Introduction to Solidity in Remix IDE

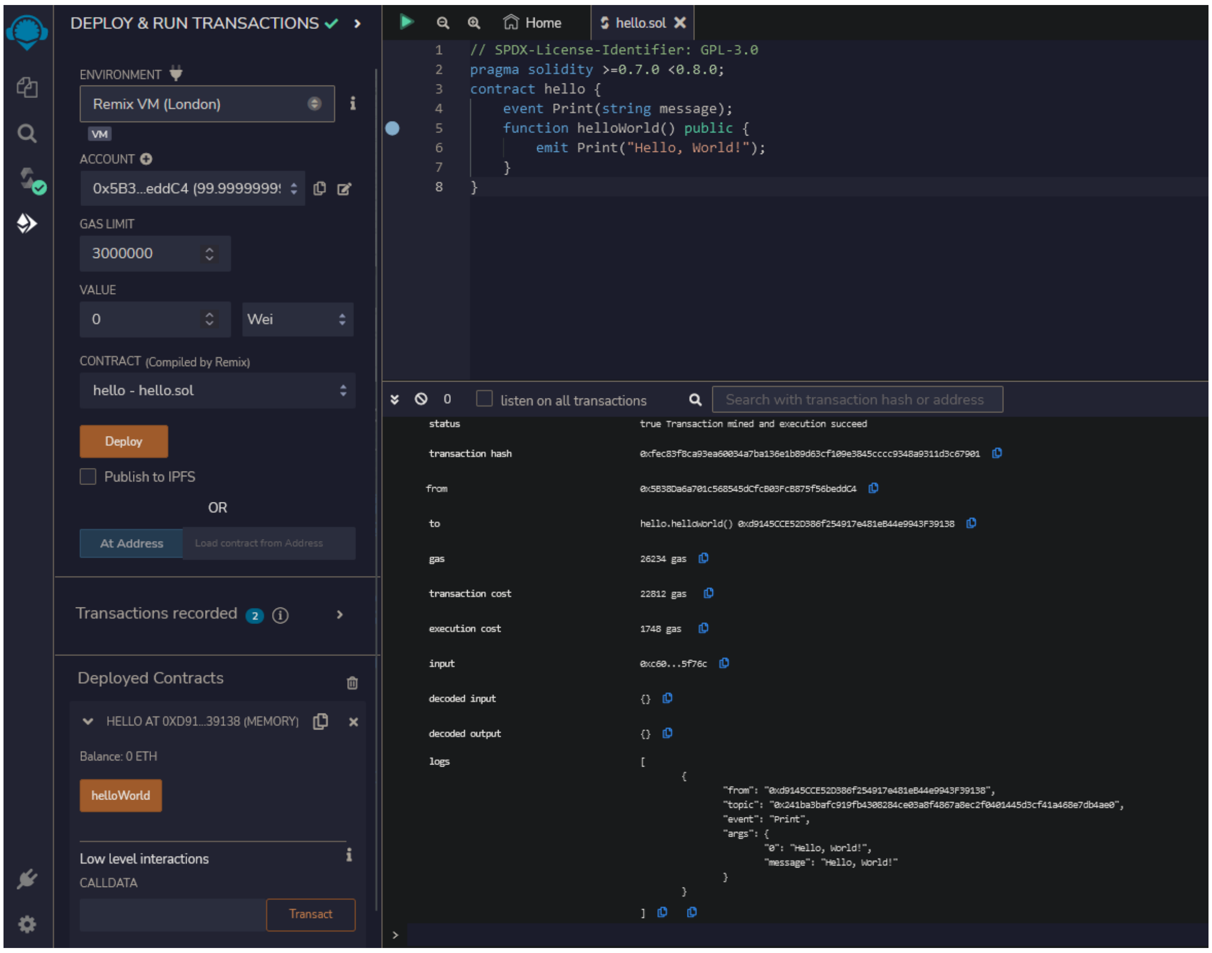
Marcos Alonso Campillo

Leon Novački

Hello World

For this first exercise we are going to run a sample HelloWorld project (HelloWorld.sol). We declare a Print Event and then call it with the “Hello world” string as the parameter. The string appears in the terminal section located at the bottom inside logs.

