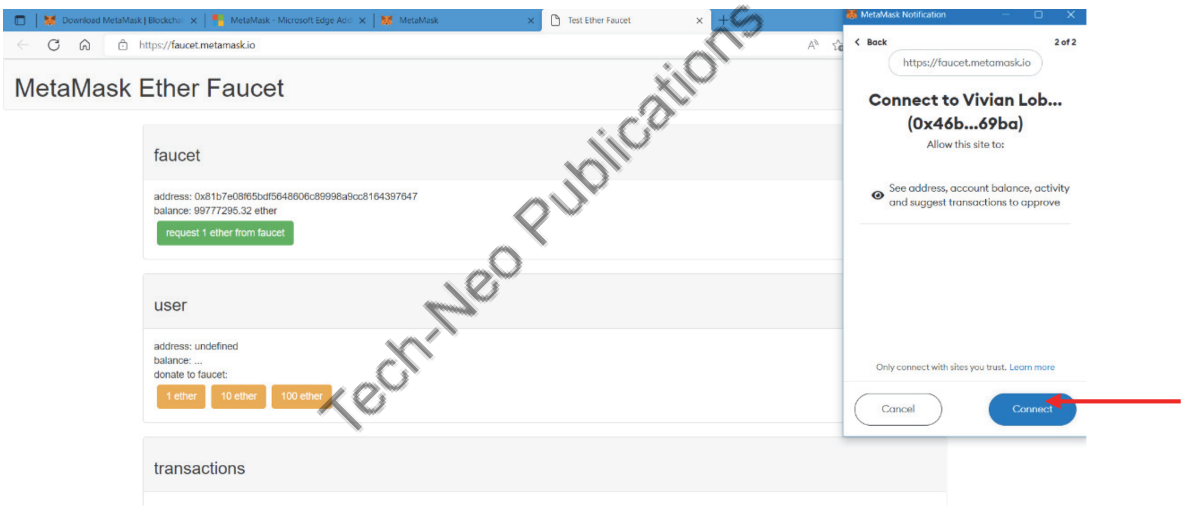
Select the account(s) to use on this site

☑ Vivian Lob... (0x46b...69ba)
0 RopstenETH

Only connect with sites you trust. Learn more

Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
Test Ether Faucet
request 1 ether from faucet
1 ether
10 ether
100 ether
New Account
Cancel
Next

- **Step 25 :** For connecting to MetaMask, you need to click on Connect



https://faucet.metamask.io

MetaMask Ether Faucet

faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99777295.32 ether

user

address: undefined
balance: ...
donate to faucet:

transactions

MetaMask Notification

2 of 2

https://faucet.metamask.io

Connect to Vivian Lob... (0x46b...69ba)

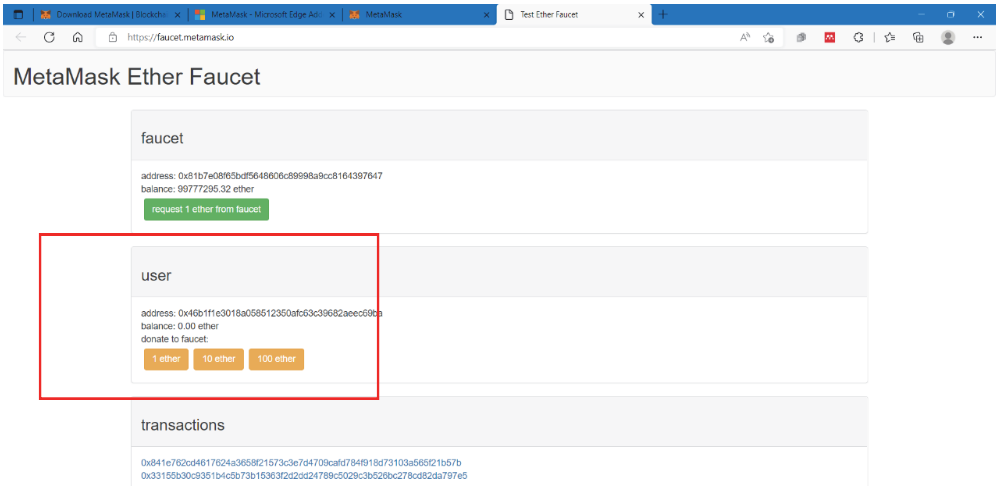
Allow this site to:

See address, account balance, activity and suggest transactions to approve

Only connect with sites you trust. Learn more

Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
Test Ether Faucet
request 1 ether from faucet
1 ether
10 ether
100 ether
Back
Cancel
Connect

- **Step 26 :** As you can see, there is 0.00 ether for given address, which is the account that is created.



https://faucet.metamask.io

MetaMask Ether Faucet

faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99777295.32 ether

user

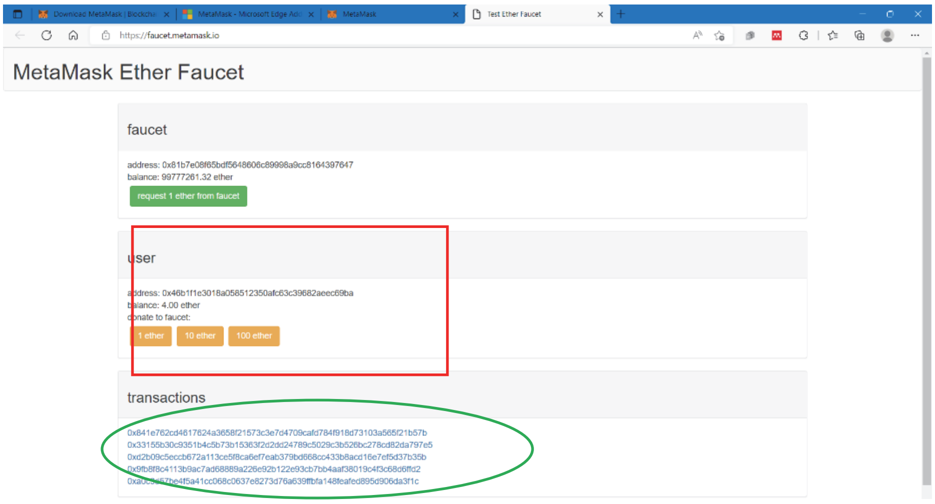
address: 0x46b1f1e3018a058512350afc63c39682aeec69ba
balance: 0.00 ether
donate to faucet:

transactions

0x841e762cd4617624a3658f21573c3e7d4709cafd784f918d73103a565f21b57b
0x33155b30c9351b4c5b73b15363f2d2dd24789c5029c3b526bc278cd82da797e5

Download MetaMask | Blockcha
MetaMask - Microsoft Edge Add
MetaMask
Test Ether Faucet
request 1 ether from faucet
1 ether
10 ether
100 ether

- **Step 27 :** However, through several requests from faucet, 4.00 ethers were obtained and the transaction address can also be seen.



MetaMask Ether Faucet

faucet

address: 0x81b7e08f65bdf5648606c89998a9cc8164397647
balance: 99777261.32 ether

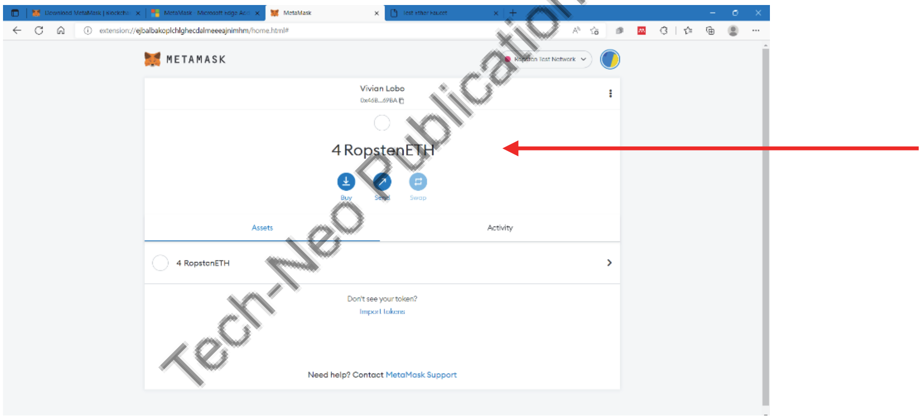
user

address: 0x46b1f1e3018a058512350afc63c39682aeec69ba
balance: 4.00 ether
donate to faucet:

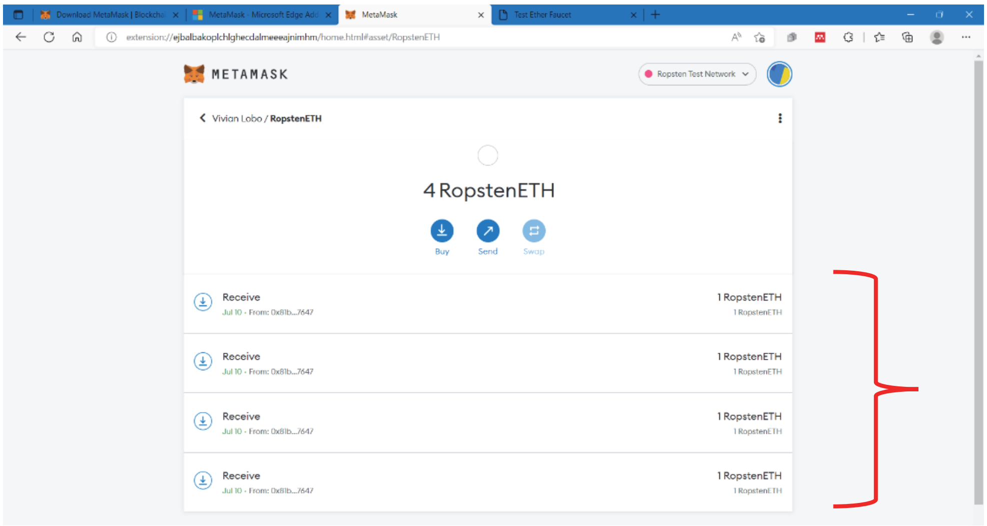
transactions

0x841e762cd4617624a3658f21573c3e7d4709cafd784f918d73103a565f21b57b
0x33155b30c9351b4c5b73b15363f2d2dd24789c5029c3b526bc278cd82da797e5
0xd2b09c5eccb672a113ce5f8ca6ef7eab379bd668cc433b8acd16e7ef5d37b35b
0x9fb8f8c4113b9ac7ad68889a226e92b122e93cb7bb4aaf38019c4f3c68d6ffd2
0xa0c3d57be4f5a41cc068c0637e8273d76a639ffbfa148feafed895d906da3f1c

- **Step 28 :** As you can see, 4 RopstenETH were received.



- **Step 29 :** Details of each RopstenETH received can be seen below.



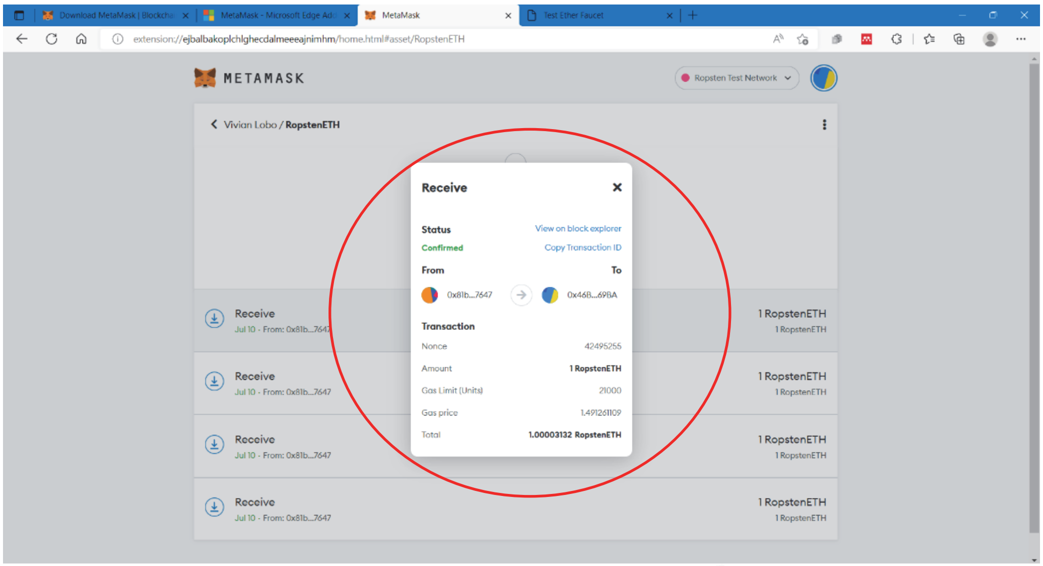
Receive 1 RopstenETH
Jul 10 · From: 0x81b...7647 1 RopstenETH

Receive 1 RopstenETH
Jul 10 · From: 0x81b...7647 1 RopstenETH

Receive 1 RopstenETH
Jul 10 · From: 0x81b...7647 1 RopstenETH

Receive 1 RopstenETH
Jul 10 · From: 0x81b...7647 1 RopstenETH

Step 30 : On clicking on each RopstenETH, the following details can be found.



Receive

Status: Confirmed
View on block explorer
Copy Transaction ID

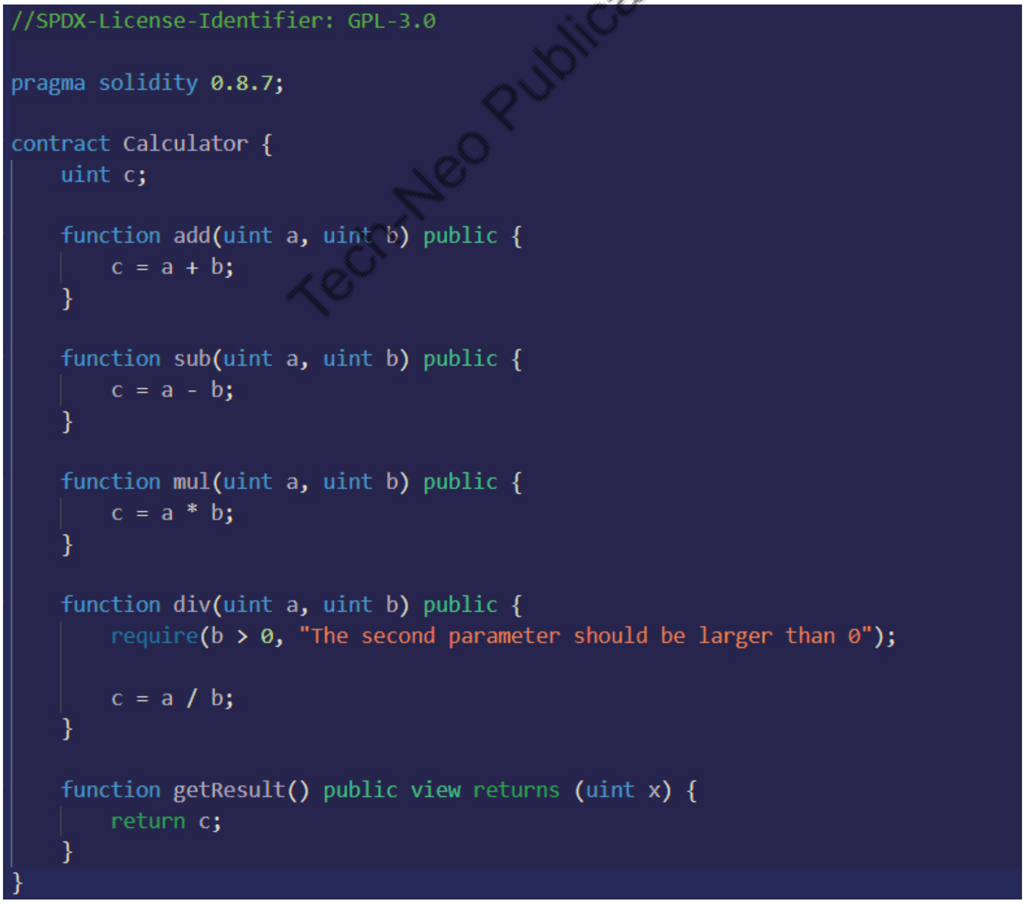
From: 0x81b...7647 → To: 0x46B...69BA

Transaction

Nonce	42495255
Amount	1 RopstenETH
Gas Limit (Units)	21000
Gas price	1.491261109
Total	1.00003132 RopstenETH

Step 31 : Writing and compiling the code in Remix IDE

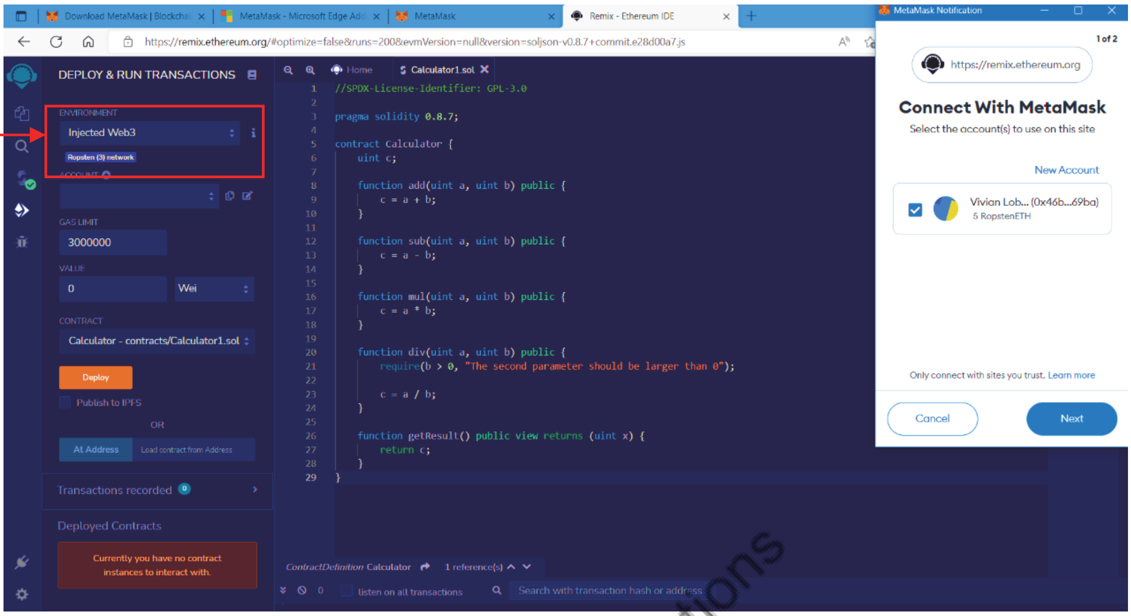
- Calculator Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

- The above code is for a simple calculator that performs basic arithmetic operations.
- The code is written in Remix IDE.

- Once the code is written, the environment needs to be changed from JavaScript VM (London)to Injected Web3. This in turn will lead to connect with MetaMask wallet.



https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Injected Web3 (Ropsten (3) network)

ACCOUNT

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Calculator - contracts/Calculator1.sol

Transactions recorded 0

Deployed Contracts

Currently you have no contract instances to interact with.

Calculator1.sol

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

ContractDefinition Calculator 1 reference(s)

MetaMask Notification

1 of 2

https://remix.ethereum.org

Connect With MetaMask

Select the account(s) to use on this site

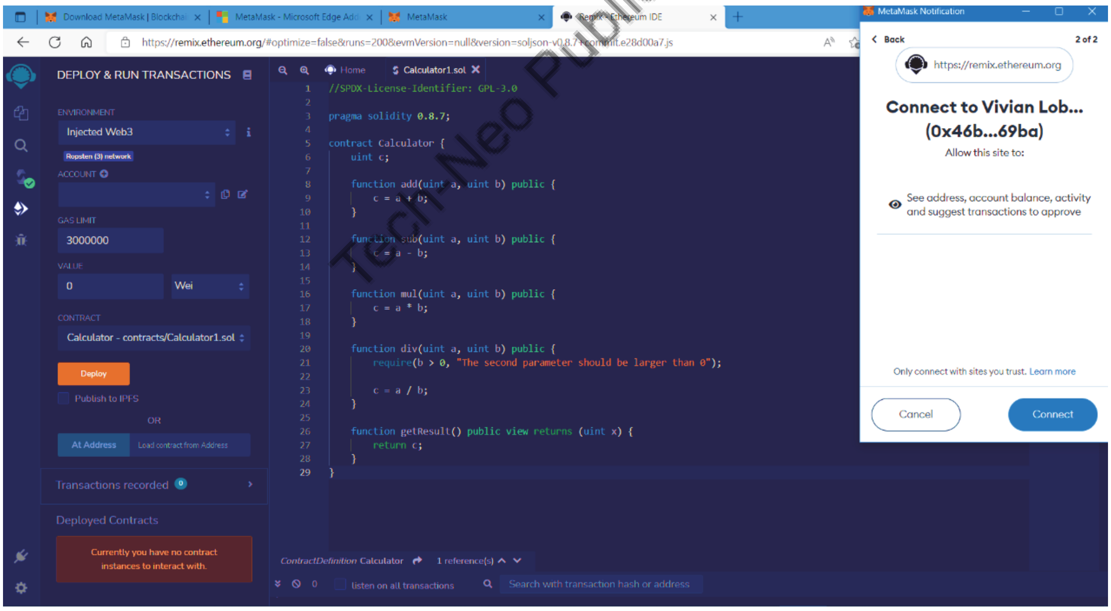
New Account

☑ Vivian Lob... (0x46b...69ba) 5 RopstenETH

Only connect with sites you trust. Learn more

Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
Remix - Ethereum IDE
Home
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
listen on all transactions
Search with transaction hash or address
Cancel
Next

- On successful compilation of the code, we connect Remix IDE to MetaMask wallet.



https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Injected Web3 (Ropsten (3) network)

ACCOUNT

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Calculator - contracts/Calculator1.sol

Transactions recorded 0

Deployed Contracts

Currently you have no contract instances to interact with.

Calculator1.sol

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

ContractDefinition Calculator 1 reference(s)

MetaMask Notification

Back 2 of 2

https://remix.ethereum.org

Connect to Vivian Lob... (0x46b...69ba)

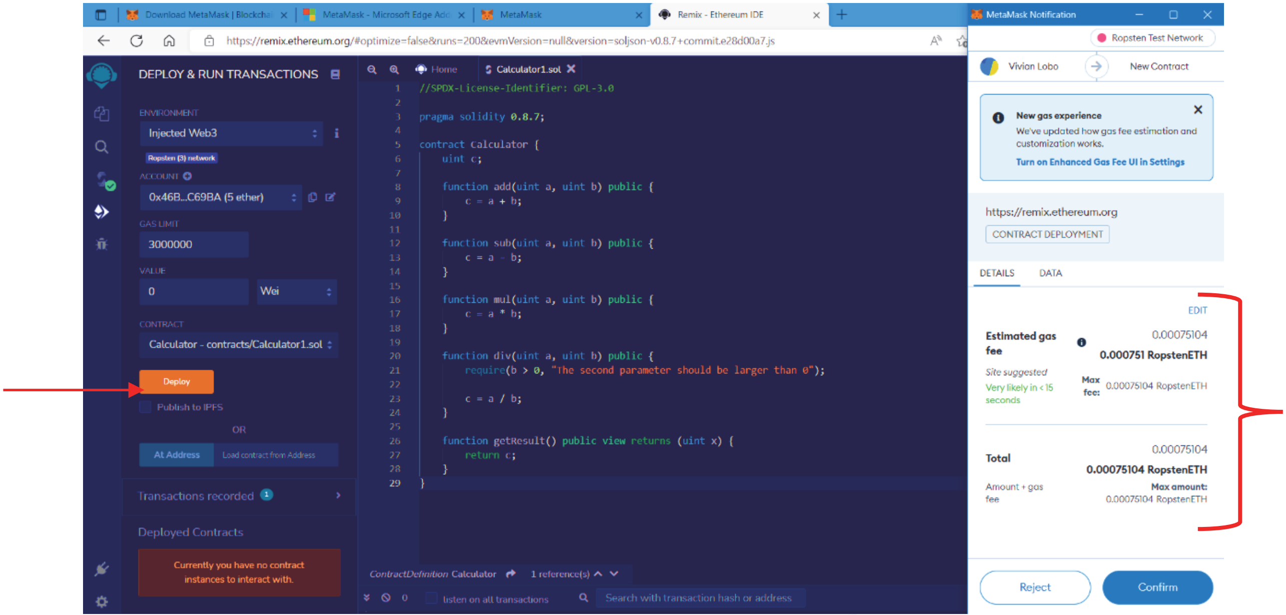
Allow this site to:

See address, account balance, activity and suggest transactions to approve

Only connect with sites you trust. Learn more

Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
Remix - Ethereum IDE
Home
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
listen on all transactions
Search with transaction hash or address
Cancel
Connect

- Once connected, there exists an Estimated Gas Fee that shows how much RopstenETH would be used out of 4 RopstenETH. Click on confirm.


Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
Remix - Ethereum IDE
https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Home
Calculator1.sol
MetaMask Notification
Reject
Confirm

ENVIRONMENT: Injected Web3

Ropsten (3) network

ACCOUNT: 0x46B...C69BA (5 ether)

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Calculator - contracts/Calculator1.sol

Transactions recorded 1

Deployed Contracts

Currently you have no contract instances to interact with.

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

ContractDefinition Calculator 1 reference(s)

listen on all transactions   Search with transaction hash or address

Ropsten Test Network

Vivian Lobo → New Contract

New gas experience
We've updated how gas fee estimation and customization works.
Turn on Enhanced Gas Fee UI in Settings

https://remix.ethereum.org
CONTRACT DEPLOYMENT

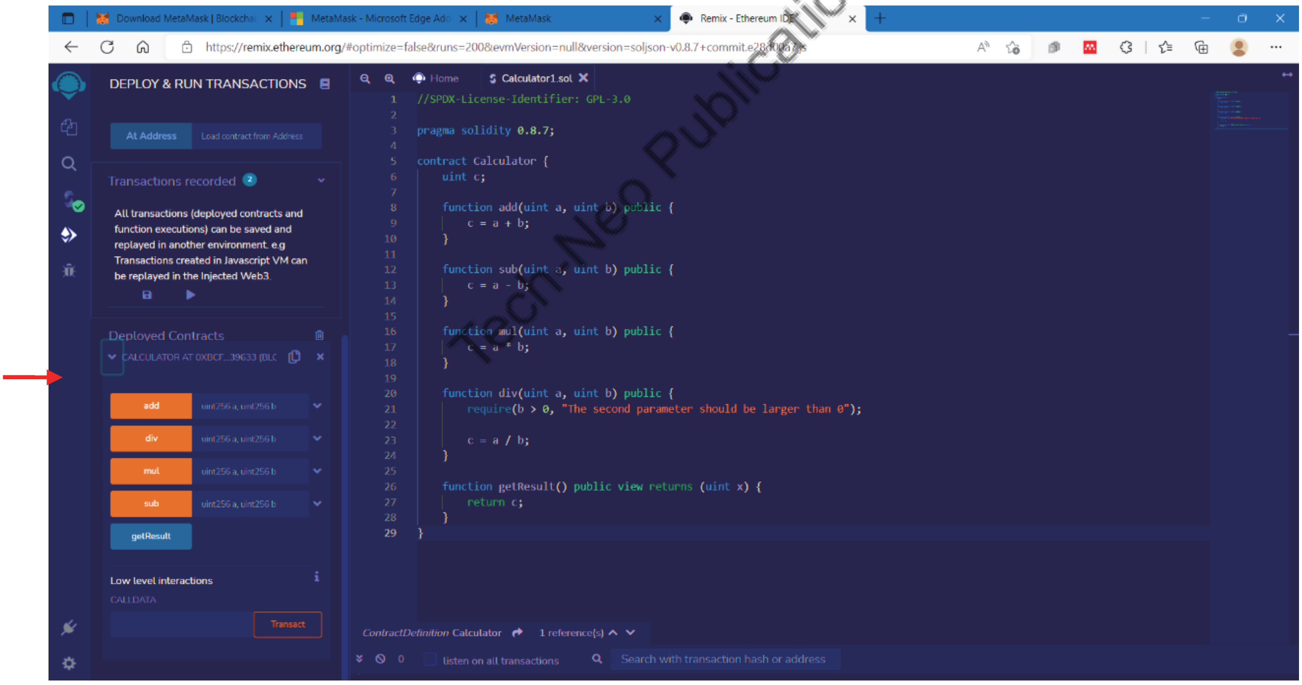
DETAILS DATA

EDIT

Estimated gas fee: 0.00075104 **0.000751 RopstenETH**
Site suggested
Very likely in < 15 seconds
Max fee: 0.00075104 RopstenETH

Total: 0.00075104 **0.00075104 RopstenETH**
Amount + gas fee
Max amount: 0.00075104 RopstenETH

- Next, click on "deploy" contact, which will result in the following output.


Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
Remix - Ethereum IDE
https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
At Address
Load contract from Address
add
div
mul
sub
getResult
Transact
Home
Calculator1.sol

Transactions recorded 2

All transactions (deployed contracts and function executions) can be saved and replayed in another environment. e.g Transactions created in Javascript VM can be replayed in the Injected Web3.

Deployed Contracts

CALCULATOR AT 0XBCF...39633 (BLC

add: uint256 a, uint256 b

div: uint256 a, uint256 b

mul: uint256 a, uint256 b

sub: uint256 a, uint256 b

Low level interactions

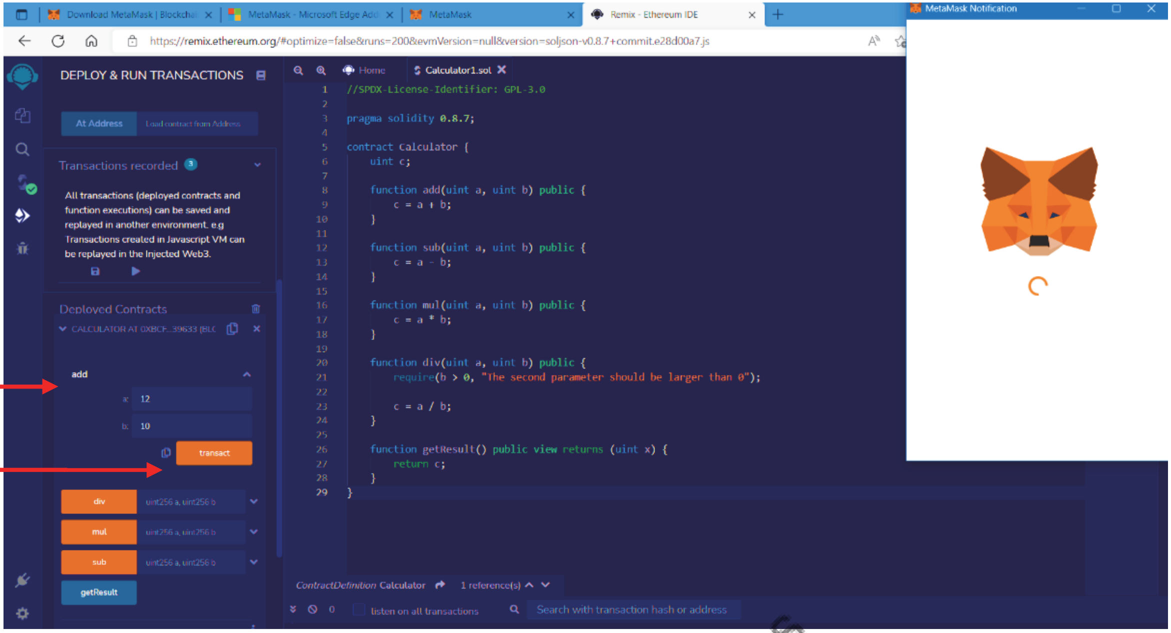
CALLDATA

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

ContractDefinition Calculator 1 reference(s)

listen on all transactions   Search with transaction hash or address

- When we consider two numbers (i.e., 12 and 10) and attempt to perform addition operation, we need to click on transact. On clicking on transact, the MetaMask wallet gets connected.


Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
Remix - Ethereum IDE
MetaMask Notification
https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
At Address
Load contract from Address
Transactions recorded 3
Home
Calculator1.sol
Deployed Contracts
CALCULATOR AT 0XBCF...39633 (BLC
add
transact
div
mul
sub
getResult
uint256 a, uint256 b
ContractDefinition Calculator
1 reference(s)
listen on all transactions
Search with transaction hash or address

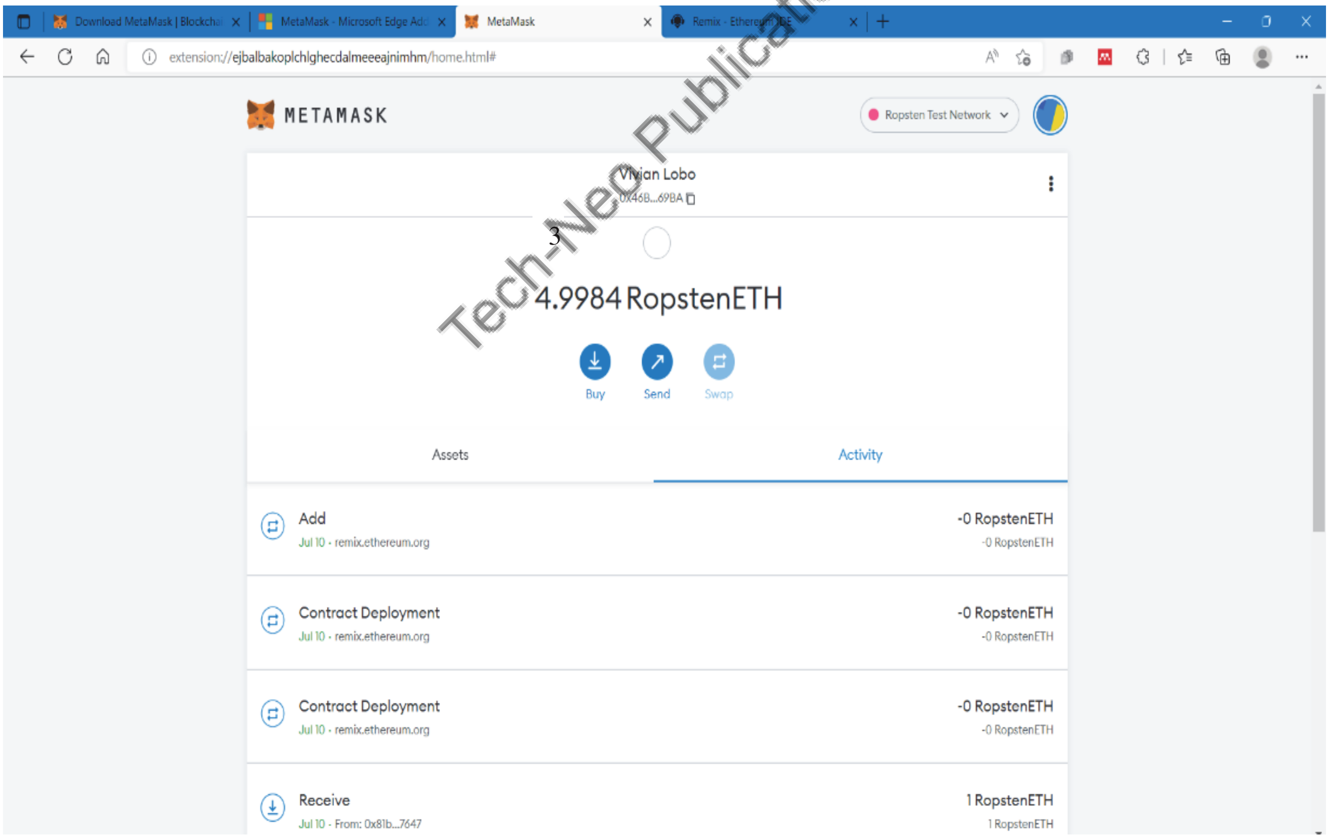
All transactions (deployed contracts and function executions) can be saved and replayed in another environment. e.g Transactions created in Javascript VM can be replayed in the Injected Web3.

a: 12

b: 10

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

- For every transaction, a certain amount of fee gets deducted from MetaMask wallet.

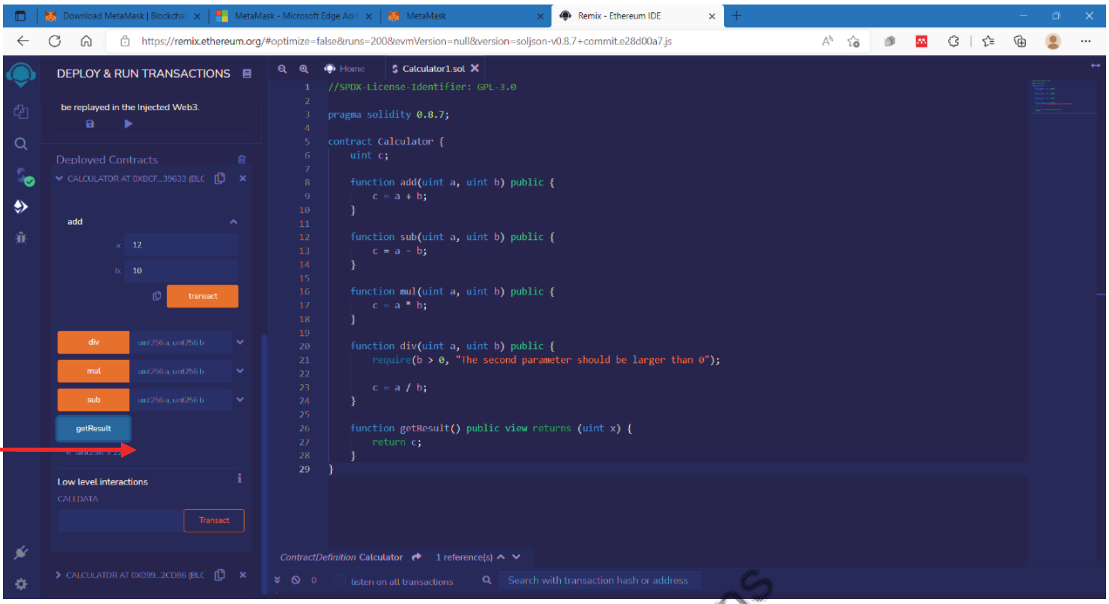

Download MetaMask | Blockchai
MetaMask - Microsoft Edge Add
MetaMask
Remix - Ethereum IDE
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#
METAMASK
Ropsten Test Network
Buy
Send
Swap
Assets
Activity

Vivian Lobo
0X46B...69BA

4.9984 RopstenETH

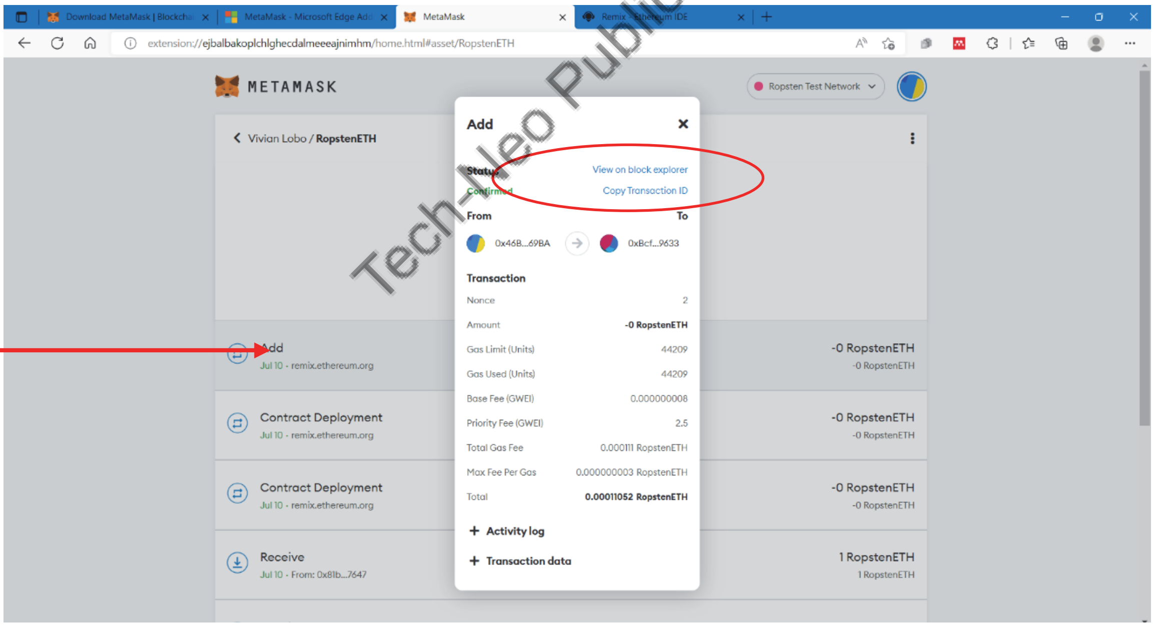
Add Jul 10 · remix.ethereum.org	-0 RopstenETH -0 RopstenETH
Contract Deployment Jul 10 · remix.ethereum.org	-0 RopstenETH -0 RopstenETH
Contract Deployment Jul 10 · remix.ethereum.org	-0 RopstenETH -0 RopstenETH
Receive Jul 10 · From: 0x81b...7647	1 RopstenETH 1 RopstenETH

- Next, the result of addition of two numbers can be seen in the *getResult()*.


Remix - Ethereum IDE
https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
be replayed in the Injected Web3.
Deployed Contracts
CALCULATOR AT 0XDCF...39633 (BLC
add
a 12
b 10
transact
div
mul
sub
getResult
Low level interactions
CALLDATA
Transact
CALCULATOR AT 0X099...2CDB6 (BLC
Home
Calculator1.sol
ContractDefinition Calculator
1 reference(s)
listen on all transactions
Search with transaction hash or address

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

- Whether the transaction is successful or not can be seen by clicking on Add, as shown below. Also, the status of the transaction can be seen on Block Explorer as well as Transaction ID can be seen.


MetaMask
extension://ejbalbakoplchlghecdalmeeeajnimhm/home.html#asset/RopstenETH
METAMASK
Ropsten Test Network
Vivian Lobo / RopstenETH

Add

Status | View on block explorer
Confirmed | Copy Transaction ID

From | To
0x46B...69BA | 0xBcf...9633

Transaction

Nonce	2
Amount	-0 RopstenETH
Gas Limit (Units)	44209
Gas Used (Units)	44209
Base Fee (GWEI)	0.000000008
Priority Fee (GWEI)	2.5
Total Gas Fee	0.000111 RopstenETH
Max Fee Per Gas	0.000000003 RopstenETH
Total	0.00011052 RopstenETH

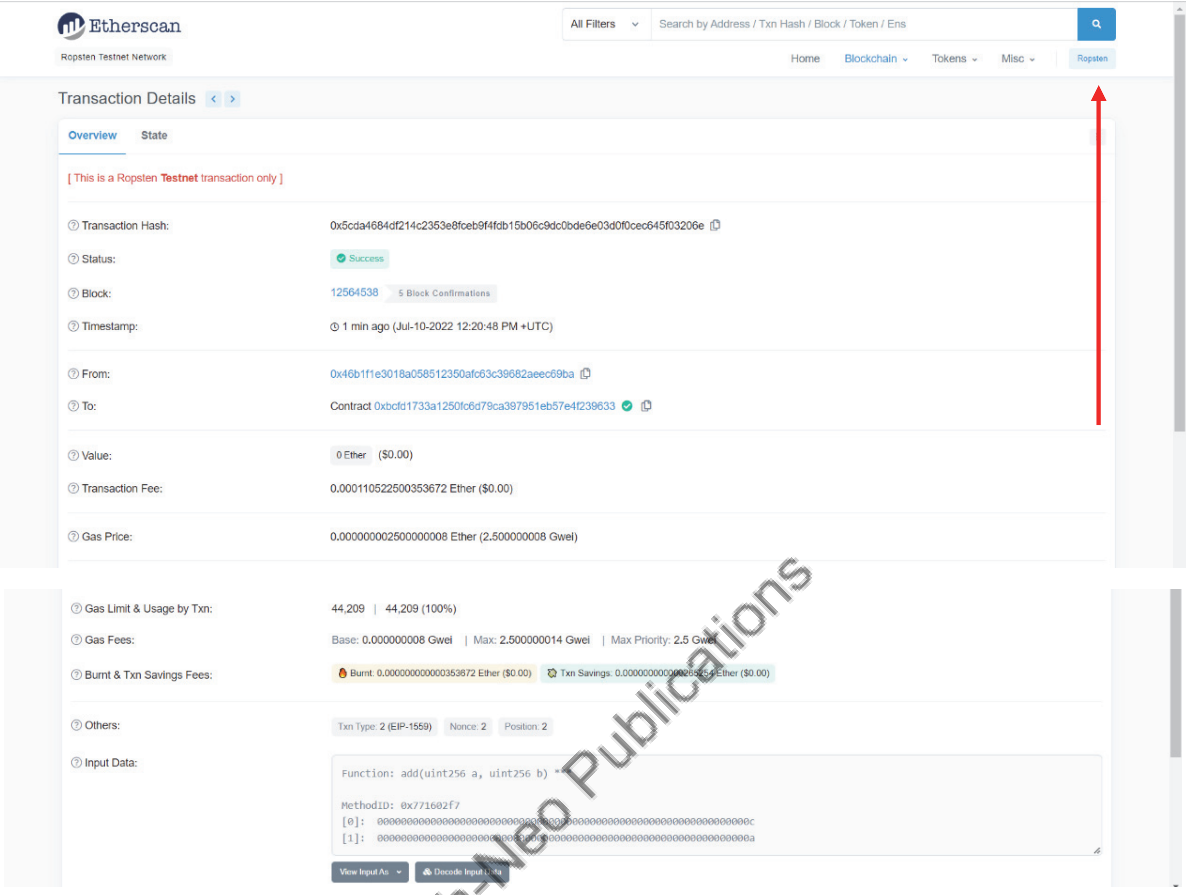
+ Activity log

+ Transaction data

Add Jul 10 · remix.ethereum.org	-0 RopstenETH -0 RopstenETH
Contract Deployment Jul 10 · remix.ethereum.org	-0 RopstenETH -0 RopstenETH
Contract Deployment Jul 10 · remix.ethereum.org	-0 RopstenETH -0 RopstenETH
Receive Jul 10 · From: 0x81b...7647	1 RopstenETH 1 RopstenETH

NOTES

- Also, on Etherscan for Ropsten Test Network, the transaction details can be seen.



