BLOCKCHAIN | Batch 1 - Day 4 (Part 1 & Part 2) Overview

- Metamask setup overview
- First smart contract
- Ethereum Nodes
- Ganache - Trufflesuite
- Data Types
- Integer
- Booleans
- Address and Balance

Metamask setup overview:

MetaMask is a cryptocurrency wallet available as a browser extension for Chrome, Firefox, Opera, and Brave. The wallet serves as a connection between your browser and the Ethereum blockchain.you can deploy various smart contracts on Robsten test networks and you can see how the transaction is happening in realtime.

Etherum Nodes:

1. Every Ethereum Network has the Network Protocol
2. The main network is for permanent
3. The test network is for Beta launches for a developer like you and me
4. Ropsten is a BETA network before making the changes in Main Network
5. EVM is going to act the same in all the Networks

Ganache-Truffle suite:

Ganache is used for setting up a personal Ethereum Blockchain for testing your Solidity contracts. It provides more features when compared to Remix.

You may download Ganache from the following URL –

https://trufflframework.com/ganache

JavaScript VM:

All the transactions will be executed in a sandbox blockchain in the browser. This means nothing will be persisted when you reload the page. The JVM is its own blockchain and on each reloads it will start a new blockchain, the old one will not be saved.

Injected Provider:

The remix will connect to an injected web3 provider. Metamask is an example of a provider that inject web3.

Web3 Provider:

The remix will connect to a remote node. You will need to provide the URL to the selected provider: geth, parity, or any Ethereum client.

Data Types:

Now let see, what are the datatypes are available in solidity

Solidity is a statically typed language, which means that the type of each variable (state and local) needs to be specified. Solidity provides several elementary types which can be combined to form complex types.

int/uint: Signed and unsigned integers of various sizes. Keywords uint8 to uint256 in steps of 8 (unsigned of 8 up to 256 bits) and int8 to int256. uint and int are aliases for uint256 and int256, respectively.

```
pragma solidity >=0.5.15 < 0.7.3;

contract datatypes{
unit8 myUnit; // 0 to 256  //unit is nothing but unsigned integer
unit256 myUnit256;//256bits
Constructor() public{   // Constructor is the default function in pragma solidity 0.5.15

myUnit256=0;
myUnit8=255;
}
function setmyDatatypes(unit256 _myUnit256 ) public{

myUnit256=_unit256;
}

function incMyUINT8() public{
myUnit256++;
}
function decMyUINT8() public{
myUnit256--;
}
}
```

once you have done this, you may see the incMyUINT8 and decMyUINT8 on the left side .. By clicking that buttons you can increase or decrease the value

Booleans:

The possible values are constants true and false. It is false in default.

```
pragma solidity >=0.5.15 < 0.7.3;

contract datatypes{

bool public Myboolean;

function setbool() public
{
```

```
Myboolean=true;
}
}
```

Address and balance:

```
pragma solidity >=0.5.15 < 0.7.3;

contract datatypes{
address public Myaddr;

function setaddr(address _addr) public
{
Myaddr=_addr;
}

function getEther() public view returns(unit) //whenever you want to return something
{
return myaddr.balance; // balance its like inbuilt global function
}
}
```