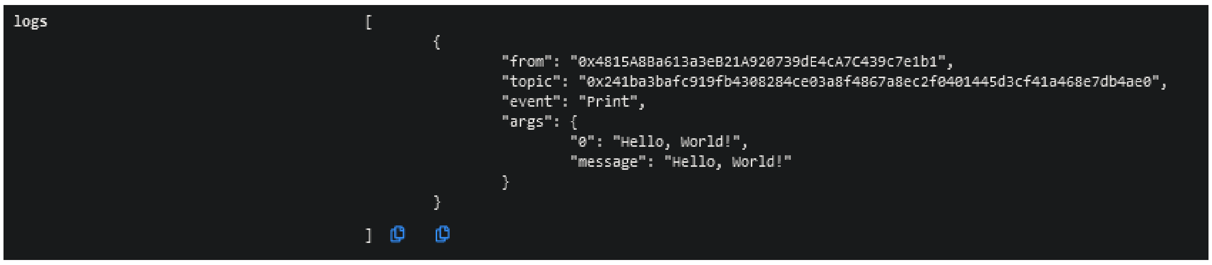
- Total gas cost: 26234 gas
- Transaction cost: 22812 gas
- Execution cost: 1748 gas



ENVIRONMENT: Remix VM (London)
ACCOUNT: 0x5B3...eddC4 (99.9999999!
GAS LIMIT: 3000000
VALUE: 0 Wei
CONTRACT (Compiled by Remix): hello - hello.sol
HELLO AT 0XD91...39138 (MEMORY)
Balance: 0 ETH
CALLDATA

```solidity
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract hello {
    event Print(string message);
    function helloWorld() public {
        emit Print("Hello, World!");
    }
}
```

```
status              true Transaction mined and execution succeed
transaction hash    0xfec83f8ca93ea60034a7ba136e1b89d63cf109e3845cccc9348a9311d3c67901
from                0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                  hello.helloWorld() 0xd9145CCE52D386f254917e481eB44e9943F39138
gas                 26234 gas
transaction cost    22812 gas
execution cost      1748 gas
input               0xc60...5f76c
decoded input       {}
decoded output      {}
logs                [
                        {
                            "from": "0xd9145CCE52D386f254917e481eB44e9943F39138",
                            "topic": "0x241ba3bafc919fb4308284ce03a8f4867a8ec2f0401445d3cf41a468e7db4ae0",
                            "event": "Print",
                            "args": {
                                "0": "Hello, World!",
                                "message": "Hello, World!"
                            }
                        }
                    ]
```