Transaction Details

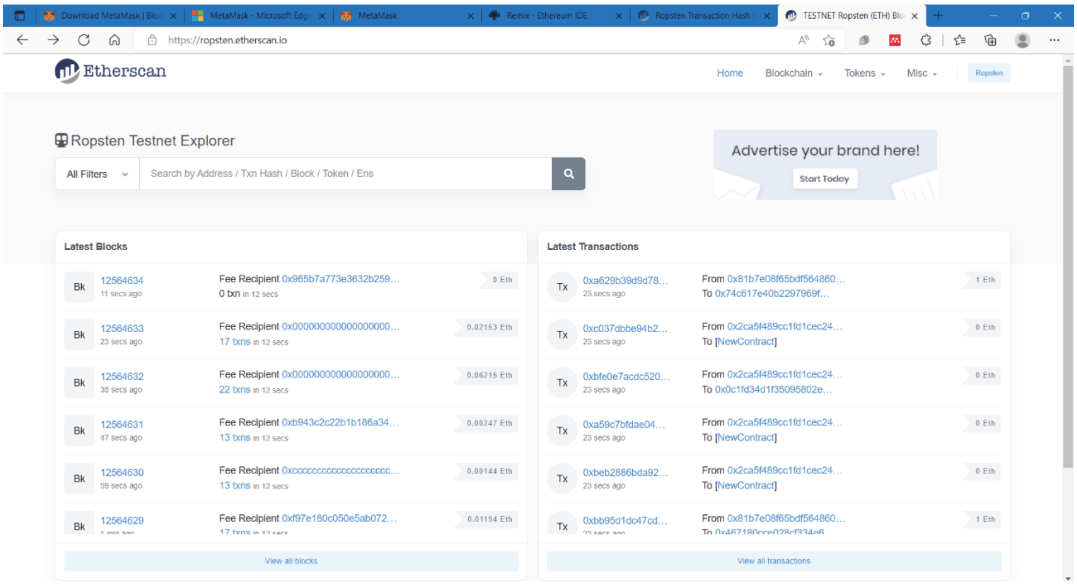
Overview | State

[This is a Ropsten **Testnet** transaction only]

Transaction Hash:	0x5cda4684df214c2353e8fceb9f4fdb15b06c9dc0bde6e03d0f0cec645f03206e
Status:	Success
Block:	12564538 5 Block Confirmations
Timestamp:	1 min ago (Jul-10-2022 12:20:48 PM +UTC)
From:	0x46b1f1e3018a058512350afc63c39682aeec69ba
To:	Contract 0xbcfd1733a1250fc6d79ca397951eb57e4f239633 ✓
Value:	0 Ether ($0.00)
Transaction Fee:	0.000110522500353672 Ether ($0.00)
Gas Price:	0.000000002500000008 Ether (2.500000008 Gwei)
Gas Limit & Usage by Txn:	44,209 | 44,209 (100%)
Gas Fees:	Base: 0.000000008 Gwei | Max: 2.500000014 Gwei | Max Priority: 2.5 Gwei
Burnt & Txn Savings Fees:	Burnt: 0.000000000000353672 Ether ($0.00) Txn Savings: 0.000000000000265254 Ether ($0.00)
Others:	Txn Type: 2 (EIP-1559) Nonce: 2 Position: 2
Input Data:	

```
Function: add(uint256 a, uint256 b) ***

MethodID: 0x771602f7
[0]:  000000000000000000000000000000000000000000000000000000000000000c
[1]:  000000000000000000000000000000000000000000000000000000000000000a
```

- On clicking on Ropsten, we can see entire Ropsten Testnet Explorer, which includes latest blocks and latest transactions.



Ropsten Testnet Explorer

All Filters | Search by Address / Txn Hash / Block / Token / Ens

Latest Blocks

Bk	12564634 11 secs ago	Fee Recipient 0x965b7a773e3632b259… 0 txn in 12 secs	0 Eth
Bk	12564633 23 secs ago	Fee Recipient 0x000000000000000000… 17 txns in 12 secs	0.02153 Eth
Bk	12564632 35 secs ago	Fee Recipient 0x000000000000000000… 22 txns in 12 secs	0.06215 Eth
Bk	12564631 47 secs ago	Fee Recipient 0xb943c2c22b1b186a34… 13 txns in 12 secs	0.00247 Eth
Bk	12564630 59 secs ago	Fee Recipient 0xcccccccccccccccccccc… 13 txns in 12 secs	0.00144 Eth
Bk	12564629 1 min ago	Fee Recipient 0xf97e180c050e5ab072… 17 txns in 12 secs	0.01194 Eth

View all blocks

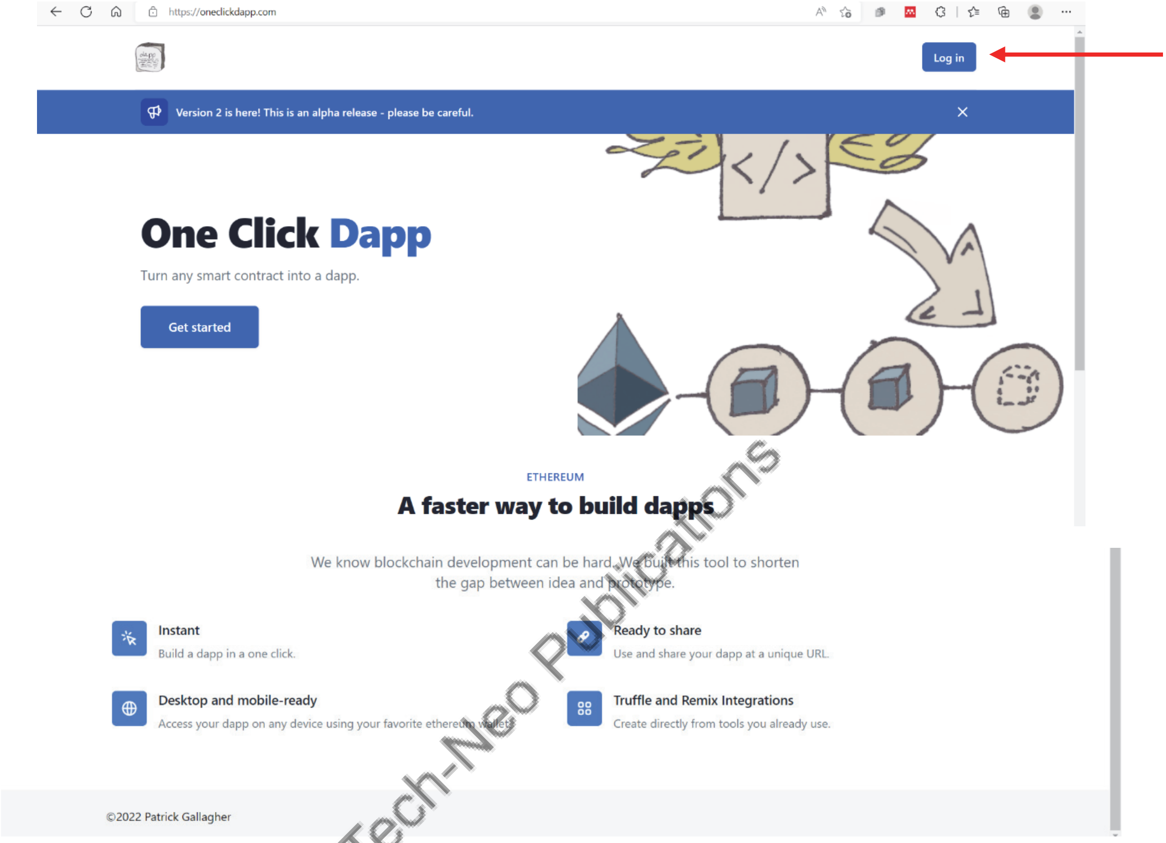
Latest Transactions

Tx	0xa629b39d9d78… 23 secs ago	From 0x81b7e08f65bdf564860… To 0x74c617e40b2297969f…	1 Eth
Tx	0xc037dbbe94b2… 23 secs ago	From 0x2ca5f489cc1fd1cec24… To [NewContract]	0 Eth
Tx	0xbfe0e7acdc520… 23 secs ago	From 0x2ca5f489cc1fd1cec24… To 0x0c1fd34d1f35095802e…	0 Eth
Tx	0xa59c7bfdae04… 23 secs ago	From 0x2ca5f489cc1fd1cec24… To [NewContract]	0 Eth
Tx	0xbeb2886bda92… 23 secs ago	From 0x2ca5f489cc1fd1cec24… To [NewContract]	0 Eth
Tx	0xbb95d1dc47cd… 23 secs ago	From 0x81b7e08f65bdf564860… To 0x467180cce028cf334e6	1 Eth

View all transactions

- Creating a simple application using ***One Click Dapp*** (https://oneclickdapp.com/). One Click Dapp turns any smart contract into a decentralized application (dapp).

Step 1 : With the help of One Click Dapp, a decentralized application can be developed. First, we need to "Log in" to Once Click Dapp.



https://oneclickdapp.com

Log in

Version 2 is here! This is an alpha release - please be careful.

One Click Dapp

Turn any smart contract into a dapp.

Get started

ETHEREUM

A faster way to build dapps

We know blockchain development can be hard. We built this tool to shorten the gap between idea and prototype.

Instant

Build a dapp in a one click.

Ready to share

Use and share your dapp at a unique URL.

Desktop and mobile-ready

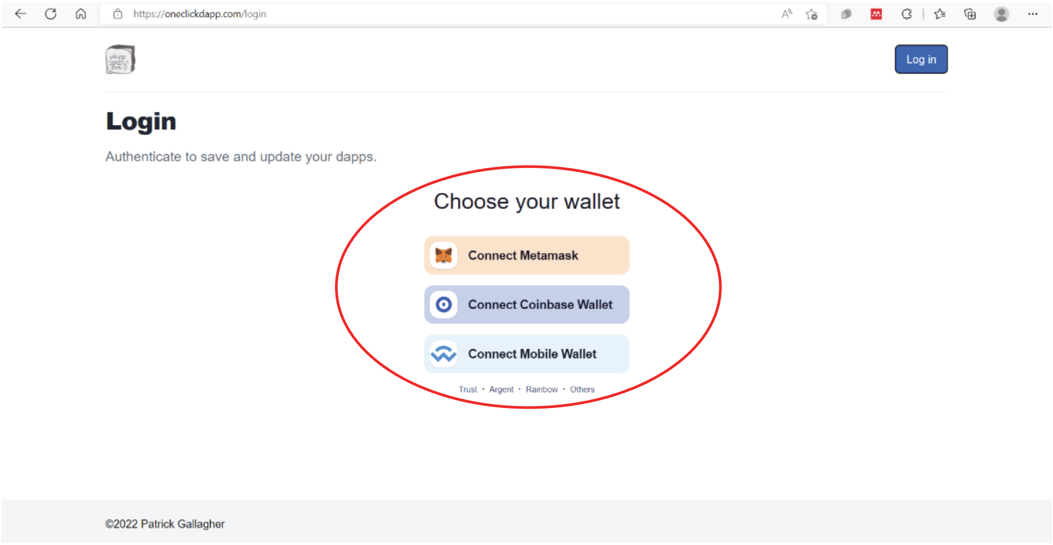
Access your dapp on any device using your favorite ethereum wallet.

Truffle and Remix Integrations

Create directly from tools you already use.

©2022 Patrick Gallagher

Step 2 : After "Log in", we need to choose a wallet from the available wallets.



https://oneclickdapp.com/login

Log in

Login

Authenticate to save and update your dapps.

Choose your wallet

Connect Metamask

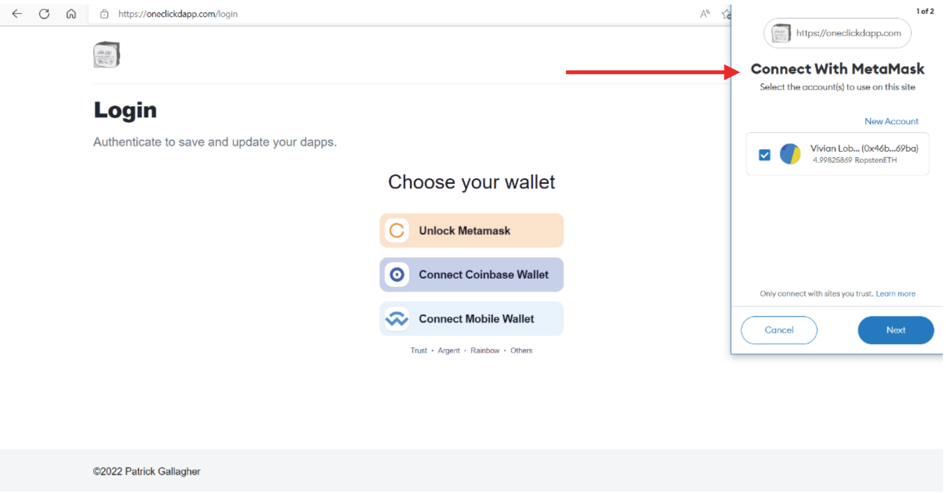
Connect Coinbase Wallet

Connect Mobile Wallet

Trust · Argent · Rainbow · Others

©2022 Patrick Gallagher

- **Step 3 :** We select MetaMask as the wallet and try to connect with it.



https://onedickdapp.com/login

Login

Authenticate to save and update your dapps.

Choose your wallet

Connect With MetaMask

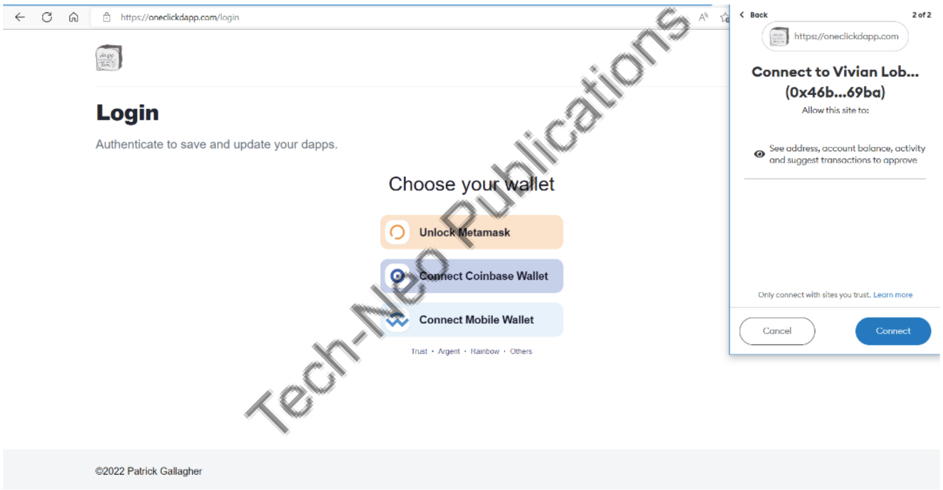
Select the account(s) to use on this site

Vivian Lob... (0x46b...69ba) 4.99825869 RopstenETH

Only connect with sites you trust. Learn more

©2022 Patrick Gallagher

- **Step 4 :** Waiting for connection to establish.



https://onedickdapp.com/login

Login

Authenticate to save and update your dapps.

Choose your wallet

Connect to Vivian Lob... (0x46b...69ba)

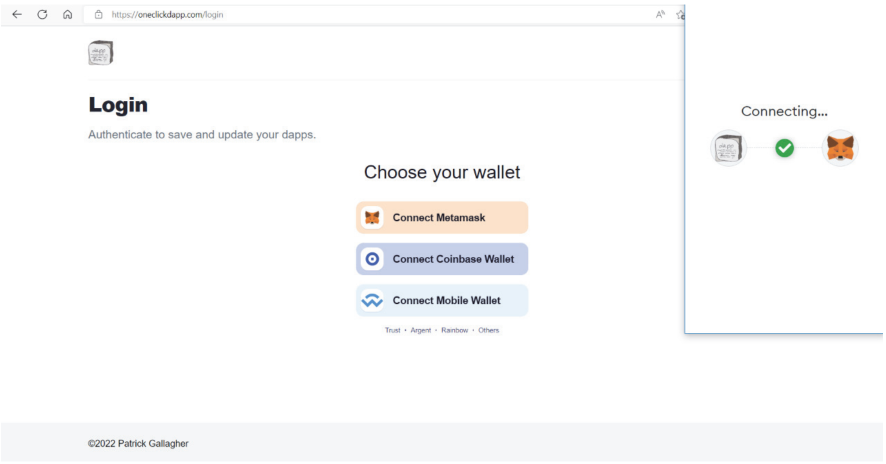
Allow this site to:

See address, account balance, activity and suggest transactions to approve

Only connect with sites you trust. Learn more

©2022 Patrick Gallagher

- **Step 5 :** One Click Dapp and MetaMask getting connected.



https://onedickdapp.com/login

Login

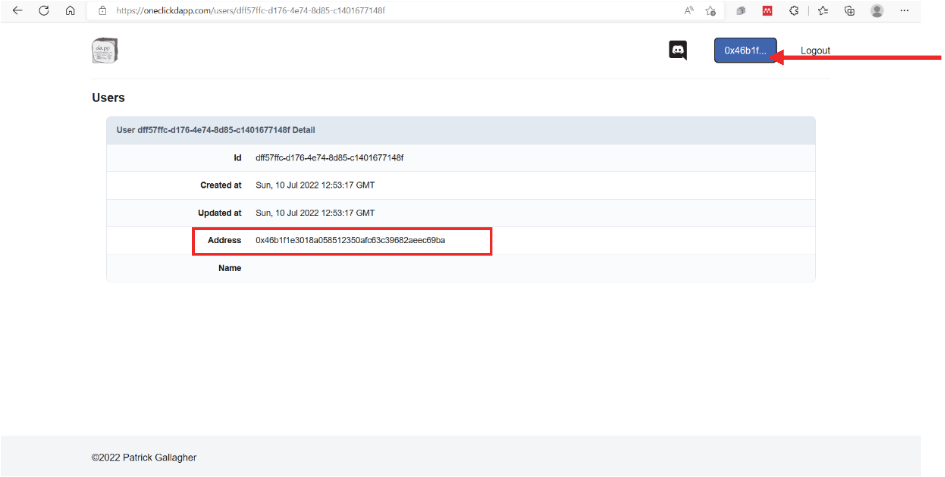
Authenticate to save and update your dapps.

Choose your wallet

Connecting...

©2022 Patrick Gallagher

- **Step 6 :** MetaMask wallet address (i.e., public key) can be seen after successful connection.



https://oneclickdapp.com/users/dff57ffc-d176-4e74-8d85-c1401677148f

0x46b1f... Logout

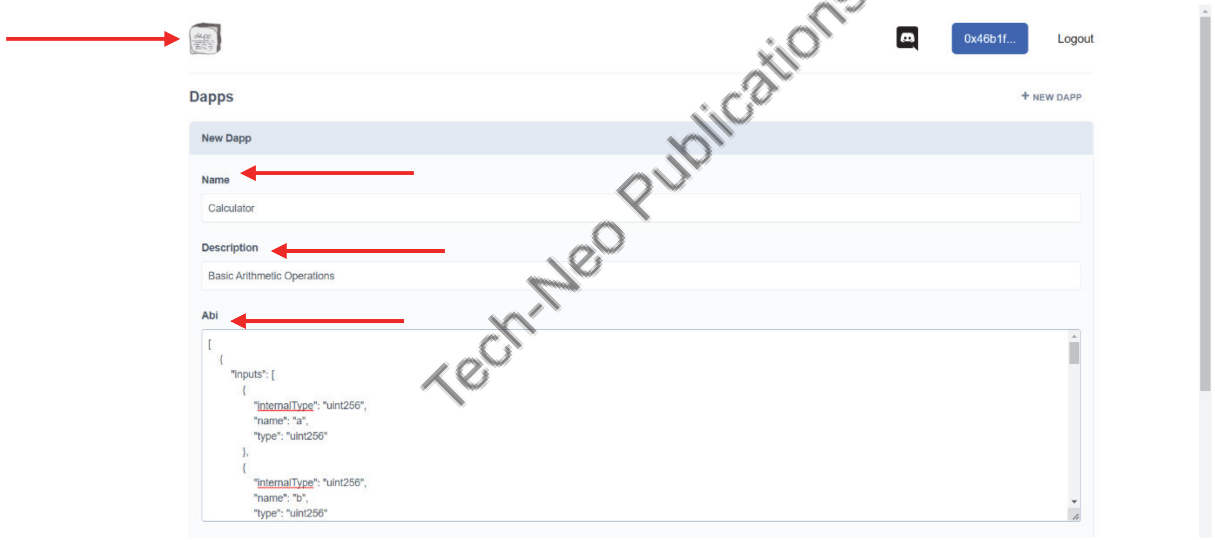
Users

User dff57ffc-d176-4e74-8d85-c1401677148f Detail

Id	dff57ffc-d176-4e74-8d85-c1401677148f
Created at	Sun, 10 Jul 2022 12:53:17 GMT
Updated at	Sun, 10 Jul 2022 12:53:17 GMT
Address	0x46b1f1e3018a058512350afc63c39682aeec69ba
Name	

©2022 Patrick Gallagher

- **Step 7:** Next, we click on the One Click Dapp and create a decentralized application wherein we need to specify the name, description, application binary interface (ABI) code, contract address, and test network name.



0x46b1f... Logout

Dapps + NEW DAPP

New Dapp

Name

Calculator

Description

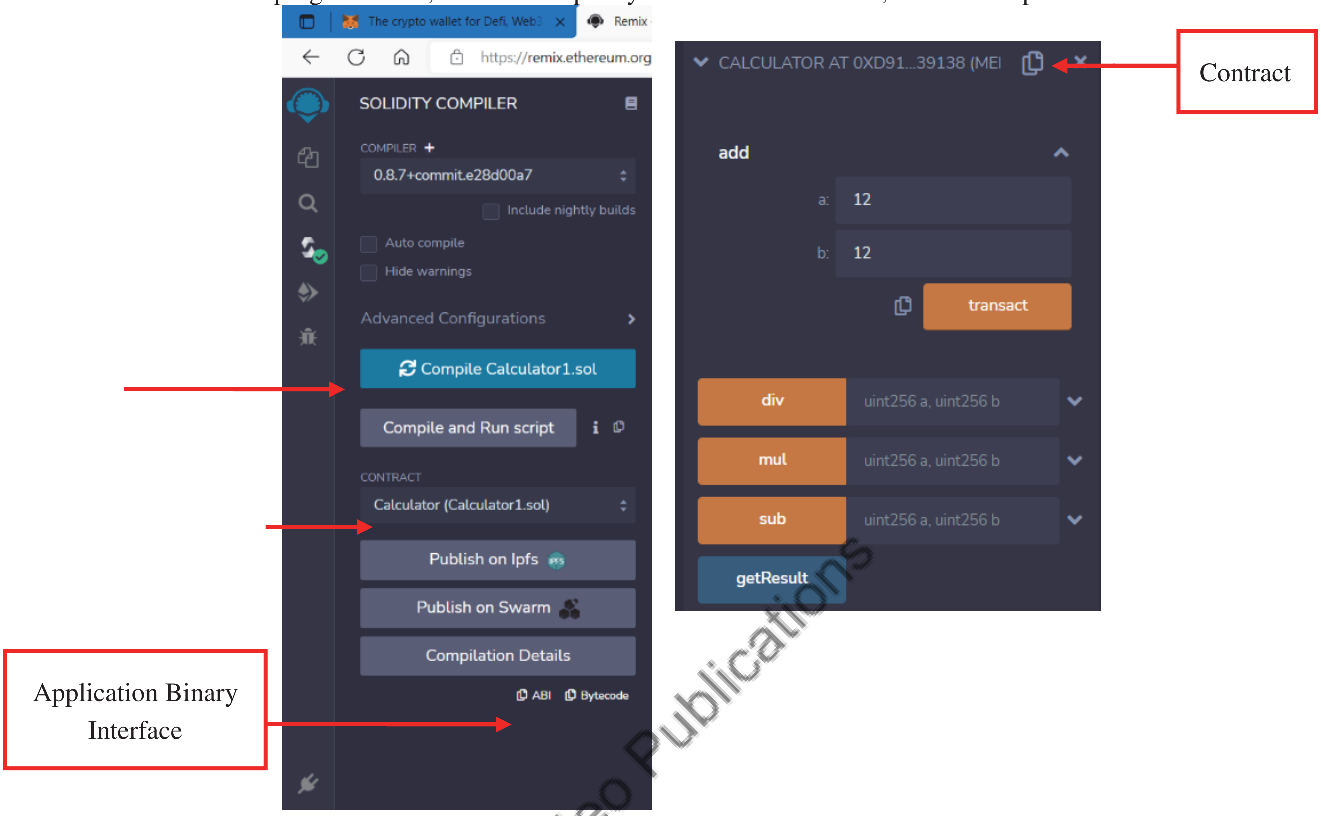
Basic Arithmetic Operations

Abi

```
[
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "a",
        "type": "uint256"
      },
      {
        "internalType": "uint256",
        "name": "b",
        "type": "uint256"
```

NOTES

- **Step 8 :** ABI code and contact address are obtained from Remix IDE where the solidity code is written for Calculator program. Also, we need to specify the test network name, which is Ropsten in our case.



COMPILER: 0.8.7+commit.e28d00a7

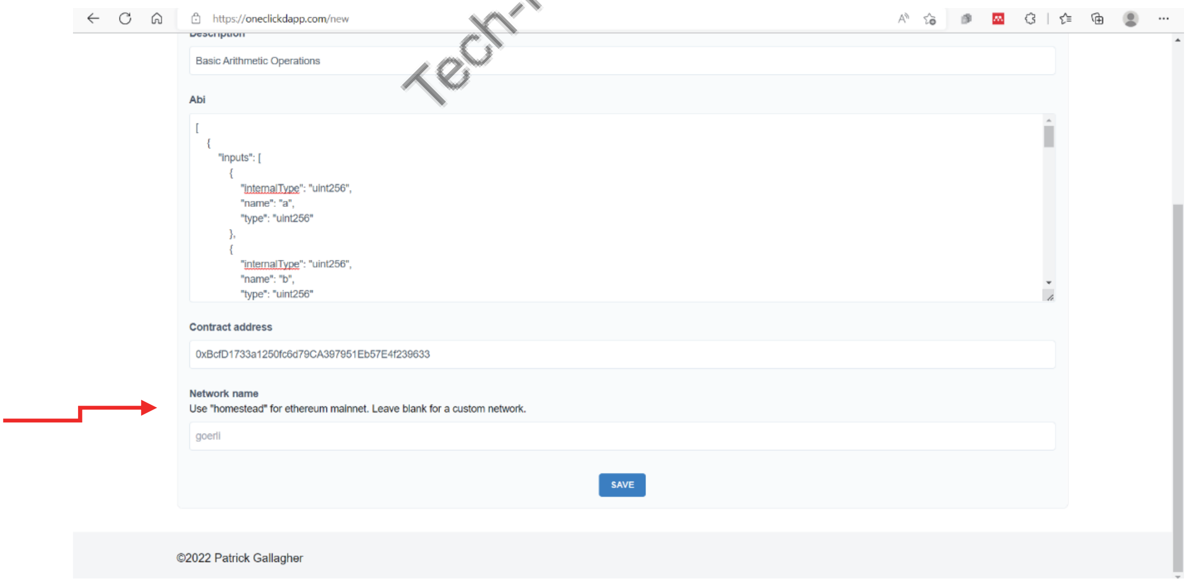
CONTRACT: Calculator (Calculator1.sol)

a: 12

b: 12

Contract

Application Binary Interface



Description

Basic Arithmetic Operations

Abi

```
[
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "a",
        "type": "uint256"
      },
      {
        "internalType": "uint256",
        "name": "b",
        "type": "uint256"
```

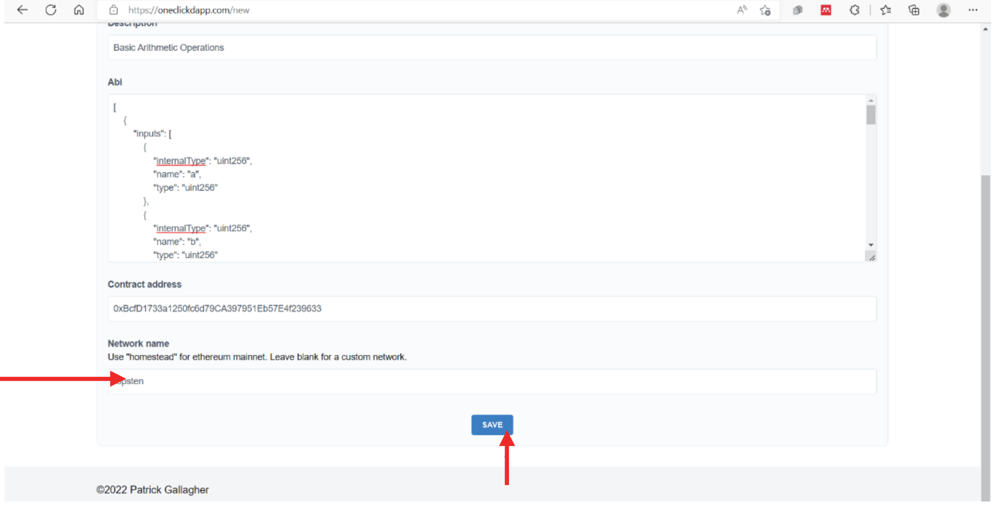
Contract address

0xBcfD1733a1250fc6d79CA397951Eb57E4f239633

Network name
Use "homestead" for ethereum mainnet. Leave blank for a custom network.

goerli

©2022 Patrick Gallagher



https://oneclickdapp.com/new

Description

Basic Arithmetic Operations

Abi

```
[
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "a",
        "type": "uint256"
      },
      {
        "internalType": "uint256",
        "name": "b",
        "type": "uint256"
```

Contract address

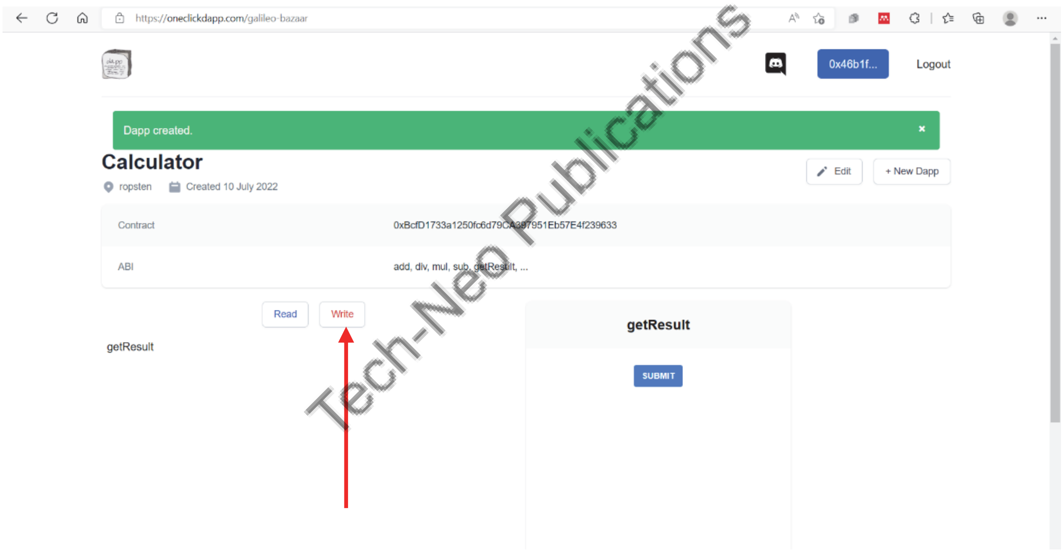
0xBcfD1733a1250fc6d79CA397951Eb57E4f239633

Network name
Use "homestead" for ethereum mainnet. Leave blank for a custom network.

psten

©2022 Patrick Gallagher

- **Step 9 :** Once done, click on SAVE button. Successful execution indicates that a Dapp has been created, as shown below.



https://oneclickdapp.com/galileo-bazaar

Dapp created.

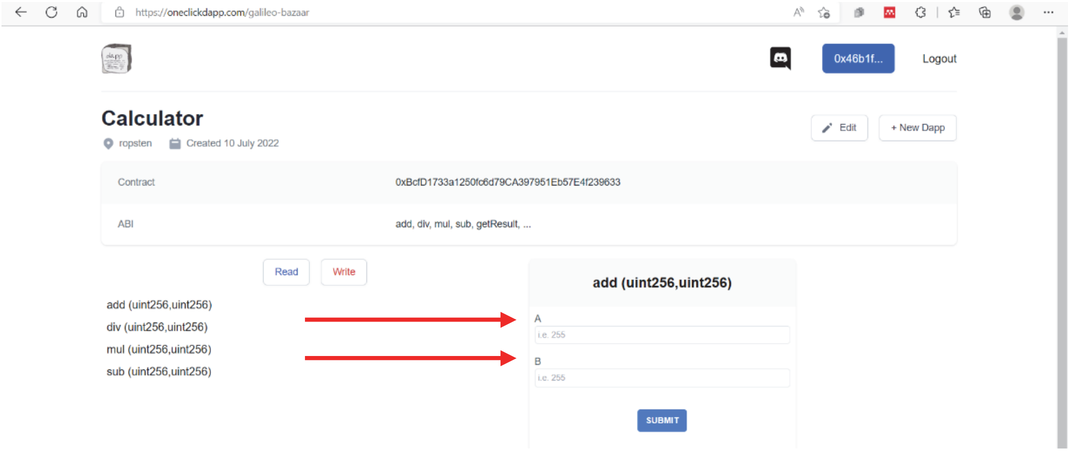
Calculator

ropsten Created 10 July 2022

Contract	0xBcfD1733a1250fc6d79CA397951Eb57E4f239633
ABI	add, div, mul, sub, getResult, ...

getResult

- **Step 10 :** Next, we click on "Write" and insert two numbers for A and B.



https://oneclickdapp.com/galileo-bazaar

Calculator

ropsten Created 10 July 2022

Contract	0xBcfD1733a1250fc6d79CA397951Eb57E4f239633
ABI	add, div, mul, sub, getResult, ...

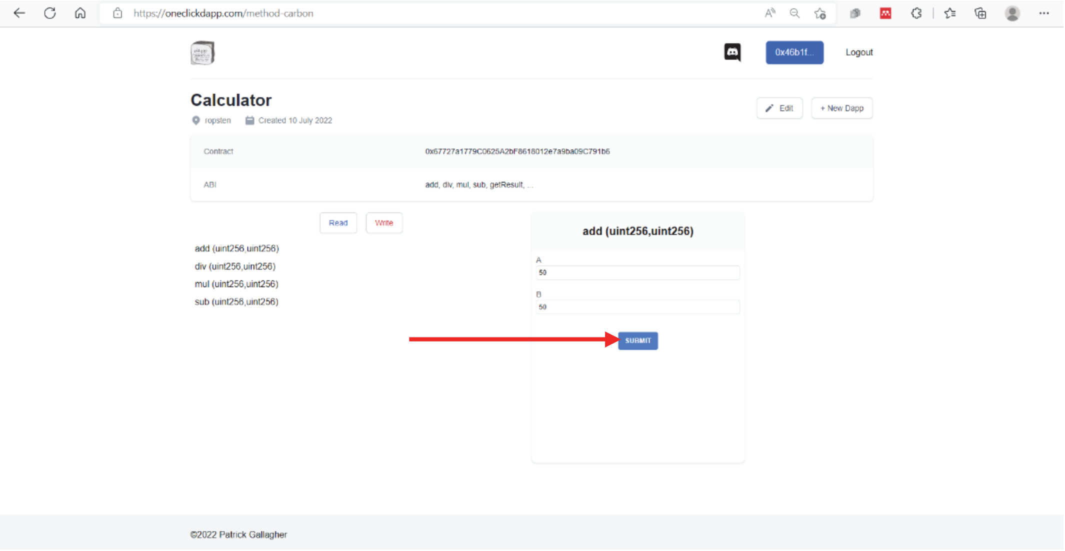
add (uint256,uint256)
div (uint256,uint256)
mul (uint256,uint256)
sub (uint256,uint256)

add (uint256,uint256)

A
i.e. 255

B
i.e. 255

- **Step 11 :** We assign A = 50 and B = 50 and then click on SUBMIT.



Calculator

ropsten Created 10 July 2022

Contract 0x67727a1779C0625A2bF8618012e7a9ba09C791b6

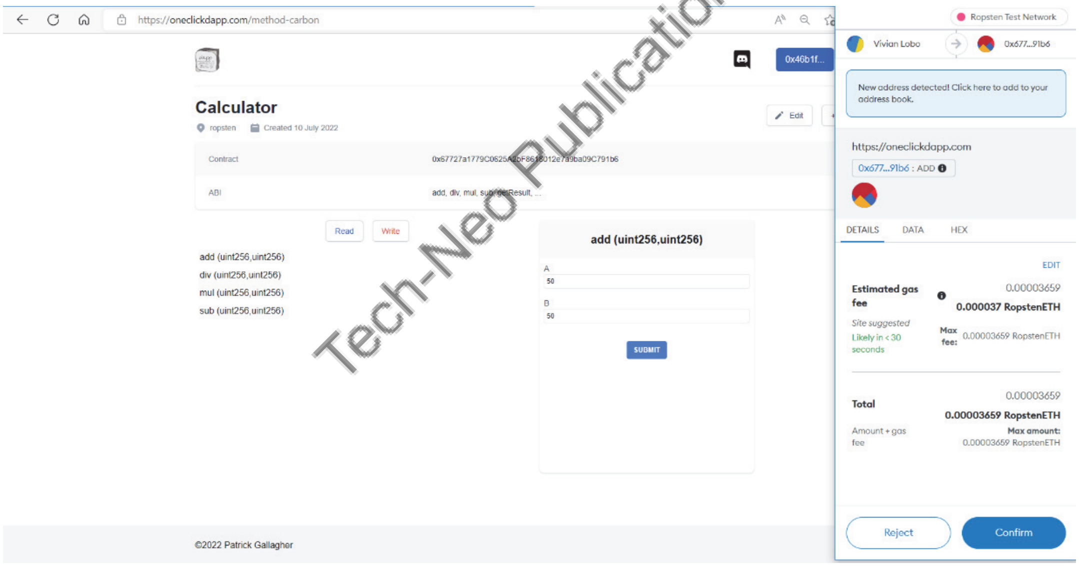
ABI add, div, mul, sub, getResult, ...

add (uint256,uint256)

A 50

B 50

- **Step 12 :** By clicking on SUBMIT, we get connected to MetaMask wallet wherein certain gas fee is estimated.



Calculator

ropsten Created 10 July 2022

Contract 0x67727a1779C0625A2bF8618012e7a9ba09C791b6

ABI add, div, mul, sub, getResult, ...

add (uint256,uint256)

A 50

B 50

New address detected! Click here to add to your address book.

https://oneclickdapp.com

0x677...91b6 : ADD

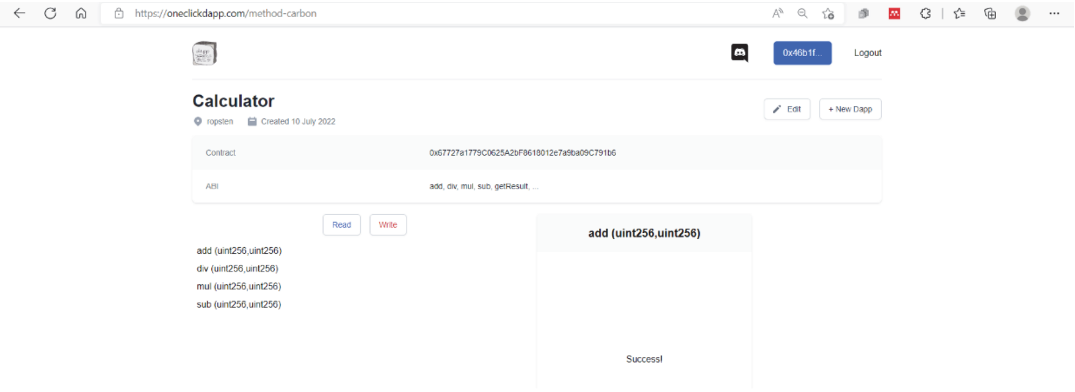
Estimated gas fee 0.00003659 0.000037 RopstenETH

Site suggested Likely in < 30 seconds Max fee: 0.00003659 RopstenETH

Total 0.00003659 0.00003659 RopstenETH

Amount + gas fee Max amount: 0.00003659 RopstenETH

- **Step 13 :** By clicking on Confirm, we obtain a Success! Message.



Calculator

ropsten Created 10 July 2022

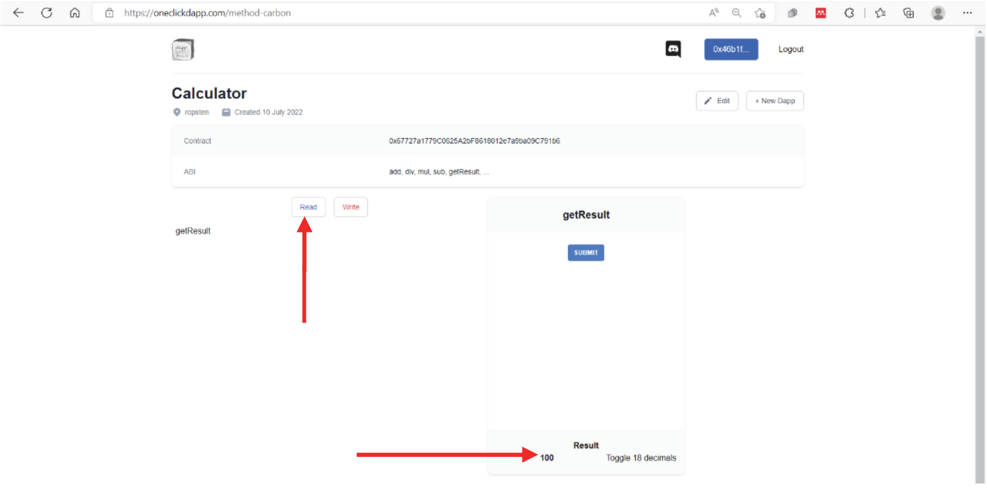
Contract 0x67727a1779C0625A2bF8618012e7a9ba09C791b6

ABI add, div, mul, sub, getResult, ...

add (uint256,uint256)

Success!

- **Step 14 :** Next, we click on Read and click on getResult() and then on SUBMIT, which gives us the result (i.e., 100)


https://oneclickdapp.com/method-carbon

Calculator

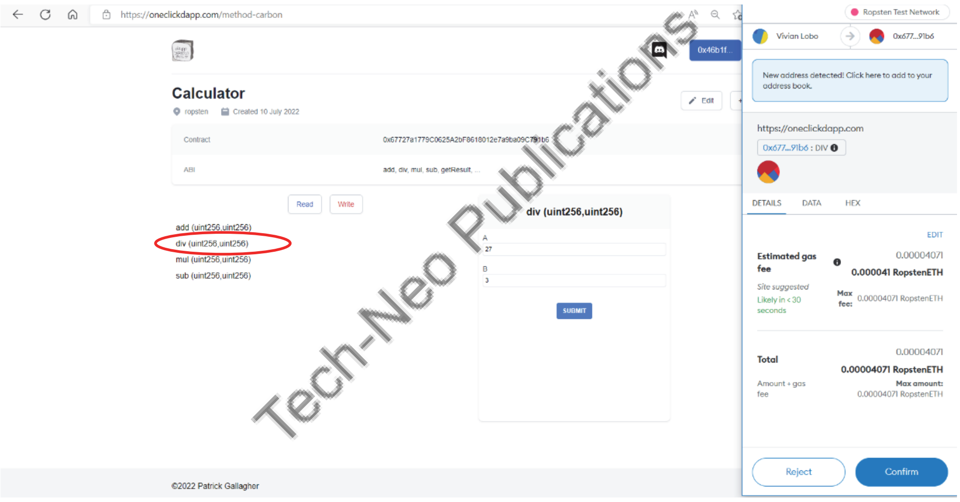
ropsten Created 10 July 2022

Contract 0x67727a1779C0625A2bF8618012e7a9ba09C791b6

ABI add, div, mul, sub, getResult, ...

Result 100

- **Step 15 :** In a similar manner, we can do it for other arithmetic operators.



https://oneclickdapp.com/method-carbon

Calculator

ropsten Created 10 July 2022

Contract 0x67727a1779C0625A2bF8618012e7a9ba09C791b6

ABI add, div, mul, sub, getResult, ...

add (uint256,uint256)
div (uint256,uint256)
mul (uint256,uint256)
sub (uint256,uint256)

div (uint256,uint256)

A 27

B 3

©2022 Patrick Gallagher

Ropsten Test Network

Vivian Lobo → 0x677...91b6

New address detected! Click here to add to your address book.

https://oneclickdapp.com

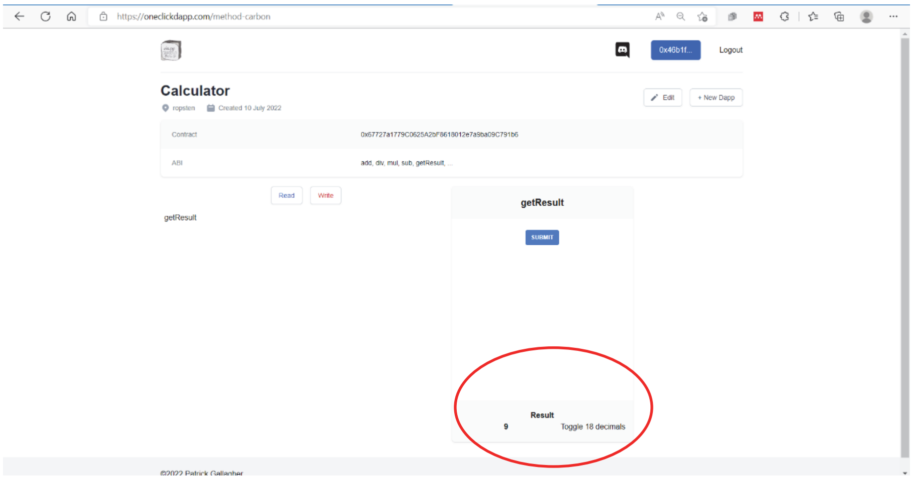
0x677...91b6 : DIV

Estimated gas fee 0.00004071 0.000041 RopstenETH

Site suggested Likely in < 30 seconds Max fee: 0.00004071 RopstenETH

Total 0.00004071 0.00004071 RopstenETH

Amount + gas fee Max amount: 0.00004071 RopstenETH


https://oneclickdapp.com/method-carbon

Calculator

ropsten Created 10 July 2022

Contract 0x67727a1779C0625A2bF8618012e7a9ba09C791b6

ABI add, div, mul, sub, getResult, ...

