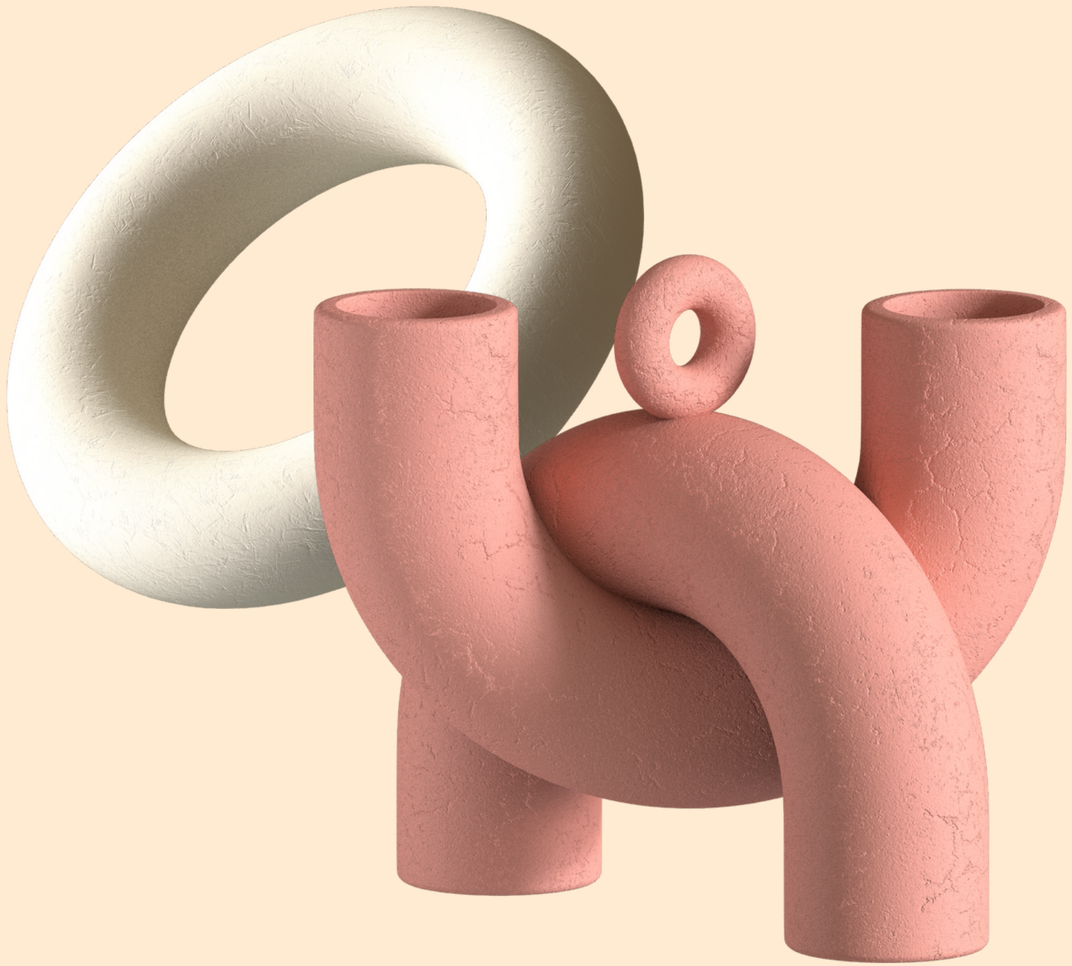
Introduction to Web 3



Brought to you by Addis software

Contents

What to Know

Introduction to Solidity

What is Solidity

- Solidity is an object-oriented, high-level language for implementing smart contracts.
- Smart contracts are programs that govern the behavior of accounts within the Ethereum state.
- Solidity is statically typed, and supports inheritance, libraries, and complex user-defined types among other features.

Data Types

- **Value Types -** A value type stores its data directly in the memory it owns. Variables of this type are duplicated whenever they appear in functions or assignments. A value type will maintain an independent copy of any duplicated variables. Therefore, a change in the value of a duplicated variable will not affect the original variable.

- **Reference Types** - Solidity reference types differ from value types in that they do not store values directly on their own. Instead, reference types store (or “reference”) the address of the data’s location and do not directly share the data.

Value Types

1. Signed integers (int)

2. Unsigned integers (uint)
   - uint256, uint32, uint16, uint8

3. Boolean

4. Addresses
   - designed to hold up to 20B, or 160 bits, which is the size of an Ethereum address.
   - address, address payable (**call, send, transfer**)

5. Enums
   - consist of user-defined data types

6. Bytes
   - refers to 8-bit signed integers
   - range — from bytes1 to bytes32

Refrence Types

1

- Arrays
- fixed, dynamic sized

2

- Byte Array
- can hold any number of bytes
- **bytes**

3

- String arrays
- are like byte arrays
- does not have an index so it lacks array members such as length, push, and pop

4

- Structs
- made up of multiple variables, which can be both value type and reference type.
- **struct**

5

- Mapping
- used to store data in the form of key-value pairs
- similarly to a hashtable or dictionary in other programming languages
- **mapping(address => bool)**

Variables

- **local -** declared inside a function. not stored on the blockchain.