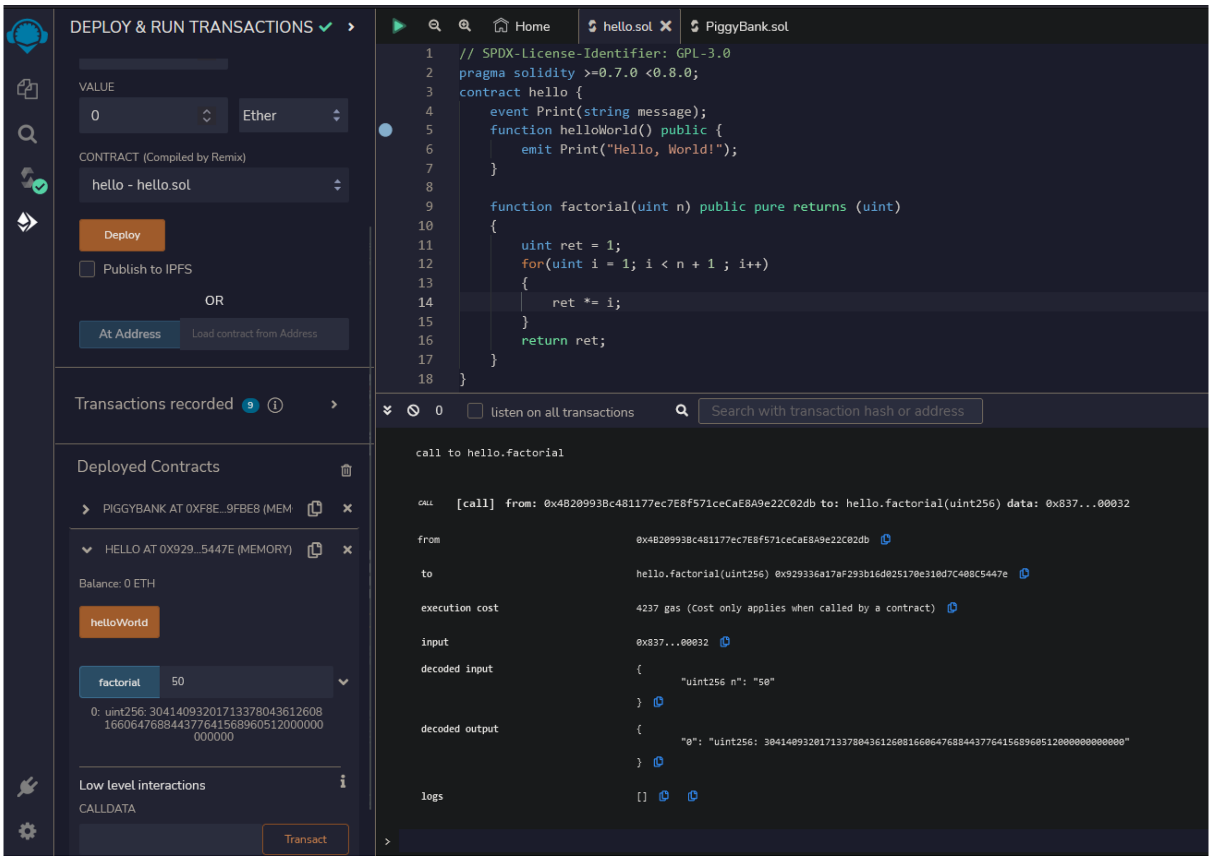


```
logs                [
                        {
                            "from": "0x4815A8Ba613a3eB21A920739dE4cA7C439c7e1b1",
                            "topic": "0x241ba3bafc919fb4308284ce03a8f4867a8ec2f0401445d3cf41a468e7db4ae0",
                            "event": "Print",
                            "args": {
                                "0": "Hello, World!",
                                "message": "Hello, World!"
                            }
                        }
                    ]
```

Factorial

In this case we need to create a function that computes the factorial of a number given. We can Input the value in the bottom left portion of the screen, the result can be obtained in the terminal inside “decoded output” or under the factorial call button.

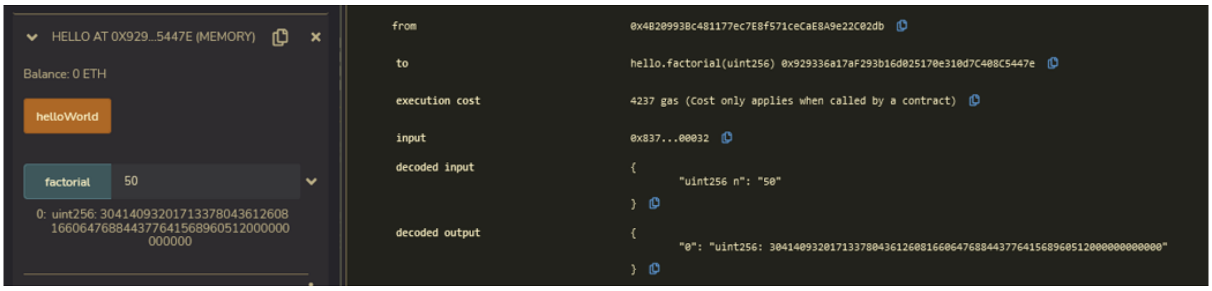
- Execution cost: 4237 gas


DEPLOY & RUN TRANSACTIONS
VALUE
0
Ether
CONTRACT (Compiled by Remix)
hello - hello.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 9
Deployed Contracts
PIGGYBANK AT 0XF8E...9FBE8 (MEM
HELLO AT 0X929...5447E (MEMORY)
Balance: 0 ETH
helloWorld
factorial
50
Low level interactions
CALLDATA
Transact
Home
hello.sol
PiggyBank.sol
listen on all transactions
Search with transaction hash or address

