under construction

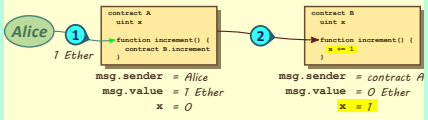
delegatecall

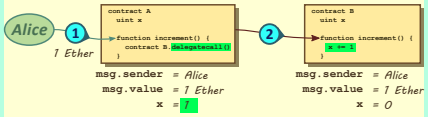
The delegatecall() method is a low level function similar to the call() method, and is used to call other contracts; when contract A executes a delegatecall() on contract B, the code inside the latter is ran inside the context of the former:

`storage` - This would refer to contract A's storage
`msg.sender` - This would refer to the caller who called contract A
`msg.value` - This would be the value that was sent to contract A

Once you deploy a smart contract to the blockchain, you cannot change the code of that contract, however using delegatecall() you are able to upgrade the contract even though you are not able to change any of the code inside it

Suppose Alice calls the increment function in contract A and sends 1 Ether

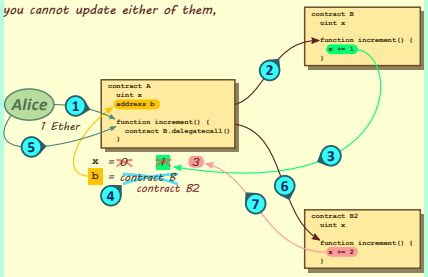




When the code in contract B is executed, it does not change the state variable inside contract B, and the state variable x inside contract A (not inside contract B) is updated, this is because delegatecall() runs the code inside contract B using the storage msg.sender & storage msg.value of contract A

How is `delegatecall()` useful?

it lets you write upgradable contracts, in the example below, contract A and contract B are already uploaded to the blockchain so you cannot update either of them,



Delegatecall

A video discussing all of this material is available here:
https://youtu.be/Yh8UL7FZwAI

This···

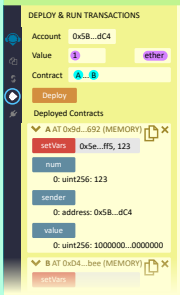
```
pragma solidity ^0.8.7;

contract A {
  uint public num;
  address public sender;
  uint public value;

  function setVars(address _contract, uint _num) public payable {
    (bool success, bytes memory data) = _contract.delegatecall(
      abi.encodeWithSignature("setVars(uint256)", _num)
    );
  }
}

contract B {
  uint public num;
  address public sender;
  uint public value;

  function setVars(uint _num) public payable {
    num = _num;
    sender = msg.sender;
    value = msg.value;
  }
}
```




```
contract B2 {
  uint public num;
  address public sender;
  uint public value;

  function setVars(uint _num) public payable {
    num = 2 * _num;
    sender = msg.sender;
    value = msg.value;
  }
}
```