Result 9

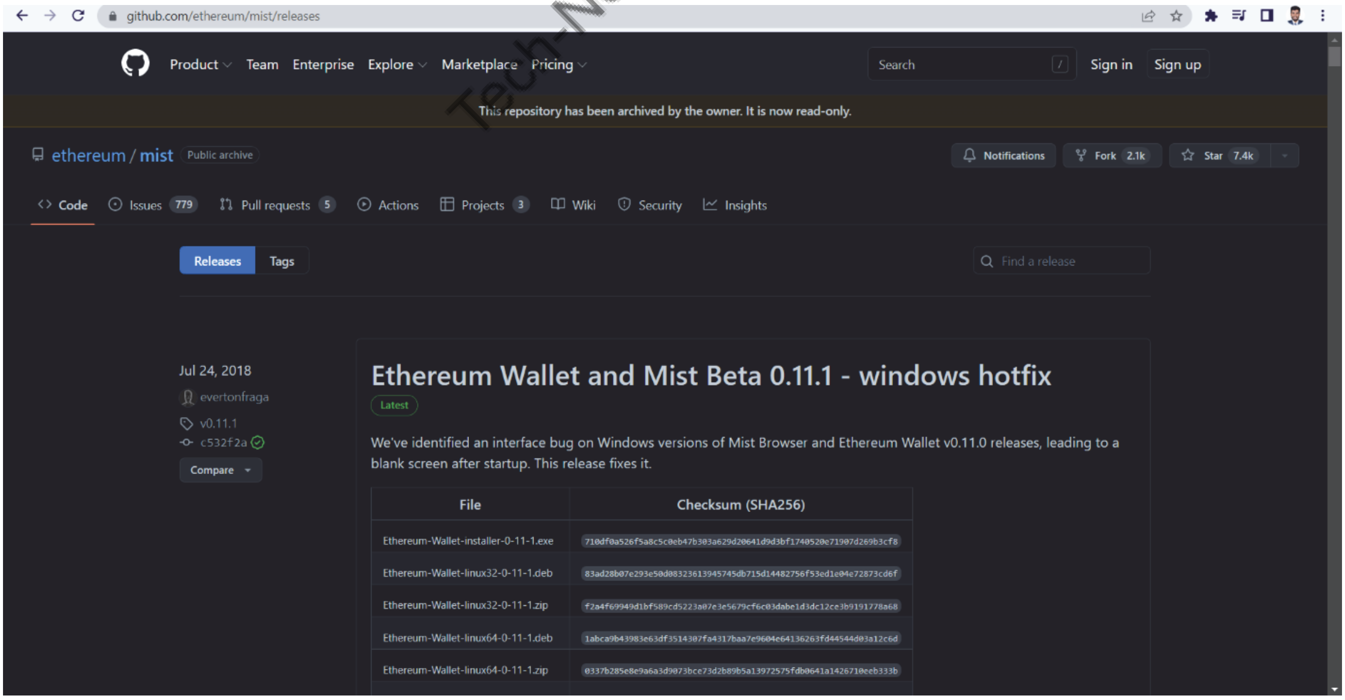
©2022 Patrick Gallagher

Ethereum Mist Wallet

- It is Ethereum's official blockchain wallet where an individual can hold his/her Ethereum assets.
- This wallet was created by a team of Ethereum Foundation so that decentralized applications and blockchain applications can be executed.
- Ethereum is the second-largest crypto by market capitalization, indicating that it is a developing ecosystem.
- Cybercriminals are often on a lurk looking to target this ecosystem.
- The Ethereum Mist Wallet was launched in 2017. It is a blockchain-based browser that serves as a platform for operating blockchain networks and decentralized applications.
- Thus, Mist is a search engine with several Dapps connected to it, and Ethereum is one of those.
- Note that Ethereum Mist wallet is compatible with Linux, Windows, and macOS.
- The purpose of the wallet is to hold assets and keep them safe until the owner feels the need for it. Ethereum Mist wallet is one of the safest Ethereum wallets that one can adapt to store his/her assets.
- Ethereum Mist wallet saves the private key on an individual's computer and not on third-party servers.
- The wallet also has multiple signature wallet options, thereby permitting multiple accounts. Thus, it is possible to have one mist wallet with multiple signatures.
- With adequate security measures such as recovery services, it is easier to monitor the signatory sign-ins and authorization of transactions.

Steps to setup an Ethereum Wallet Mist

- **Step 1:** Download Mist by visiting https://github.com/ethereum/mist/releases

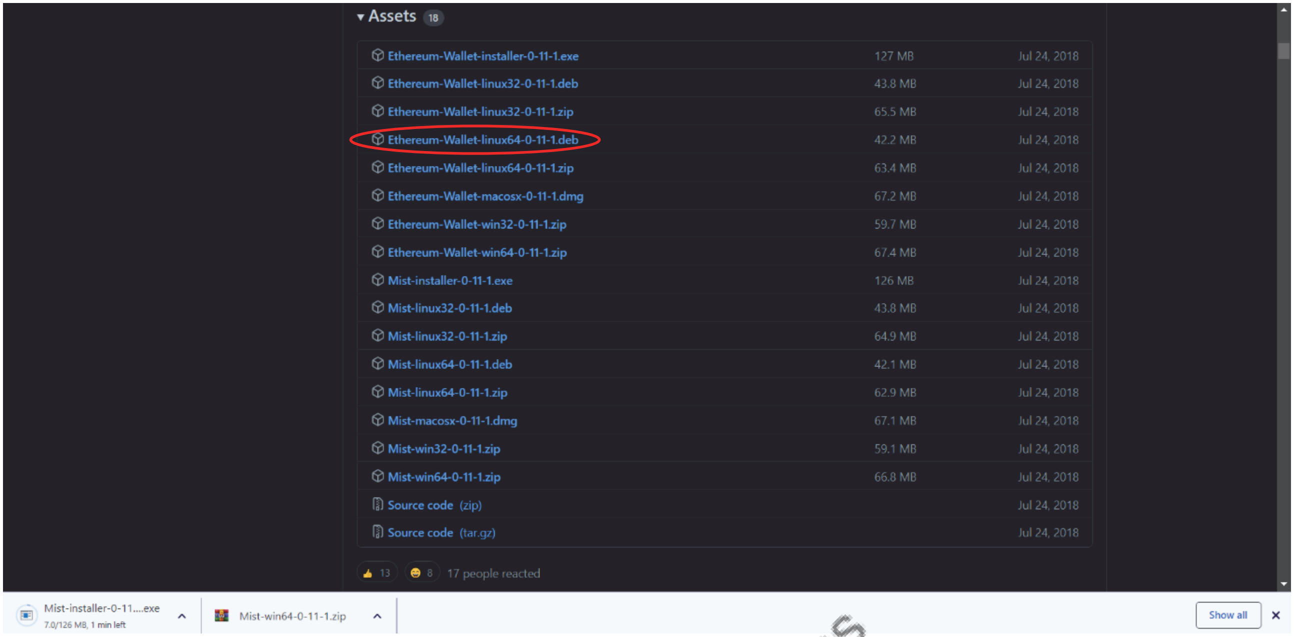


This repository has been archived by the owner. It is now read-only.

Ethereum Wallet and Mist Beta 0.11.1 - windows hotfix

We've identified an interface bug on Windows versions of Mist Browser and Ethereum Wallet v0.11.0 releases, leading to a blank screen after startup. This release fixes it.

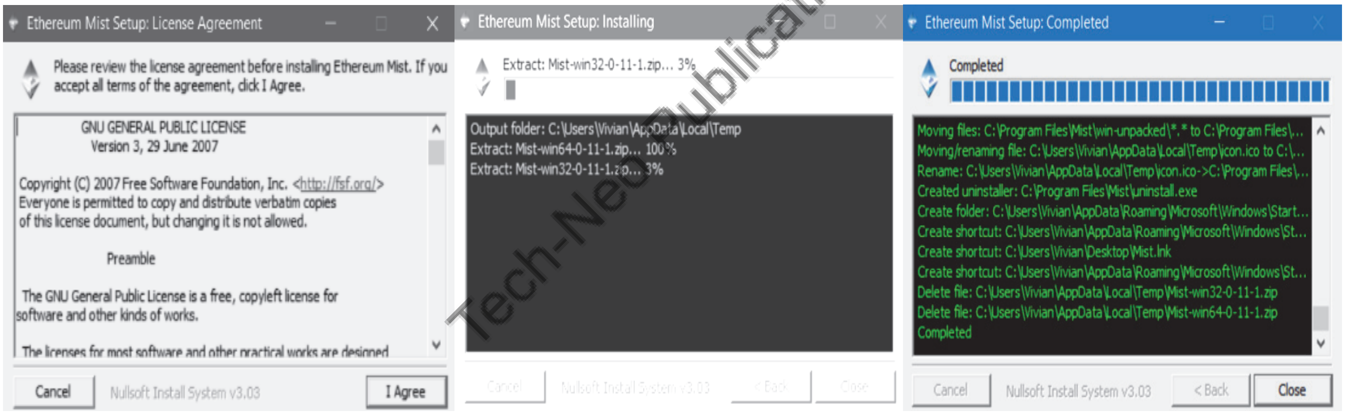
File	Checksum (SHA256)
Ethereum-Wallet-installer-0-11-1.exe	710df0a526f5a8c5c0eb47b303a629d20641d9d3bf1740520e71907d269b3cf8
Ethereum-Wallet-linux32-0-11-1.deb	83ad28b07e293e50d08323613945745db715d14482756f53ed1e04e72873cd6f
Ethereum-Wallet-linux32-0-11-1.zip	f2a4f69949d1bf589cd5223a07e3e5679cf6c03dabe1d3dc12ce3b9191778a68
Ethereum-Wallet-linux64-0-11-1.deb	1abca9b43983e63df3514307fa4317baa7e9604e64136263fd44544d03a12c6d
Ethereum-Wallet-linux64-0-11-1.zip	0337b285e8e9a6a3d9073bce73d2b89b5a13972575fdb0641a1426710eeb333b



▾ Assets 18

Ethereum-Wallet-installer-0-11-1.exe	127 MB	Jul 24, 2018
Ethereum-Wallet-linux32-0-11-1.deb	43.8 MB	Jul 24, 2018
Ethereum-Wallet-linux32-0-11-1.zip	65.5 MB	Jul 24, 2018
Ethereum-Wallet-linux64-0-11-1.deb	42.2 MB	Jul 24, 2018
Ethereum-Wallet-linux64-0-11-1.zip	63.4 MB	Jul 24, 2018
Ethereum-Wallet-macosx-0-11-1.dmg	67.2 MB	Jul 24, 2018
Ethereum-Wallet-win32-0-11-1.zip	59.7 MB	Jul 24, 2018
Ethereum-Wallet-win64-0-11-1.zip	67.4 MB	Jul 24, 2018
Mist-installer-0-11-1.exe	126 MB	Jul 24, 2018
Mist-linux32-0-11-1.deb	43.8 MB	Jul 24, 2018
Mist-linux32-0-11-1.zip	64.9 MB	Jul 24, 2018
Mist-linux64-0-11-1.deb	42.1 MB	Jul 24, 2018
Mist-linux64-0-11-1.zip	62.9 MB	Jul 24, 2018
Mist-macosx-0-11-1.dmg	67.1 MB	Jul 24, 2018
Mist-win32-0-11-1.zip	59.1 MB	Jul 24, 2018
Mist-win64-0-11-1.zip	66.8 MB	Jul 24, 2018
Source code (zip)		Jul 24, 2018
Source code (tar.gz)		Jul 24, 2018

- **Step 2 :** Follow the standard normal procedure for completion of the setup of Ethereum Mist Wallet.



Ethereum Mist Setup: License Agreement

Please review the license agreement before installing Ethereum Mist. If you accept all terms of the agreement, click I Agree.

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed

Ethereum Mist Setup: Installing

Extract: Mist-win32-0-11-1.zip... 3%

```
Output folder: C:\Users\Vivian\AppData\Local\Temp
Extract: Mist-win64-0-11-1.zip... 100%
Extract: Mist-win32-0-11-1.zip... 3%
```

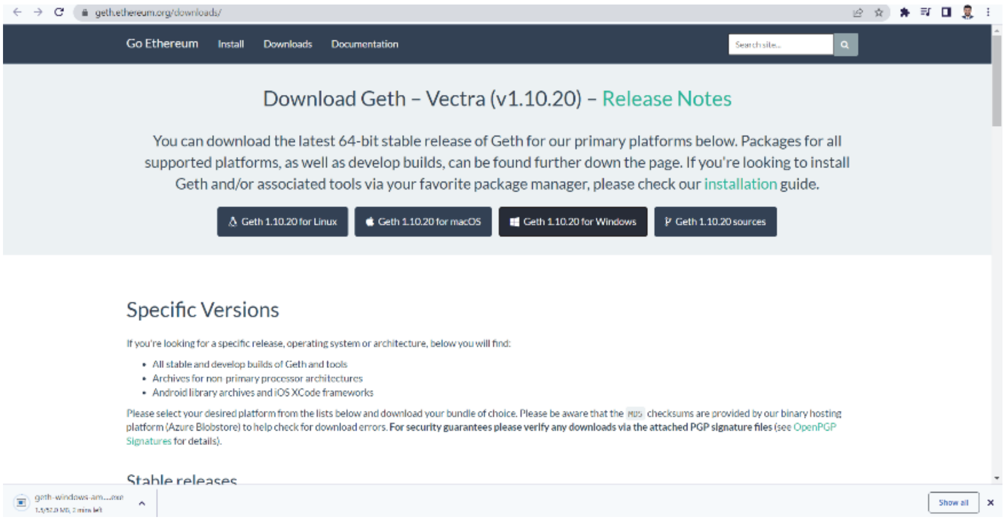
Ethereum Mist Setup: Completed

Completed

```
Moving files: C:\Program Files\Mist\win-unpacked\*.* to C:\Program Files\...
Moving/renaming file: C:\Users\Vivian\AppData\Local\Temp\icon.ico to C:\...
Rename: C:\Users\Vivian\AppData\Local\Temp\icon.ico->C:\Program Files\...
Created uninstaller: C:\Program Files\Mist\uninstall.exe
Create folder: C:\Users\Vivian\AppData\Roaming\Microsoft\Windows\Start...
Create shortcut: C:\Users\Vivian\AppData\Roaming\Microsoft\Windows\St...
Create shortcut: C:\Users\Vivian\Desktop\Mist.lnk
Create shortcut: C:\Users\Vivian\AppData\Roaming\Microsoft\Windows\St...
Delete file: C:\Users\Vivian\AppData\Local\Temp\Mist-win32-0-11-1.zip
Delete file: C:\Users\Vivian\AppData\Local\Temp\Mist-win64-0-11-1.zip
Completed
```

- **Step 3 :** The Mist icon appears. On clicking, the following window opens.

- **Step 4:** In case there is a need to install Geth for accessing Mist, then Geth can be downloaded and installed from https://geth.ethereum.org/downloads/



Download Geth – Vectra (v1.10.20) – Release Notes

You can download the latest 64-bit stable release of Geth for our primary platforms below. Packages for all supported platforms, as well as develop builds, can be found further down the page. If you're looking to install Geth and/or associated tools via your favorite package manager, please check our installation guide.

Specific Versions

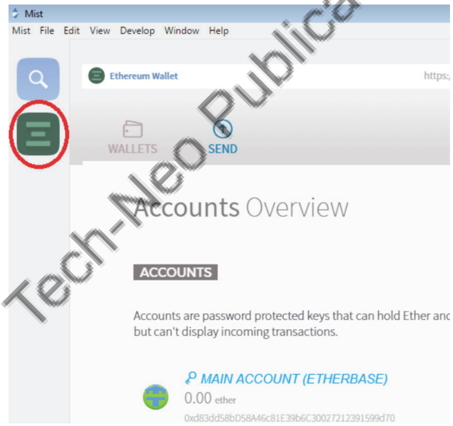
If you're looking for a specific release, operating system or architecture, below you will find:

- All stable and develop builds of Geth and tools
- Archives for non primary processor architectures
- Android library archives and iOS XCode frameworks

Please select your desired platform from the lists below and download your bundle of choice. Please be aware that the MD5 checksums are provided by our binary hosting platform (Azure Blobstore) to help check for download errors. **For security guarantees please verify any downloads via the attached PGP signature files** (see OpenPGP Signatures for details).

Stable releases

- **Step 5 :** Next, the Mist Wallet opens only after a password has been set for the account, as shown below. Note that the password cannot be reset, so remember the password.



Accounts Overview

Accounts are password protected keys that can hold Ether and but can't display incoming transactions.

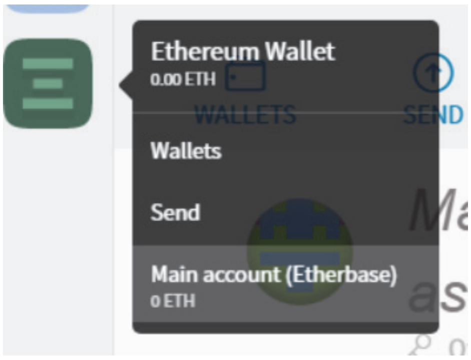
0.00 ether

0xd83dd58bD58A46c81E39b6C30027212391599d70

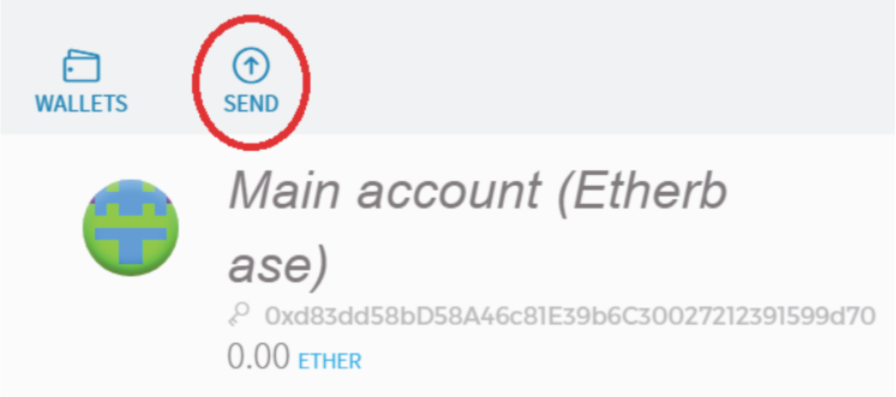
- **Step 6:** Click "MAIN ACCOUNT" and below it you can view and copy your Ethereum Mist wallet address.
- **Step 7:** Paste the address in the relevant exchange or wallet to send funds to your wallet.

How to send payments on Ethereum Mist wallet ?

- **Step 1 :** Launch your wallet and select the account from which you want to send ETH**.**



- **Step 2 :** Click on "Send"

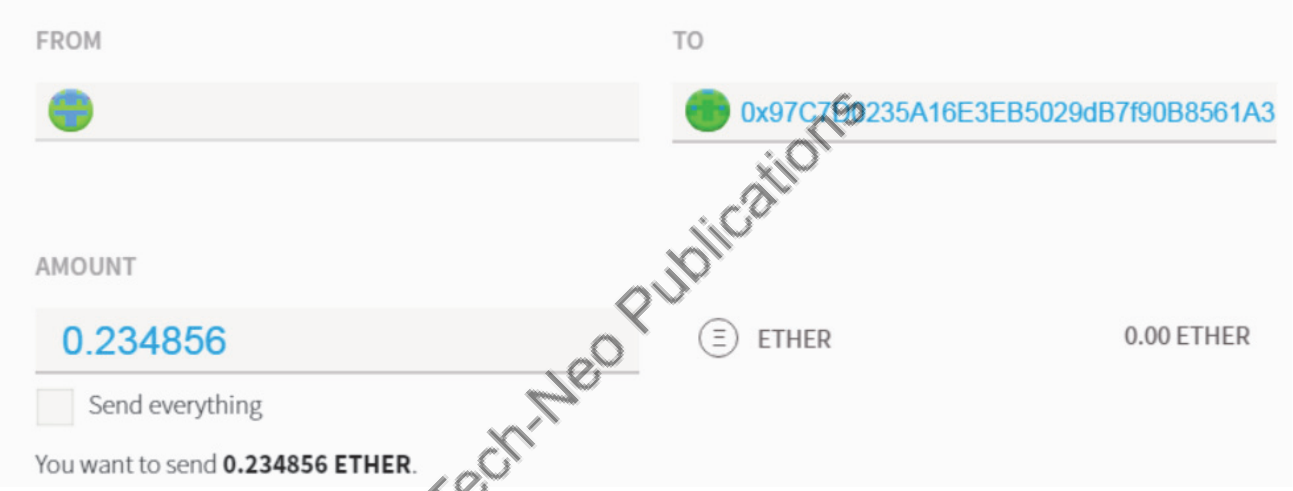


Main account (Etherbase)

0xd83dd58bD58A46c81E39b6C30027212391599d70

0.00 ETHER

- **Step 3 :** In the "To" box, enter your recipient's wallet address and then enter the amount



FROM

TO

0x97C7D0235A16E3EB5029dB7f90B8561A3

AMOUNT

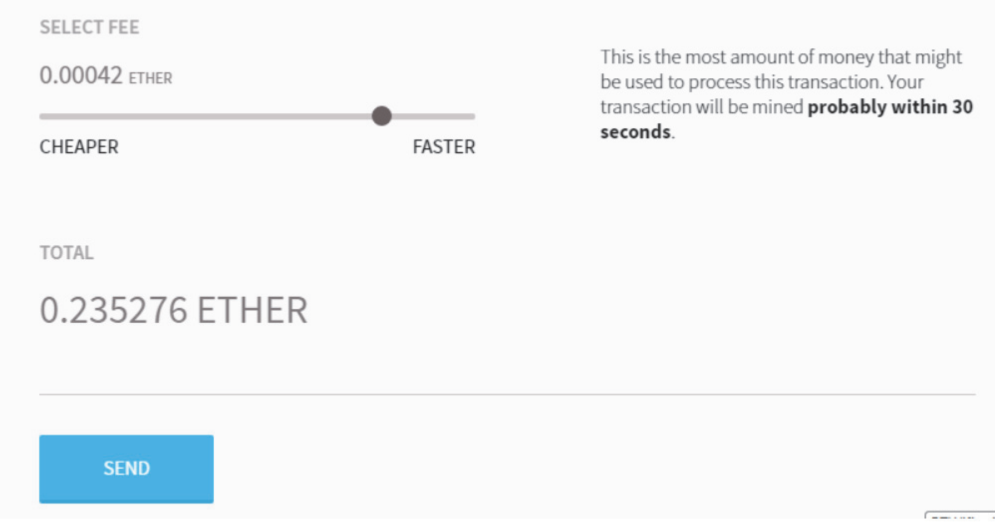
0.234856

□ Send everything

You want to send **0.234856 ETHER**.

ETHER 0.00 ETHER

Step 4 : Select gas fee or leave it as it is and then click "Send"



SELECT FEE

0.00042 ETHER

This is the most amount of money that might be used to process this transaction. Your transaction will be mined **probably within 30 seconds**.

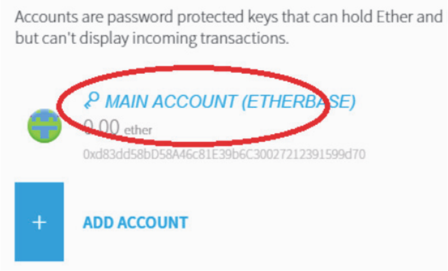
TOTAL

0.235276 ETHER

- **Step 5 :** Enter your password to confirm transaction and then hit "Send"

How to receive payments on Ethereum Mist wallet?

- **Step 1 :** Click on the Accounts Overview button
- **Step 2 :** Click "Main Account" and then copy your wallet address and send it to your counterparty.



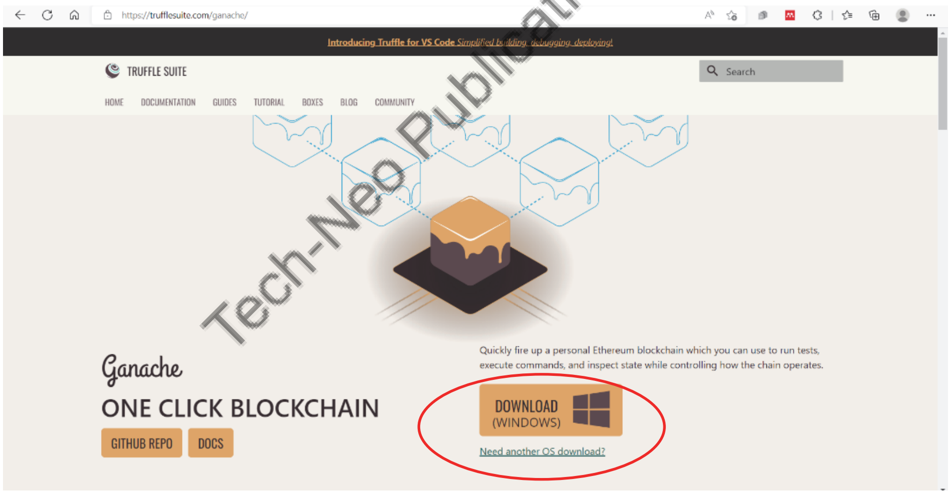
Accounts are password protected keys that can hold Ether and but can't display incoming transactions.

MAIN ACCOUNT (ETHERBASE)
0.00 ether
0xd83dd58bD58A46c81E39b6C30027212391599d70

Ganache for Ethereum Blockchain

It is a personal blockchain for Ethereum. Ganache was formerly known as TestRPC, which can be used to deploy smart contracts, develop applications, and run tests. It also possesses features like advanced mining controls and configuring advanced mining options in a single check. It provides a link to examine all blocks and transactions.

- **Step 1 :** From Truffle Suite (i.e., https://trufflesuite.com/), Ganache can be downloaded (i.e. https://trufflesuite.com/ganache/).

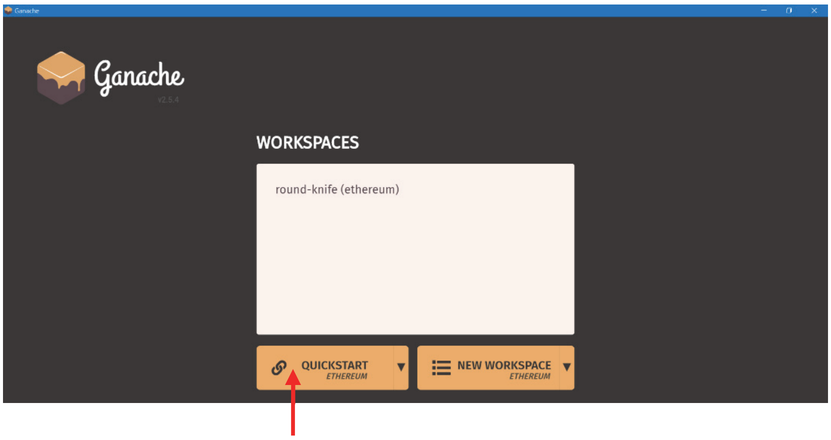


https://trufflesuite.com/ganache/

Introducing Truffle for VS Code Simplified building, debugging, deploying!

Quickly fire up a personal Ethereum blockchain which you can use to run tests, execute commands, and inspect state while controlling how the chain operates.

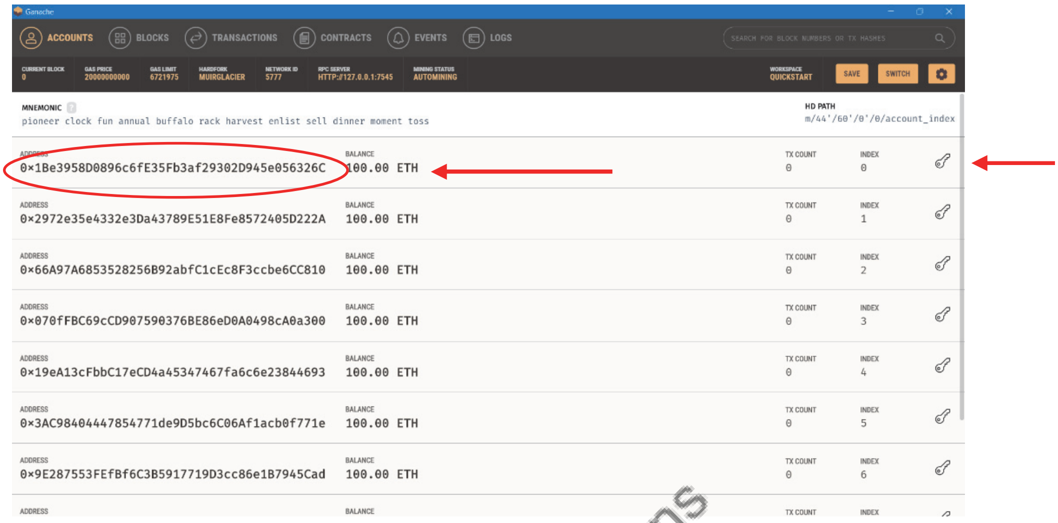
- **Step 2 :** On successfully downloading and installing Ethereum, the following GUI of Ganache is obtained. We select QUICKSTART ETHEREUM.



WORKSPACES

round-knife (ethereum)

- **Step 3 :** Ganache has pre-determined set of accounts (i.e., 10 accounts) that can be used when we write codes in solidity wherein test ethers (100 ETH) are made available for each of these accounts. Each account possesses its own ADDRESS and Private key to facilitate transactions.



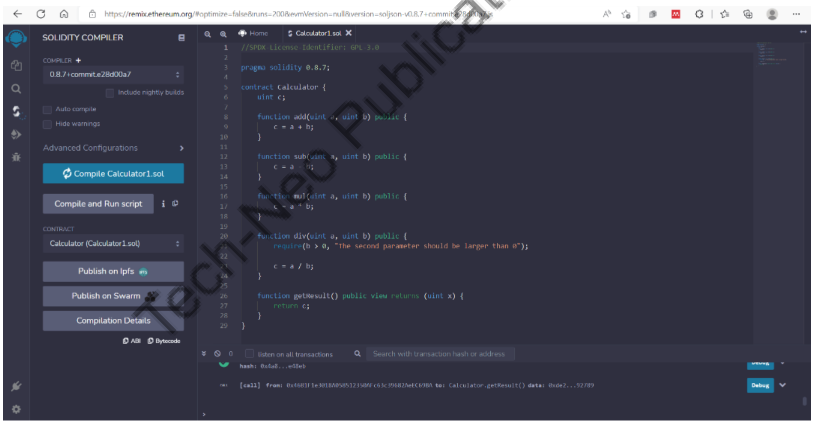
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
0	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	QUICKSTART

MNEMONIC
pioneer clock fun annual buffalo rack harvest enlist sell dinner moment toss

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×1Be3958D0896c6fE35Fb3af29302D945e056326C	100.00 ETH	0	0
0×2972e35e4332e3Da43789E51E8Fe8572405D222A	100.00 ETH	0	1
0×66A97A6853528256B92abfC1cEc8F3ccbe6CC810	100.00 ETH	0	2
0×070fFBC69cCD907590376BE86eD0A0498cA0a300	100.00 ETH	0	3
0×19eA13cFbbC17eCD4a45347467fa6c6e23844693	100.00 ETH	0	4
0×3AC98404447854771de9D5bc6C06Af1acb0f771e	100.00 ETH	0	5
0×9E287553FEfBf6C3B5917719D3cc86e1B7945Cad	100.00 ETH	0	6

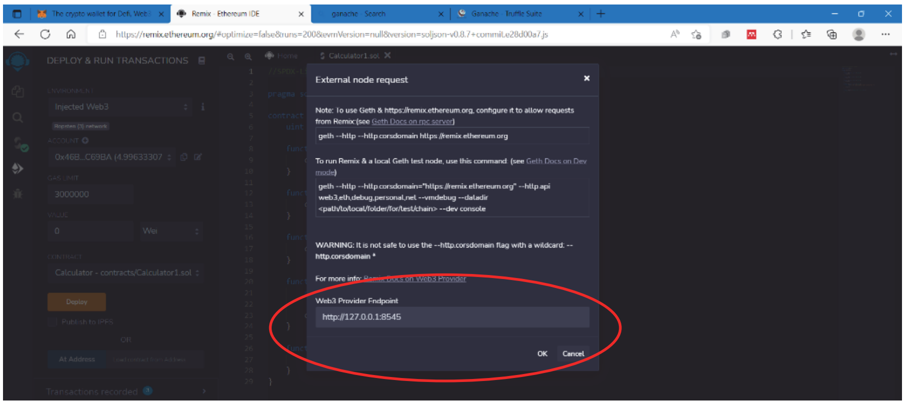
- **Step 4 :** Simple calculator program written in Remix IDE



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

```
hash: 0x4a8...e48eb
[call] from: 0x4681F1e3018A058512350AFc63c39682AeEC69BA to: Calculator.getResult() data: 0xde2...92789
```

- **Step 5 :** Connecting Remix IDE with Ganache. Once the simple calculator code is written, then under ENVIRONMENT, we need to select **Web3 Provider** to connect Remix IDE to Ganache. On selecting Web3 Provider, a pop-up occurs wherein a window opens as asks for "External node request". Under this, we need to specify Web3 Provider Endpoint.


The crypto wallet for Defi, Web3
Remix - Ethereum IDE
ganache - Search
Ganache - Truffle Suite
https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
ACCOUNT
0x468...C69BA (4.99633307
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
Calculator - contracts/Calculator1.sol
Deploy
Publish to IPFS
OR
At Address
Transactions recorded
Home
Calculator1.sol

External node request

Note: To use Geth & https://remix.ethereum.org, configure it to allow requests from Remix:(see Geth Docs on rpc server)

```
geth --http --http.corsdomain https://remix.ethereum.org
```

To run Remix & a local Geth test node, use this command: (see Geth Docs on Dev mode)

```
geth --http --http.corsdomain="https://remix.ethereum.org" --http.api
web3,eth,debug,personal,net --vmdebug --datadir
<path/to/local/folder/for/test/chain> --dev console
```

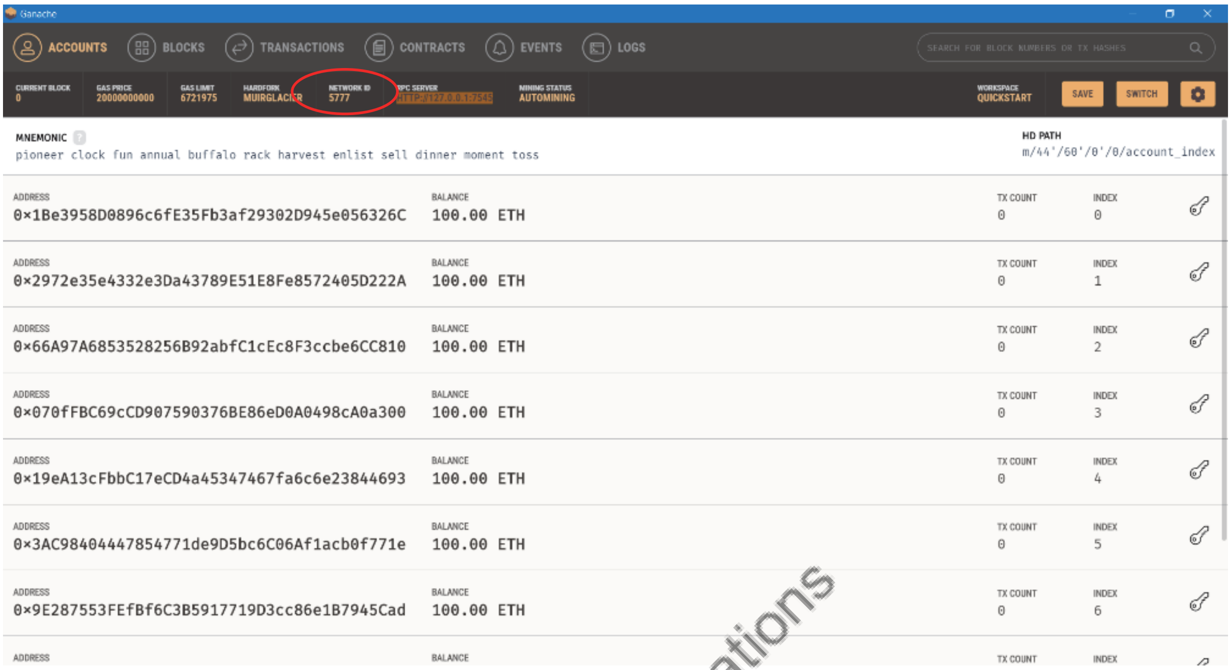
WARNING: It is not safe to use the --http.corsdomain flag with a wildcard: --http.corsdomain *

For more info: Remix Docs on Web3 Provider

Web3 Provider Endpoint: http://127.0.0.1:8545

OK Cancel

- **Step 6 :** In Ganache, under RPC server, we can see the Web3 Provider Endpoint (i.e., HTTP://127.0.0.1:7545).



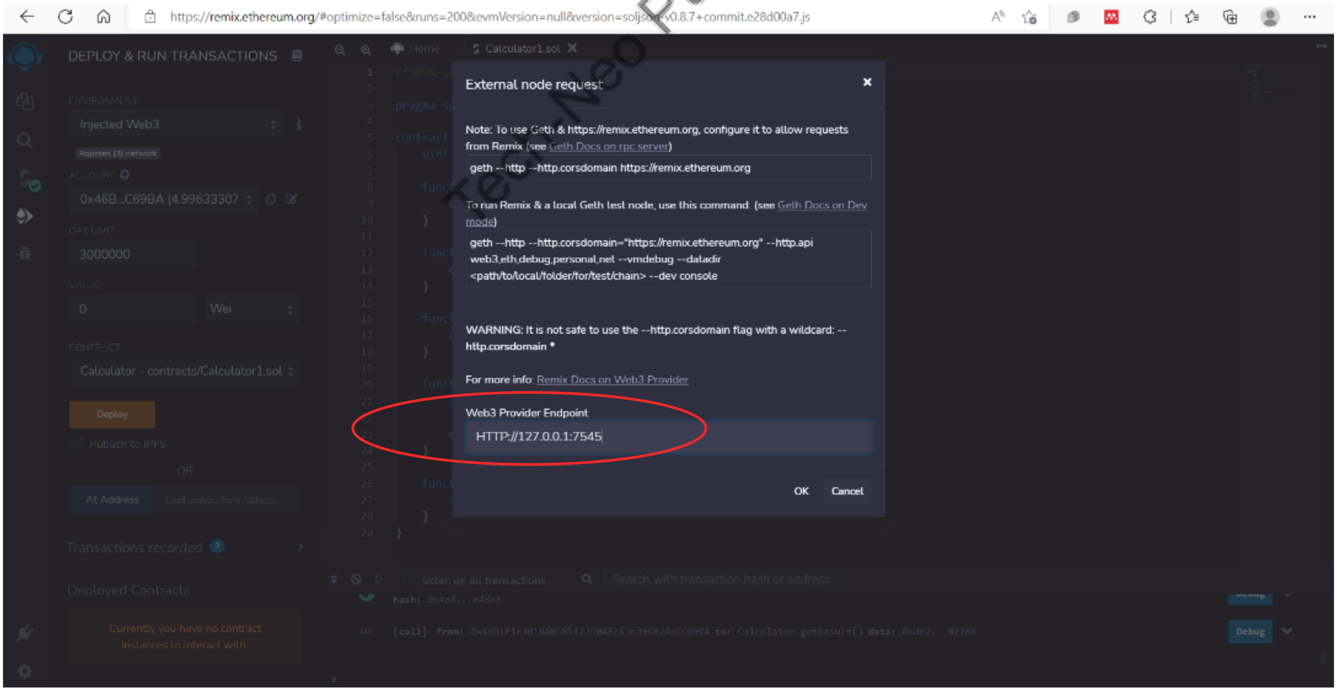
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
0	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	QUICKSTART

MNEMONIC
pioneer clock fun annual buffalo rack harvest enlist sell dinner moment toss

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×1Be3958D0896c6fE35Fb3af29302D945e056326C	100.00 ETH	0	0
0×2972e35e4332e3Da43789E51E8Fe8572405D222A	100.00 ETH	0	1
0×66A97A6853528256B92abfC1cEc8F3ccbe6CC810	100.00 ETH	0	2
0×070fFBC69cCD907590376BE86eD0A0498cA0a300	100.00 ETH	0	3
0×19eA13cFbbC17eCD4a45347467fa6c6e23844693	100.00 ETH	0	4
0×3AC98404447854771de9D5bc6C06Af1acb0f771e	100.00 ETH	0	5
0×9E287553FEfBf6C3B5917719D3cc86e1B7945Cad	100.00 ETH	0	6

- **Step 7 :** The RPC server from Ganache is specified in the Web3 Provider Endpoint, as shown below. Next, we click on OK.


https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
Home
Calculator1.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
Deployed Contracts
Currently you have no contract instances to interact with
listen on all transactions
Search with transaction hash or address
Debug

ENVIRONMENT
Injected Web3
Ropsten (3) network

ACCOUNT
0x46B...C69BA (4.99633307

GAS LIMIT
3000000

VALUE
0 Wei

CONTRACT
Calculator - contracts/Calculator1.sol

External node request

Note: To use Geth & https://remix.ethereum.org, configure it to allow requests from Remix (see Geth Docs on rpc server)

```
geth --http --http.corsdomain https://remix.ethereum.org
```

To run Remix & a local Geth test node, use this command: (see Geth Docs on Dev mode)

```
geth --http --http.corsdomain="https://remix.ethereum.org" --http.api
web3,eth,debug,personal,net --vmdebug --datadir
<path/to/local/folder/for/test/chain> --dev console
```

WARNING: It is not safe to use the --http.corsdomain flag with a wildcard: --http.corsdomain *

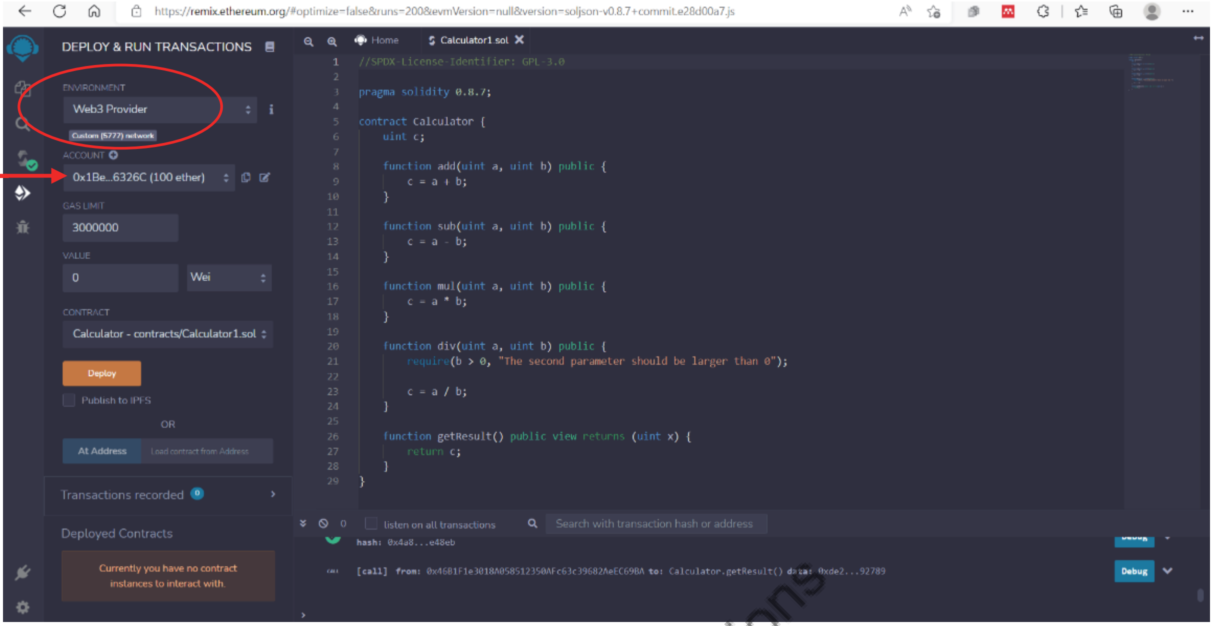
For more info: Remix Docs on Web3 Provider

Web3 Provider Endpoint
HTTP://127.0.0.1:7545

OK Cancel

```
[call] from: 0x46B1F1e3018A0585123S0AFc63c39G02AcCC69BA to: Calculator.getResult() data: 0xde2...97789
```

- **Step 8 :** Now, Remix IDE is successfully connected with Ganache. Next, we deploy the contract.



https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Web3 Provider

Custom (5777) network

ACCOUNT: 0x1Be...6326C (100 ether)

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Calculator - contracts/Calculator1.sol

Deploy

□ Publish to IPFS

OR

At Address | Load contract from Address

Transactions recorded 0

Deployed Contracts

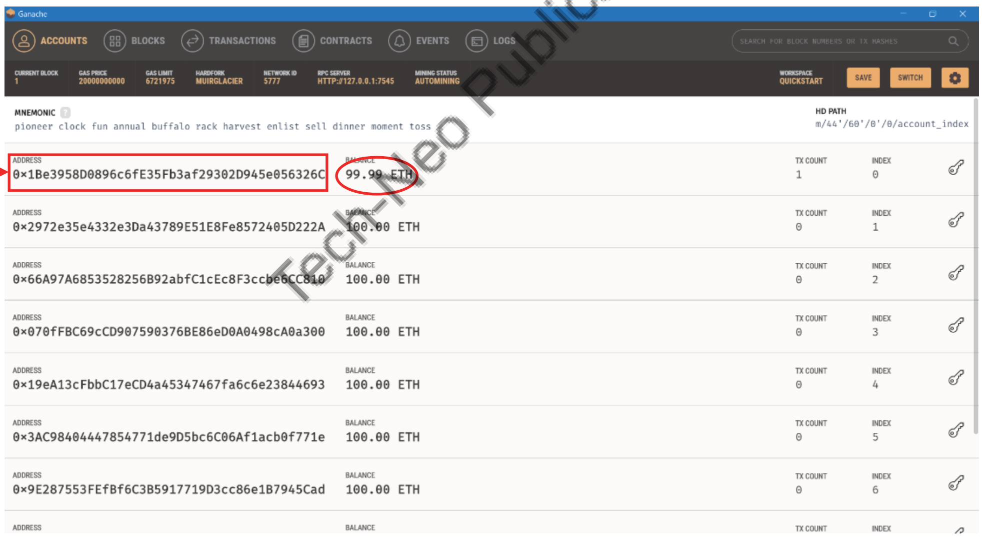
Currently you have no contract instances to interact with.