0: uint256: 30414093201713378043612608166064768844377641568960512000000000000

```solidity
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract hello {
    event Print(string message);
    function helloWorld() public {
        emit Print("Hello, World!");
    }

    function factorial(uint n) public pure returns (uint)
    {
        uint ret = 1;
        for(uint i = 1; i < n + 1 ; i++)
        {
            ret *= i;
        }
        return ret;
    }
}
```

```
call to hello.factorial

CALL  [call] from: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db to: hello.factorial(uint256) data: 0x837...00032

from                0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
to                  hello.factorial(uint256) 0x929336a17aF293b16d025170e310d7C408C5447e
execution cost      4237 gas (Cost only applies when called by a contract)
input               0x837...00032
decoded input       {
                        "uint256 n": "50"
                    }
decoded output      {
                        "0": "uint256: 30414093201713378043612608166064768844377641568960512000000000000"
                    }
logs                []
```

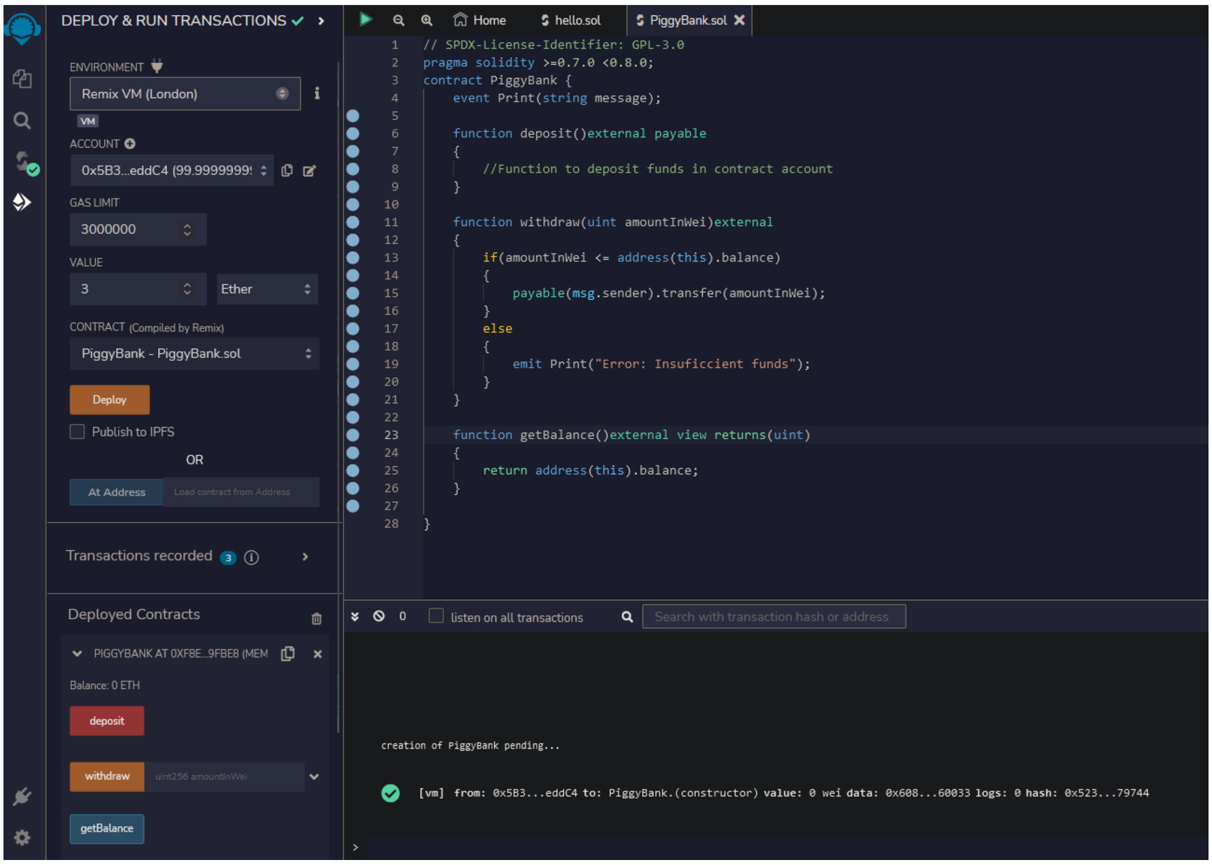


0: uint256: 30414093201713378043612608166064768844377641568960512000000000000

```
from                0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
to                  hello.factorial(uint256) 0x929336a17aF293b16d025170e310d7C408C5447e
execution cost      4237 gas (Cost only applies when called by a contract)
input               0x837...00032
decoded input       {
                        "uint256 n": "50"
                    }
decoded output      {
                        "0": "uint256: 30414093201713378043612608166064768844377641568960512000000000000"
                    }
```

