

## Practical – 3


**Aim: Ethereum – With respect to Ethereum, carry out following:**

- **Install Geth Client & Configure Ethereum Nodes**
- **Manage Accounts and Account states.**
- **Enable Mining and checking balance in Ether**
- **Setting up Metamask and Testing Fund Transfer – work with Ethereum Ecosystem**
- **AddParameters to Cryptocurrency**
- **Check Balance Before Transfer**
- **Adding Transfer Event for Logging.**

**Ganache installation:**

Ganache				
ACCOUNTS	BLOCKS	TRANSACTIONS	CONTRACTS	EVENTS
LOGS	SEARCH FOR BLOCK NUMBERS OR TX HASHES			
CURRENT BLOCK 13	GAS PRICE 20000000000	GAS LIMIT 6721975	HARDFORK MUIRGLACIER	NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545	MINING STATUS AUTOMINING	WORKSPACE QUICKSTART	SAVE	SWITCH
MNEMONIC ? sing obtain assault bridge sad sight month put reopen window invite robot		HD PATH m/44'/60'/0'/0/account_index		
ADDRESS 0x8d058e72cd913E541B212C1495c64B4742344bb2	BALANCE 79.98 ETH	TX COUNT 13	INDEX 0	
ADDRESS 0xFddAbEBa613067f140036d2f8571Ee04c53EcFb5	BALANCE 100.00 ETH	TX COUNT 0	INDEX 1	
ADDRESS 0x9274384865896b6dC47583495F183eFB74C0085F	BALANCE 100.00 ETH	TX COUNT 0	INDEX 2	
ADDRESS 0x73Ced982c15B784F01663a4B235bF6fF343E2Bc4	BALANCE 100.00 ETH	TX COUNT 0	INDEX 3	
ADDRESS 0x933480639716018c4FFDc3e2dFd7419065FBb100	BALANCE 100.00 ETH	TX COUNT 0	INDEX 4	
ADDRESS 0x6D9005F9dee695685DEd5FADcE0744b1C9091CB7	BALANCE 100.00 ETH	TX COUNT 0	INDEX 5	
ADDRESS 0x7a3729D51A38E80E820693D7A40F63665fbd1ce1	BALANCE 100.00 ETH	TX COUNT 0	INDEX 6	
ADDRESS	BALANCE	TX COUNT	INDEX	

Ganache is a personal blockchain for rapid Ethereum and Corda distributed application development. You can use Ganache across the entire development cycle; enabling you to develop, deploy, and test your dApps in a safe and deterministic environment.

**Smart contract:**

```
1 pragma solidity 0.4.23;
2
3 contract Bank
4 {
5     int bal;
6
7     constructor() public
8     {
9         bal = 1;
10    }
11
12    event Transfer(int balance, string message);
13
14    function getBalance() view public returns (int)
15    {
16        return bal;
17    }
18
19    function withdraw(int amt) public
20    {
21        if(bal - amt < 0)
22        {
23            revert("Not enough balance to withdraw");
24        }
25        bal = bal - amt;
26        emit Transfer(bal, "Withdraw successful");
27    }
28
29    function deposit(int amt) public
30    {
31        bal = bal + amt;
32        emit Transfer(bal, "Deposit successful");
33    }
34 }
```

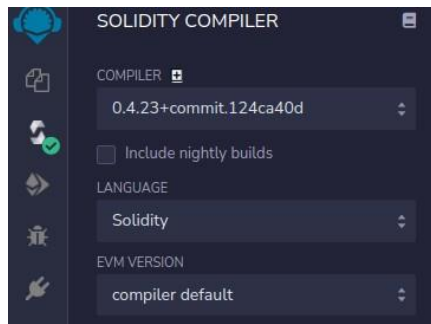
As seen in the code above, the smart contract consists of three functions:

1. getBalance: returns the balance in the account
2. deposit: deposits the given amount into the account
3. withdraw: withdraws the given amount from the account