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Calculator {
    uint c;

    function add(uint a, uint b) public {
        c = a + b;
    }

    function sub(uint a, uint b) public {
        c = a - b;
    }

    function mul(uint a, uint b) public {
        c = a * b;
    }

    function div(uint a, uint b) public {
        require(b > 0, "The second parameter should be larger than 0");

        c = a / b;
    }

    function getResult() public view returns (uint x) {
        return c;
    }
}
```

□ listen on all transactions | Search with transaction hash or address

```
hash: 0x4a8...e48eb
[call]  from: 0x4681F1e3018A058512350AFc63c39682AeEC69BA to: Calculator.getResult() data: 0xde2...92789
```

- **Step 9 :** As it can be seen that after deployment, for a specific address, certain ethers get deducted, as shown below.



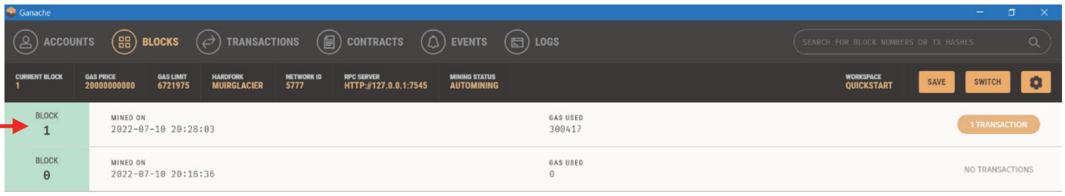
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
1	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	QUICKSTART

MNEMONIC
pioneer clock fun annual buffalo rack harvest enlist sell dinner moment toss

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0x1Be3958D0896c6fE35Fb3af29302D945e056326C	99.99 ETH	1	0
0x2972e35e4332e3Da43789E51E8Fe8572405D222A	100.00 ETH	0	1
0x66A97A6853528256B92abfC1cEc8F3ccbe6CC810	100.00 ETH	0	2
0x070fFBC69cCD907590376BE86eD0A0498cA0a300	100.00 ETH	0	3
0x19eA13cFbbC17eCD4a45347467fa6c6e23844693	100.00 ETH	0	4
0x3AC98404447854771de9D5bc6C06Af1acb0f771e	100.00 ETH	0	5
0x9E287553FEfBf6C3B5917719D3cc86e1B7945Cad	100.00 ETH	0	6

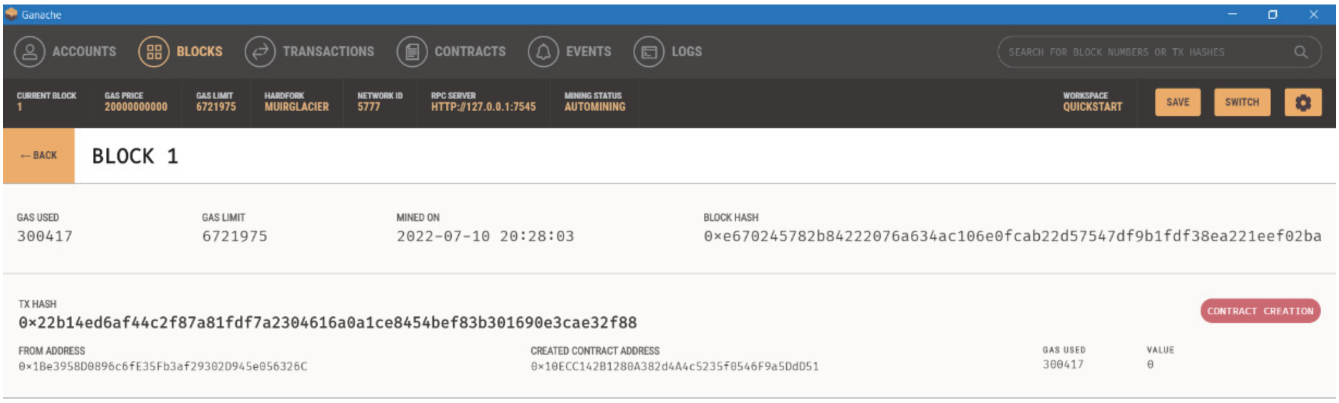
- **Step 10 :** Furthermore, on clicking on the address, we can see information such as block no., date and time of mining, and gas used.



CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
1	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	QUICKSTART

BLOCK	MINED ON	GAS USED	
1	2022-07-10 20:28:03	300417	1 TRANSACTION
0	2022-07-10 20:18:36	0	NO TRANSACTIONS

- **Step 11 :** Moreover, on clicking on BLOCK 1, additional information can be viewed such as gas used, gas limit, mined on, block hash, and transaction (Tx) hash.



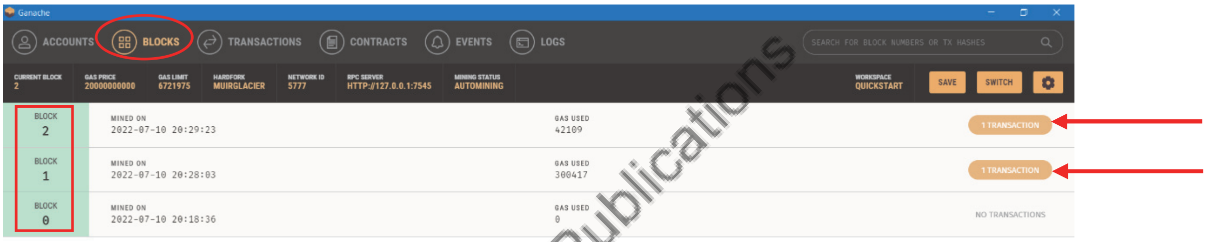
CURRENT BLOCK 1 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE QUICKSTART

BLOCK 1

GAS USED 300417
GAS LIMIT 6721975
MINED ON 2022-07-10 20:28:03
BLOCK HASH 0×e670245782b84222076a634ac106e0fcab22d57547df9b1fdf38ea221eef02ba

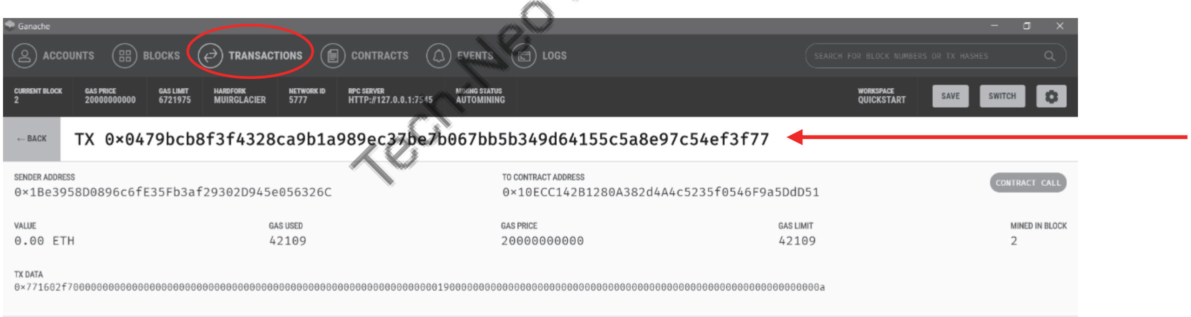
TX HASH 0×22b14ed6af44c2f87a81fdf7a2304616a0a1ce8454bef83b301690e3cae32f88
FROM ADDRESS 0×1Be3958D0896c6fE35Fb3af29302D945e056326C
CREATED CONTRACT ADDRESS 0×10ECC142B1280A382d4A4c5235f0546F9a5DdD51
GAS USED 300417
VALUE 0

- **Step 12 :** In a similar manner, we can add several blocks, as shown below.



CURRENT BLOCK 2 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE QUICKSTART

BLOCK 2	MINED ON 2022-07-10 20:29:23	GAS USED 42109	1 TRANSACTION
BLOCK 1	MINED ON 2022-07-10 20:28:03	GAS USED 300417	1 TRANSACTION
BLOCK 0	MINED ON 2022-07-10 20:18:36	GAS USED 0	NO TRANSACTIONS



CURRENT BLOCK 2 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE QUICKSTART

TX 0×0479bcb8f3f4328ca9b1a989ec37be7b067bb5b349d64155c5a8e97c54ef3f77

SENDER ADDRESS 0×1Be3958D0896c6fE35Fb3af29302D945e056326C
TO CONTRACT ADDRESS 0×10ECC142B1280A382d4A4c5235f0546F9a5DdD51

VALUE 0.00 ETH
GAS USED 42109
GAS PRICE 20000000000
GAS LIMIT 42109
MINED IN BLOCK 2

TX DATA
0×771602f70000000000000000000000000000000000000000000000000000000000000019000000000000000000000000000000000000000000000000000000000000000a

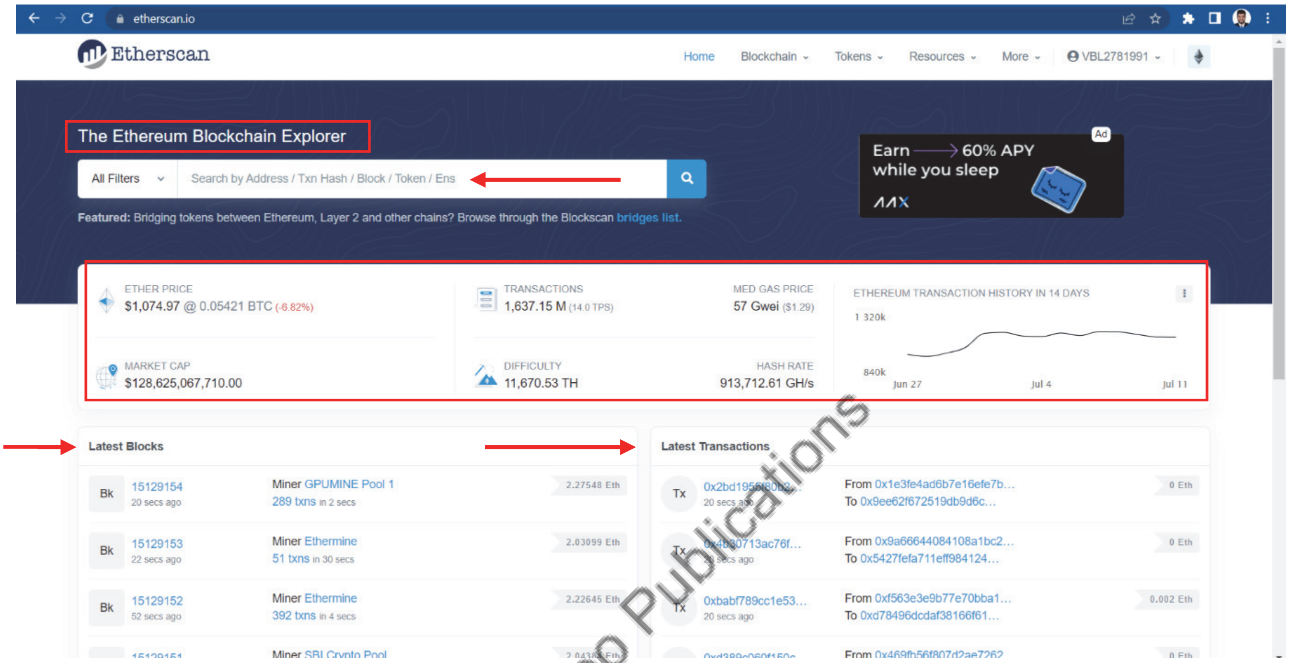
Exploring etherscan.io

- **Etherscan**
- Etherscan permitsan individual to view the assets held on any public Ethereum wallet address.
- On Etherscan by entering any Ethereum address into the search box,one can see the current balance and transaction history of the wallet under consideration.
- Etherscan displays any gas fees and smart contracts involving that address.
- Etherscan can be used to :
  1. Calculate Ethereum gas fees with the Etherscan gas tracker.
  2. Lookup and verify smart contracts.
  3. View crypto assets held in or associated with a public wallet address.
  4. Observe live transactions taking place on the Ethereum blockchain.
  5. Lookup a single transaction made from any Ethereum wallet.

6. Discover which smart contracts have a verified source code and security audit.
7. Keep track of how many smart contracts a user has authorized with their wallet.
8. Review and revoke access to a wallet for any decentralized applications (DApps).

Etherscan : Ethereum Blockchain Explorer(https://etherscan.io/)

- Contracts, transactions, and accounts



etherscan.io

Etherscan

The Ethereum Blockchain Explorer

All Filters | Search by Address / Txn Hash / Block / Token / Ens

Featured: Bridging tokens between Ethereum, Layer 2 and other chains? Browse through the Blockscan bridges list.

ETHER PRICE
$1,074.97 @ 0.05421 BTC (-6.82%)

TRANSACTIONS
1,637.15 M (14.0 TPS)

MED GAS PRICE
57 Gwei ($1.29)

MARKET CAP
$128,625,067,710.00

DIFFICULTY
11,670.53 TH

HASH RATE
913,712.61 GH/s

ETHEREUM TRANSACTION HISTORY IN 14 DAYS

Latest Blocks

Bk 15129154 20 secs ago — Miner GPUMINE Pool 1 — 289 txns in 2 secs — 2.27548 Eth

Bk 15129153 22 secs ago — Miner Ethermine — 51 txns in 30 secs — 2.03099 Eth

Bk 15129152 52 secs ago — Miner Ethermine — 392 txns in 4 secs — 2.22645 Eth

15129151 — Miner SBI Crypto Pool — 2.0438 Eth

Latest Transactions

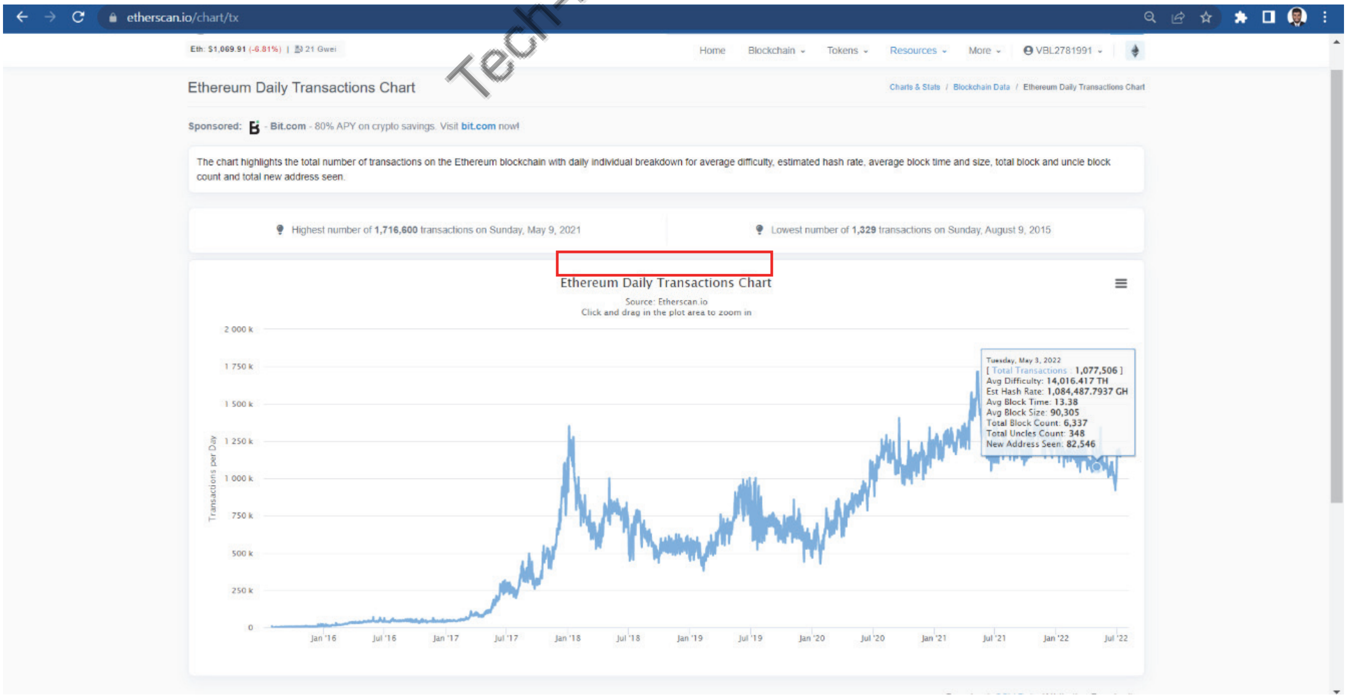
Tx 0x2bd1956f80b2… 20 secs ago — From 0x1e3fe4ad6b7e16efe7b… To 0x9ee62f672519db9d6c… — 0 Eth

Tx 0x4b30713ac76f… 20 secs ago — From 0x9a66644084108a1bc2… To 0x5427fefa711eff984124… — 0 Eth

Tx 0xbabf789cc1e53… 20 secs ago — From 0xf563e3e9b77e70bba1… To 0xd78496dcdaf38166f61… — 0.002 Eth

0xd389c060f150c — From 0x469fb56f807d2ae7262 — 0 Eth

Ethereum daily transactions chart



etherscan.io/chart/tx

Eth: $1,069.91 (-6.81%) | 21 Gwei

Ethereum Daily Transactions Chart

Charts & Stats / Blockchain Data / Ethereum Daily Transactions Chart

Sponsored: Bit.com - 80% APY on crypto savings. Visit bit.com now!

The chart highlights the total number of transactions on the Ethereum blockchain with daily individual breakdown for average difficulty, estimated hash rate, average block time and size, total block and uncle block count and total new address seen.

Highest number of **1,716,600** transactions on Sunday, May 9, 2021

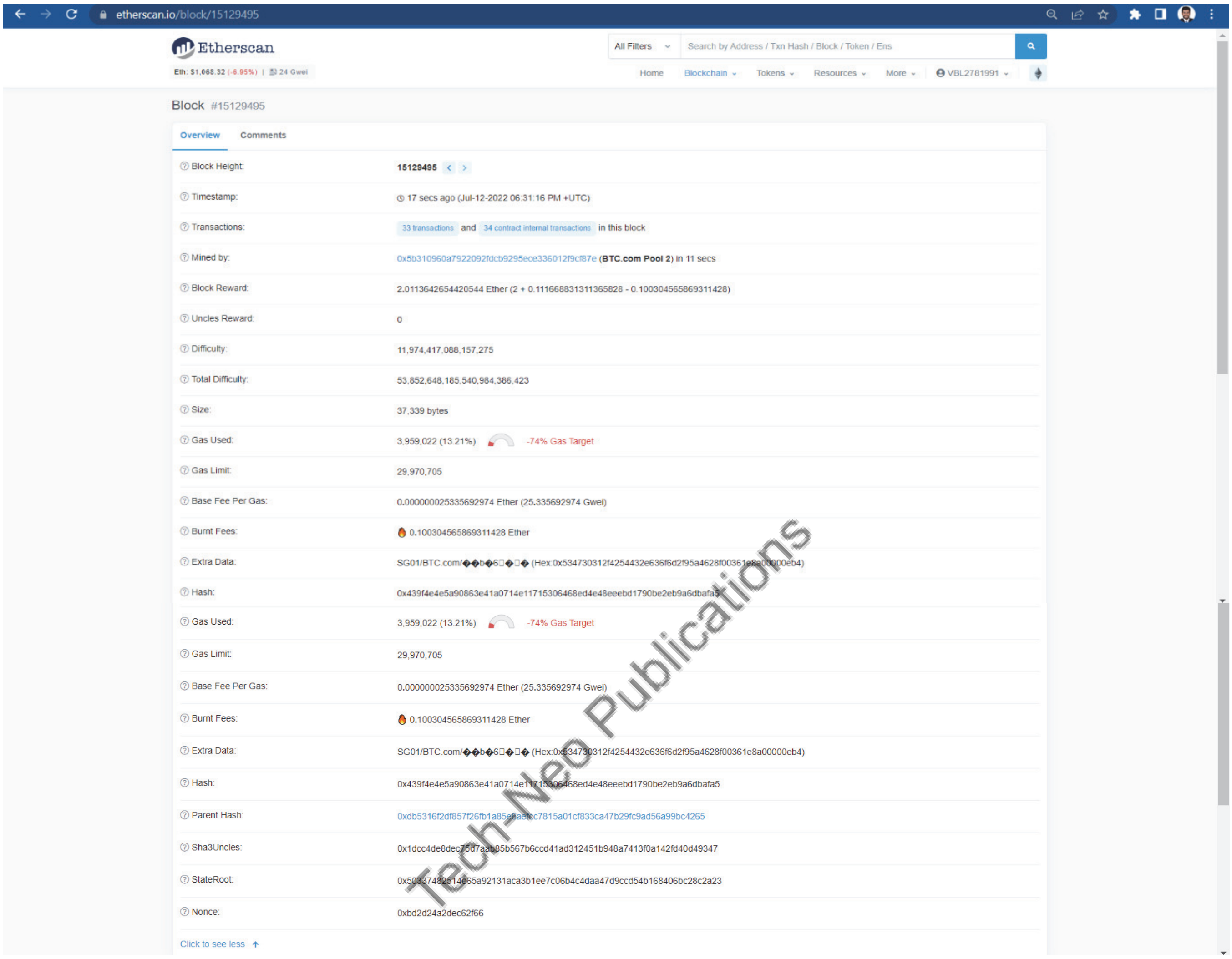
Lowest number of **1,329** transactions on Sunday, August 9, 2015

Ethereum Daily Transactions Chart

Source: Etherscan.io
Click and drag in the plot area to zoom in

Tuesday, May 3, 2022
[Total Transactions : 1,077,506]
Avg Difficulty: 14,016.417 TH
Est Hash Rate: 1,084,487.7937 GH
Avg Block Time: 13.38
Avg Block Size: 90,305
Total Block Count: 6,337
Total Uncles Count: 348
New Address Seen: 82,546

Example of an Ether block Structure


etherscan.io/block/15129495
Etherscan
Eth: $1,068.32 (-6.95%) | 24 Gwei
All Filters
Search by Address / Txn Hash / Block / Token / Ens
Home
Blockchain
Tokens
Resources
More
VBL2781991
Overview
Comments

Block #15129495

Block Height:	15129495
Timestamp:	17 secs ago (Jul-12-2022 06:31:16 PM +UTC)
Transactions:	33 transactions and 34 contract internal transactions in this block
Mined by:	0x5b310960a7922092fdcb9295ece336012f9cf87e (**BTC.com Pool 2**) in 11 secs
Block Reward:	2.0113642654420544 Ether (2 + 0.111668831311365828 - 0.100304565869311428)
Uncles Reward:	0
Difficulty:	11,974,417,088,157,275
Total Difficulty:	53,852,648,185,540,984,386,423
Size:	37,339 bytes
Gas Used:	3,959,022 (13.21%) -74% Gas Target
Gas Limit:	29,970,705
Base Fee Per Gas:	0.000000025335692974 Ether (25.335692974 Gwei)
Burnt Fees:	🔥 0.100304565869311428 Ether
Extra Data:	SG01/BTC.com/��b�6□�□� (Hex:0x534730312f4254432e636f6d2f95a4628f00361e8a00000eb4)
Hash:	0x439f4e4e5a90863e41a0714e11715306468ed4e48eeebd1790be2eb9a6dbafa5
Gas Used:	3,959,022 (13.21%) -74% Gas Target
Gas Limit:	29,970,705
Base Fee Per Gas:	0.000000025335692974 Ether (25.335692974 Gwei)
Burnt Fees:	🔥 0.100304565869311428 Ether
Extra Data:	SG01/BTC.com/��b�6□�□� (Hex:0x534730312f4254432e636f6d2f95a4628f00361e8a00000eb4)
Hash:	0x439f4e4e5a90863e41a0714e11715306468ed4e48eeebd1790be2eb9a6dbafa5
Parent Hash:	0xdb5316f2df857f26fb1a85e8aefcc7815a01cf833ca47b29fc9ad56a99bc4265
Sha3Uncles:	0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347
StateRoot:	0x50337482514e65a92131aca3b1ee7c06b4c4daa47d9ccd54b168406bc28c2a23
Nonce:	0xbd2d24a2dec62f66

Click to see less

☞ Comparison of Bitcoin and Ethereum

GQ. Compare Bitcoin and Ethereum.

Sr. No.	Parameters	Bitcoin	Ethereum
1.	Founder	Satoshi Nakamoto	Vitalik Buterin and others
2.	Purpose	Cryptocurrency	Network Software
3.	Release date	January 2009	July 2015
4.	Average block time	~10 minutes	~15 seconds
5.	Coin symbol	BTC	ETH
6.	Tokens	Not available	Available
7.	Monetary policy	Hardcoded	Non-hardcoded
8.	Emission rate	Halving policy followed	Occasional
9.	Backward compatibility	Available	Not available
10.	Block limitation	1 MB per block	No limit

3.3 SMART CONTRACTS

GQ. What is a smart contract ? Explain its characteristics.

3.3.1 Introduction

- The term was coined by Nick Szabo (a computer scientist and cryptographer) in 1996. Szabo claimed that smart contacts (i.e., whenever we are establishing smart contracts between multiple parties) can be realized with the help of a public ledger. Thus, blockchain can also be a pioneering technology to realize smart contracts.
- A smart contract basically provides a decentralized platform which can be utilized to avoid the intermediaries (i.e., middleman) in a contract. When two persons are coming to a contract, there is a legal advisor who is controling such type of a contract. So, with the help of such type of blockchain environment, intermediaries/middleman can be avoided.

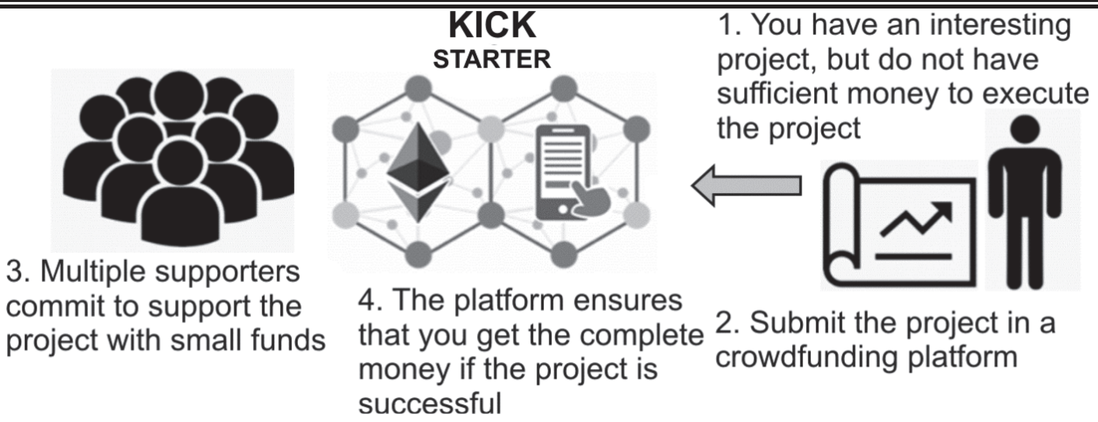
3.3.2 Smart Contract

❒ **Definition :** A smart contract is an automated computerized protocol used for digitally facilitating, verifying, or enforcing the negotiation or performance of a legal contract by avoiding intermediates and directly validating the contract over a decentralized platform.

☞ **Contracts in a centralized platform by means of crowdfunding**

GQ. How crowdfunding platforms can be managed using smart contracts ?

- Kickstarter is a crowd funding company the uses the following principle. For instance, you (or a group of people) want to execute some kind of an interesting project but there is no sufficient amount of money or fund.
- An individual or a group of people what they do is they submit the project to a crowd funding company like Kickstarter to acquire funds. Now, there are multiple supporters who can support with some small funds for a particular project.
- The project that is submitted by an individual or a group of people could be supported by an individual (i.e., offering the entire fund) or a group of supporters by offering small funds.
- The task of Kickstarter platform is to ensure that when certain milestone of a project is achieved, an individual or a group of people is eligible to receive the fund.
- Kickstarter ensures that whenever a supporter is providing the fund, the fund is received to an intended project as and when the project completes certain milestone (i.e., project executers are getting the fund).
- However, if the project is not completed successfully or in between the project gets scrapped, then the fund is sent back to the supporter.

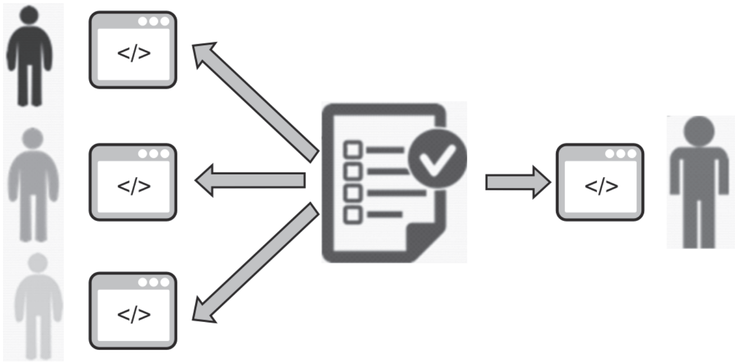

KICK
STARTER
1. You have an interesting project, but do not have sufficient money to execute the project
2. Submit the project in a crowdfunding platform
3. Multiple supporters commit to support the project with small funds
4. The platform ensures that you get the complete money if the project is successful

(1B20)**Fig. 3.3.1 : Crowdfunding**

- In this type of an architecture, the relationship of trust is essential on the crowdfunding platform between the product team and the supporters. The product team expects the money to be get paid based on the project progress (i.e., whenever a particular milestone for a project is reached Kickstarter should provide the fund).
- The supporters expect the money to go to the right project and if the project is scrapped in between, they will get back their money. Here, the crowdfunding platform (i.e., Kickstarter) acts as a middleman that takes a huge amount of money from the product team as well as the supporters to manage the entire process*(as there is a huge amount of risk factor that is involved like if the project is left half completed or the supporter would not support the project further)*, which is indeed a problem in a centralized platform.

☞ How crowdfunding platforms can be managed using smart contracts ?

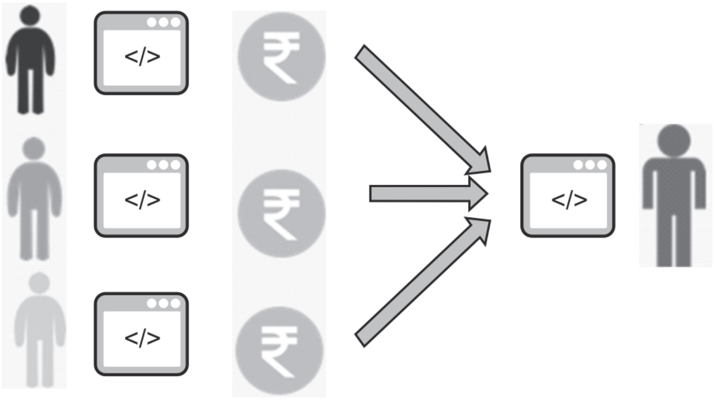
- There are a list of supporters on one side and on the other side there is a product team.
- The contract between the supporters and the product team is written in a code, which is available to all the stakeholders (i.e., to the supporters as well as the product team).



(1B21)**Fig. 3.3.2**

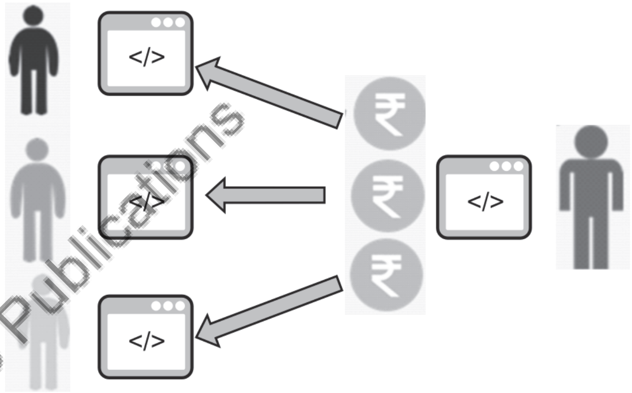
NOTES

- Now, everyone can verify the contract. Also, if we put the contact inside a blockchain, then everyone will be able to verify that contract but no one will be able to tamper it.
- So, this gives an interesting idea that such type of a smart contract platform can be realized using a blockchain.



(1B22) Fig. 3.3.3

- If certain goals of the project are reached, then the code automatically transfers the money from supporters to the production team.
- If the project goals (contracts) fail, then the code can transfer the money back to the supporters.



(1B23) Fig. 3.3.4

- In this example, instead of placing a transaction or data inside the blockchain, we are placing the code which will be automatically verified by every stakeholders. They will not be able to tamper the code. They will not be able to deny the code in between, but as and when the code runs by verifying the events or actions that have been executed, the contract can get executed over time and fulfill the initial agreements that have been made.

☞ Characteristics of a Smart Contract

The characteristics of a smart contract are as follows :

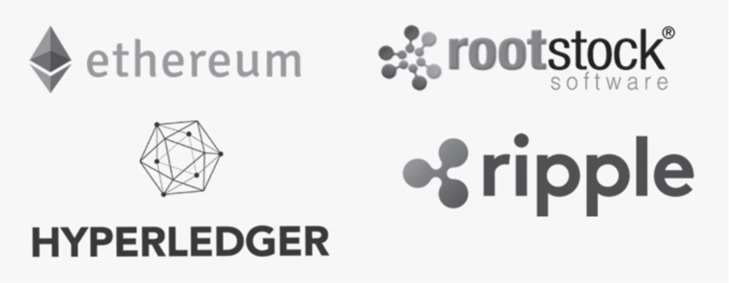
- **Immutable :** No party will be able to change the contract once it is fixed and written to the public ledger (i.e., the Blockchain).
- **Distributed :** There is no need of a middleman like Kickstarter who handles all the risk. The code will get automatically executed. If the promise is not fulfilled, then automatically the code will execute some steps based on the contracts. In other words,all the steps of the contract can be validated by every participating party, i.e., no one can claim later that the contract was not validated.

☞ Blockchain is a suitable platform for executing a smart contract because of the following reasons

- Based on the blockchain architecture, blocks are immutable.
- Information is open (i.e., everyone can check and validate the information inside a blockchain).

☞ Various types of smart contract platforms

GQ. Elaborate on different types of smart contracts.



(1B24)**Fig. 3.3.5 : Types of smart contract platforms**

☞ Types of Smart contracts

Smart contracts can be categorized into four types based on the applications :

- **Smart legal contracts :** These contracts possess several legal contract templates. They simply execute the contracts as per the templates used.
- **Decentralized applications :** They are also called as DApps. They run point-to-point network of computers instead of a single computer. It is not essential that DApps should run on top of a blockchain network.DApps are blockchain-enabled websites. On the other hand, a smart contract is what permits a DApp to connect it to a blockchain. DApp covers from end to end (i.e., both front end and back end). If a decentralized application on a smart contract system needs to be created, then several smart contracts must be combined and third-party systems have to be relied upon for the front-end.
- **Distributed autonomous organization (DAO) :** It is an organization exemplified by logic encoded as a transparent program controlled by shareholders and not influenced by a central authority. The financial transaction records and program logics of DAO are maintained on a blockchain. DAOs make use of blockchain technology to maintain a secure advanced record in the form of a digital ledger (DL) to trace financial transactions over the Internet. DAOs make use of distributed databases and is tamperproof as it involves timestamps, which excludes the presence of a trusted third-party in a financial transaction. DAO is an entity that is independently present on the Internet. However, it requires human intervention to perform certain tasks.

☞ Characteristics of independent DAOs

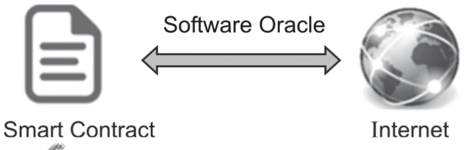
- **Tokens of transactions :** DAOs require tokens that are used for rewarding certain activities. Funding occurs directly by a specific organization that creates the tokens.
- **Independent :** The code, once deployed, is not under the control of its creators and cannot be influenced by outsiders. Thus, DAOs are open source, which offer a transparent and indestructible system.
- **Consensus :** Most of the stakeholders must agree on the same decision for not only withdrawing but also moving funds from DAOs.
- **Contractors :** The task of building a product, writing a code, or developing a hardware do not lie in the hands of DAOs. They need to hire a contractor, and so, contractors get appointed through voting.
- **Proposals :** In DAOs, the principle way of taking decisions is through proposals. To avoid overloading of the network with proposals, DAOs require monetary deposit, which discourages people from spamming the network.
- **Voting :** Voting takes place only after submitting the proposal. DAOs permit people to exchange economic values with others through various activities like investing, money raising, lending, and borrowing without any intermediaries.

- **Smart contracting devices :** Any failure in IoT on the Internet will result in the leak of personal data. A security breach occurs primarily in authentication, connection, and transaction. With the help of blockchain that manages and access data from IoT devices, a hacker has to bypass an additional layer of security in addition to some robust encryption standards that are available. Moreover, a single point of failure cannot take place because of the decentralized nature.
- In the context of smart contracts and blockchain, an oracle can be defined as an agent that verifies real-world occurrences and submits details to a blockchain so that smart contracts can use it. For example, fund managers can release funds only when certain conditions are satisfied/met. Moreover, an oracle has to sign on a smart contract for releasing funds.

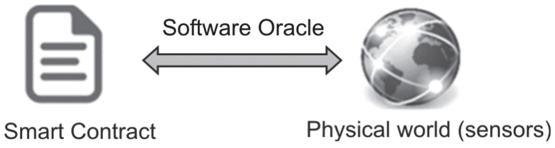
☞ Types of Oracles

GQ. What is an oracle? State and explain different types of oracles.

1. **Software Oracles :** They manage information that is commonly available on the Internet. For instance, flight details, commodity prices, etc. Such data is usually acquired from online sources (i.e., airways sites and e-commerce sites). Software oracles fetches relevant information and sends it to a smart contract.
2. **Hardware Oracles :** Certain smart contracts require input from the physical word (i.e., sensors), which is offered by hardware oracles. For instance, hardware oracles can be used to track a car's details, such as date, time, speed, and direction, when it is crossing a specific junction through sensors. Another example could be the use of RFID sensors for keeping a track of truck movements that carry goods.
3. **Inbound and Outbound Oracles :** Inbound oracles offer a smart contact with data from external world (i.e., the use of an automatic purchase order for an item in an inventory management system when that item's stock value hits a certain threshold value), whereas outbound oracles allows a smart contract to send data to the outside world.
4. **Consensus-based Oracles :** The prediction market relies heavily on oracle for predicting future outcomes. Depending on a single source of information is not advisable and risky too. In order to avoid market manipulation, prediction markets implement a rating system for oracles. An amalgamation of different oracles is used for improving security.

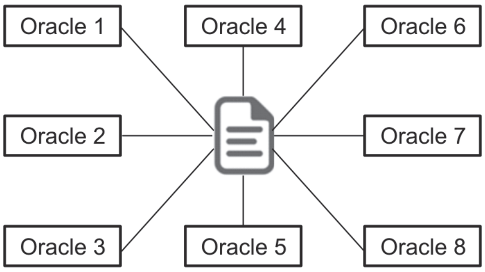


(1B25)Fig. 3.3.6 : Software Oracle



(1B26)Fig. 3.3.7 : Hardware Oracle

(1B27)Fig. 3.3.8 : Inbound and Outbound Oracles



(1B28)Fig. 3.3.9 : Consensus-based Oracles

3.4 SMART CONTRACTS IN ETHEREUM

GQ. Write a short note on : Smart contracts in Ethereum.

- A contract is a legal record that unites two parties to perform transactions either immediately or at a later instance. Example of this would be an individual getting to a contract with an insurance company to secure his/her wellness, a person investing in a property that is belonging to someone else, and an organization attempting to sell its shares to some other organizations.
- On the other hand, a smart contract is the one that is executed, deployed, and implemented in an Ethereum environment. Smart contracts result in the digitization of legal contracts. Smart contracts store data. The data that is stored in smart contracts could be used to describe facts, associations, accounts, or other information to execute logic to obtain real-world contracts.
- A smart contract can be thought of as a small program that consists of functions/methods. With the help of a smart contract, an instance can be created and functions/methods can be invoked for viewing and updating contract data in conjunction with logic implementation.
- Smart contracts enable blockchain to be useful in finance and trade, supply chain, banking, etc. Assume that there is crowd funding (*means to raise money/funds from individuals or a group of individuals*) taking place to build/develop a huge project. The project can be raised with the following features :
  - **Project :** Crowdfunding for a huge project
  - **Amount :** 10,000 US dollars
  - **Duration :** 60 days
- **Decision :** If the specified amount is raised within the given duration, then the funds/money would be given for developing the huge project. Nonetheless, if the specified amount is not raised in the given duration, then the funds/money would go back to the respective donor.

- **Step 1 :** Crowdfunding project launched via Ethereum.
- **Step 2 :** As this is public funding, the platform is open to all *(i.e., anybody can join the network, mine, or buy ethers).*
- **Step 3 :** Stop further funding for the project.
- **Step 4 :** If the crowdfunding project meets the specified amount, then transfer the fund/money to the organization. If not, then revert the funds to the respective donors and ensure that the right amount is transferred to the right owner.

3.5 SMART CONTRACTS IN INDUSTRY

GQ. State and explain some use cases of smart contracts in industry.

Smart contracts in blockchain are a new technology that appears to be influencing industries such as finance, healthcare, supply chain and logistics, retail, government, and so on. Some of the use cases are detailed shortly below.

☞ Healthcare industry

- There is a massive opportunity for blockchain revolution to lead digital transformation in the healthcare industry. There are numerous applications for this technology, ranging from medical records to pharmaceutical supply chains to smart contracts for payment distribution.
- Here are some examples of how blockchain and smart contracts can impact the way healthcare services are delivered. Every healthcare system is reliant on medical records. Medical records, on the other hand, tend to become more detailed

and involved with each doctor’s or medical consultant’s visit. It is difficult for healthcare providers to obtain access to these records because each hospital, clinic, or doctor’s office stores them differently.

- Individual businesses, such as medical chain and medicos, have devised solutions to this issue. The goal is to provide patients control over their entire medical history and to provide a single point of contact for patients and physicians, bringing security to health record access. Authorized parties can access, receive, and modify their records using the Smart Contract functionality. These documentation changes are verified and authenticated by the blockchain, making them more secure than traditional medical records.
- Prescriptions can be shared and managed with a proper timestamp that cannot be tampered with. This increases the safety of medical records. Physical prescriptions are not required to be carried because they can be accessed by physicians, patients, hospital or clinic staff, and others with appropriate access. This solution, when combined with big data and artificial intelligence, has the potential to predict and manage epidemics, lead to more informed research, and characterize our efforts toward better health.

☞ Manufacturing and supply chain

- From its inception, the supply chain industry has maintained the highest levels of product safety, security, and stability. Because pharmaceutical manufacturing and distribution partners are spread across multiple countries and regions, it is difficult to track and trace the information of each drug or medical device. Furthermore, different countries have different drug laws. Because real-time tracking is currently unavailable, records can be easily altered.
- Furthermore, non-legitimate vendors contribute to the drug black market by fabricating records. Blockchain smart contracts can be used to secure and transparently monitor a supply chain. This can be used to investigate time delays and human errors, which are both extremely costly to the pharmaceutical industry.
- It can be used to track costs, labour, and even waste throughout the supply chain. It can be used to verify the authenticity of a product and track it back to its source in order to combat the counterfeit drug market, which costs the industry approximately $200 billion USD per year.
- Blockpharma and Modus, for example, are working to improve logistics efficiency. Modum, in particular, works in accordance with EU laws, which require proof that the medicine has not been subjected to conditions such as unusually high or low temperatures, which could compromise its quality.
- Sensors that monitor the climatic conditions of the products while they are in transit and physically record them on the blockchain, which will be programmed with smart contracts, will be included in these solutions.

☞ Banking and financial service industry

- **Syndicated loans :** Banks and financial institutions are increasingly using smart contracts to manage standard loans. Consequently, syndicated loans, in which multiple lenders provide loans to multiple borrowers on the same loan terms, stand to benefit significantly from the use of smart contracts. All steps, including syndication, diligence, underwriting, and servicing of syndicated loans, can be completed more quickly using smart contracts. With multiple entities involved in syndicated loans, smart contracts’ on-chain/off-chain information makes it much easier to build relationships, identities, and maintain security. Smart contracts thus act to drive effectiveness and efficiency in this scenario. In this scenario, smart contracts act to drive effectiveness and efficiency.

- **Securities :** Private companies can benefit from smart contracts to simplify capitalization table management. Currently, the security's issuer must transfer title to a number of guardians, each of whom will have sub-custodians until it reaches the investor. Various other challenges faced by the securities segment are listed below :

  1. Securities are paper-based
  2. Manual company registration procedures
  3. Increased cost
  4. Increased counterparty risk
  5. Increased latency.

  The following are the advantages of using smart contracts logic :

  1. Security is delivered to an investor directly from an issuer.
  2. As a result of securities on a distributed ledger, entire workflows have been digitized.
  3. Private security markets provide benefits more quickly than public securities markets.
  4. The cryptographic signature of a ledger entry replaces the state's seal on paper stock certificates.
  5. While issuers will need to know who owns their securities, some buy-side firms will keep this information confidential.

☞ Trade and finance

- Smart contracts facilitate the smooth transfer of goods across borders by enabling smart credit and trade payment initiation, as well as increased liquidity of finance-related assets. Smart contracts provide a risk-mitigated payment method. It also increases the efficiency of all parties involved in the process, including buyers, suppliers, and financial institutions. The current challenges we face are :

  o Obtaining a letter of credit is a time-consuming and expensive process.

  o Because of paperwork and physical document handling, there is various coordination among multiple parties.

  o These manual de-linked processes may result in fraud and duplication of funding.

  Using smart contracts logic includes the following benefits :

  o Automated compliance and tracking of letter of credit conditions allows for quick payment and approval commencement.

  o Enhanced efficiency in creating, changing, and supporting trade, title, and transportation-related contract arrangements.

  o Financial asset liquidity has increased as a result of the simplified transport process and a reduction in fraud.

  Some of the considerations while implementing smart contracts in trade and finance are :

  o Standard smart contract templates for each industry should be implemented, and guidelines should be put in place to ensure wider acceptance and adoption.

  o Legal implications for potential smart contract execution fall-out must be decided and addressed, particularly for defaults and dispute resolution.

  o The legal implications of potential smart contract execution fallout must be decided and addressed, particularly in terms of defaults and dispute resolution.

☞ Derivatives

- Because most over-the-counter (OTC) derivatives have asset servicing handled independently by each counter party, post-trading has many redundant and time-consuming processes.
- The vast majority of transaction contracts are paper-based and include terms, trade agreements, and/or post-trade confirmations. Smart contracts can improve post-trade processes by eliminating duplicate processes performed by each

counter party for record keeping and verifying trades, as well as executing appropriate trade levels and other lifecycle events.

- Implementing standard rules and regulations in smart contracts optimizes OTC derivatives post-trade processing. Some of the considerations while building smart contracts for derivatives are:
  - To manage large-scale protocol changes to current contracts as a result of regulatory reform, contract changes, or other unforeseen events, proper governance must be established.
  - For OTC derivatives, proper agreement on lifecycle events is required (e.g., an external source of data).
  - Oracles required to feed smart contracts with information to/from the blockchain network should be integrated and governed.

☞ Smart contracts in other industries

- There are many other use cases of smart contracts in other industries, as given below:
  - **Legal :** Smart contracts stored in a blockchain track contract parties, terms, transfer of ownership, and delivery of goods or services in the absence of the disadvantageous need for a legal intervention.
  - **Government :** Smart contracts on a blockchain provides promise as a system to maintain personal identifying information, criminal histories and "e-citizenship" authorized by biometrics.
  - **Food :** The blockchain concept may be utilized to trace a product's origin, batch, processing, expiration, storage conditions, and shipment in the food supply chain business.
  - **Insurance :** When autonomous vehicles such as driverless automobiles and smart devices communicate status updates with insurance carriers via the blockchain, overall costs are reduced because there is no need for auditing and verifying data.
  - **Education :** Universities and educational institutions can use blockchain to record grade or credentials data related to tests, degrees, and transcripts and to provide such data to the government or firms looking to hire students.

☞ Travel and hospitality

Hotel guests and airline passengers can use blockchain to maintain their authenticated "single travel ID." This will aid them in the processing of travel documents, identity cards, loyalty programmes, personal preferences, and payment information.

3.6 INTRODUCTION TO PROGRAMMING : SOLIDITY PROGRAMMING

- Smart contacts can be written in several programming languages like Serpent, Vyper, Lisp like language (LLL), and Solidity. Solidity a Turning complete and general purpose language—allows for programming smart contracts in a high-level, object-oriented language. The introduction of solidity based on Ethereum Virtual Machine (EVM) makes it easy to deploy with blockchain.
- Solidity programming is similar to JavaScript. It is a case sensitive and object-oriented programming (OOP) language. Even though it is anOOP language, it supports limited features (i.e., data types of variables should be defined and known at compile time). Also, function name and variable name should be written in the same manner as they are defined for successful execution.
- Solidity programs are written and savedwith a *.sol*extension, and statements are terminated in solidity programming with a semicolon (;). Any text editor can be used to open and read solidity programming files.

- A solidity programming file broadly consists of the following structure :

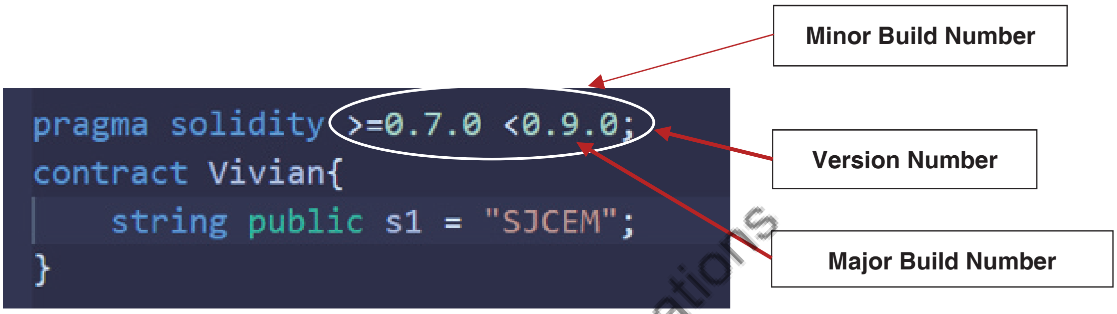
1. Pragma

It is the 1st line of code that is generally written within any solidity file. *Pragma* usually specifies which version of the compiler has to be used for a solidity file. The syntax for *pragma* is

pragma solidity <<version number>>;

Using *pragma*, one can select a desired compiler version and get your target code executed accordingly.

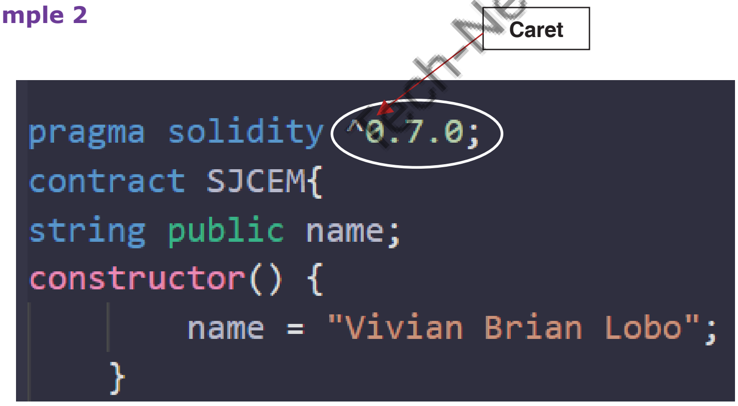
☞ **Example 1**