PiggyBank

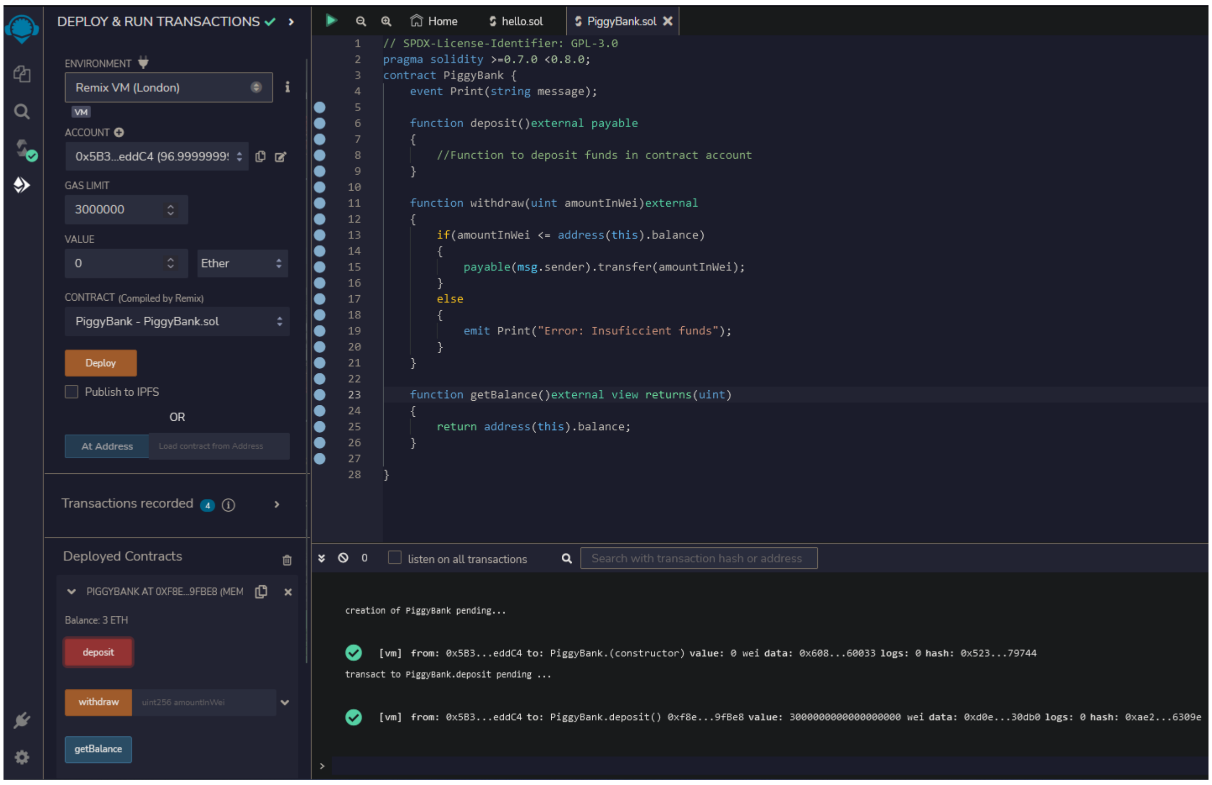
This contract will serve as a personal money reserve. It must have three functionalities: deposit funds, withdraw funds and check account balance.