- **state -** declared outside a function. stored on the blockchain.

- **global** - provides information about the blockchain.
  https://docs.soliditylang.org/en/v0.8.15/units-and-globalvariables.html

Visibility

- **Private -** only to inside contract.
- **Public -** to any contract.
- **Internal** - only inside contract and children.
- **external -** only from outside contract. (this.externalFunction())

Functions

function name(**arg** 1, **arg** 2**,...**) visibility behaviour returns(**types**){

}

behaviour

- view - don't modify variable, read-only
- pure - don't modify or read state
- payable - denotes a function that can receive ether

The Fallback function

function () external payable{

}

It has

- no name
- no arguments
- don't return
- only external visibility
- only payable behaviour

Called When

- a call to a function that doesn't exist
- sending ether to that contract

Guard functions

- **assert()**
- **require()**
- **revert()**

example

require(msg.value > 1); // value must be greater than 1

Function modifiers

- Modifiers are code that can be run before and / or after a function call.

Are used to -

- Restrict access
- Validate inputs
- Guard against reentrancy hack

Gas

- a fee which is required to conduct a transaction on the Ethereum blockchain.
- Sender pays the gas
- Minors(who adds block to the blockchain), recives the gas
- Cost of gas depended on difficulty of computation
- Limit the amount of computation a transaction can do (Prevent spamming)

Gas Limit

- max gas you are willing to use.

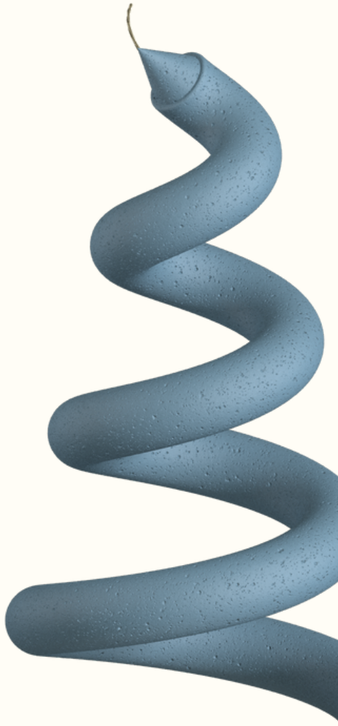
Gas Price

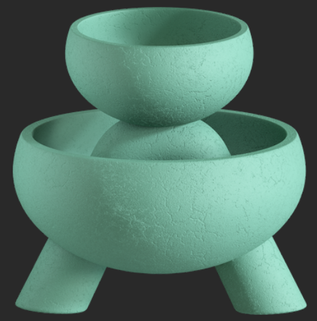
- how much you are willing tp pay for each gas.

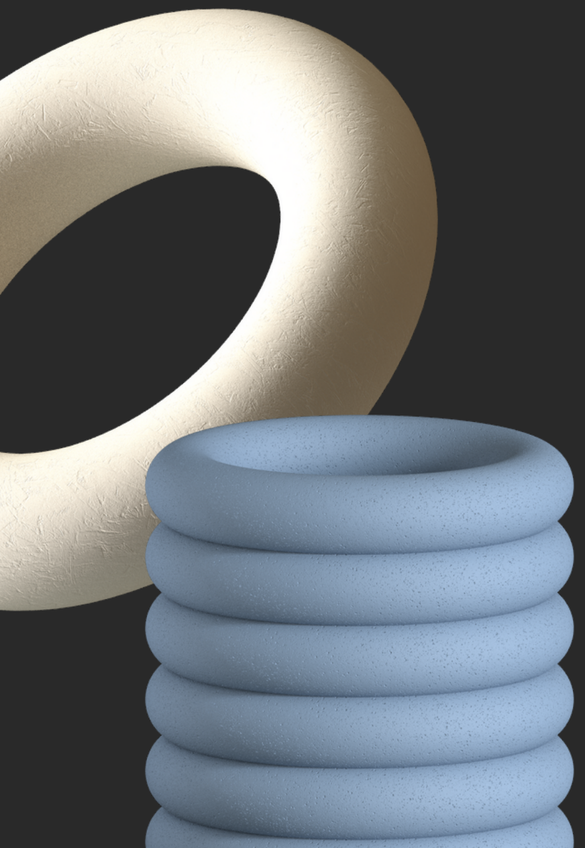
- The higher the gas limit, the higher the computation you can process.
- Higher gas price, short waiting time, and vice versa

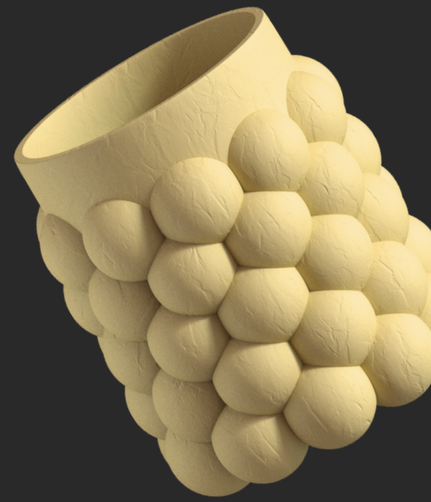
https://ethereum.org/en/developers/docs/evm/opcodes/

DEMO TIME...









Thank You!