```
pragma solidity >=0.7.0 <0.9.0;
contract Vivian{
    string public s1 = "SJCEM";
}
```

The version number comprises a major and minor build number wherein the major build number here is 9 and the minor build number here is 0. Here, >=0.7.0 <0.9.0 indicates that one can use any solidity compiler having version number that is greater than equal to 0.7.0 and less than 0.9.0.

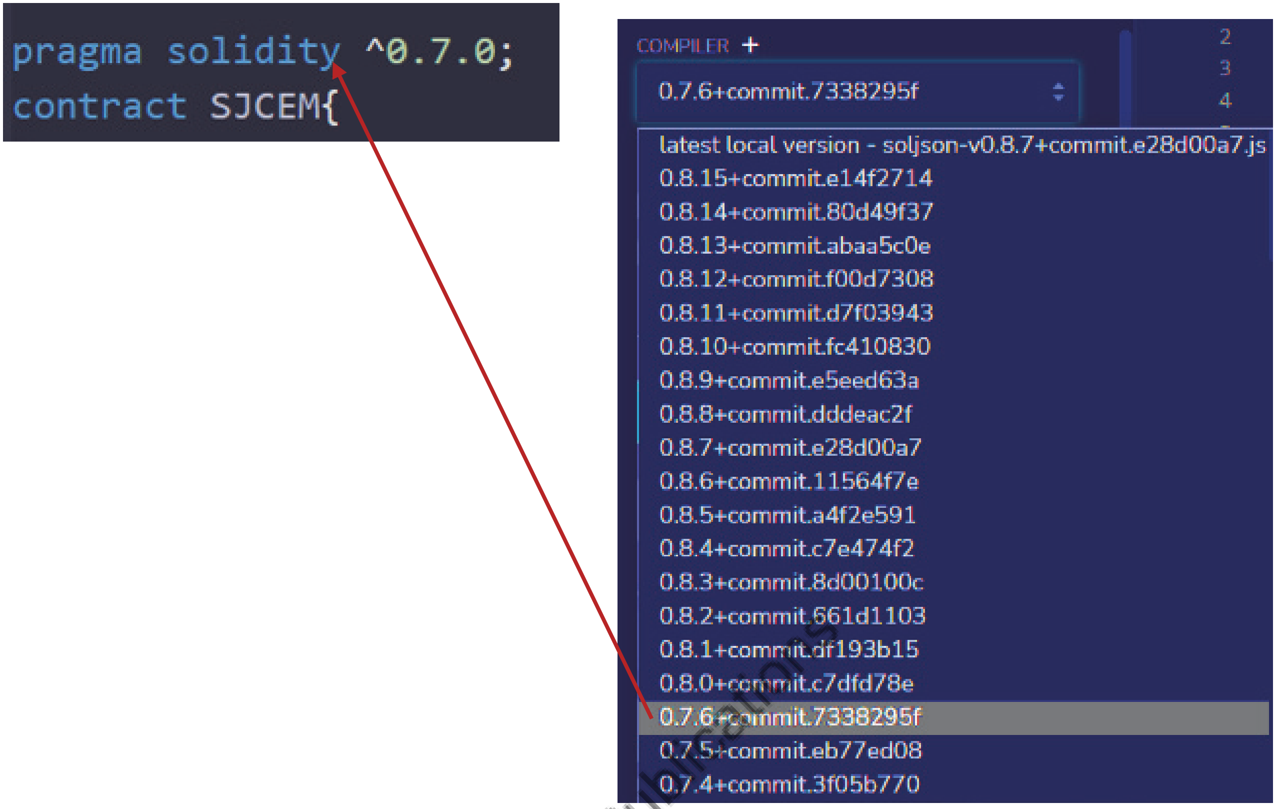
☞ **Example 2**



```
pragma solidity ^0.7.0;
contract SJCEM{
string public name;
constructor() {
        name = "Vivian Brian Lobo";
    }
```

The ^ character is called a *caret*, which is optional but plays a significant role in deciding the version number of a solidity compiler.

Note : Caret refers to the latest version within a specified major build number (i.e., ^0.7.0 refers to the major build number with 0.7.6 being the latest version).



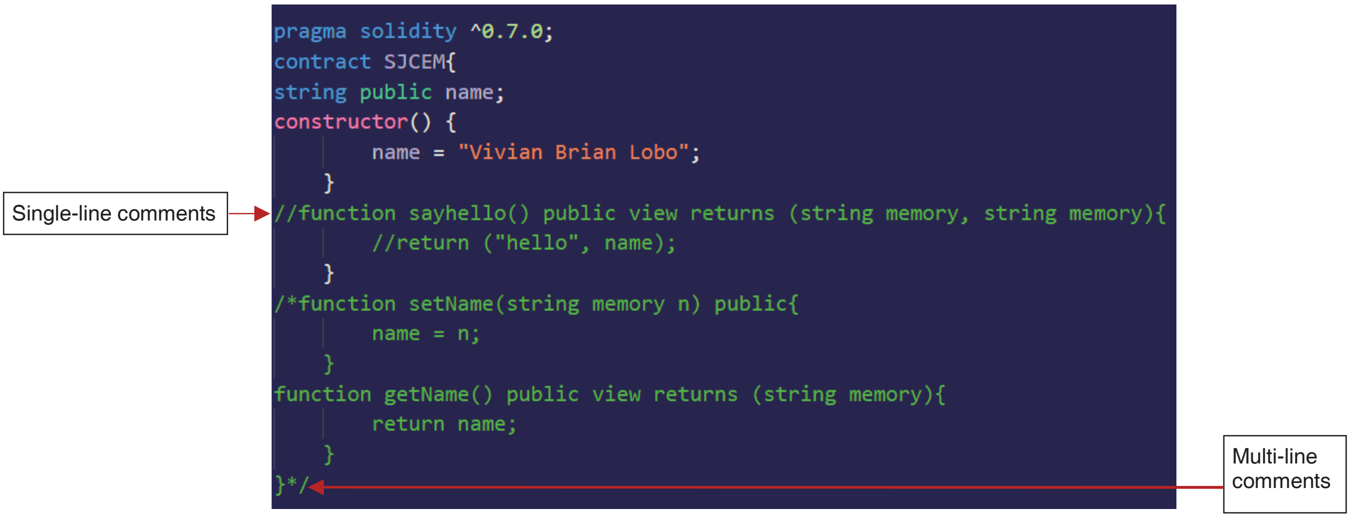
```
pragma solidity ^0.7.0;
contract SJCEM{
```

```
0.7.6+commit.7338295f
latest local version - soljson-v0.8.7+commit.e28d00a7.js
0.8.15+commit.e14f2714
0.8.14+commit.80d49f37
0.8.13+commit.abaa5c0e
0.8.12+commit.f00d7308
0.8.11+commit.d7f03943
0.8.10+commit.fc410830
0.8.9+commit.e5eed63a
0.8.8+commit.dddeac2f
0.8.7+commit.e28d00a7
0.8.6+commit.11564f7e
0.8.5+commit.a4f2e591
0.8.4+commit.c7e474f2
0.8.3+commit.8d00100c
0.8.2+commit.661d1103
0.8.1+commit.df193b15
0.8.0+commit.c7dfd78e
0.7.6+commit.7338295f
0.7.5+commit.eb77ed08
0.7.4+commit.3f05b770
```

- In a similar manner, for ^0.8.0, the latest version would be 0.8.15.
- The caret will not target any other major build apartfrom the one that is provided.
- The solidity file will compile only with a compiler with 7 asthe major build. It will not compile with any other majorbuild.

☞ 2. Comments

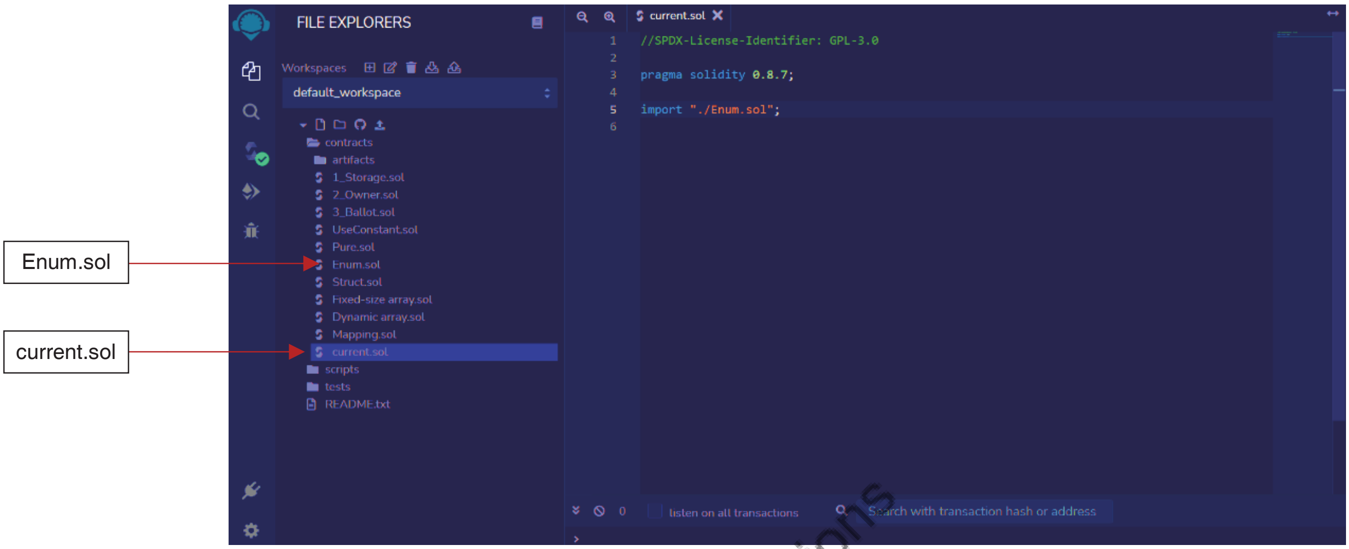
Like any other programming language, solidity offers the provision to add single-line (//) and multi-line comments (/*…*/).



```
pragma solidity ^0.7.0;
contract SJCEM{
string public name;
constructor() {
        name = "Vivian Brian Lobo";
    }
//function sayhello() public view returns (string memory, string memory){
        //return ("hello", name);
    }
/*function setName(string memory n) public{
        name = n;
    }
function getName() public view returns (string memory){
        return name;
    }
}*/
```

☞ 3. Import

With the help of *import* keyword, other solidity files can be imported. This keyword helps in accessing other solidity file's code within the current solidity file, which helps in writing modular solidity code.



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

import "./Enum.sol";
```

For instance, *current.sol* is the solidity file wherein *Enum.sol* file needs to be imported, which can be done by using *import* keyword.

Note : While using **http://remix.ethereum.org/** and writing code in solidity, it is essential that the license identifier (i.e., **//SPDX-License-Identifier: GPL-3.0**) should be written before specifying the version number of the solidity compiler.

☞ Basics

Structure of a smart contract

- The primary goal of solidity programming is to write smart contracts for Ethereum. Smart contracts are the basic unit of deployment and execution for Ethereum Virtual Machines (EVMs). Smart contracts mainly consists of two important aspects, i.e., *variables and functions*.
- A smart contract consists of state variables, structure definitions, modifier definitions, event declarations, enumeration definitions, and function definitions

State variable

- A variable that is declared in a contract that is not inside any function is called a state variable.
- It stores the current address of the contract. A state variable's allotted memory is statically assigned, meaning that it cannot change while the contract is in effect.
- State variables can possess qualifiers such as *internal*, *private*, *public*, and *constant*.
  - ○ ***Internal :*** If nothing is specified to a state variable, then by default, it is termed to be internal. This means that a variable can be accessed inside the current contract. Accessibility from outside the contract is not permitted. However, such variables can be viewed from outside.
  - ○ ***private :*** It is similar to *internal* but stringent. Private state variables can only be used in contracts declaring them. Moreover, they cannot be used even within derived contracts.
  - ○ ***public :*** With the help of *public*, state variables can be accessed directly.
  - ○ ***constant :*** Using *constant* makes state variables unalterable.

☞ Functions

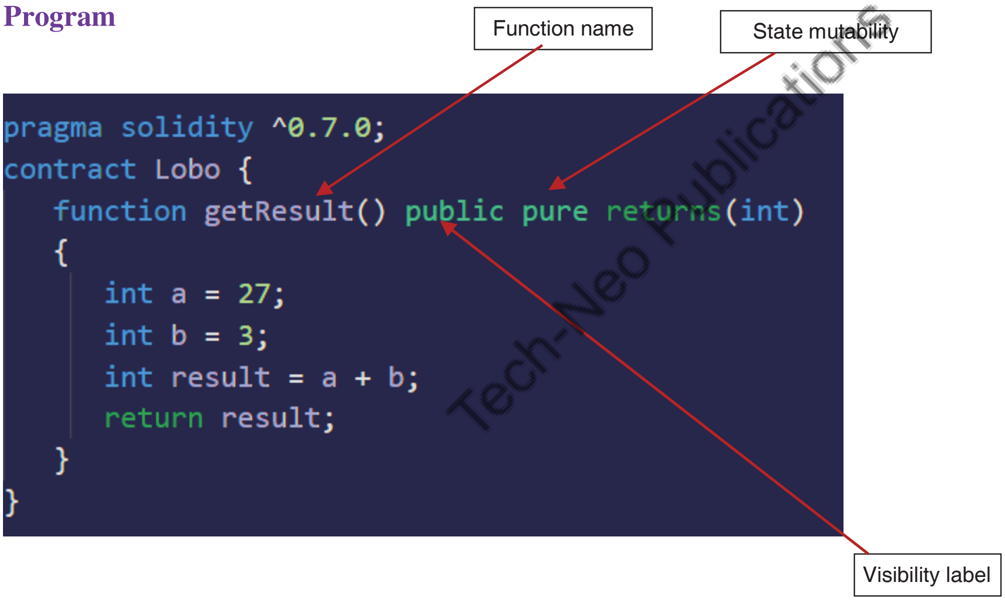
Functions play a crucial role in any programming language, and the same is applicable for solidity programming. Ethereum maintains a state variable's current state and executes transactions to change values in state variables. When a function in a smart contract is invoked, it results in transaction creation.

- Functions are the means to not only read but also write values from/to state variables.
- On-demand execution of code can be performed through functions by simply calling them.
- Functions can accept arguments or parameters,execute its logic, and optionally return values to the caller.

In solidity programming, the most common way of defining a function is by using the keyword *function* followed by a function name, a list of arguments or parameters, and a set of statements.

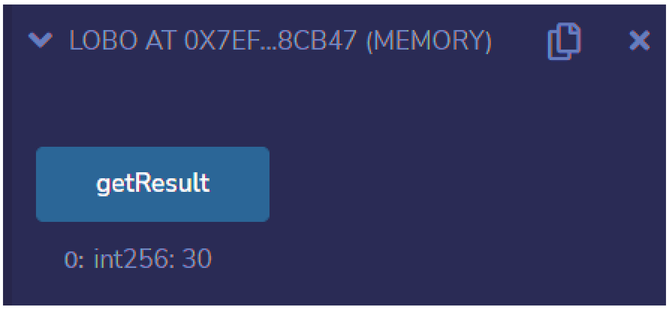
☞ Example 1

Program



```
pragma solidity ^0.7.0;
contract Lobo {
   function getResult() public pure returns(int)
   {
      int a = 27;
      int b = 3;
      int result = a + b;
      return result;
   }
}
```

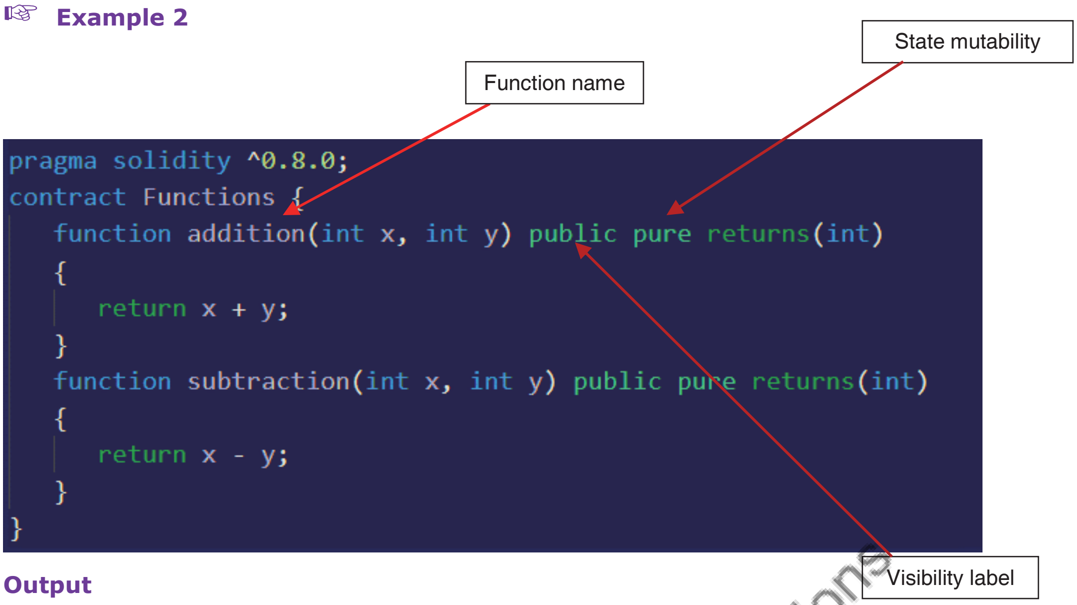
Output



```
LOBO AT 0X7EF...8CB47 (MEMORY)

getResult

0: int256: 30
```

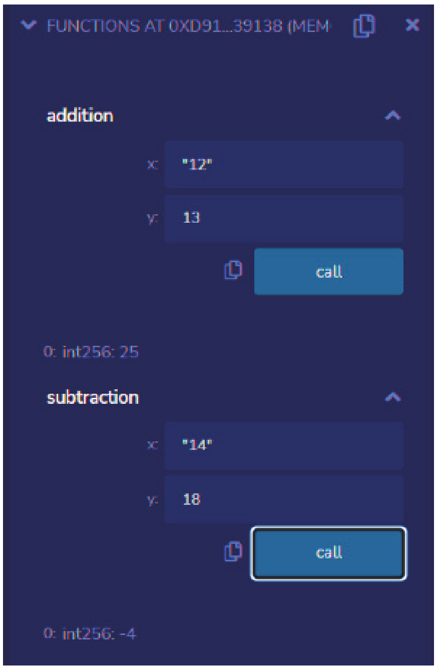
The above program is a simple example that defines a function called *getResult()* that takes no arguments or parameters.

☞ **Example 2**



```
pragma solidity ^0.8.0;
contract Functions {
    function addition(int x, int y) public pure returns(int)
    {
        return x + y;
    }
    function subtraction(int x, int y) public pure returns(int)
    {
        return x - y;
    }
}
```

Output



```
FUNCTIONS AT 0XD91...39138 (MEM

addition
        x: "12"
        y: 13
                call
0: int256: 25

subtraction
        x: "14"
        y: 18
                call
0: int256: -4
```

The above program is a simple example that define functions *addition()* and *subtraction()* that takes two integer arguments or parameters (i.e., *x* and *y*).

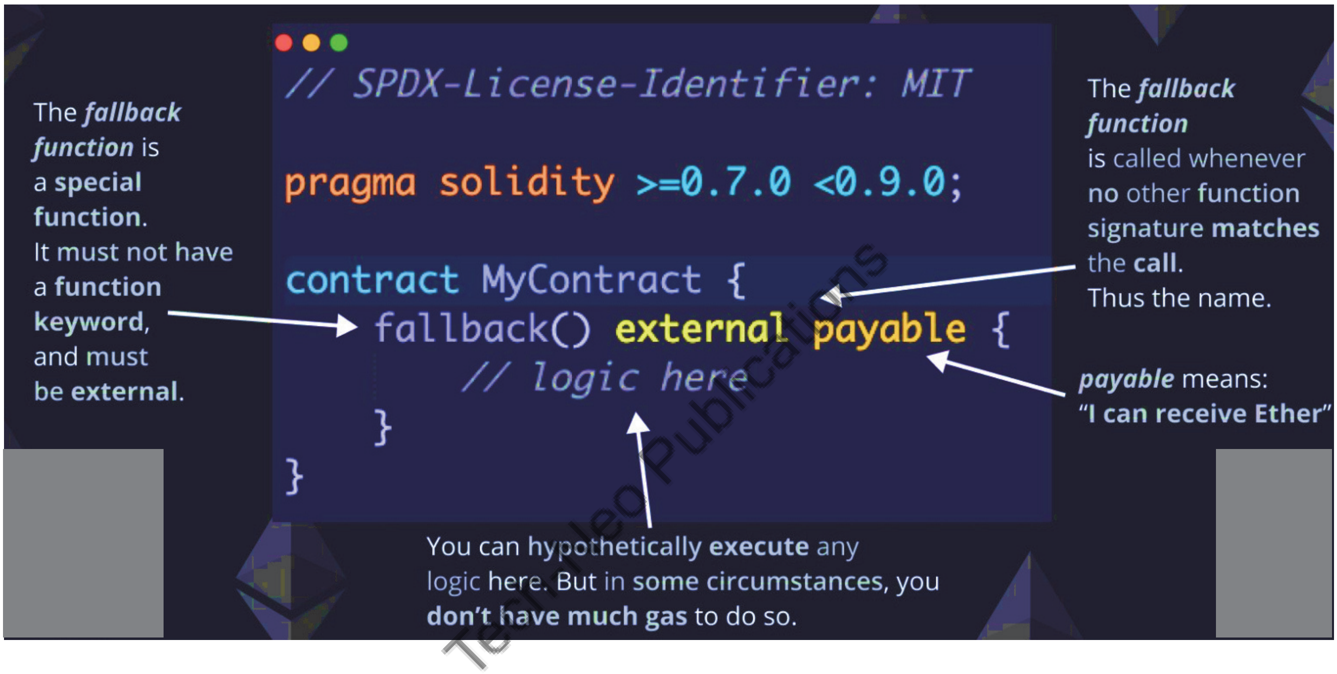
Fall back function

- If none of the other functions match the function identification or no data was provided with the function call, the solidity fallback function is called. A contract can only have one unnamed function, which is executed anytime the contract gets plainether without any data.
- To receive ether and add it to the contract's total balance, the fallback function must be marked payable. If such a function does not exist, then the contract will fail to receive anether via ordinary transactions and will throw an error.

☞ Properties

1. It has no name or arguments.
2. If it is not marked *payable*, then the contract will throw an exception if it receives plain ether without data.
3. It cannot return anything
4. It is also executed if the caller meant to call a function that is not available
5. It is mandatory to mark it external.

Example



```
// SPDX-License-Identifier: MIT

pragma solidity >=0.7.0 <0.9.0;

contract MyContract {
    fallback() external payable {
        // logic here
    }
}
```

The ***fallback function*** is a **special function.** It must not have a **function keyword,** and must **be external.**

The ***fallback function*** is called whenever **no other function signature matches** the **call.** Thus the name.

payable means: **"I can receive Ether"**

You can hypothetically **execute** any logic here. But **in some circumstances**, you **don't have much gas** to do so.

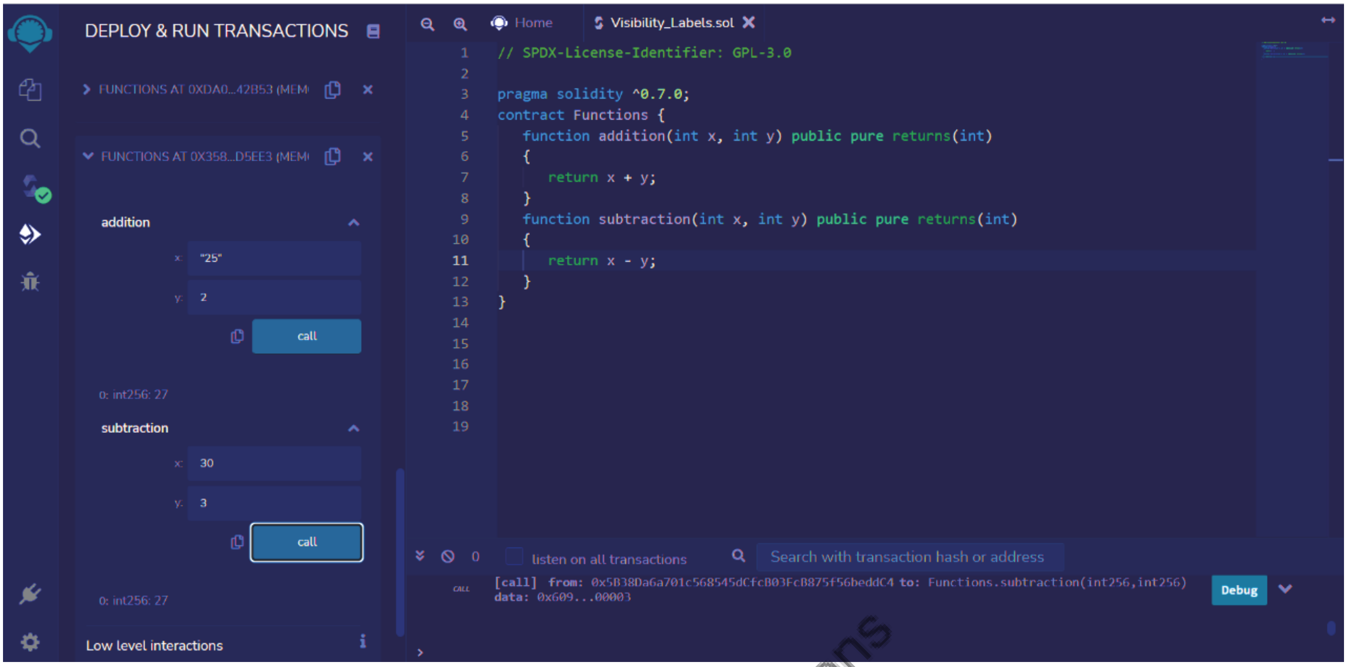
☞ Visibility and Activity Qualifiers

- There exists several quantifiers that affect not only the behavior of a function but also its execution. Functions possess visibility quantifiers and quantifiers pertaining to what actions/activities can be executed inside a function (i.e., activity quantifiers).
- Data can be returned in functions using the keyword *return*.

☞ Visibility Qualifiers

A function's visibility can be any one of the following :

- **public :** It makes a function access directly from outside (i.e., externally). Public functions become a part of the smart contract interface and can be called from within as well as from outside.



DEPLOY & RUN TRANSACTIONS
FUNCTIONS AT 0XDA0...42B53 (MEM
FUNCTIONS AT 0X358...D5EE3 (MEM
Home
Visibility_Labels.sol
Low level interactions
Debug

addition

x: "25"

y: 2

call

0: int256: 27

subtraction

x: 30

y: 3

call

0: int256: 27

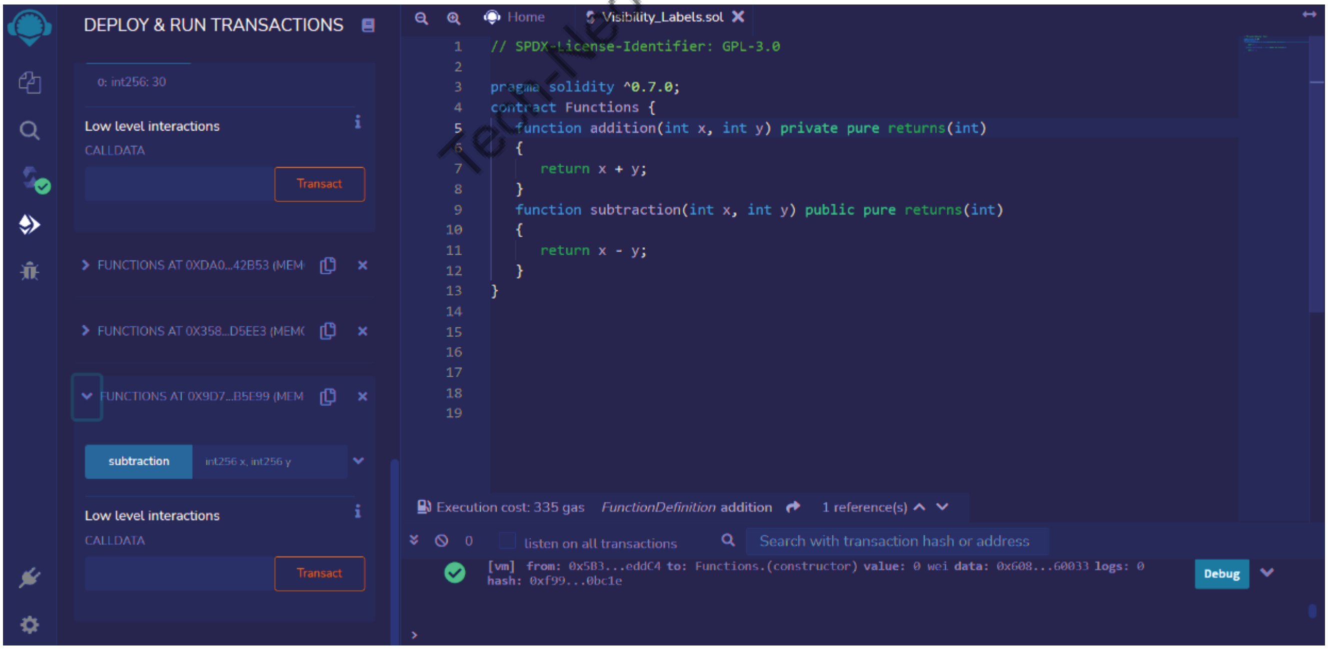
```
// SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.7.0;
contract Functions {
    function addition(int x, int y) public pure returns(int)
    {
        return x + y;
    }
    function subtraction(int x, int y) public pure returns(int)
    {
        return x - y;
    }
}
```

listen on all transactions   Search with transaction hash or address

```
CALL  [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Functions.subtraction(int256,int256)
      data: 0x609...00003
```

> From the above figure, it is evident that functions *addition()* and *subtraction()* in a smart contract *Functions* can be accessed externally after it has been successfully deployed.

- **private :** They can be used only if a contract is explicitly declared with a keyword *private*. Private functions are not a part of a smart contract's interface, and they cannot be used even within derived contracts.



DEPLOY & RUN TRANSACTIONS
Home
Visibility_Labels.sol
FUNCTIONS AT 0XDA0...42B53 (MEM
FUNCTIONS AT 0X358...D5EE3 (MEM
FUNCTIONS AT 0X9D7...B5E99 (MEM
Transact
Debug

0: int256: 30

Low level interactions

CALLDATA

subtraction   int256 x, int256 y

Low level interactions

CALLDATA

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.7.0;
contract Functions {
    function addition(int x, int y) private pure returns(int)
    {
        return x + y;
    }
    function subtraction(int x, int y) public pure returns(int)
    {
        return x - y;
    }
}
```

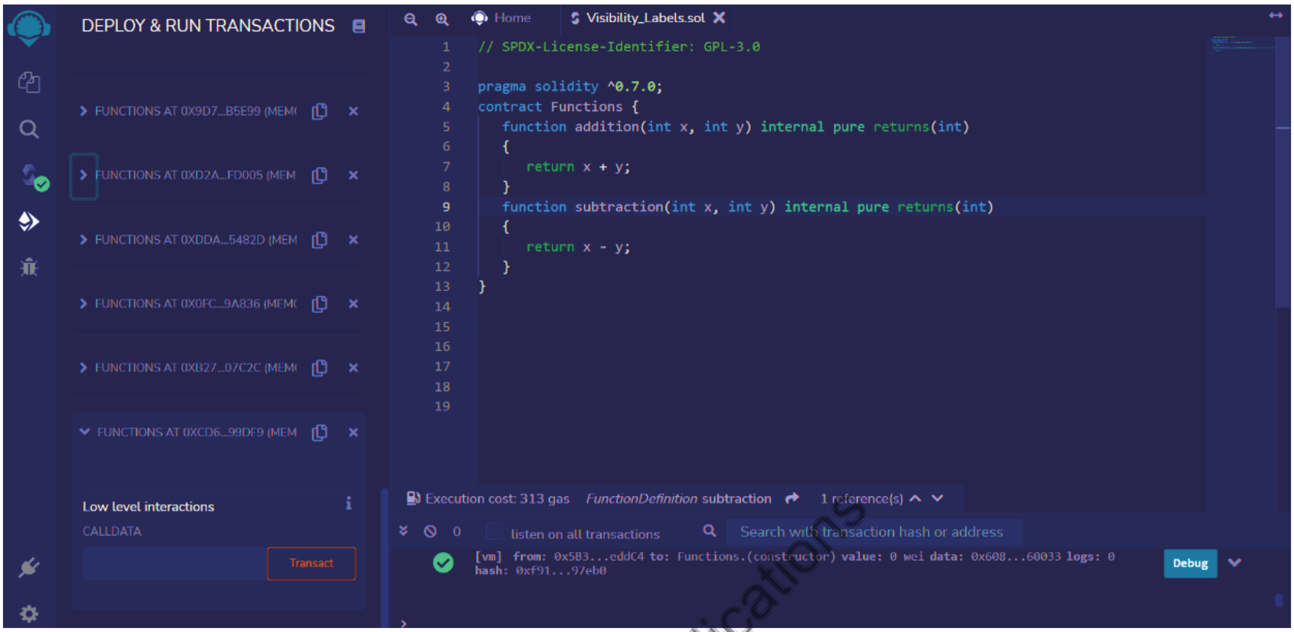
Execution cost: 335 gas   *FunctionDefinition* addition   1 reference(s)

listen on all transactions   Search with transaction hash or address

```
[vm] from: 0x5B3...eddC4 to: Functions.(constructor) value: 0 wei data: 0x608...60033 logs: 0
hash: 0xf99...0bc1e
```

> From the above figure, it is evident that function *addition()* is declared *private* and *subtraction()* is declared public. Thus, only *subtract()* can be accessed at the interface of the smart contract.

- **internal :** This function can only be used within the smart contract and any contract that inherits from it. Such functions cannot be accessed from outside and are not part of a smart contract's interface. Note that if nothing is specified, then by default *internal* qualifier is used.

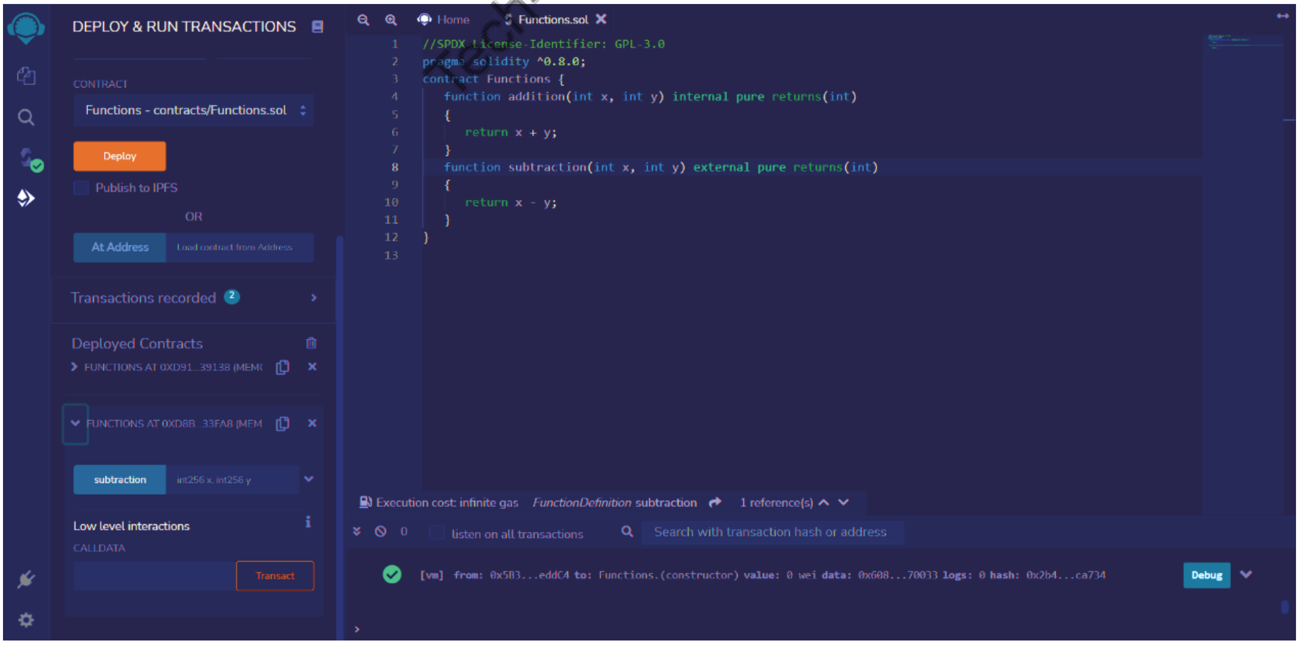

DEPLOY & RUN TRANSACTIONS
FUNCTIONS AT 0X9D7...B5E99 (MEM
FUNCTIONS AT 0XD2A...FD005 (MEM
FUNCTIONS AT 0XDDA...5482D (MEM
FUNCTIONS AT 0X0FC...9A836 (MEM
FUNCTIONS AT 0XB27...07C2C (MEM
FUNCTIONS AT 0XCD6...99DF9 (MEM
Low level interactions
CALLDATA
Transact
Home
Visibility_Labels.sol
Debug

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.7.0;
contract Functions {
    function addition(int x, int y) internal pure returns(int)
    {
        return x + y;
    }
    function subtraction(int x, int y) internal pure returns(int)
    {
        return x - y;
    }
}
```

```
Execution cost: 313 gas   FunctionDefinition subtraction   1 reference(s)
0   listen on all transactions   Search with transaction hash or address
[vm] from: 0x5B3...eddC4 to: Functions.(constructor) value: 0 wei data: 0x608...60033 logs: 0
hash: 0xf91...97eb0
```

From the above figure, it is evident that both functions *addition()* and *subtraction()* are declared internal, and so, they cannot be accessed from outside.

- **external :** It makes a function access directly from outside but not inside. Such functions become part of the contracts interface.



```
//SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.0;
contract Functions {
  function addition(int x, int y) internal pure returns(int)
  {
    return x + y;
  }
  function subtraction(int x, int y) external pure returns(int)
  {
    return x - y;
  }
}
```

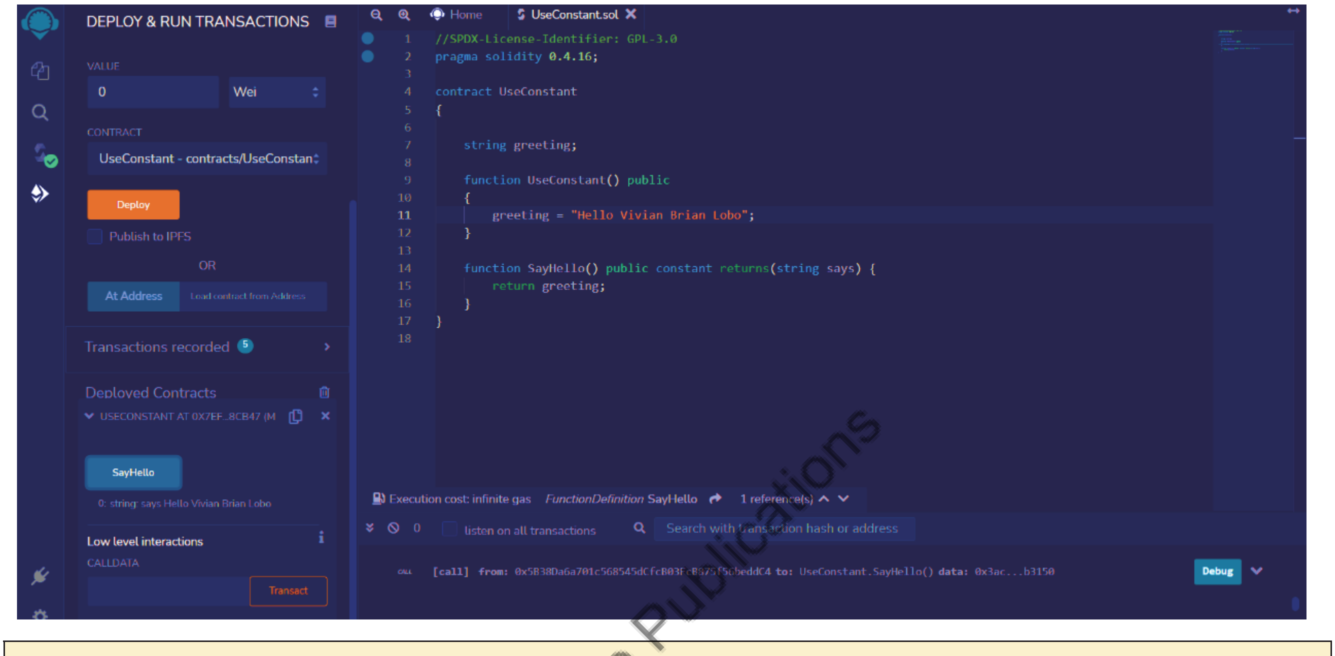
```
Execution cost: infinite gas   FunctionDefinition subtraction   1 reference(s)
0   listen on all transactions   Search with transaction hash or address
[vm] from: 0x5B3...eddC4 to: Functions.(constructor) value: 0 wei data: 0x608...70033 logs: 0 hash: 0x2b4...ca734
```

From the above figure, it is evident that function *subtraction()* is declared external, which is accessible from the smart contract's interface, whereas function *addition()* cannot be accessed.

☞ Activity Qualifiers

A function’s behavior can be changed through the following activity qualifiers :

- **constant :** They can only read state variables and return back to the caller but cannot modify any variable, invoke an event, create another smart contract, and call another function.



VALUE: 0 Wei

CONTRACT: UseConstant - contracts/UseConstant

USECONSTANT AT 0X7EF...8CB47 (M

0: string: says Hello Vivian Brian Lobo

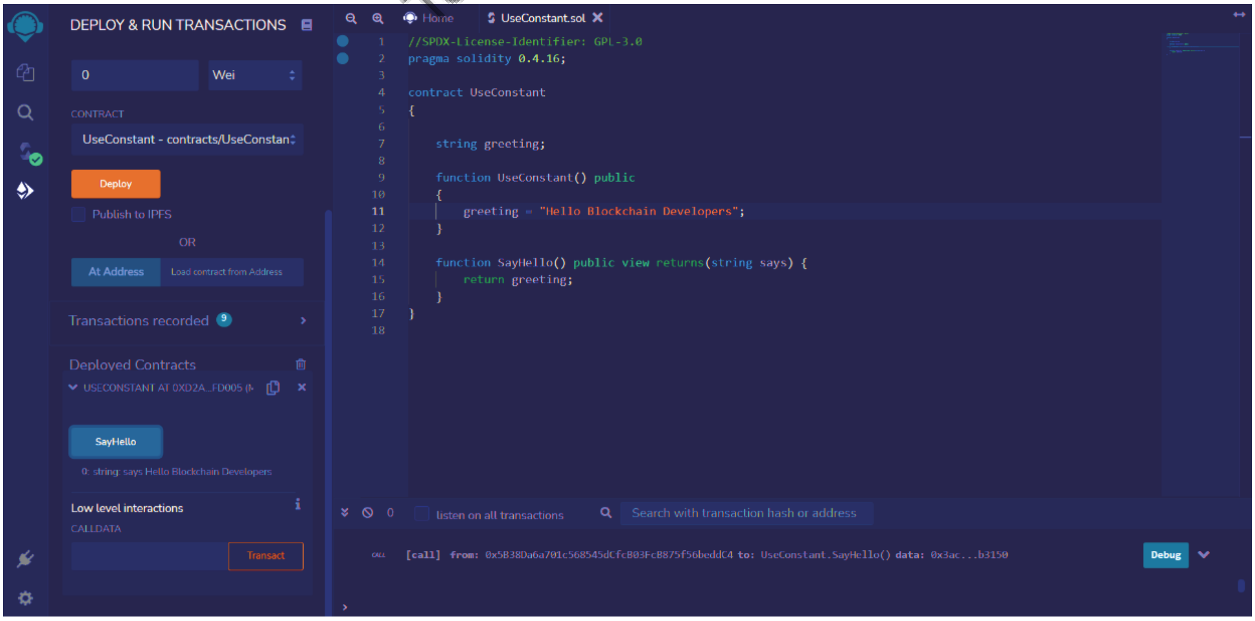
```
//SPDX-License-Identifier: GPL-3.0
pragma solidity 0.4.16;

contract UseConstant
{

    string greeting;

    function UseConstant() public
    {
        greeting = "Hello Vivian Brian Lobo";
    }

    function SayHello() public constant returns(string says) {
        return greeting;
    }
}
```

Execution cost: infinite gas *FunctionDefinition* SayHello 1 reference(s)

```
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: UseConstant.SayHello() data: 0x3ac...b3150
```

From the above figure, it is evident that function *SayHello()* cannot modify the state of a blockchain. It can only read the state variable in this case.