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Remix VM (London)
VM
ACCOUNT
0x5B3...eddC4 (99.9999999!
GAS LIMIT
3000000
VALUE
3
Ether
CONTRACT (Compiled by Remix)
PiggyBank - PiggyBank.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 3
Deployed Contracts
PIGGYBANK AT 0XF8E...9FBE8 (MEM
Balance: 0 ETH
deposit
withdraw
uint256 amountInWei
getBalance
Home
hello.sol
PiggyBank.sol
listen on all transactions
Search with transaction hash or address

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract PiggyBank {
    event Print(string message);

    function deposit()external payable
    {
        //Function to deposit funds in contract account
    }

    function withdraw(uint amountInWei)external
    {
        if(amountInWei <= address(this).balance)
        {
            payable(msg.sender).transfer(amountInWei);
        }
        else
        {
            emit Print("Error: Insuficcient funds");
        }
    }

    function getBalance()external view returns(uint)
    {
        return address(this).balance;
    }

}
```

```
creation of PiggyBank pending...

[vm] from: 0x5B3...eddC4 to: PiggyBank.(constructor) value: 0 wei data: 0x608...60033 logs: 0 hash: 0x523...79744
```

When we deploy the contract we will be presented with the 3 functions but only one with a possible input. We are going to start with a 3 ETH deposit in the contracts account as seen in the picture above. When the transaction is complete the user's account balance will be reduced by 3 ETH minus the total gas cost.

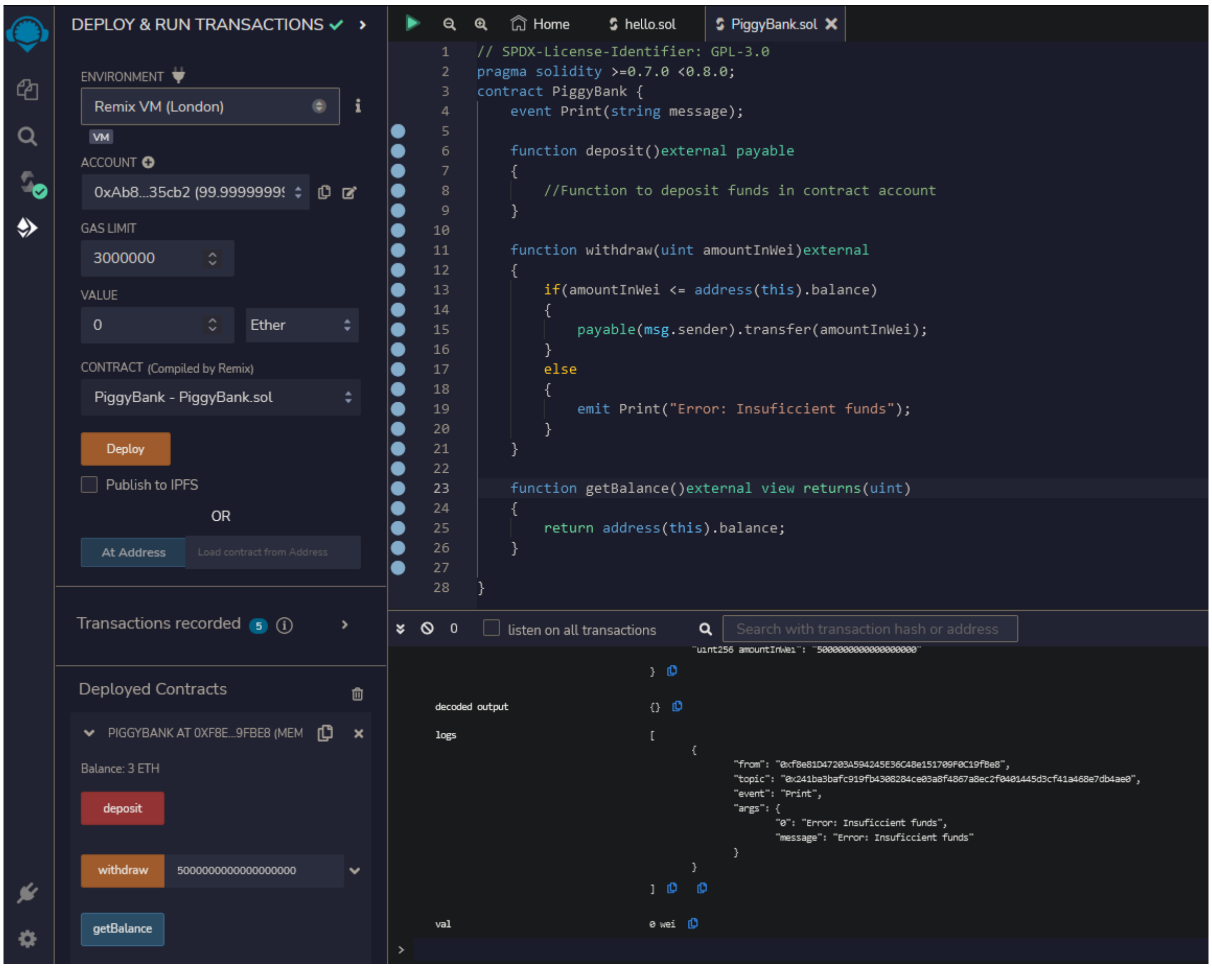


ENVIRONMENT: Remix VM (London)
ACCOUNT: 0x5B3...eddC4 (96.9999999
GAS LIMIT: 3000000
VALUE: 0 Ether
CONTRACT (Compiled by Remix): PiggyBank - PiggyBank.sol
Deploy
Publish to IPFS
OR
At Address — Load contract from Address
Transactions recorded 4
Deployed Contracts
PIGGYBANK AT 0XF8E...9FBE8 (MEM
Balance: 3 ETH
deposit
withdraw — uint256 amountInWei
getBalance

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract PiggyBank {
    event Print(string message);

    function deposit()external payable
    {
        //Function to deposit funds in contract account
    }

    function withdraw(uint amountInWei)external
    {
        if(amountInWei <= address(this).balance)
        {
            payable(msg.sender).transfer(amountInWei);
        }
        else
        {
            emit Print("Error: Insuficcient funds");
        }
    }

    function getBalance()external view returns(uint)
    {
        return address(this).balance;
    }

}
```

```
creation of PiggyBank pending...

[vm] from: 0x5B3...eddC4 to: PiggyBank.(constructor) value: 0 wei data: 0x608...60033 logs: 0 hash: 0x523...79744
transact to PiggyBank.deposit pending ...

[vm] from: 0x5B3...eddC4 to: PiggyBank.deposit() 0xf8e...9fBe8 value: 3000000000000000000 wei data: 0xd0e...30db0 logs: 0 hash: 0xae2...6309e
```