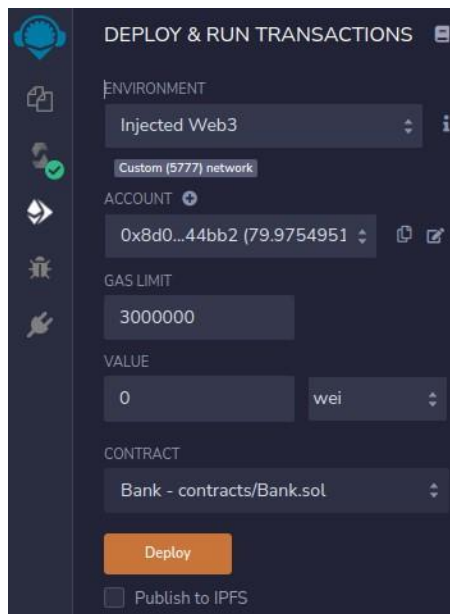
When withdrawing, if the balance is less than the amount, an error is thrown.

Also, there is a Transfer event which takes balance and a message and it is emitted when successful deposit and withdraw take place.

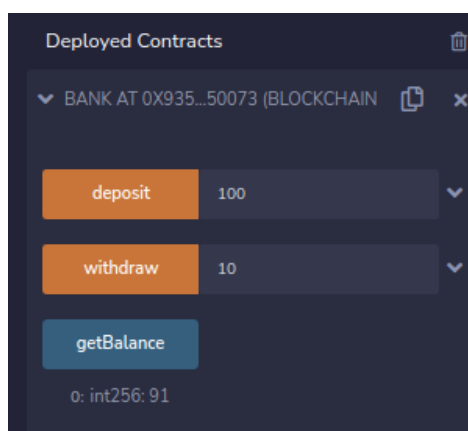
## Compile:



## Deploy:



## Run:



Get balance:

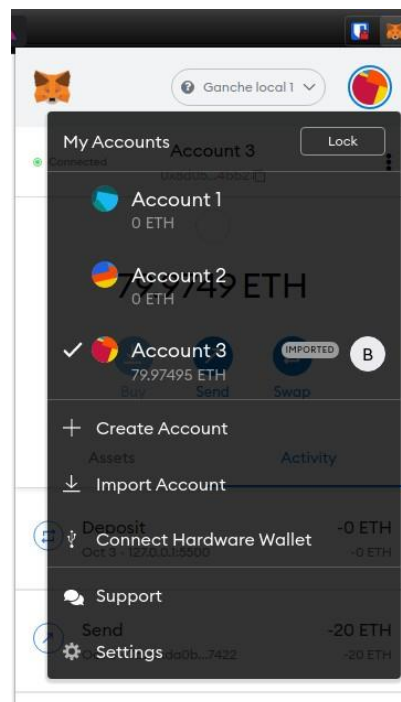
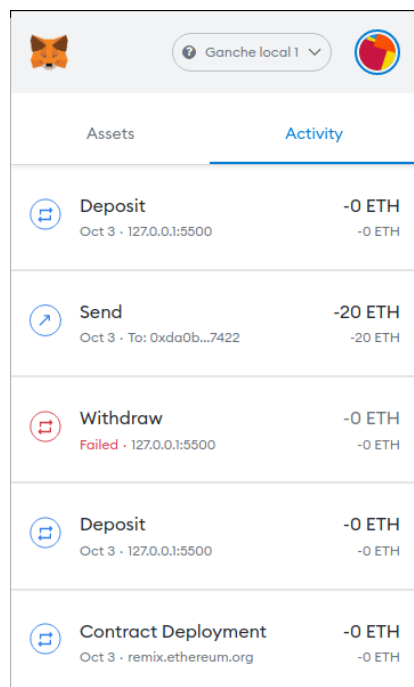
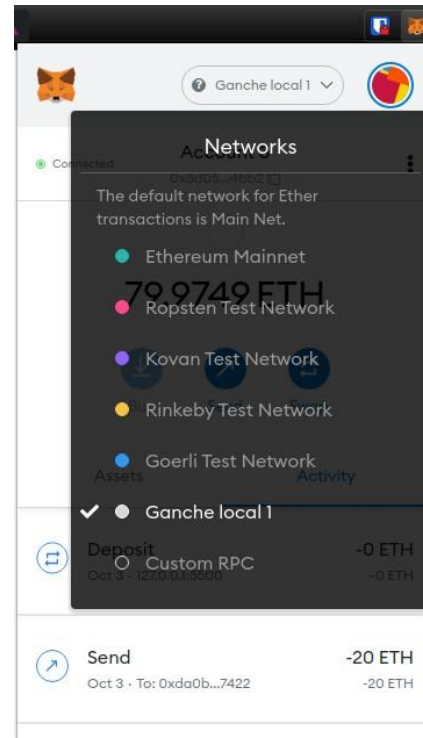
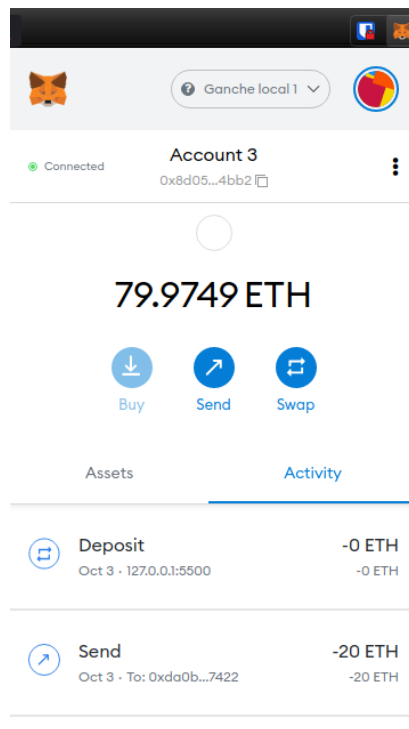
The screenshot shows a web interface for a bank account titled "BANK AT 0X935...50073 (BLOCKCHAIN)". It features three main buttons: "deposit", "withdraw", and "getBalance". The "deposit" and "withdraw" buttons are orange, while "getBalance" is blue. Below the buttons, the balance is displayed as "0: int256: 491".