- **view :** It can be considered asa replacement or an alias for *constant*. It indicates that a function will not modify the storage state in any way.



0 Wei

CONTRACT: UseConstant - contracts/UseConstant

USECONSTANT AT 0XD2A...FD005 (M

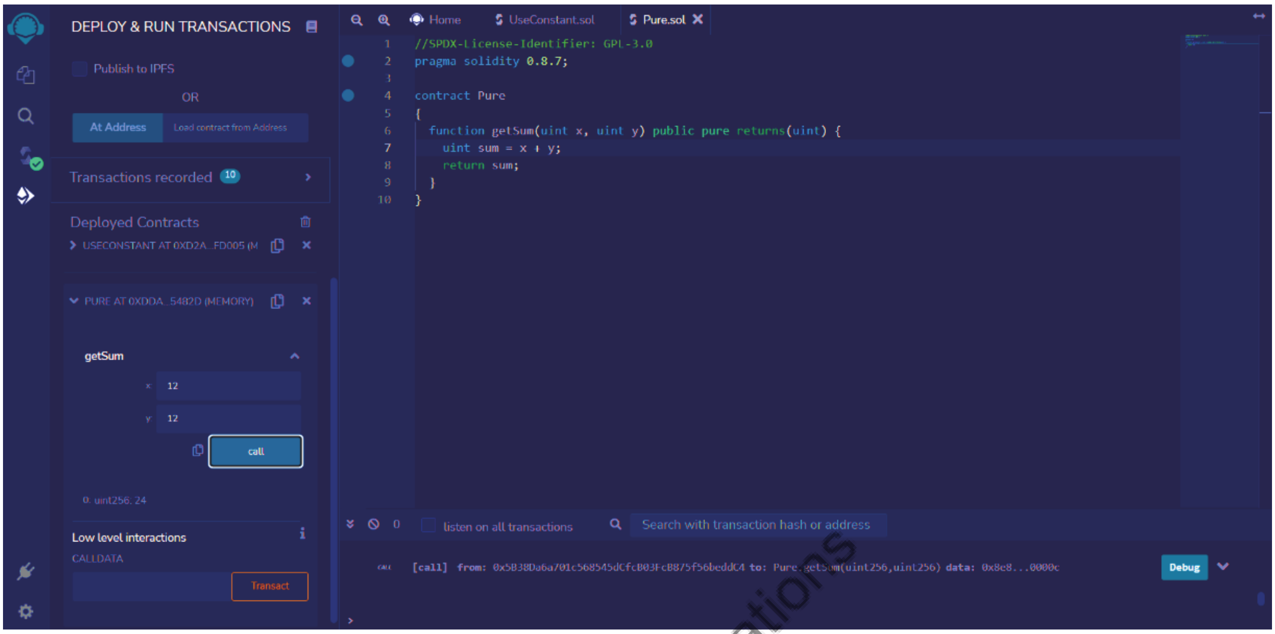
0: string: says Hello Blockchain Developers

```
//SPDX-License-Identifier: GPL-3.0
pragma solidity 0.4.16;

contract UseConstant
{

    string greeting;

    function UseConstant() public
    {
        greeting = "Hello Blockchain Developers";
    }

    function SayHello() public view returns(string says) {
        return greeting;
    }
}
```

```
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: UseConstant.SayHello() data: 0x3ac...b3150
```

Here, the function *SayHello()* is denoted by an activity qualifier *view*.

- **pure :** They are even stricter than *view*. It can neither make any changes to state variables nor read them. They cannot make any calculations that use any state variables as its component.


DEPLOY & RUN TRANSACTIONS
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 10
Deployed Contracts
USECONSTANT AT 0XD2A...FD005 (M
PURE AT 0XDDA...5482D (MEMORY)
getSum
x: 12
y: 12
call
0: uint256: 24
Low level interactions
CALLDATA
Transact
Home
UseConstant.sol
Pure.sol
listen on all transactions
Search with transaction hash or address
Debug

```
//SPDX-License-Identifier: GPL-3.0
pragma solidity 0.8.7;

contract Pure
{
  function getSum(uint x, uint y) public pure returns(uint) {
    uint sum = x + y;
    return sum;
  }
}
```

```
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Pure.getSum(uint256,uint256) data: 0x8e8...0000c
```

Here, the function *getSum()* does not read or modify any state variables, and so, it is considered *pure*.

- **payable :** Functions declared with the *payable* keyword can only accept Ether from a caller. The call will fail in case Ether is not provided by a sender. A function can only accept an Ether if it is marked as *payable*.

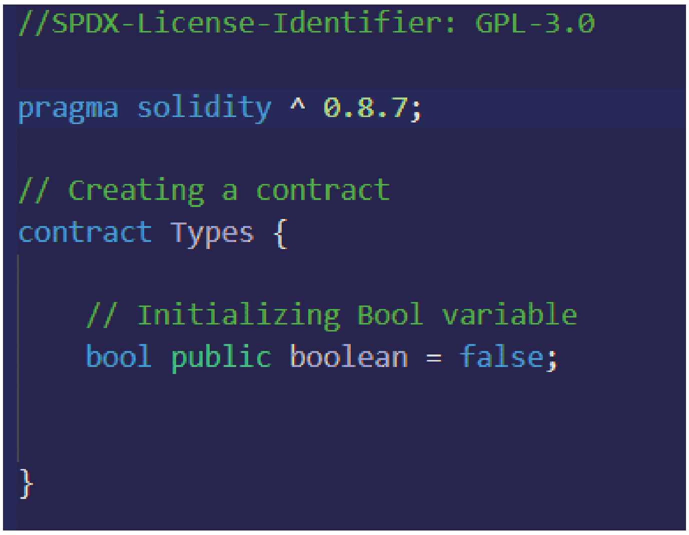
☞ Address and Address Payable

The distinction between address and address payable was introduced in Solidity version 0.5.0. The idea was to make the distinction between addresses that can receive money, and those who can't (used for other purposes). Simply speaking, an address payable can receive Ether, while a plain address cannot.

☞ Data types in solidity

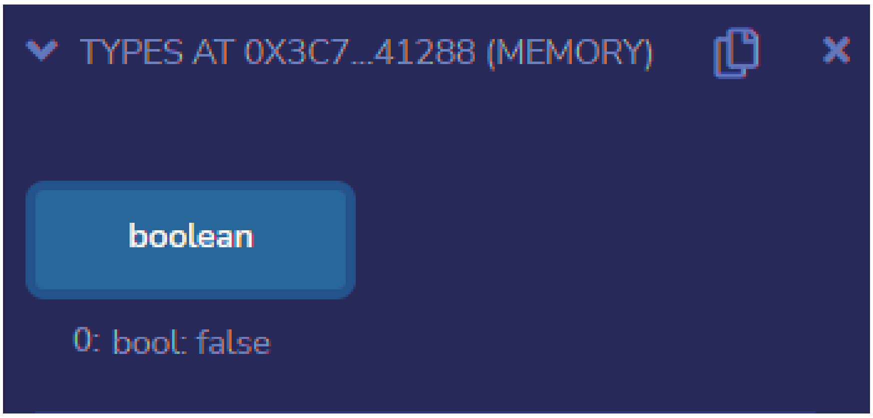
- **bool :** It can hold true or false as its value.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity ^ 0.8.7;

// Creating a contract
contract Types {

    // Initializing Bool variable
    bool public boolean = false;


}
```

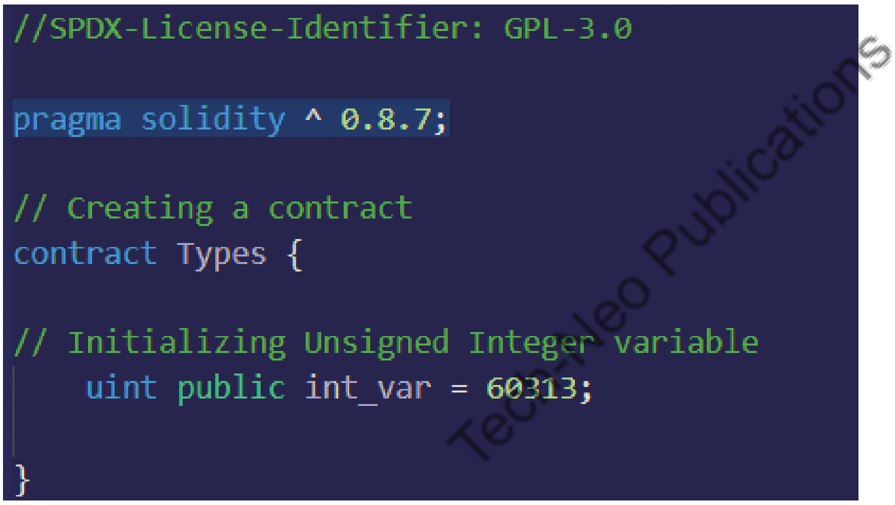
Output



```
0: bool: false
```

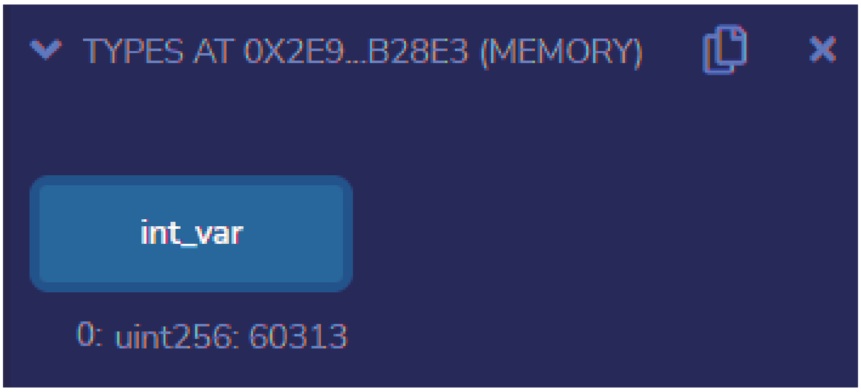
- **uint :** They are unsigned integers that can have only 0 and +ve values only.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity ^ 0.8.7;

// Creating a contract
contract Types {

// Initializing Unsigned Integer variable
    uint public int_var = 60313;

}
```

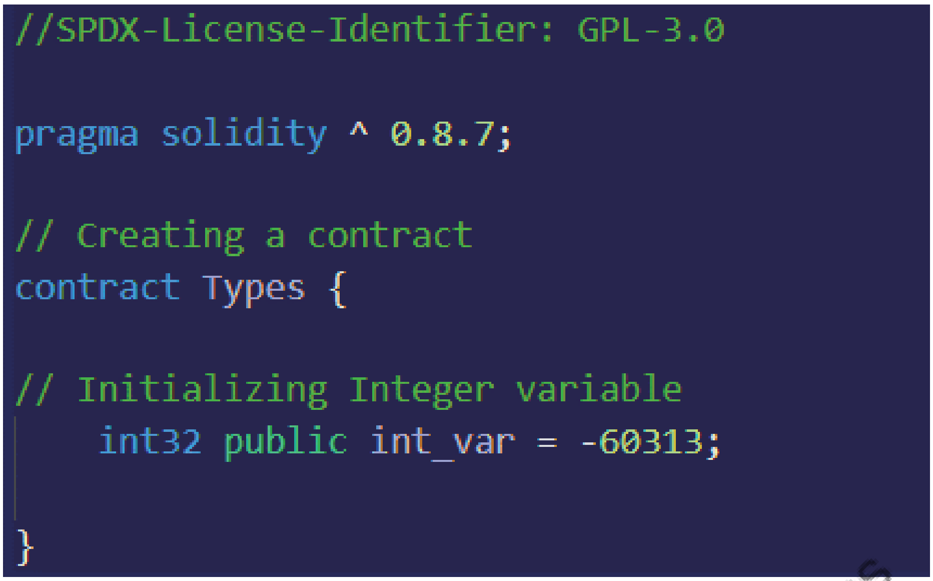
Output



```
0: uint256: 60313
```

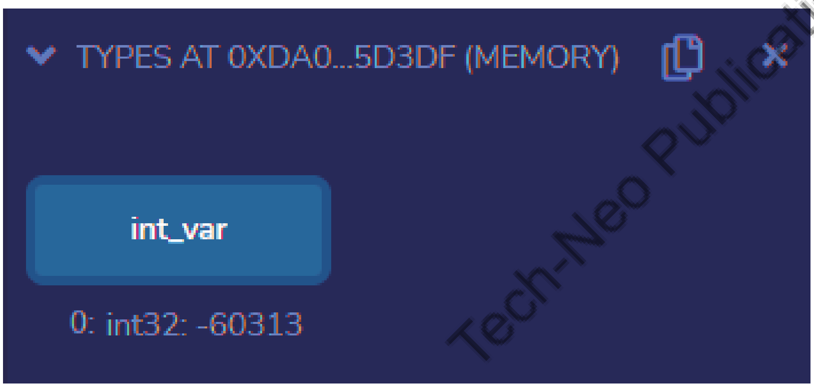
- **int :** Theyare signed integers that can hold +ve as well as –ve values.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity ^ 0.8.7;

// Creating a contract
contract Types {

// Initializing Integer variable
    int32 public int_var = -60313;

}
```

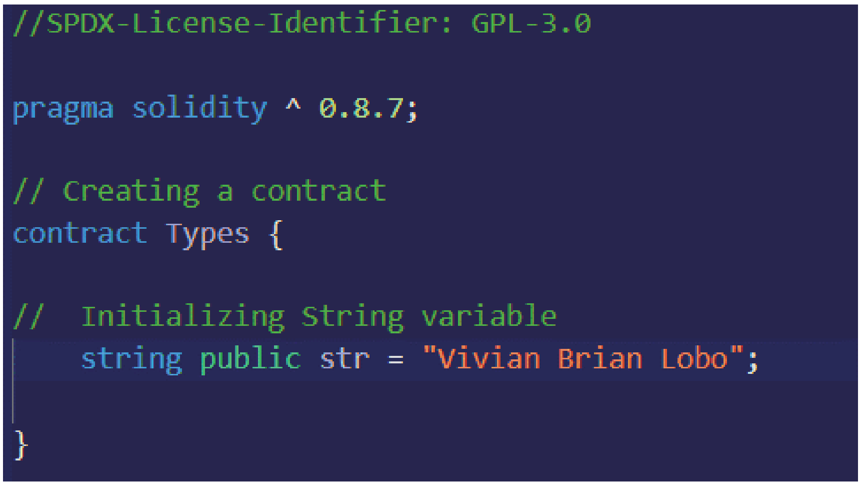
Output



```
0: int32: -60313
```

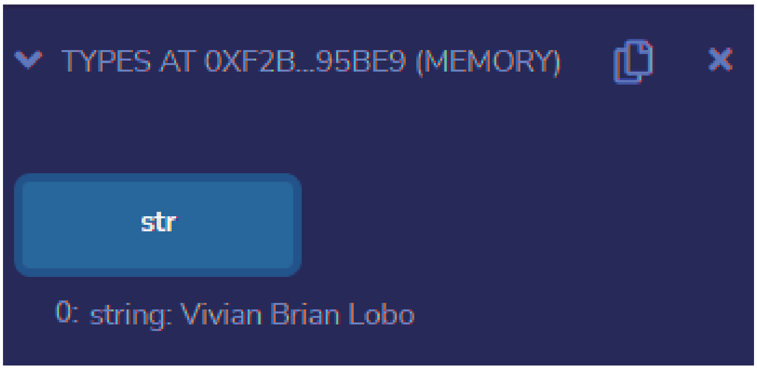
- **string :** A combination of characters is known as strings. In solidity programming, the keyword **string** is used to declare a string.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity ^ 0.8.7;

// Creating a contract
contract Types {

//  Initializing String variable
    string public str = "Vivian Brian Lobo";

}
```

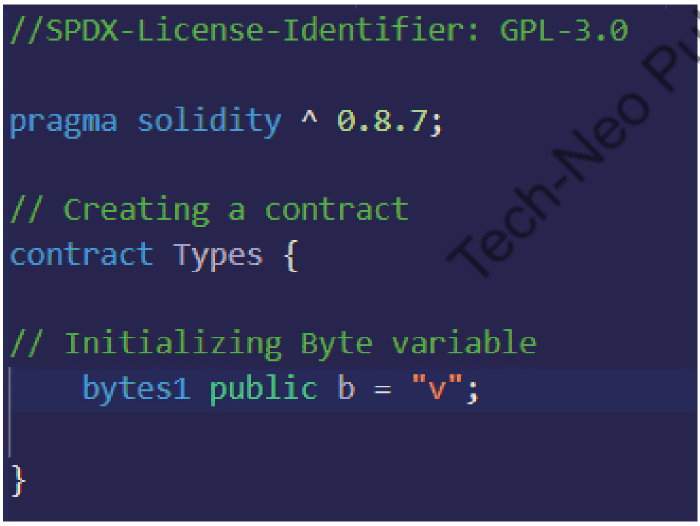
Output



```
0: string: Vivian Brian Lobo
```

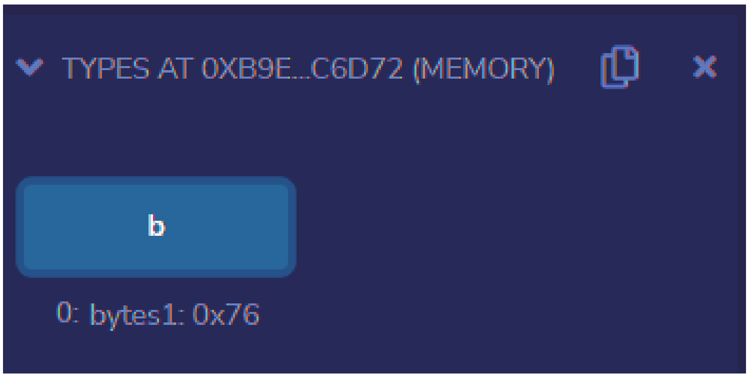
- **bytes :** It refers to 8 bit signed integers. Everything in memory is stored in bits consisting of binary values—0 and 1. Solidity provides the byte data type to store information in binary format. In general, programming languages have a single data type for representing bytes. However, solidity offers different types of byte datatype. It provides data types in the range from bytes1 to bytes32 to represent varying byte lengths. These are called fixed-sized byte arrays and are implemented as value types. The bytes1 data type represents 1 byte and bytes2 represents 2 bytes. The default value for byte is 0×00, and it gets initialized with this value. Solidity also has a byte type that is an alias to bytes1.
- A byte can be assigned character values as follows :

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity ^ 0.8.7;

// Creating a contract
contract Types {

// Initializing Byte variable
    bytes1 public b = "v";

}
```

Output

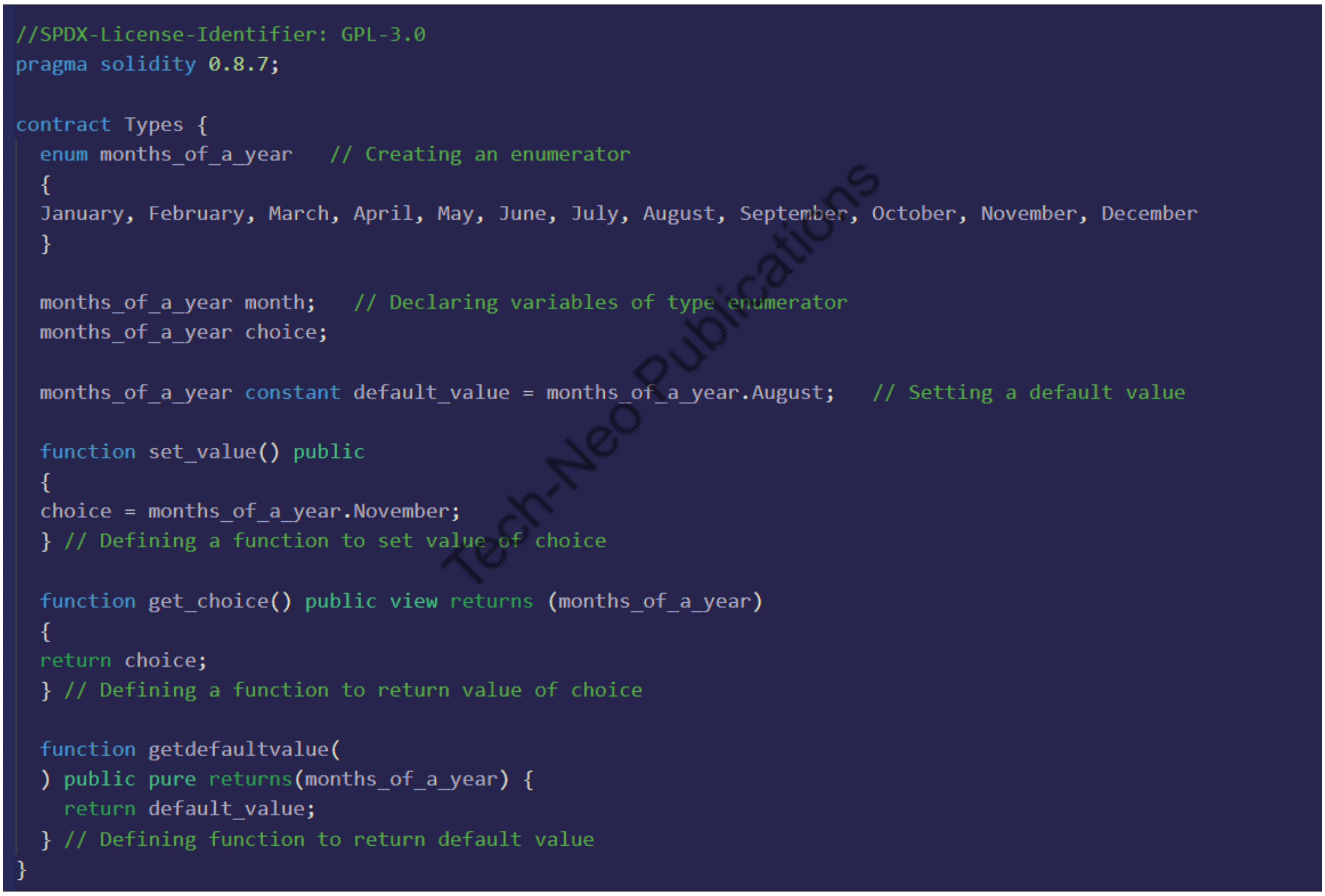


```
0: bytes1: 0x76
```

- ***In a similar manner, we have***
- *bytes1 public vv = 0x65;   // A byte can be assigned byte values in hexadecimal format*
- *bytes1 public rr = 28;   // A byte can be assigned integer values in decimal format*
- *bytes1 public ss = -22;   // A byte can be assigned negative integer values in decimal format*

- **Enums :** They are a technique to create user-defined data types and are generally used to provide names for integral constants, which improves the readability and maintenance of a smart contract. Enums are predetermined values that limit a variable to one of a small number of options. Integer numbers beginning at zero are used to represent options.Enums can also have a default value. The number of defects in a code can be decreased by using enums.

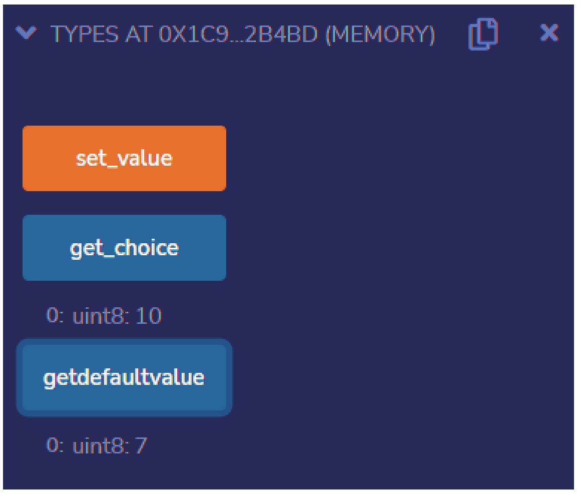
Program



```
//SPDX-License-Identifier: GPL-3.0
pragma solidity 0.8.7;

contract Types {
  enum months_of_a_year   // Creating an enumerator
  {
  January, February, March, April, May, June, July, August, September, October, November, December
  }

  months_of_a_year month;   // Declaring variables of type enumerator
  months_of_a_year choice;

  months_of_a_year constant default_value = months_of_a_year.August;   // Setting a default value

  function set_value() public
  {
  choice = months_of_a_year.November;
  } // Defining a function to set value of choice

  function get_choice() public view returns (months_of_a_year)
  {
  return choice;
  } // Defining a function to return value of choice

  function getdefaultvalue(
  ) public pure returns(months_of_a_year) {
    return default_value;
  } // Defining function to return default value
}
```

In the example, the contract *Types* consist of an enumerator months_of_a_year, and functions are defined to *set* and *get* a variable's value of the type enumerator.

Output

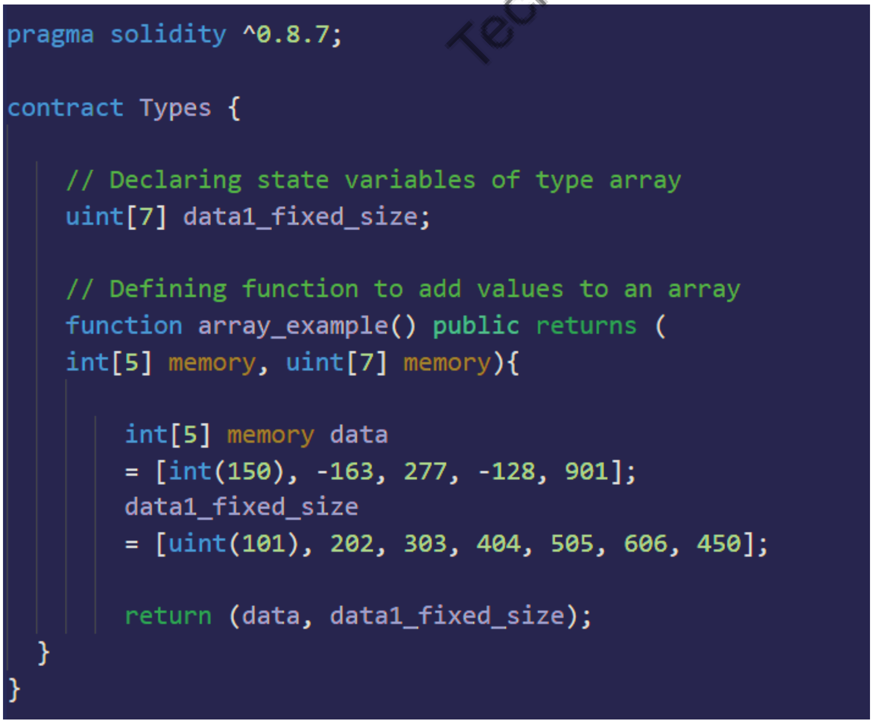


```
TYPES AT 0X1C9...2B4BD (MEMORY)
0: uint8: 10
0: uint8: 7
```

☞ Arrays : Fixed and Dynamic Arrays

- **Arrays :** A fixed collection of identically typed elements are stored in arrays. Each element in an array has a unique place known as an index.
- An array of a specific size needs to be declared, and then we need to put the elements in the array and make use of the index to retrieve the array's elements rather than establishing several separate variables of the same type. Arrays in solidity can either be fixed or dynamic in terms of its size. The first element in an array is represented by the lowest index, and the last element is represented by the highest index.
  1. **Fixed-size array :** An array's size ought to be known in advance. The total number of elements must not be greater than an array's size. If an array's size is not supplied, then an array of sufficient size needs to be built to accommodate the initialization.

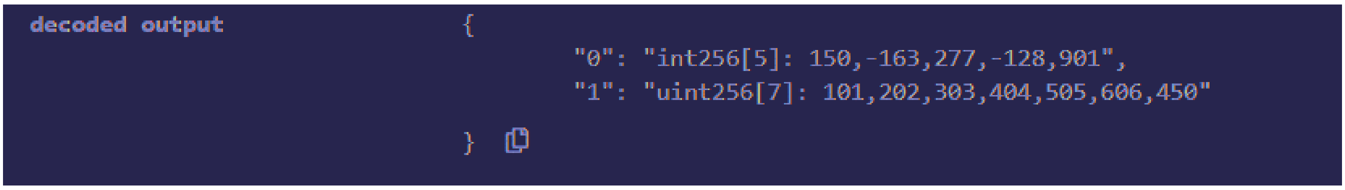
Program



```
pragma solidity ^0.8.7;

contract Types {

    // Declaring state variables of type array
    uint[7] data1_fixed_size;

    // Defining function to add values to an array
    function array_example() public returns (
    int[5] memory, uint[7] memory){

        int[5] memory data
        = [int(150), -163, 277, -128, 901];
        data1_fixed_size
        = [uint(101), 202, 303, 404, 505, 606, 450];

        return (data, data1_fixed_size);
  }
}
```

In the example, the contract *Types* are created to demonstrate how to declare and initialize *fixed-size arrays*.

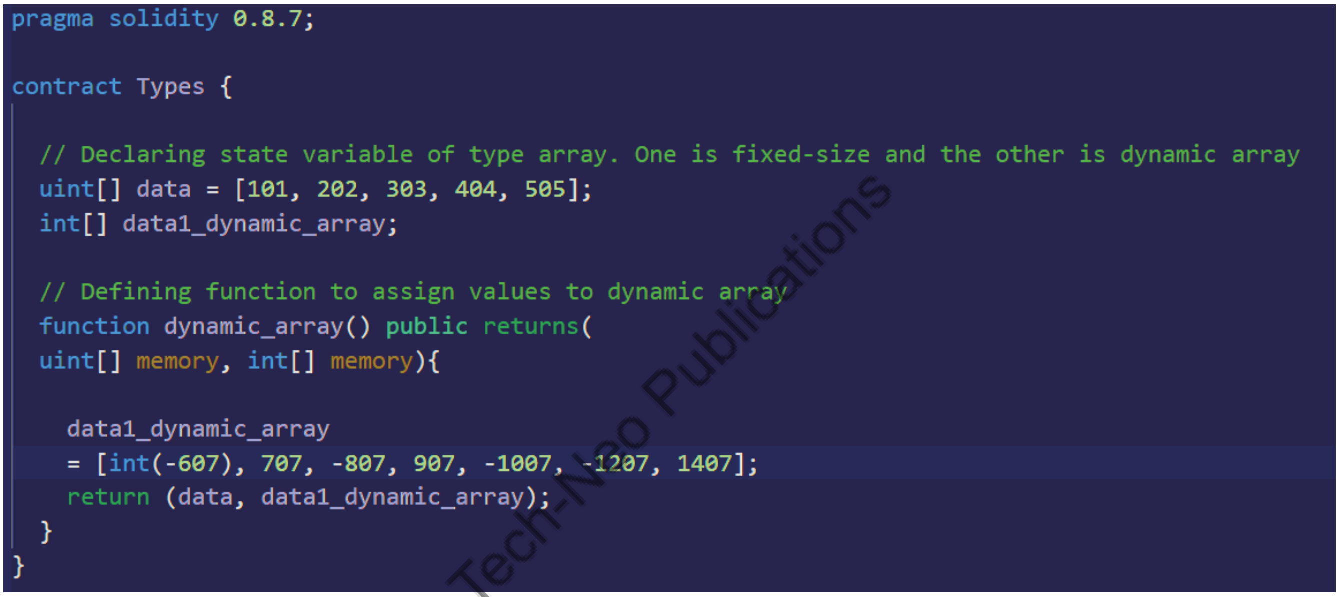
Output



```
decoded output                {
                                  "0": "int256[5]: 150,-163,277,-128,901",
                                  "1": "uint256[7]: 101,202,303,404,505,606,450"
                              }
```

2. **Dynamic array :** When an array is declared, its size is not known in advance. An array's size changes when new elements are added, and it will be calculated during runtime.

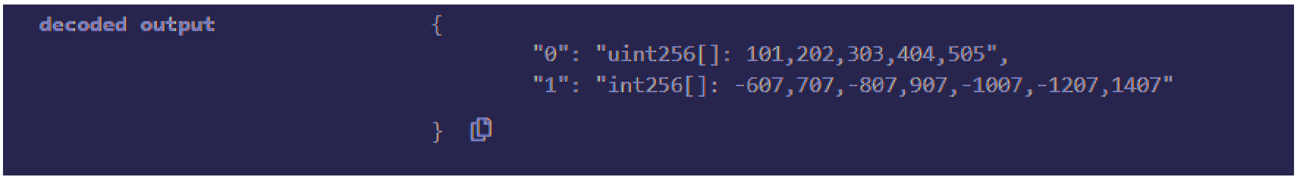
Program



```
pragma solidity 0.8.7;

contract Types {

  // Declaring state variable of type array. One is fixed-size and the other is dynamic array
  uint[] data = [101, 202, 303, 404, 505];
  int[] data1_dynamic_array;

  // Defining function to assign values to dynamic array
  function dynamic_array() public returns(
  uint[] memory, int[] memory){

    data1_dynamic_array
    = [int(-607), 707, -807, 907, -1007, -1207, 1407];
    return (data, data1_dynamic_array);
  }
}
```

In the example, the contract *Types* are created to demonstrate how to declare and initialize *dynamic arrays*.

Output



```
decoded output              {
                                "0": "uint256[]: 101,202,303,404,505",
                                "1": "int256[]: -607,707,-807,907,-1007,-1207,1407"
                            }
```

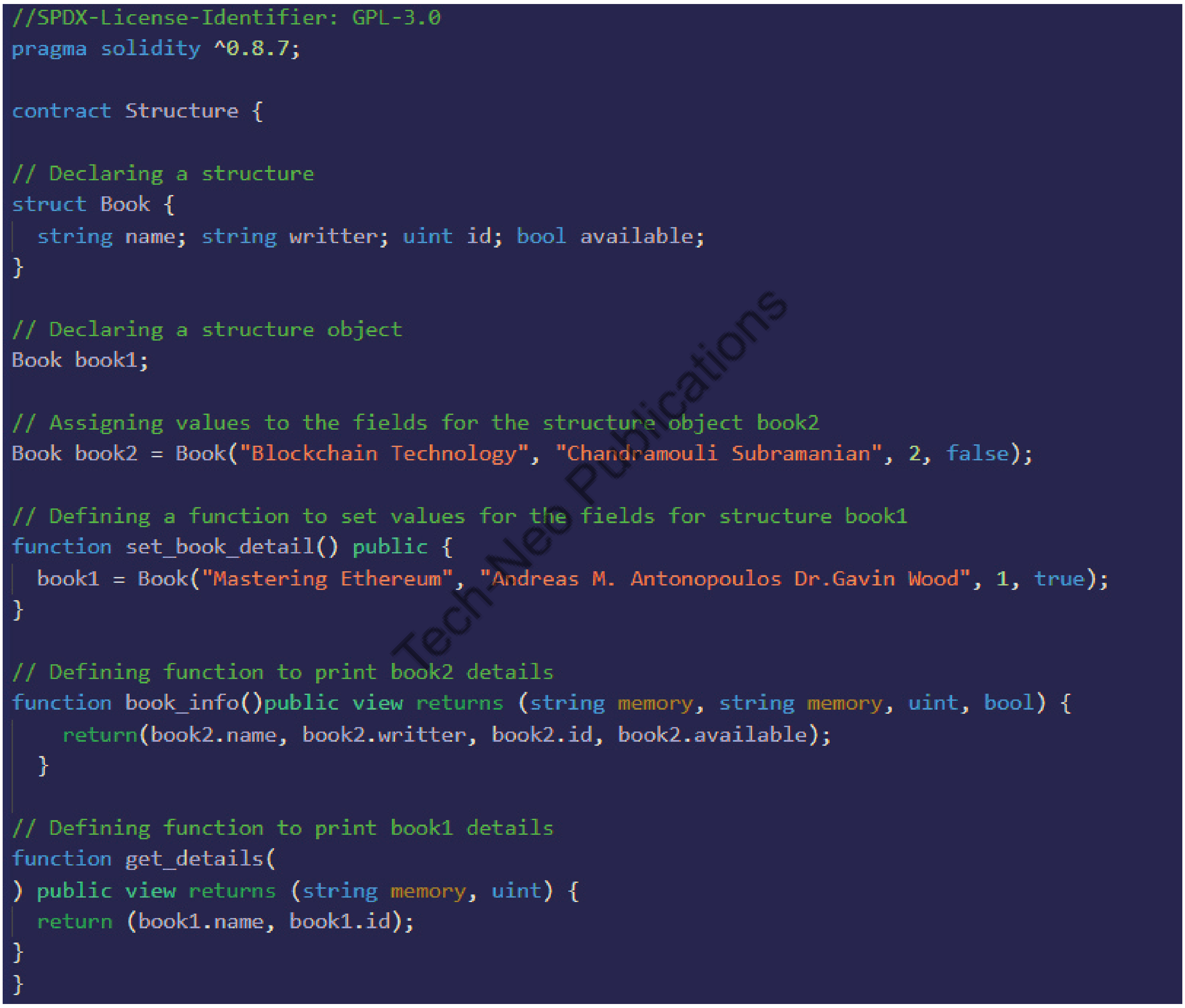
☞ Special Dynamically Sized Arrays

Solidity provides two special arrays:

- **bytes array :** It is a dynamic array that can hold any number of bytes(i.e., they can hold an arbitrary length of raw byte data). As bytesis treated as an array in solidity programming, it can have a length of zero and one can append a byte to the end.
- **string array :** It is a dynamic array of UTF-8 data. *string* in solidity programming does not provide functions to obtain string length or perform concatenation or comparison between two strings. A string can be converted to a byte array using *bytes(<string>)*.

- **struct :** Solidity allows an individual to create complicated datatypes that have multiple properties, which can be done by using struct.One can define his/her own datatype by creating a struct. Structs are helpful for collecting related data into one category. Structures may be imported into one contract from another after being declared outside of it. It often serves as a record representation. The struct keyword, which generates a new data type, is used to define a structure. The "dot operator" is used to delineate the space between the struct variable and the element one wants to access when accessing any element of the structure. To define the variable of structure data type, structure name is used.

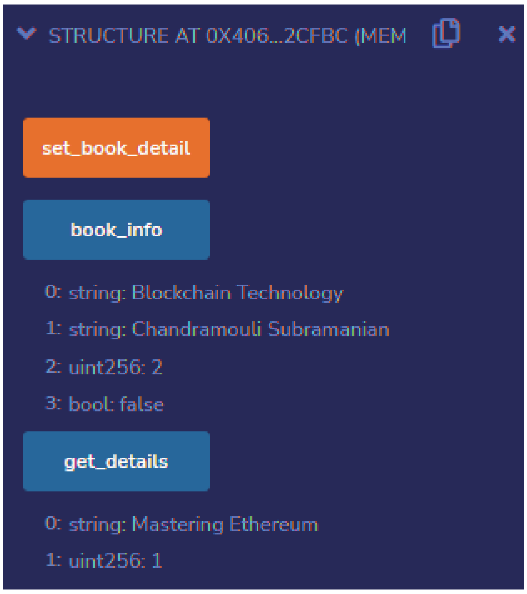
Program



```
//SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.7;

contract Structure {

// Declaring a structure
struct Book {
  string name; string writter; uint id; bool available;
}

// Declaring a structure object
Book book1;

// Assigning values to the fields for the structure object book2
Book book2 = Book("Blockchain Technology", "Chandramouli Subramanian", 2, false);

// Defining a function to set values for the fields for structure book1
function set_book_detail() public {
  book1 = Book("Mastering Ethereum", "Andreas M. Antonopoulos Dr.Gavin Wood", 1, true);
}

// Defining function to print book2 details
function book_info()public view returns (string memory, string memory, uint, bool) {
    return(book2.name, book2.writter, book2.id, book2.available);
  }

// Defining function to print book1 details
function get_details(
) public view returns (string memory, uint) {
  return (book1.name, book1.id);
}
}
```

In the example, the contract *Structure* consists of a structure Book, and functions are defined to *set* and *get* values of the elements of the structure.

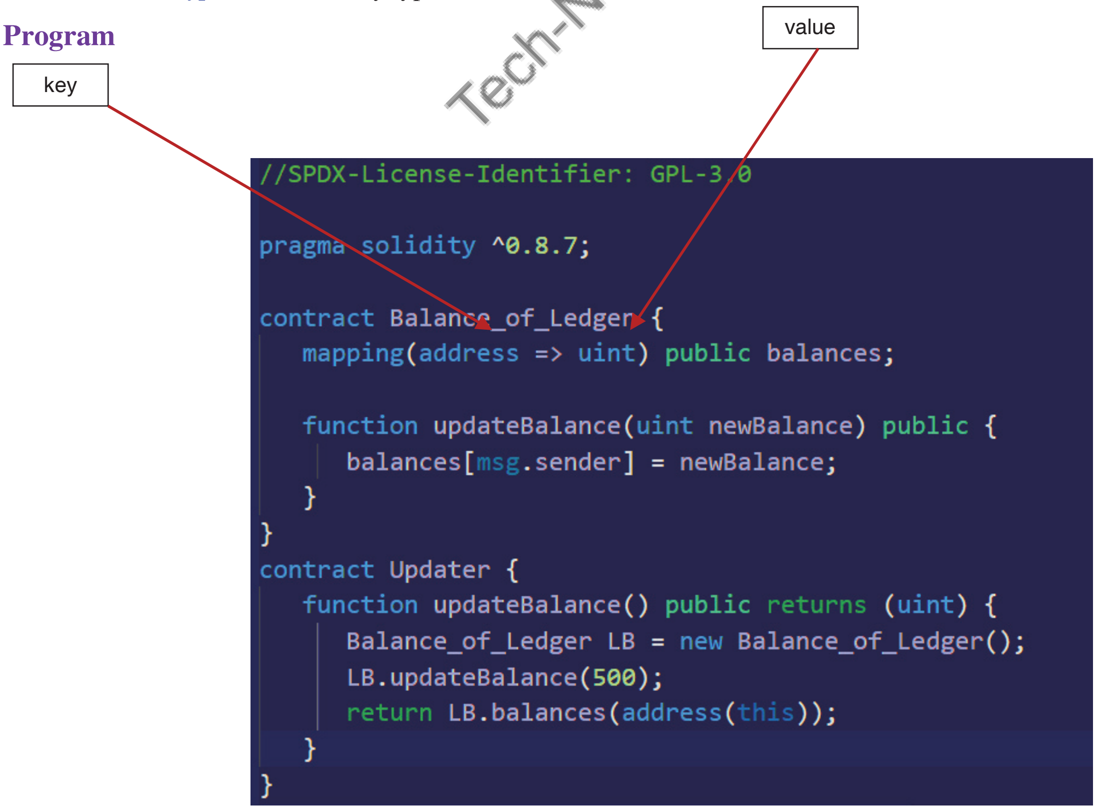
Output



```
STRUCTURE AT 0X406...2CFBC (MEM

set_book_detail

book_info

0: string: Blockchain Technology
1: string: Chandramouli Subramanian
2: uint256: 2
3: bool: false

get_details

0: string: Mastering Ethereum
1: uint256: 1
```

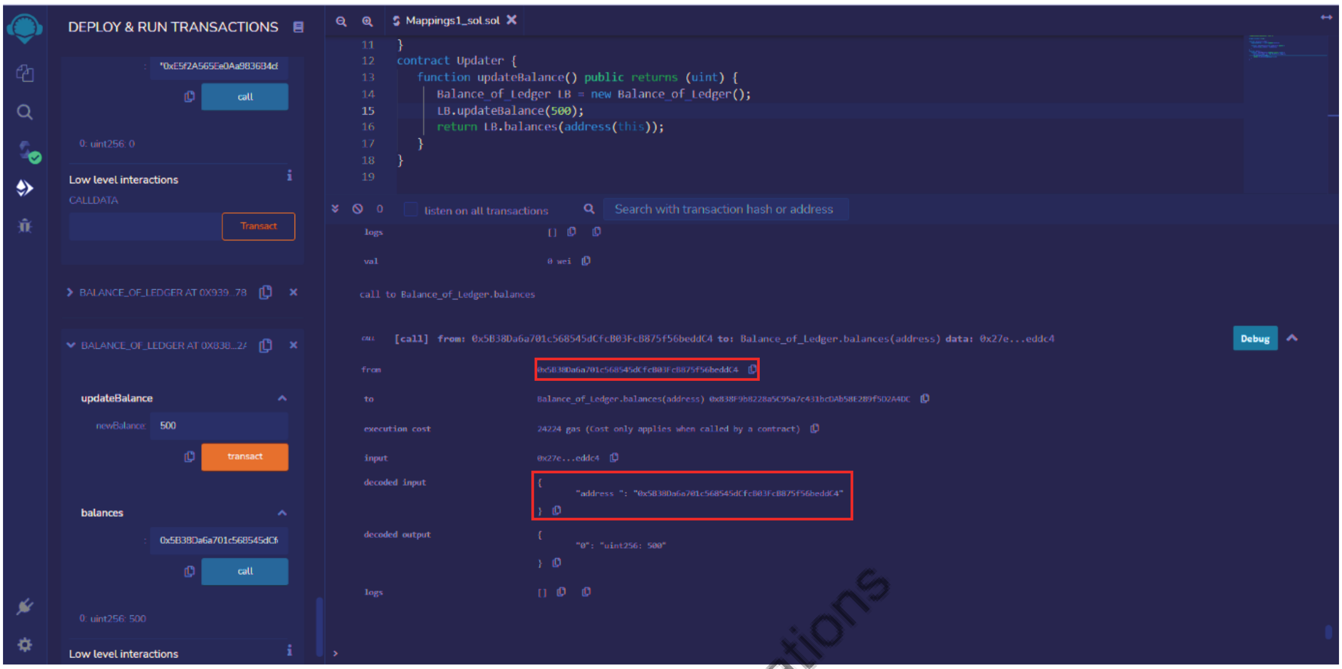
- **Mapping :** They store the data in the form of key–value pair. A key can be any of built-in data types, although reference types are not permitted, and the value can be of any type. Mostly, mappings are used to link the specific Ethereum address to the relevant value type. They act like a hash table or dictionary in any other programming language.Mapping is defined as any other variable type, which accepts a key type and a value type.
  - ○ ***key type :*** It can be any built-in types as well as bytes and string. No reference type or complex objects are allowed.
  - ○ ***value type :*** It can be any type.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.7;

contract Balance_of_Ledger {
   mapping(address => uint) public balances;

   function updateBalance(uint newBalance) public {
      balances[msg.sender] = newBalance;
   }
}
contract Updater {
   function updateBalance() public returns (uint) {
      Balance_of_Ledger LB = new Balance_of_Ledger();
      LB.updateBalance(500);
      return LB.balances(address(this));
   }
}
```

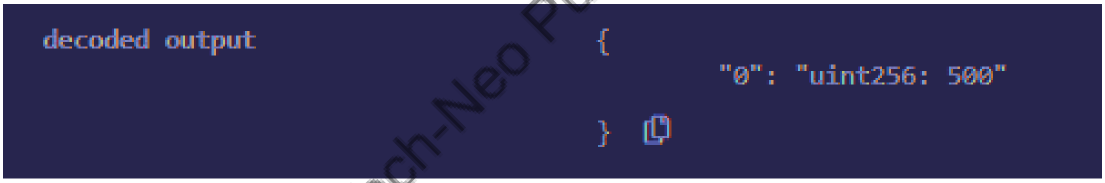
Output



```
}
contract Updater {
    function updateBalance() public returns (uint) {
      Balance_of_Ledger LB = new Balance_of_Ledger();
      LB.updateBalance(500);
      return LB.balances(address(this));
    }
}
```

```
logs              []
val               0 wei
call to Balance_of_Ledger.balances
call  [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Balance_of_Ledger.balances(address) data: 0x27e...eddc4
from              0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                Balance_of_Ledger.balances(address) 0x838F9b8228a5C95a7c431bcDAb58E289f5D2A4DC
execution cost    24224 gas (Cost only applies when called by a contract)
input             0x27e...eddc4
decoded input     {
                       "address ": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4"
                  }
decoded output    {
                       "0": "uint256: 500"
                  }
logs              []
```

Click on *updateBalance* button to set the value as 500 and then in the *balances* button add *from* address and click on *call* button, will show the **decoded output** as



```
decoded output          {
                             "0": "uint256: 500"
                        }
```

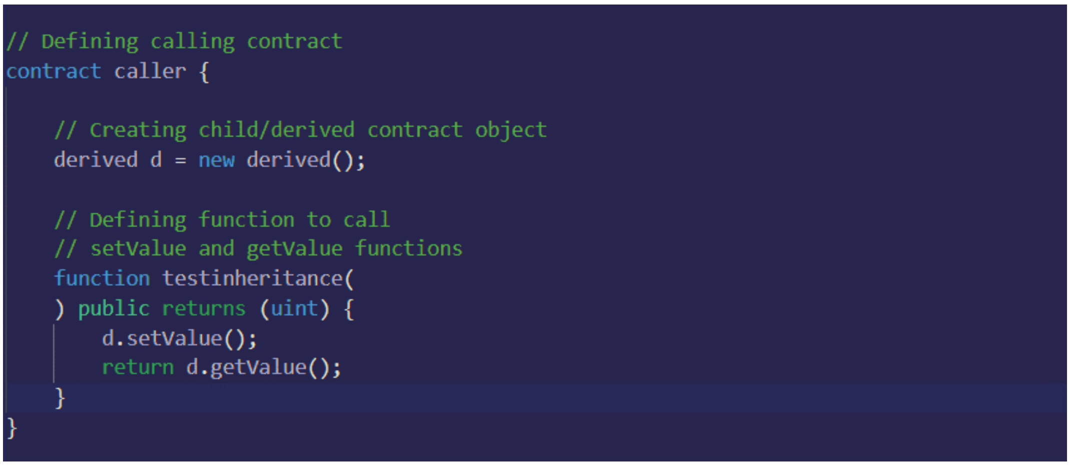
☞ Inheritance

- One of the most important features of the object-oriented programming language is inheritance.
- It is a method of extending a program's functionality by separating the code, reducing dependency, and increasing the re-usability of existing code.
- Solidity supports smart contract inheritance, allowing multiple contracts to be inherited into a single contract.
- A base contract is the contract from which other contracts inherit features, whereas a derived contract is the contract that inherits the features.
- They are simply known as parent–child contracts. In solidity programming, inheritance is restricted to public and internal modifiers.
- Solidity has the "is" keyword to inherit the base/parent contract to the derived/child contract.

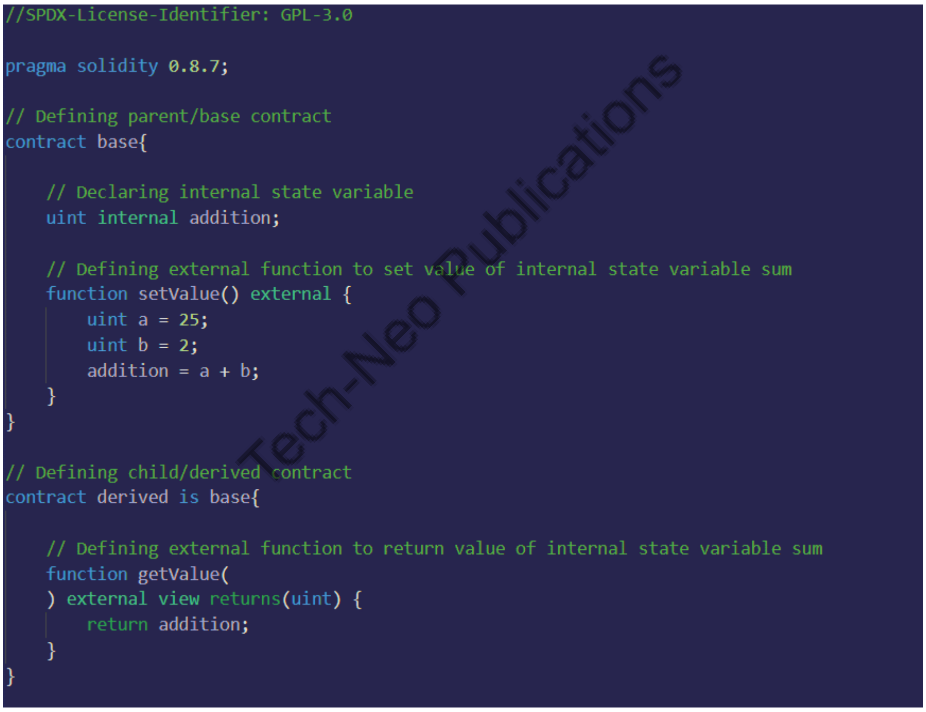
Difference types of inheritance

- **Single inheritance**
  - In this type of inheritance, the functions and variables of one parent/base contract are inherited to only one child/derived contract.

Program

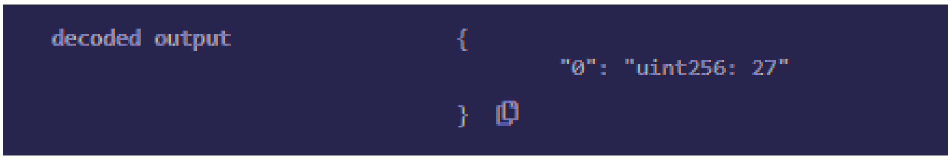


```
// Defining calling contract
contract caller {

    // Creating child/derived contract object
    derived d = new derived();

    // Defining function to call
    // setValue and getValue functions
    function testinheritance(
    ) public returns (uint) {
        d.setValue();
        return d.getValue();
    }
}
```



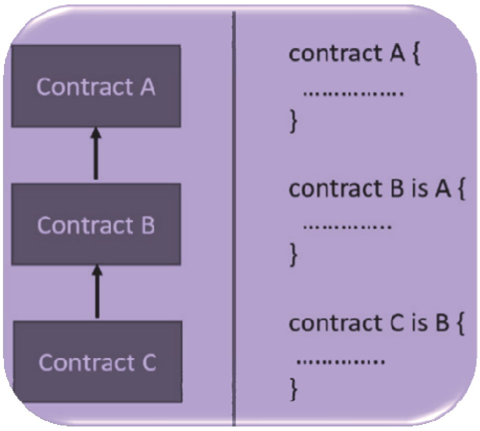
```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

// Defining parent/base contract
contract base{

    // Declaring internal state variable
    uint internal addition;

    // Defining external function to set value of internal state variable sum
    function setValue() external {
        uint a = 25;
        uint b = 2;
        addition = a + b;
    }
}

// Defining child/derived contract
contract derived is base{

    // Defining external function to return value of internal state variable sum
    function getValue(
    ) external view returns(uint) {
        return addition;
    }
}
```

Output

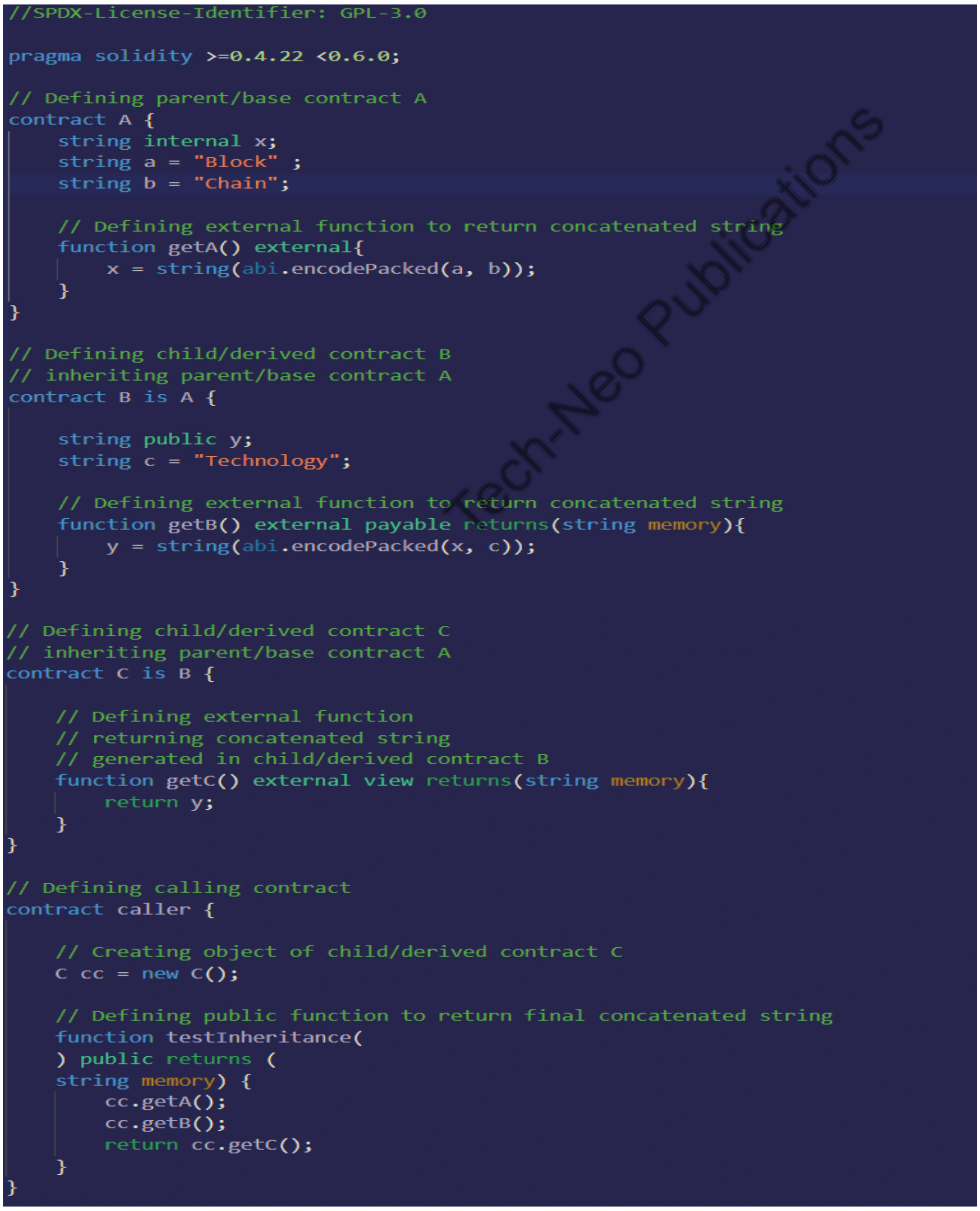


```
decoded output          {
                                "0": "uint256: 27"
                        }
```

- **Multilevel inheritance**
  - It is similar to single inheritance, but the difference is that it has levels of the parent–child relationship. The child contract derived from a parent also serves as a parent to the contract derived from it.

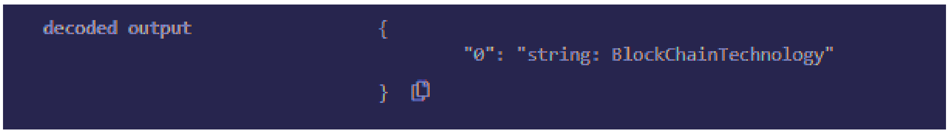


Program



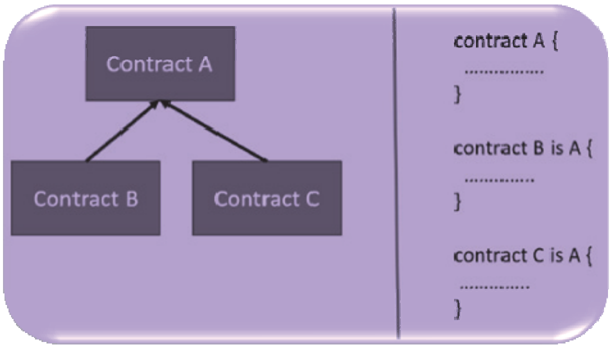
```
//SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.4.22 <0.6.0;

// Defining parent/base contract A
contract A {
    string internal x;
    string a = "Block" ;
    string b = "Chain";

    // Defining external function to return concatenated string
    function getA() external{
        x = string(abi.encodePacked(a, b));
    }
}

// Defining child/derived contract B
// inheriting parent/base contract A
contract B is A {

    string public y;
    string c = "Technology";

    // Defining external function to return concatenated string
    function getB() external payable returns(string memory){
        y = string(abi.encodePacked(x, c));
    }
}

// Defining child/derived contract C
// inheriting parent/base contract A
contract C is B {

    // Defining external function
    // returning concatenated string
    // generated in child/derived contract B
    function getC() external view returns(string memory){
        return y;
    }
}

// Defining calling contract
contract caller {

    // Creating object of child/derived contract C
    C cc = new C();

    // Defining public function to return final concatenated string
    function testInheritance(
    ) public returns (
    string memory) {
        cc.getA();
        cc.getB();
        return cc.getC();
    }
}
```

Output



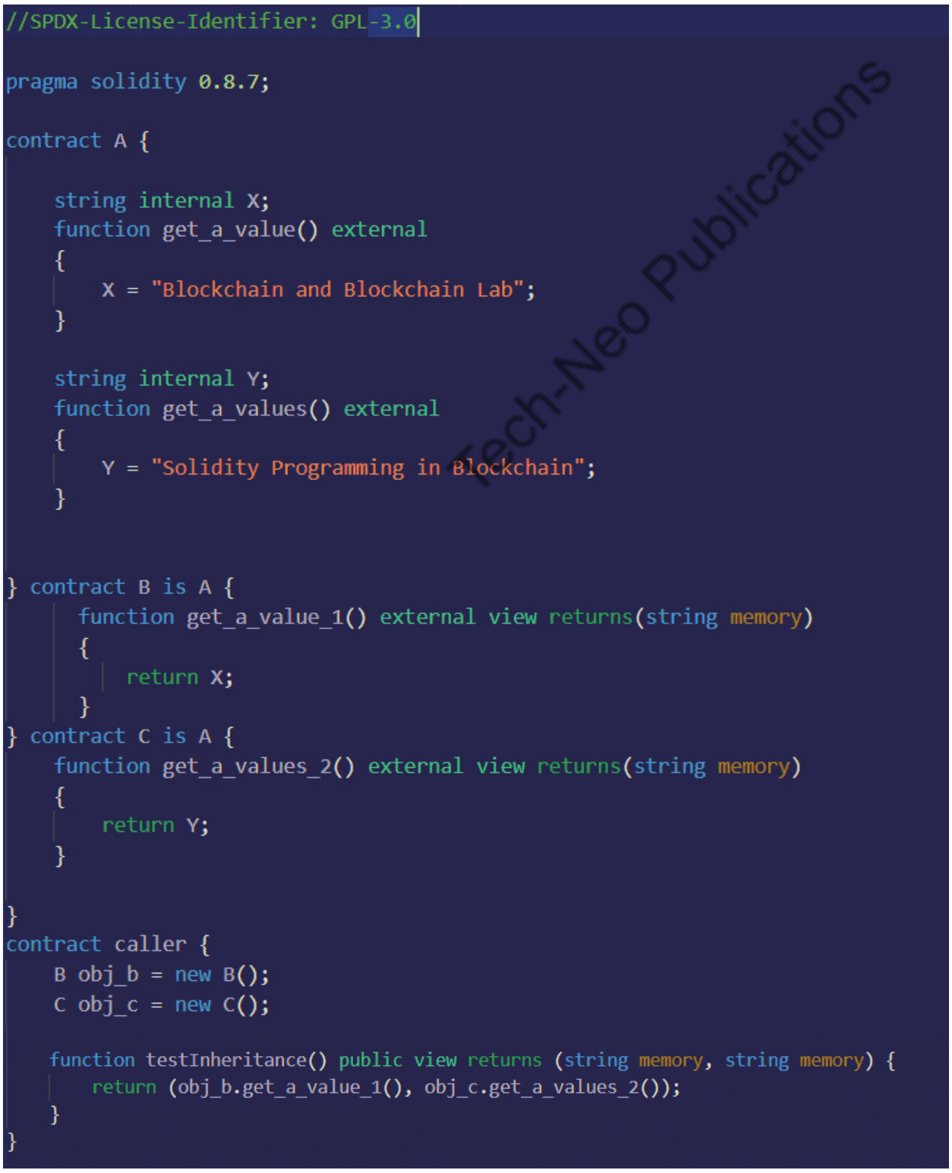
```
decoded output            {
                              "0": "string: BlockChainTechnology"
                          }
```

- **Hierarchical inheritance**
  - In this type of inheritance, a parent contract has more than one child contract. It is commonly used when the same functionality must be used in multiple places.



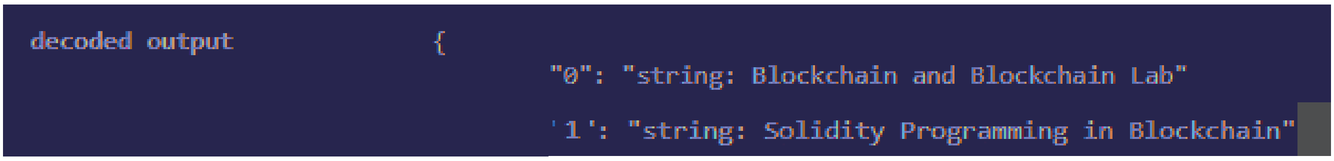
```
contract A {
  ................
}

contract B is A {
  ..............
}

contract C is A {
  ..............
}
```

Program



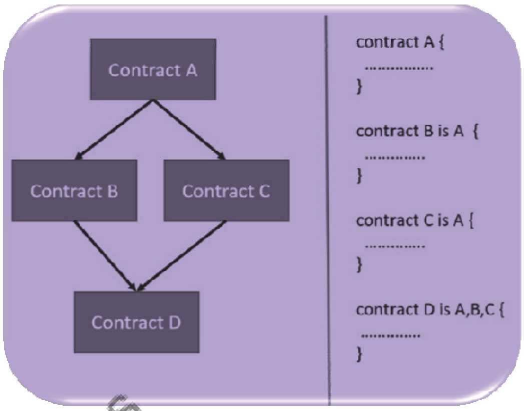
```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract A {

    string internal X;
    function get_a_value() external
    {
        X = "Blockchain and Blockchain Lab";
    }

    string internal Y;
    function get_a_values() external
    {
        Y = "Solidity Programming in Blockchain";
    }


} contract B is A {
      function get_a_value_1() external view returns(string memory)
      {
          return X;
      }
} contract C is A {
    function get_a_values_2() external view returns(string memory)
    {
        return Y;
    }

}
contract caller {
    B obj_b = new B();
    C obj_c = new C();

    function testInheritance() public view returns (string memory, string memory) {
        return (obj_b.get_a_value_1(), obj_c.get_a_values_2());
    }
}
```

Output



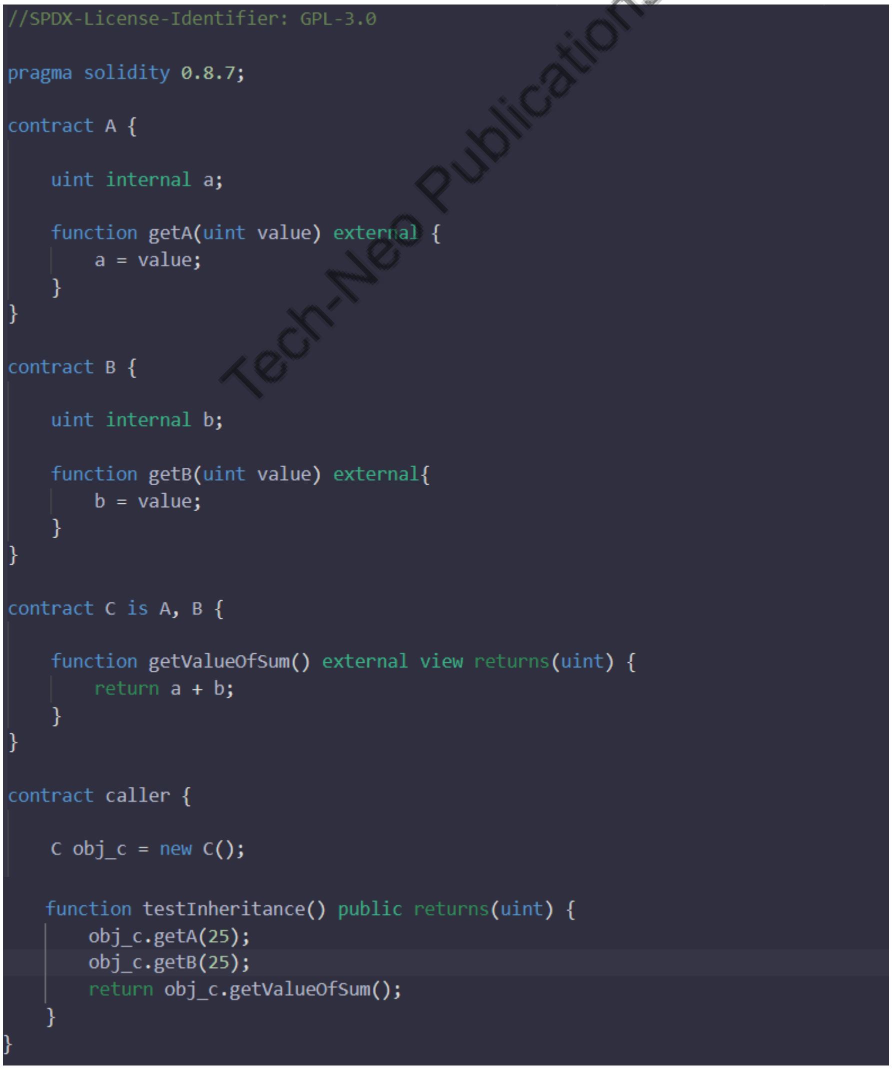
```
decoded output               {
                                   "0": "string: Blockchain and Blockchain Lab"
                                   '1': "string: Solidity Programming in Blockchain"
```

- **Multiple inheritance**
  - In this type of inheritance, a single contract can be inherited from multiple contracts.
  - A parent contract can have multiple children, whereas a child contract can have multiple parents.



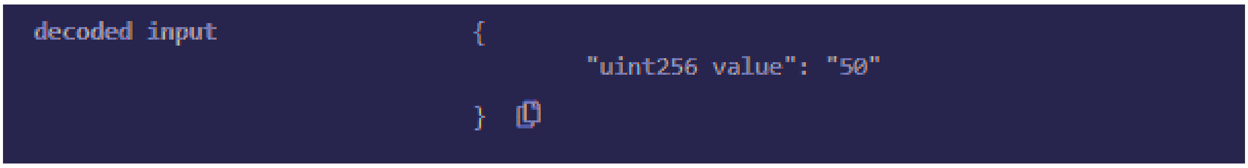
```
contract A {
  ..............
}
contract B is A {
  ..............
}
contract C is A {
  ..............
}
contract D is A,B,C {
  ..............
}
```

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract A {

    uint internal a;

    function getA(uint value) external {
        a = value;
    }
}

contract B {

    uint internal b;

    function getB(uint value) external{
        b = value;
    }
}

contract C is A, B {

    function getValueOfSum() external view returns(uint) {
        return a + b;
    }
}

contract caller {

    C obj_c = new C();

   function testInheritance() public returns(uint) {
       obj_c.getA(25);
       obj_c.getB(25);
       return obj_c.getValueOfSum();
   }
}
```

Output



```
decoded input          {
                           "uint256 value": "50"
                       }
```

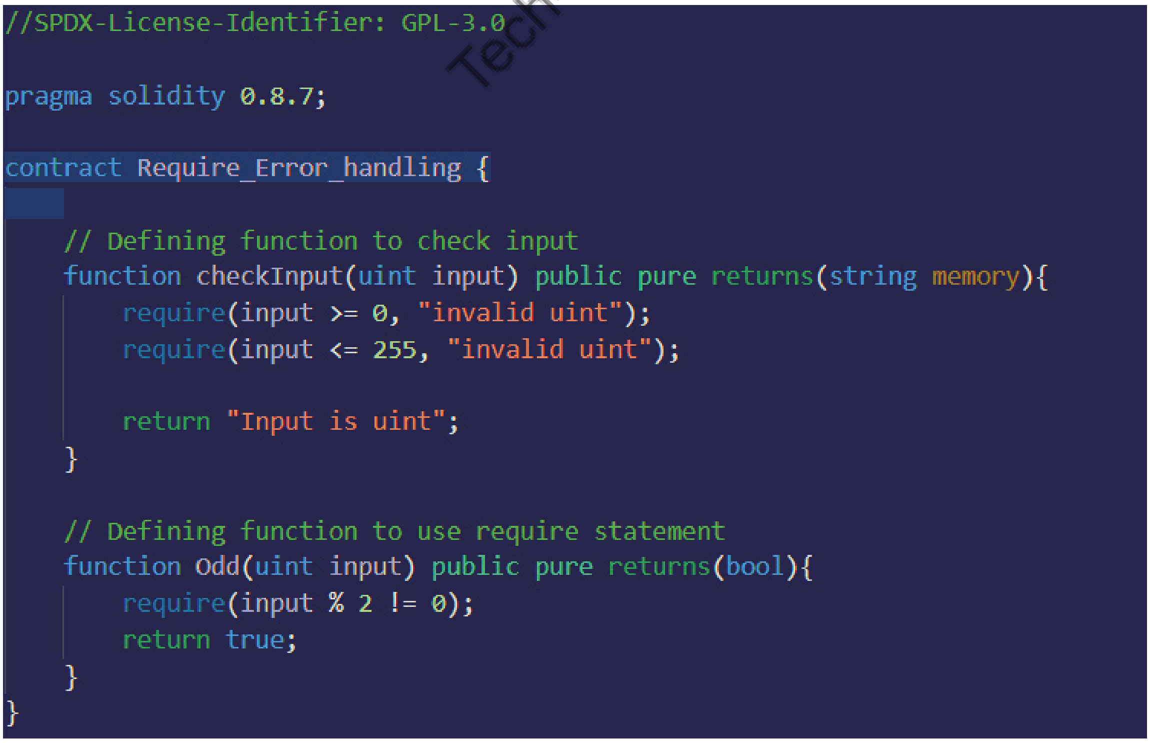
☞ Error Handling

- Solidity has a plethora of error handling functions.
- Errors can occur during compile or runtime.
- Solidity is compiled to byte code, and a syntax error check occurs at compile time, whereas runtime errors are difficult to detect and occur primarily during contract execution.
- Out-of-gas error, data type overflow error, divide by zero error, array-out-of-index error, and so on are examples of runtime errors.
- Solidity had a single throw statement until version 4.10; so to handle errors, multiple if...else statements must be implemented for checking the values and throwing errors, which consumes more gas.
- After version 4.10, new error handling constructs *require, assert, and revert* statements were added, and the throw function was made absolute.

Require

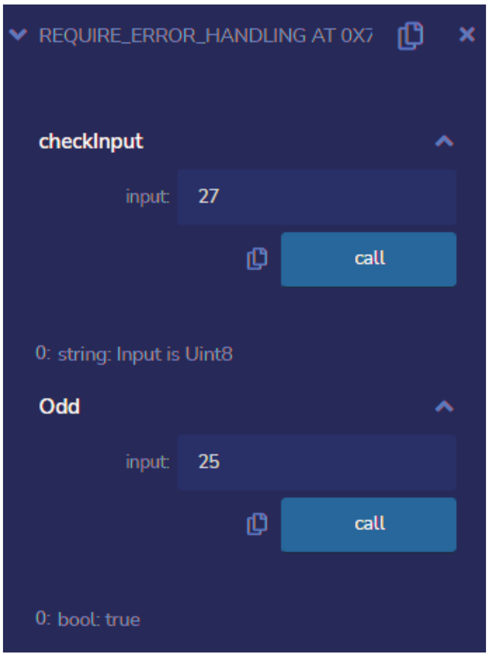
- The 'require' statement declares the prerequisites for running a function, i.e., the constraints that must be met before executing a code. It takes a single argument and after evaluation returns a Boolean value.
- It also has a custom string message option. If false, then an exception is thrown, and the execution is terminated. The unused gas is returned to the caller, and the state is reset to its original setting.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Require_Error_handling {

    // Defining function to check input
    function checkInput(uint input) public pure returns(string memory){
        require(input >= 0, "invalid uint");
        require(input <= 255, "invalid uint");

        return "Input is uint";
    }

    // Defining function to use require statement
    function Odd(uint input) public pure returns(bool){
        require(input % 2 != 0);
        return true;
    }
}
```

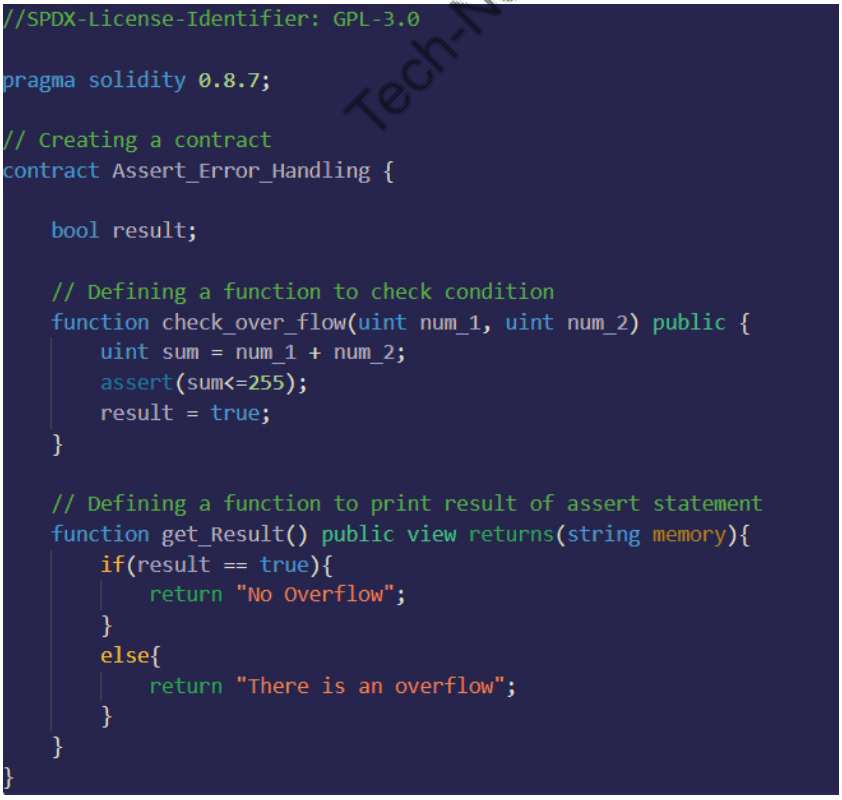
Output



```
REQUIRE_ERROR_HANDLING AT 0X7

checkInput
    input: 27
    call

0: string: Input is Uint8

Odd
    input: 25
    call

0: bool: true
```

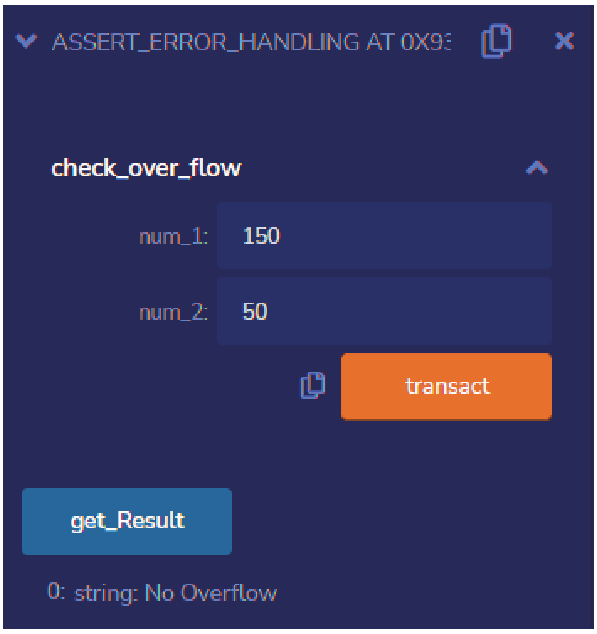
- **Assert :** Its syntax is similar to the ‘require’ statement and returns a Boolean value. Depending on the return value, a programme will either continue or throw an exception. Instead of returning unused gas, the assert statement consumes the entire gas supply and then returns the state to its original state. Before executing the contract, *assert* is used to validate the current state and function conditions.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

// Creating a contract
contract Assert_Error_Handling {

    bool result;

    // Defining a function to check condition
    function check_over_flow(uint num_1, uint num_2) public {
        uint sum = num_1 + num_2;
        assert(sum<=255);
        result = true;
    }

    // Defining a function to print result of assert statement
    function get_Result() public view returns(string memory){
        if(result == true){
            return "No Overflow";
        }
        else{
            return "There is an overflow";
        }
    }
}
```

Output



num_1: 150

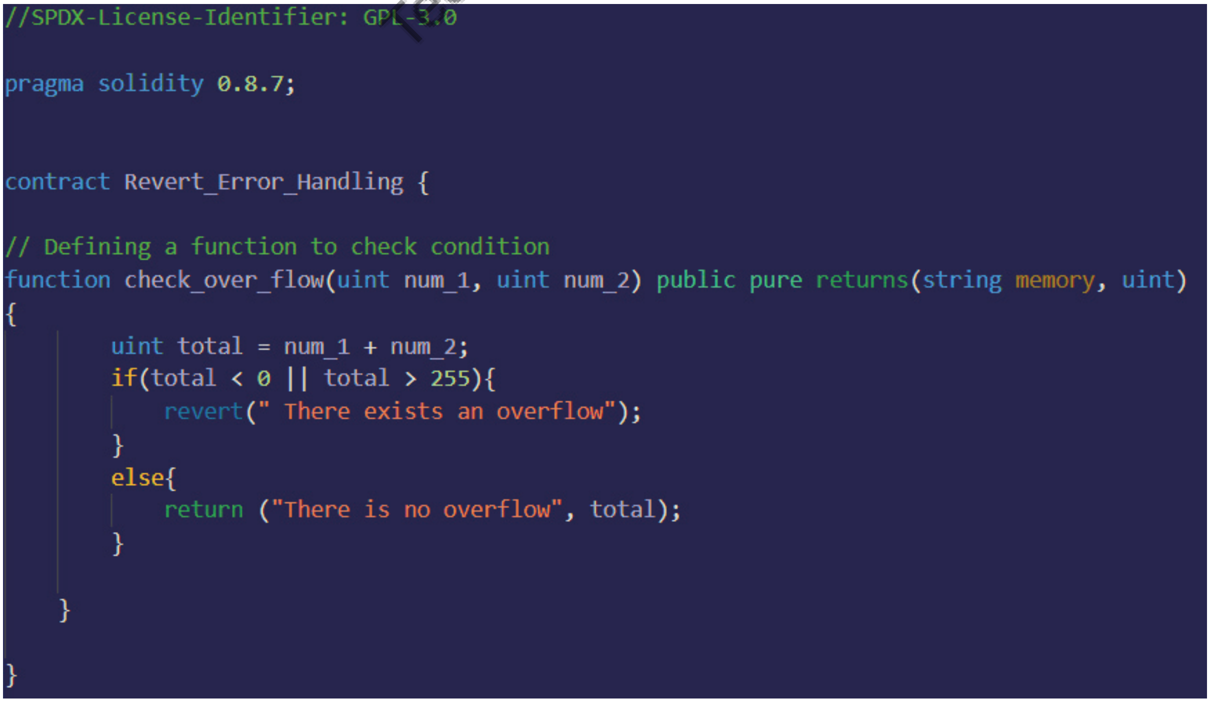
num_2: 50

```
0: string: No Overflow
```

Revert

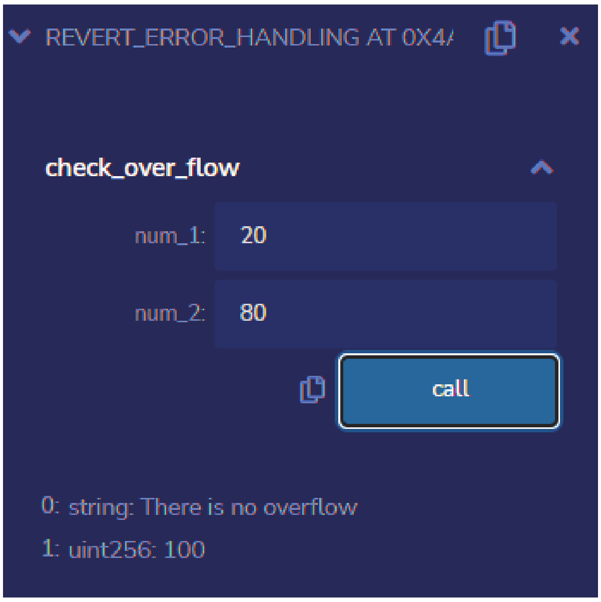
- This is comparable to the 'require' statement. It does not evaluate any condition or rely on any state or statement. It is used to throw exceptions, show errors, and revert a function call. This statement includes a string message that describes the problem with the exception's information.
- A revert statement implies that an exception is thrown, the unused gas is returned, and the state is reset to its original state. Revert is used to handle the same types of exceptions that require does, but with slightly more complex logic.

Program



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;


contract Revert_Error_Handling {

// Defining a function to check condition
function check_over_flow(uint num_1, uint num_2) public pure returns(string memory, uint)
{
        uint total = num_1 + num_2;
        if(total < 0 || total > 255){
            revert(" There exists an overflow");
        }
        else{
            return ("There is no overflow", total);
        }

    }

}
```

Output



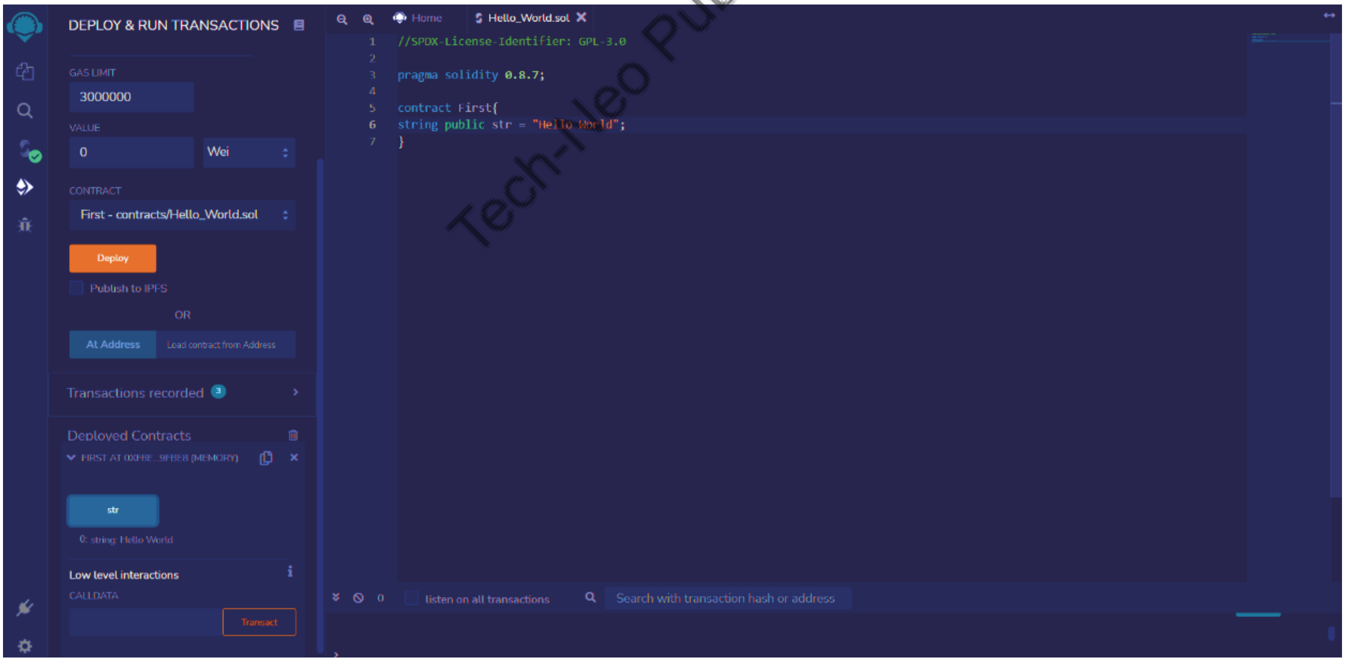
num_1: 20

num_2: 80

```
0: string: There is no overflow
1: uint256: 100
```

☞ Sample programs in solidity programming

1. Print Hello World in Solidity



DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 3
Deployed Contracts
FIRST AT 0XF8E...9FBE8 (MEMORY)
str
Low level interactions
Transact
Home
Hello_World.sol
listen on all transactions
Search with transaction hash or address

GAS LIMIT: 3000000

VALUE: 0 Wei

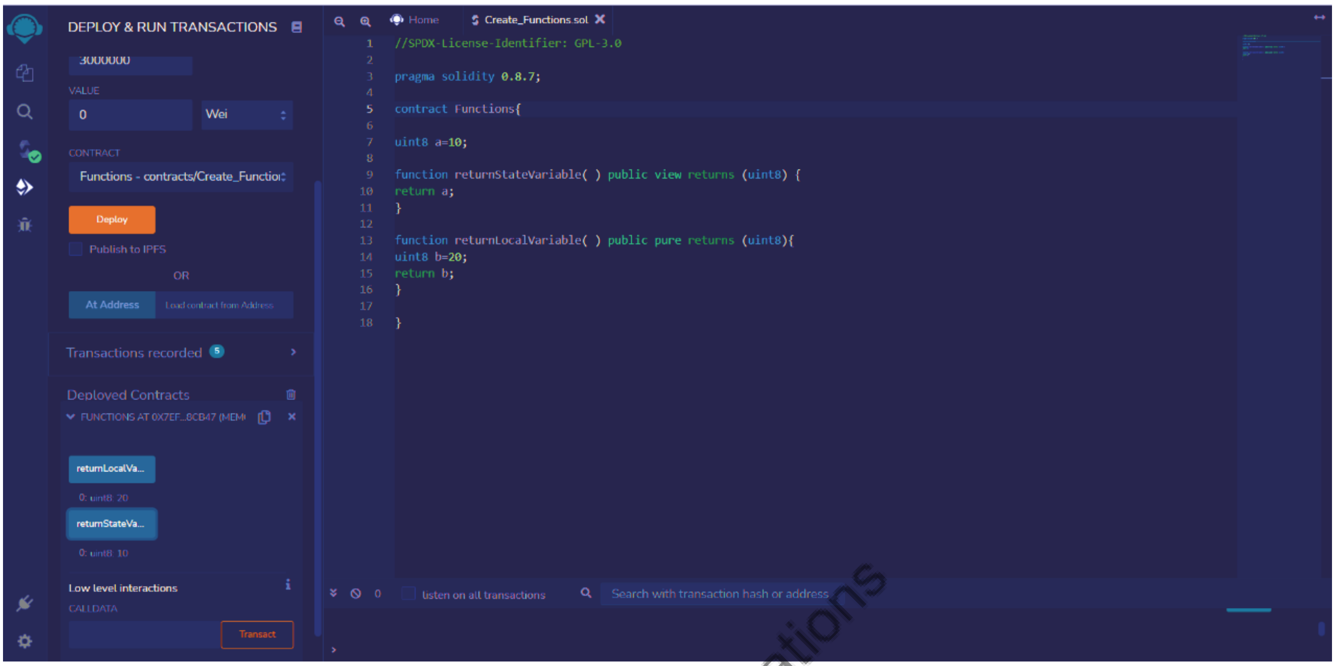
CONTRACT: First - contracts/Hello_World.sol