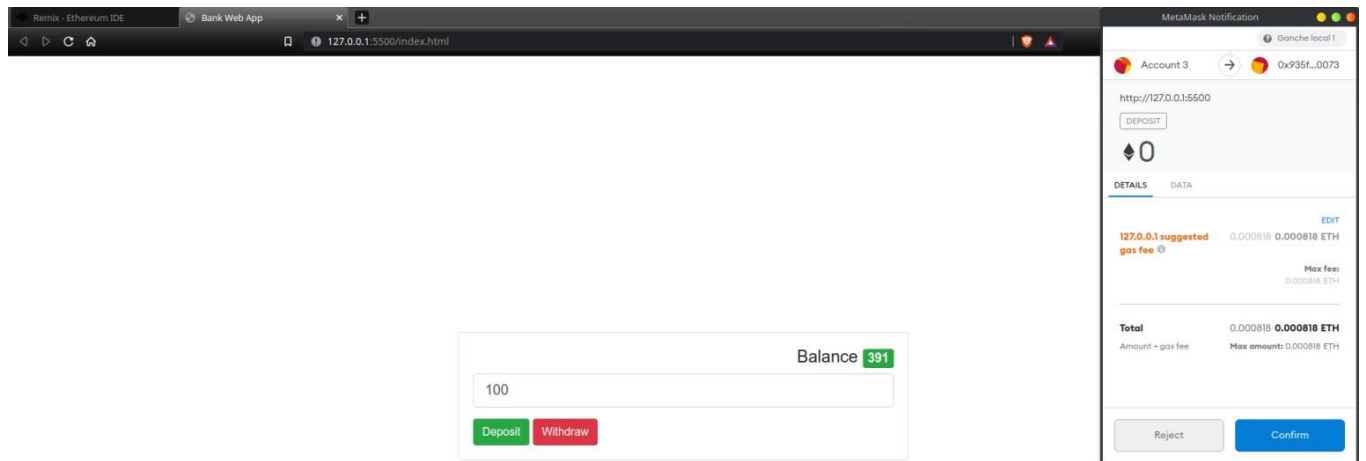
Deposit 100:

The screenshot shows the same web interface, but the "deposit" button is highlighted with an orange border, and the value "100" is entered in the adjacent input field. The "withdraw" button and "getBalance" button remain visible. The balance at the bottom is still "0: int256: 491".

Get balance:

The screenshot shows the same web interface, but the "getBalance" button is highlighted with a blue border. The balance at the bottom has updated to "0: int256: 591".

**Metamask:**

**Frontend for the smart contract:**

Balance **491**

Amount

Deposit Withdraw