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract First{
string public str = "Hello World";
}
```

CALLDATA

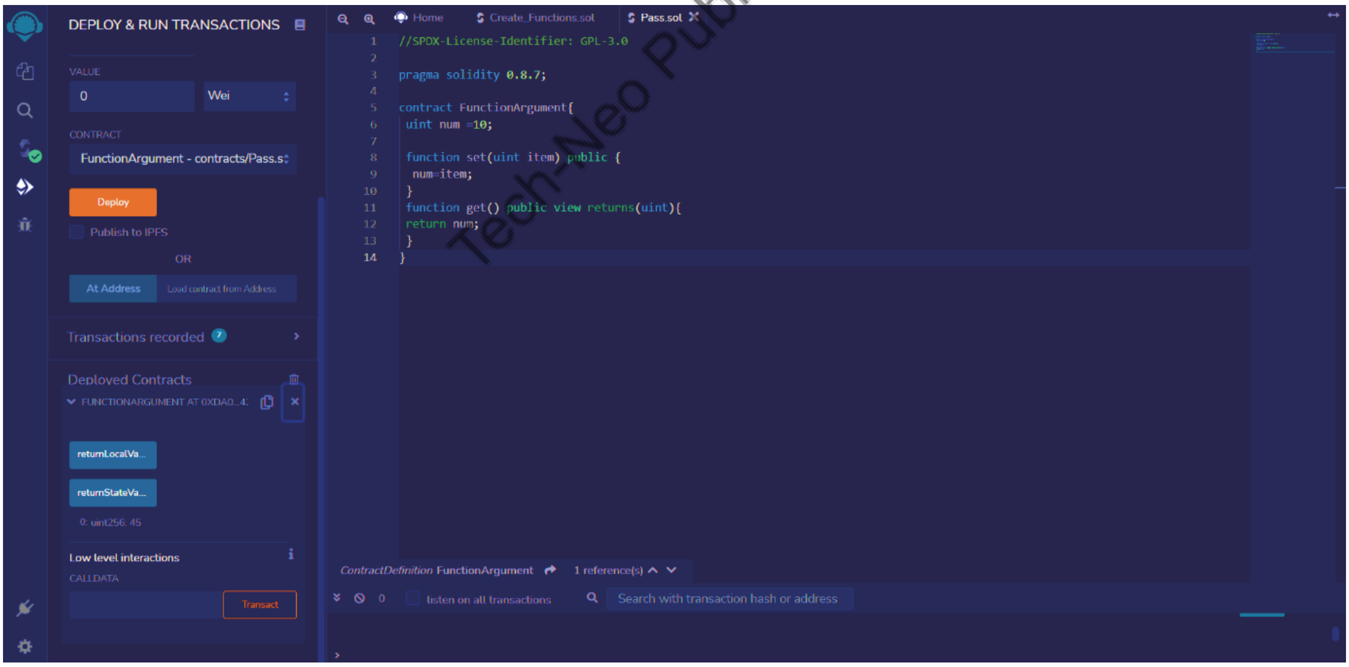
```
0: string: Hello World
```

2. Create functions



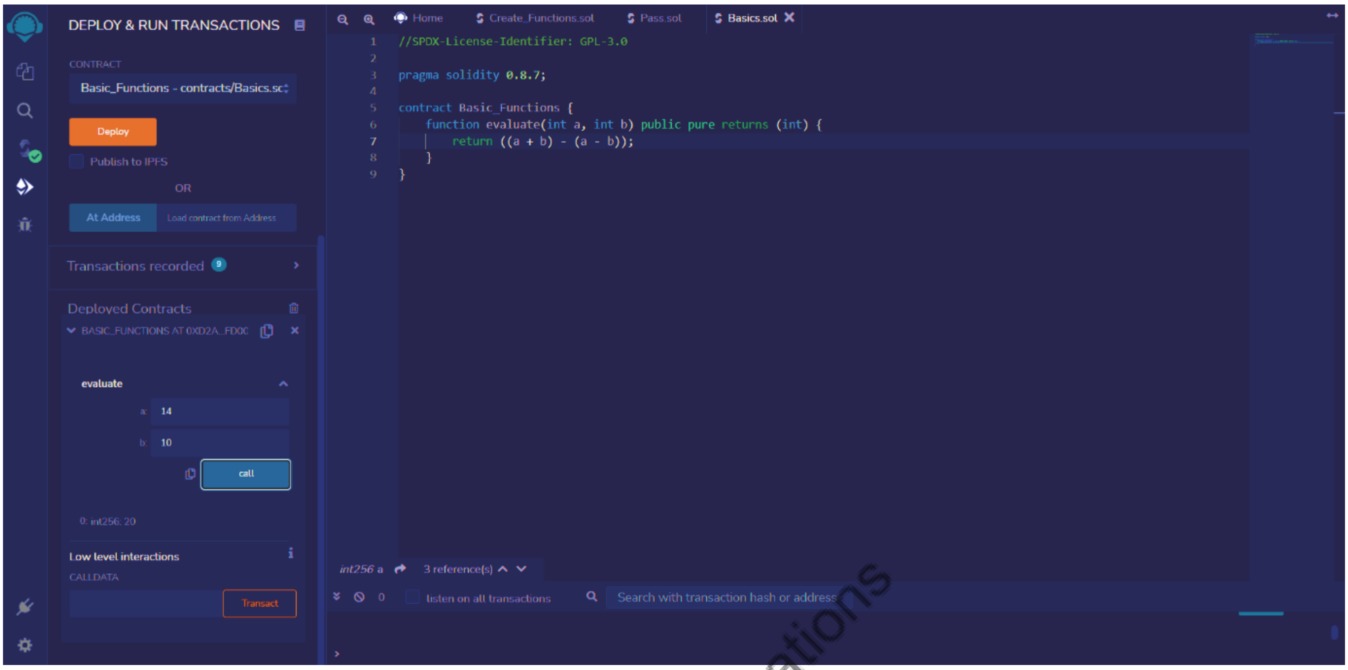
```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Functions{

uint8 a=10;

function returnStateVariable( ) public view returns (uint8) {
return a;
}

function returnLocalVariable( ) public pure returns (uint8){
uint8 b=20;
return b;
}

}
```

3. Pass an argument to function



```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract FunctionArgument{
 uint num =10;

 function set(uint item) public {
  num=item;
 }
 function get() public view returns(uint){
 return num;
 }
}
```

4. Basic solidity programming : Subtract the difference between two numbers from the sum of two numbers.



CONTRACT: Basic_Functions - contracts/Basics.sol

evaluate

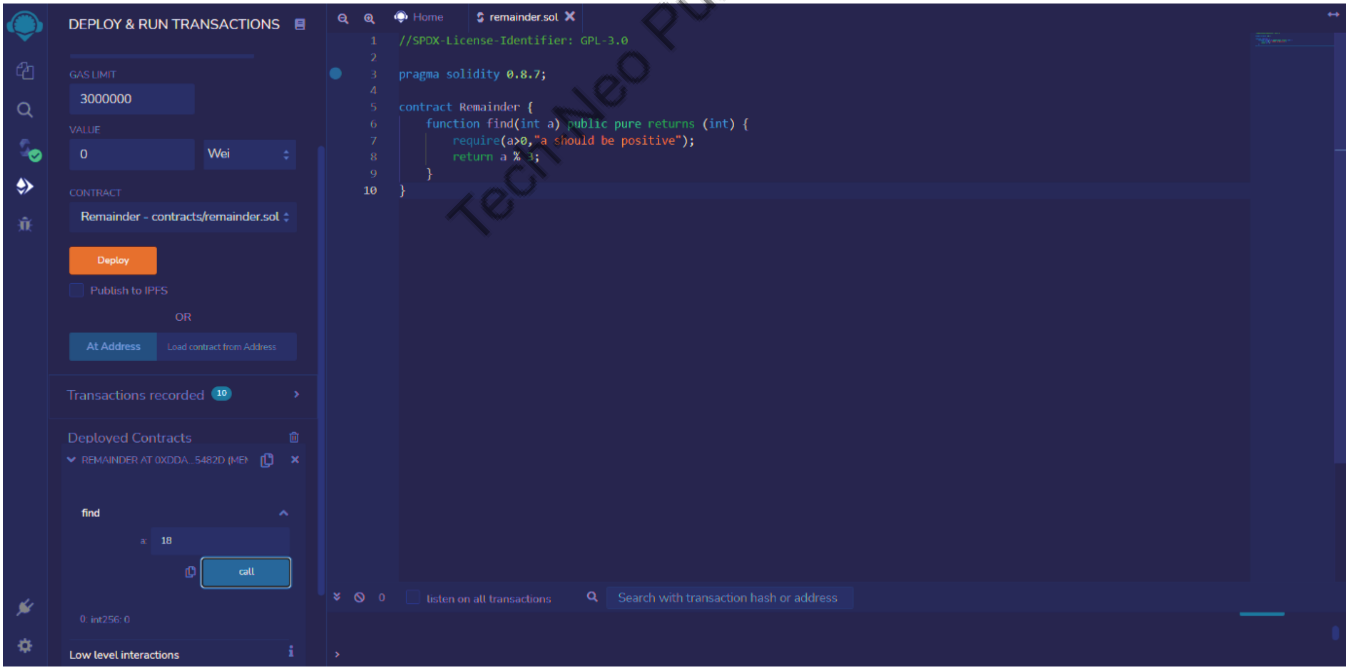
a: 14

b: 10

```
0: int256: 20
```

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Basic_Functions {
    function evaluate(int a, int b) public pure returns (int) {
        return ((a + b) - (a - b));
    }
}
```

5. Find remainder


DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 10
Deployed Contracts
REMAINDER AT 0XDDA...5482D (MEM
call
Low level interactions
Home
remainder.sol
listen on all transactions
Search with transaction hash or address

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Remainder - contracts/remainder.sol

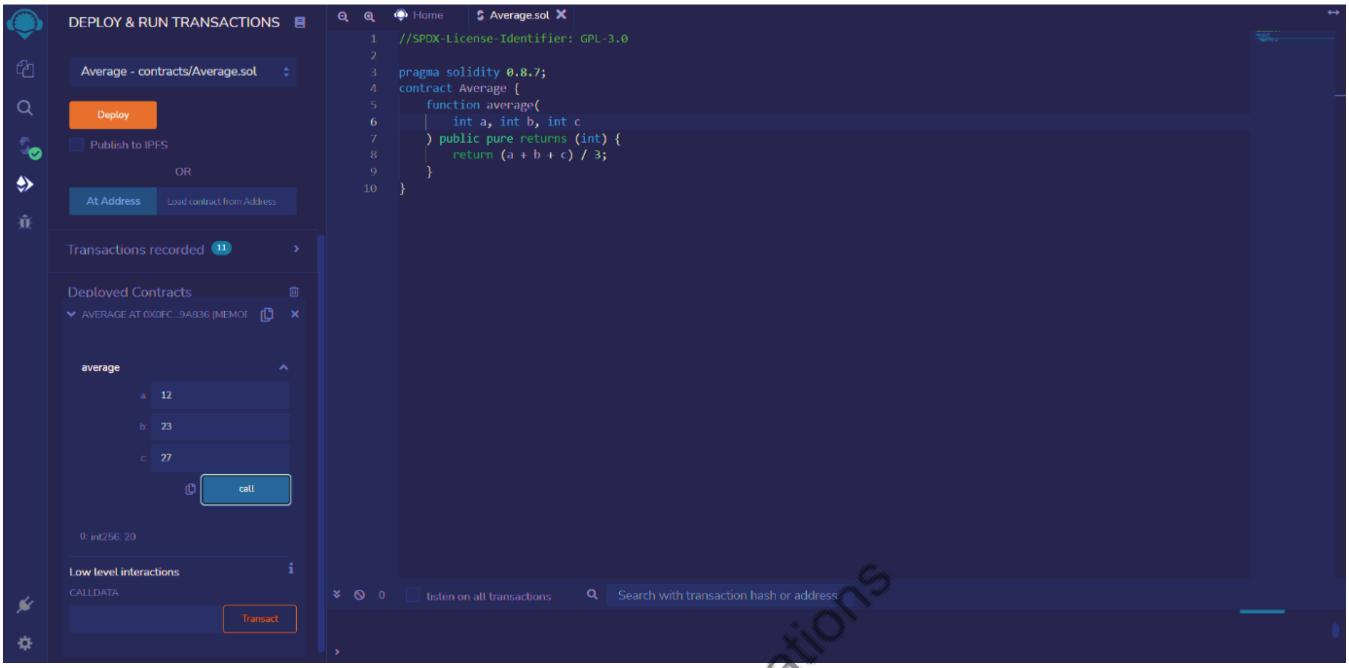
find

a: 18

```
0: int256: 0
```

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Remainder {
    function find(int a) public pure returns (int) {
        require(a>0,"a should be positive");
        return a % 3;
    }
}
```

6. Find average of three numbers


DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 11
Deployed Contracts
AVERAGE AT 0X0FC...9A836 (MEMOF
call
Low level interactions
CALLDATA
Transact
Home
Average.sol
listen on all transactions
Search with transaction hash or address

Average - contracts/Average.sol

average

a: 12

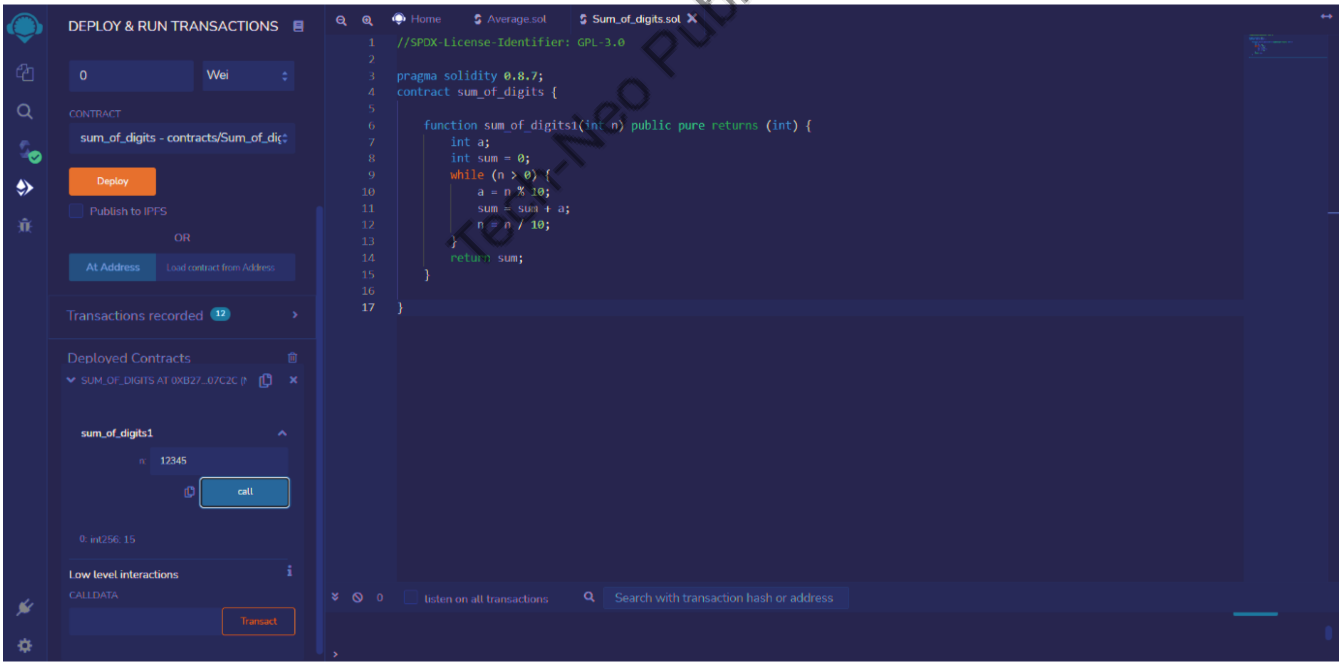
b: 23

c: 27

0: int256: 20

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract Average {
    function average(
        int a, int b, int c
    ) public pure returns (int) {
        return (a + b + c) / 3;
    }
}
```

7. Find the sum of digits



0 Wei

sum_of_digits - contracts/Sum_of_dig

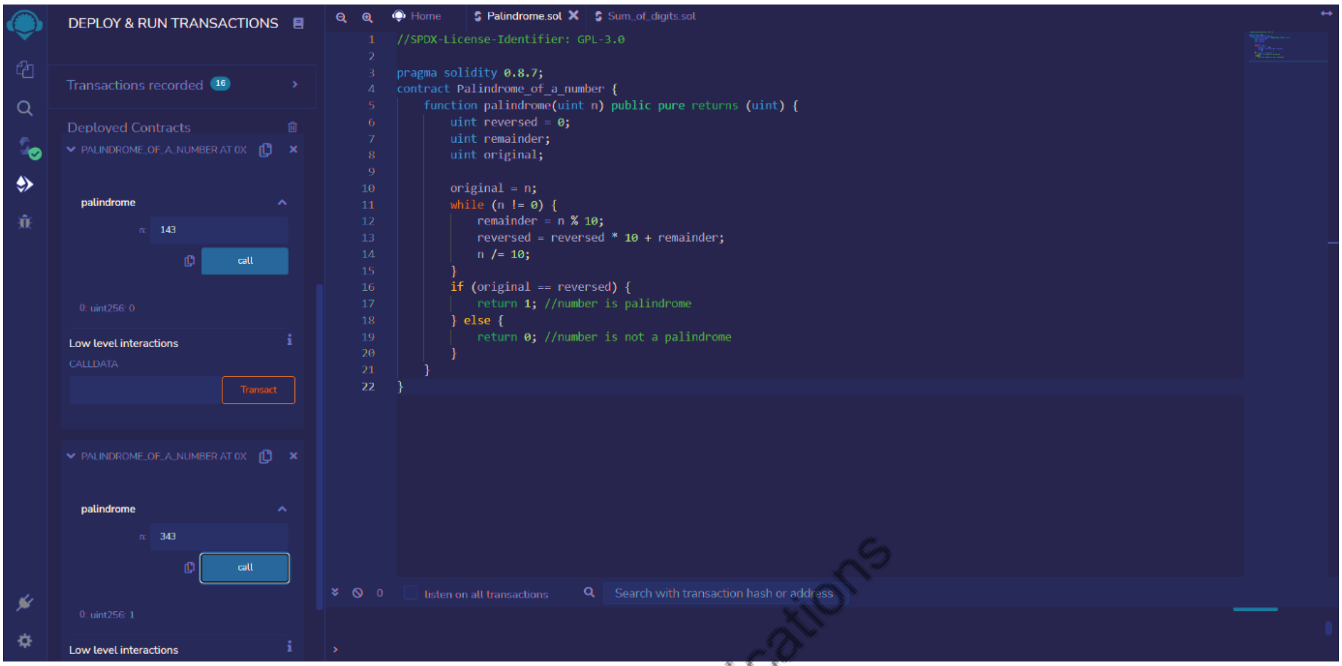
sum_of_digits1

n: 12345

0: int256: 15

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract sum_of_digits {

    function sum_of_digits1(int n) public pure returns (int) {
        int a;
        int sum = 0;
        while (n > 0) {
            a = n % 10;
            sum = sum + a;
            n = n / 10;
        }
        return sum;
    }

}
```

8. Palindrome of a number

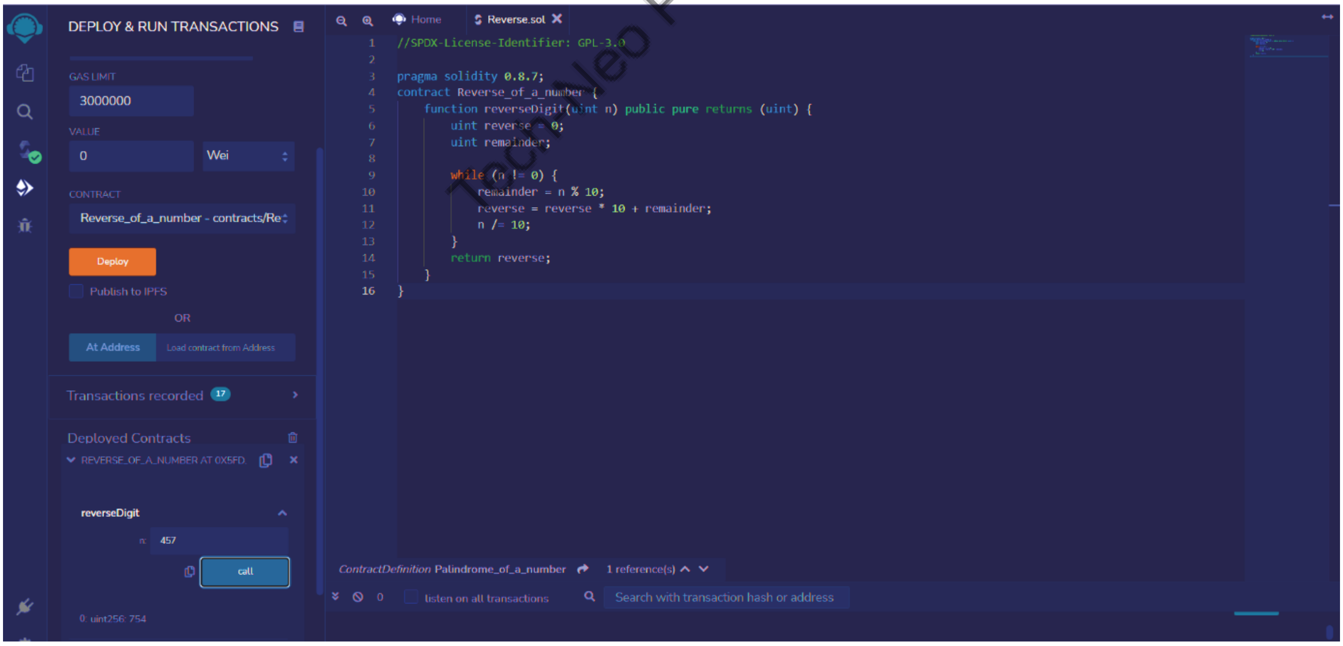

DEPLOY & RUN TRANSACTIONS
Home
Palindrome.sol
Sum_of_digits.sol
Transactions recorded 16
Deployed Contracts
PALINDROME_OF_A_NUMBER AT 0X
Low level interactions
CALLDATA
Transact
listen on all transactions
Search with transaction hash or address

palindrome
n: 143
call
0: uint256: 0

palindrome
n: 343
call
0: uint256: 1

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract Palindrome_of_a_number {
    function palindrome(uint n) public pure returns (uint) {
        uint reversed = 0;
        uint remainder;
        uint original;

        original = n;
        while (n != 0) {
            remainder = n % 10;
            reversed = reversed * 10 + remainder;
            n /= 10;
        }
        if (original == reversed) {
            return 1; //number is palindrome
        } else {
            return 0; //number is not a palindrome
        }
    }
}
```

9. Reverse of a number

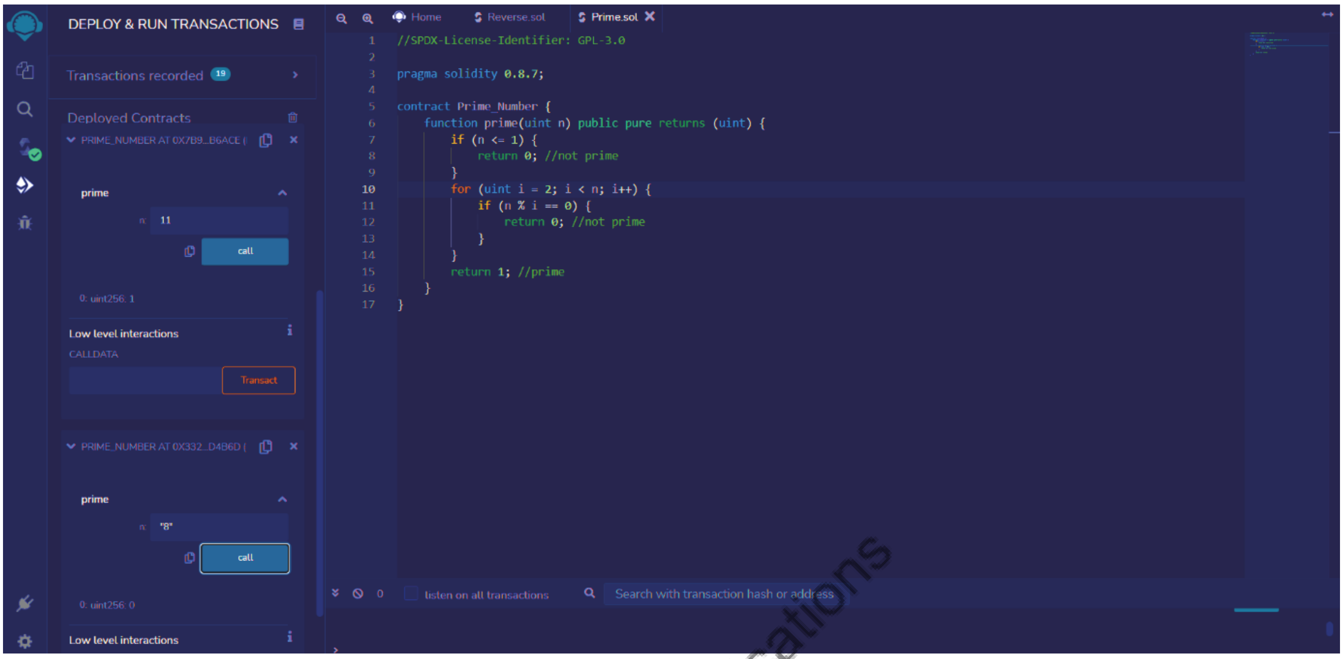

DEPLOY & RUN TRANSACTIONS
Home
Reverse.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 17
Deployed Contracts
REVERSE_OF_A_NUMBER AT 0X5FD
ContractDefinition Palindrome_of_a_number
1 reference(s)
listen on all transactions
Search with transaction hash or address

GAS LIMIT: 3000000
VALUE: 0 Wei
CONTRACT: Reverse_of_a_number - contracts/Re

reverseDigit
n: 457
call
0: uint256: 754

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract Reverse_of_a_number {
    function reverseDigit(uint n) public pure returns (uint) {
        uint reverse = 0;
        uint remainder;

        while (n != 0) {
            remainder = n % 10;
            reverse = reverse * 10 + remainder;
            n /= 10;
        }
        return reverse;
    }
}
```

10. To check whether a number is prime or not


DEPLOY & RUN TRANSACTIONS
Transactions recorded 19
Deployed Contracts
PRIME_NUMBER AT 0X7B9...B6ACE (
prime
call
Low level interactions
CALLDATA
Transact
PRIME_NUMBER AT 0X332...D4B6D (
prime
call
Low level interactions
Home
Reverse.sol
Prime.sol
listen on all transactions
Search with transaction hash or address

n: 11

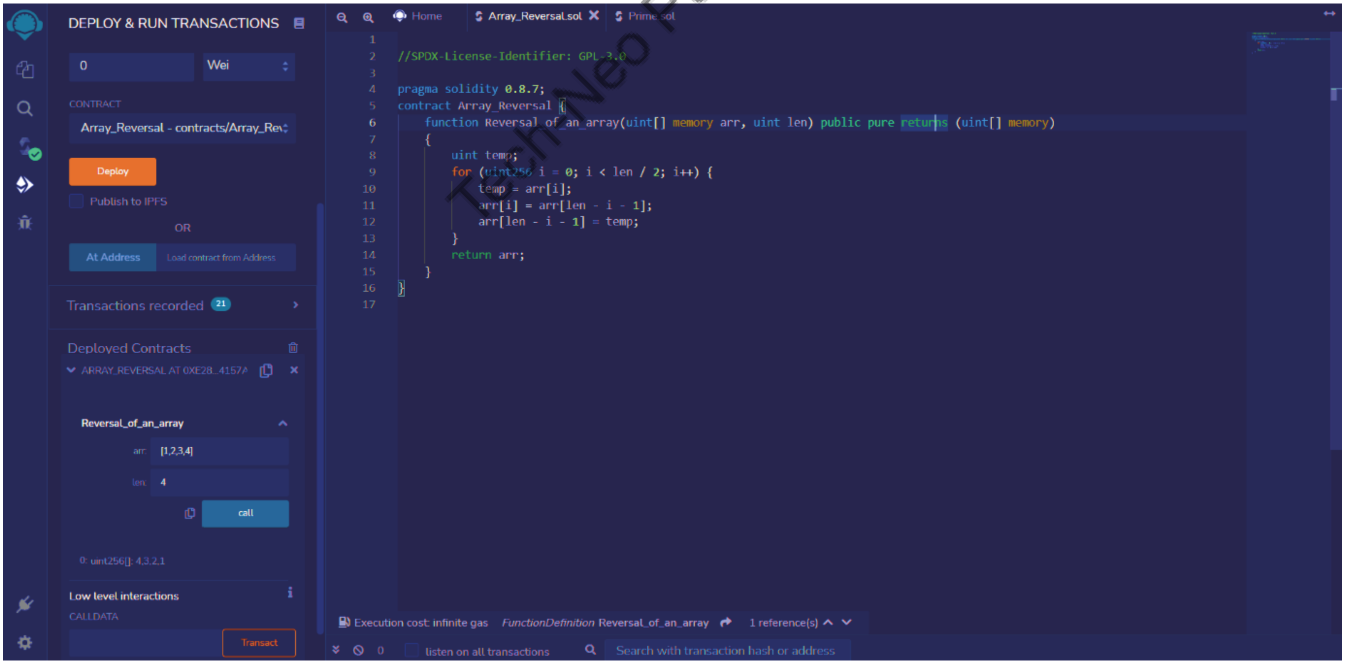
0: uint256: 1

n: "8"

0: uint256: 0

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Prime_Number {
    function prime(uint n) public pure returns (uint) {
        if (n <= 1) {
            return 0; //not prime
        }
        for (uint i = 2; i < n; i++) {
            if (n % i == 0) {
                return 0; //not prime
            }
        }
        return 1; //prime
    }
}
```

11. Reversal of an array



0 Wei

Array_Reversal - contracts/Array_Rev

arr: [1,2,3,4]

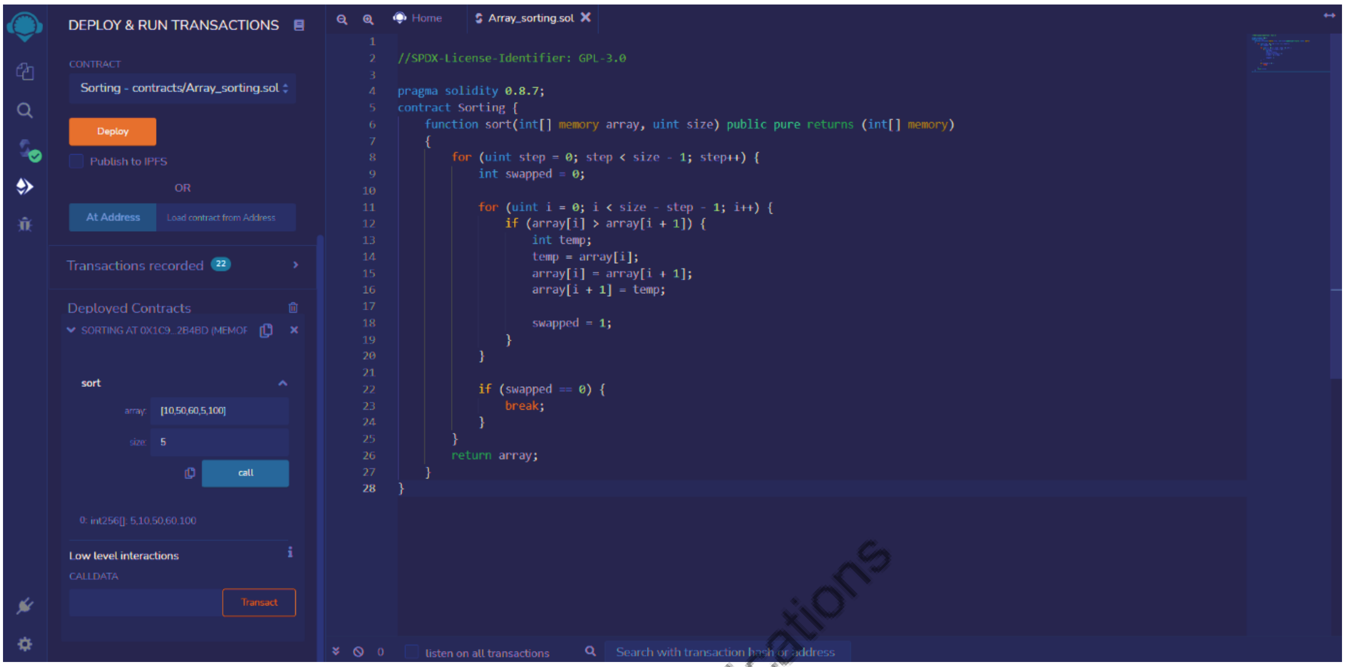
len: 4

0: uint256[]: 4,3,2,1

Execution cost: infinite gas FunctionDefinition Reversal_of_an_array 1 reference(s)

```

//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract Array_Reversal {
    function Reversal_of_an_array(uint[] memory arr, uint len) public pure returns (uint[] memory)
    {
        uint temp;
        for (uint256 i = 0; i < len / 2; i++) {
            temp = arr[i];
            arr[i] = arr[len - i - 1];
            arr[len - i - 1] = temp;
        }
        return arr;
    }
}
```

12. Sorting of an array



Sorting - contracts/Array_sorting.sol

SORTING AT 0X1C9...2B4BD (MEMOF

sort

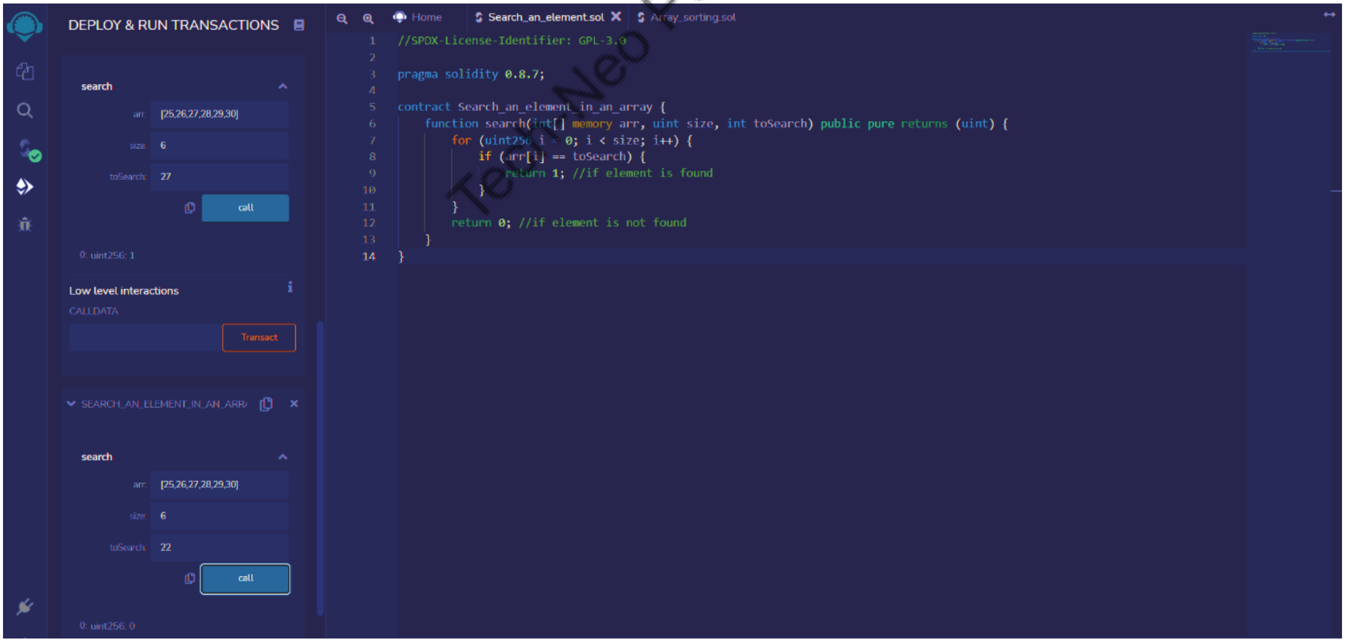
array: [10,50,60,5,100]

size: 5

0: int256[]: 5,10,50,60,100

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract Sorting {
    function sort(int[] memory array, uint size) public pure returns (int[] memory)
    {
        for (uint step = 0; step < size - 1; step++) {
            int swapped = 0;

            for (uint i = 0; i < size - step - 1; i++) {
                if (array[i] > array[i + 1]) {
                    int temp;
                    temp = array[i];
                    array[i] = array[i + 1];
                    array[i + 1] = temp;

                    swapped = 1;
                }
            }

            if (swapped == 0) {
                break;
            }
        }
        return array;
    }
}
```

13. To search an element in an array



search

arr: [25,26,27,28,29,30]

size: 6

toSearch: 27

0: uint256: 1

SEARCH_AN_ELEMENT_IN_AN_ARR/

search

arr: [25,26,27,28,29,30]

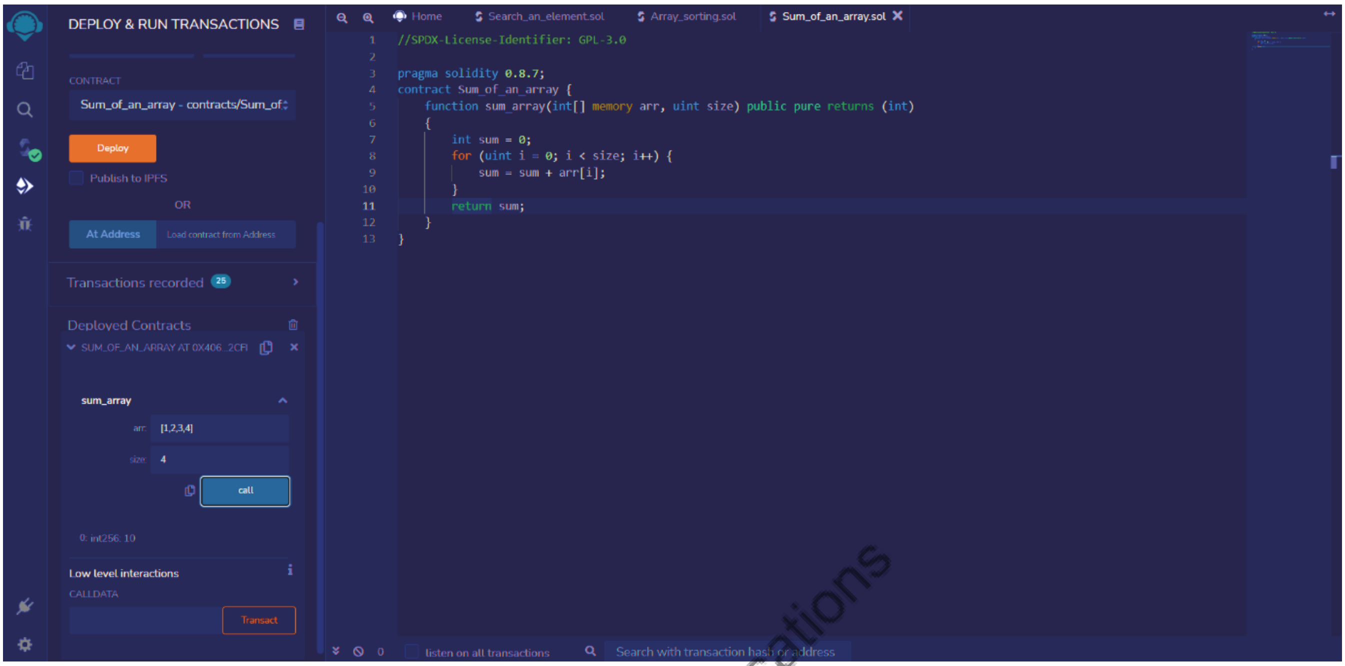
size: 6

toSearch: 22

0: uint256: 0

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Search_an_element_in_an_array {
    function search(int[] memory arr, uint size, int toSearch) public pure returns (uint) {
        for (uint256 i = 0; i < size; i++) {
            if (arr[i] == toSearch) {
                return 1; //if element is found
            }
        }
        return 0; //if element is not found
    }
}
```

14. Sum of an array



CONTRACT
Sum_of_an_array - contracts/Sum_of_

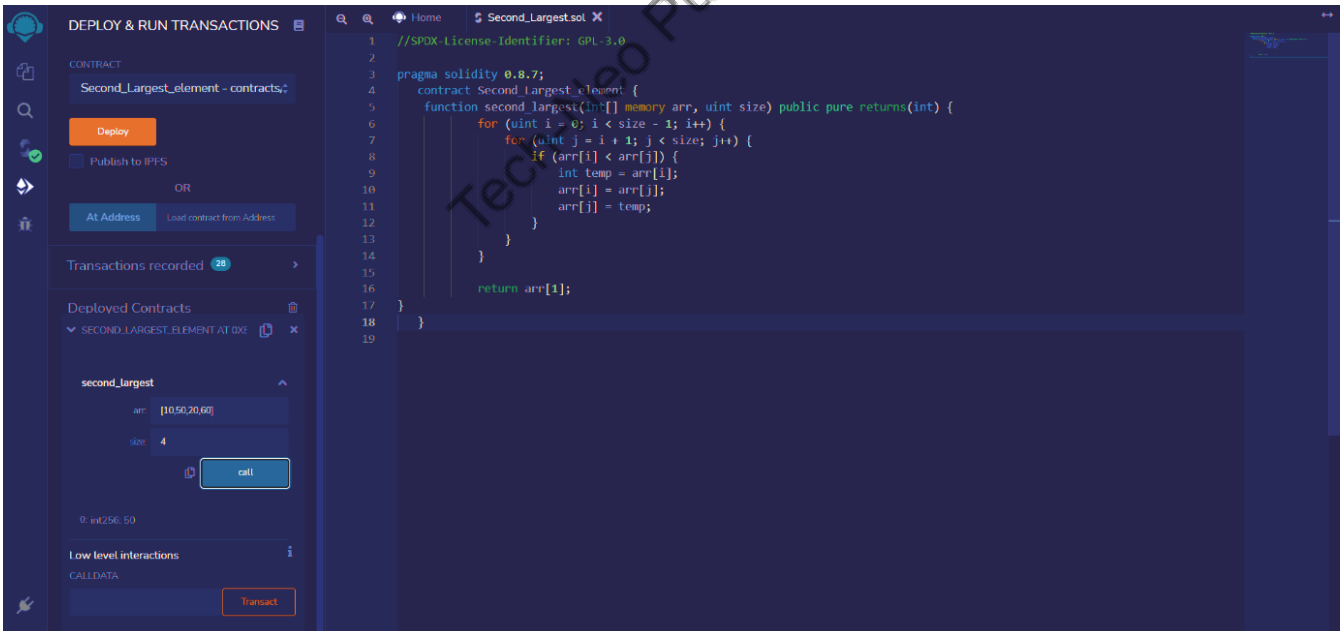
SUM_OF_AN_ARRAY AT 0X406...2CF8

sum_array
arr: [1,2,3,4]
size: 4

0: int256: 10

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
contract Sum_of_an_array {
    function sum_array(int[] memory arr, uint size) public pure returns (int)
    {
        int sum = 0;
        for (uint i = 0; i < size; i++) {
            sum = sum + arr[i];
        }
        return sum;
    }
}
```

15. To find the second largest element in an array



CONTRACT
Second_Largest_element - contracts/

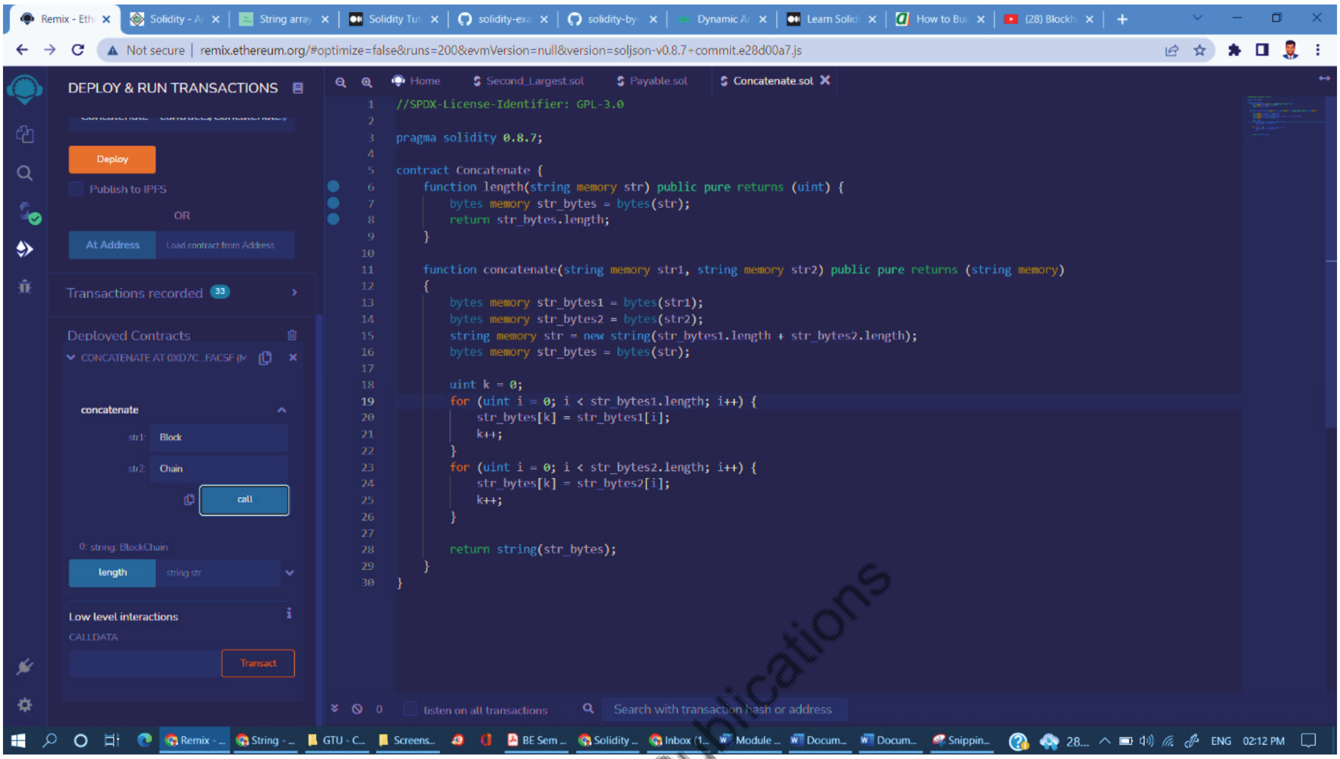
SECOND_LARGEST_ELEMENT AT 0XE

second_largest
arr: [10,50,20,60]
size: 4

0: int256: 50

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;
   contract Second_Largest_element {
    function second_largest(int[] memory arr, uint size) public pure returns(int) {
            for (uint i = 0; i < size - 1; i++) {
                for (uint j = i + 1; j < size; j++) {
                    if (arr[i] < arr[j]) {
                        int temp = arr[i];
                        arr[i] = arr[j];
                        arr[j] = temp;
                    }
                }
            }

            return arr[1];
}
   }
```

16. To concatenate two strings


remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 33
Deployed Contracts
CONCATENATE AT 0XD7C...FAC5F
concatenate
call
length
Low level interactions
CALLDATA
Transact
Home
Second_Largest.sol
Payable.sol
Concatenate.sol
listen on all transactions
Search with transaction hash or address

str1: Block

str2: Chain

0: string: BlockChain

length: string str

```
//SPDX-License-Identifier: GPL-3.0

pragma solidity 0.8.7;

contract Concatenate {
    function length(string memory str) public pure returns (uint) {
        bytes memory str_bytes = bytes(str);
        return str_bytes.length;
    }

    function concatenate(string memory str1, string memory str2) public pure returns (string memory)
    {
        bytes memory str_bytes1 = bytes(str1);
        bytes memory str_bytes2 = bytes(str2);
        string memory str = new string(str_bytes1.length + str_bytes2.length);
        bytes memory str_bytes = bytes(str);

        uint k = 0;
        for (uint i = 0; i < str_bytes1.length; i++) {
            str_bytes[k] = str_bytes1[i];
            k++;
        }
        for (uint i = 0; i < str_bytes2.length; i++) {
            str_bytes[k] = str_bytes2[i];
            k++;
        }

        return string(str_bytes);
    }
}
```

Try it out for yourself

1. There is a series S where the next term is the sum of pervious three terms. Given the first three terms of the series a, b, and c, respectively, you have to output the n^th^ term of the series.

   S(n) = a for n=1

   S(n) = b for n=2

   S(n) = c for n=3

   S(n) = S(n-1) + S(n-2) + S(n-3) for n>3

   Create a function n_th_term(uint n, uint a, uint b, uint c) where n is the n^th^ term to find and a, b, and c are the three terms of the series.
2. ***X raised to Y :*** Create a function power(uint x, uint y). This power() will calculate x raised to the power of y and return it.
3. Create a function even(array, length of array). This even() will take two arguments, i.e., a dynamic uint type array and length of the array. The even() will multiply each element of array with 2.
4. Create a function distinct(array, length of array). This distinct() will take two arguments, i.e., a dynamic uint type array and length of the array. The distinct() will return the number of distinct elements in an array.
5. Find the sum of the series 1 + x + x^2+ x^3+ … + x^n. Create a function expression(x, n).The expression() will find the sum of the above expression.
6. Create a function hcf(num1, num2). This hcf() will take two arguments uint type number1 and number2.The hcf() will find the of number1 and number2.

Chapter Ends…

❑❑❑