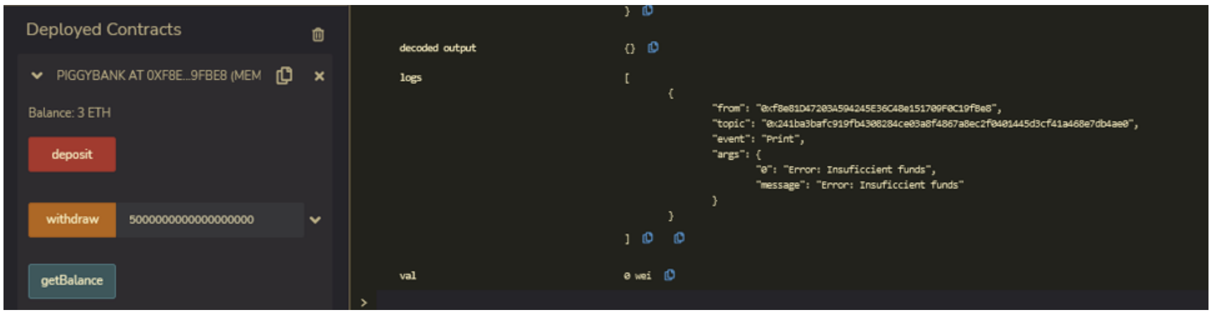
For this example we will change the user's account to try and withdraw a greater amount than what it is available.



ENVIRONMENT: Remix VM (London)
ACCOUNT: 0xAb8...35cb2 (99.9999999
GAS LIMIT: 3000000
VALUE: 0 Ether
CONTRACT (Compiled by Remix): PiggyBank - PiggyBank.sol
Deploy
Publish to IPFS
OR
At Address — Load contract from Address
Transactions recorded 5
Deployed Contracts
PIGGYBANK AT 0XF8E...9FBE8 (MEM
Balance: 3 ETH
deposit
withdraw — 5000000000000000000
getBalance

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract PiggyBank {
    event Print(string message);

    function deposit()external payable
    {
        //Function to deposit funds in contract account
    }

    function withdraw(uint amountInWei)external
    {
        if(amountInWei <= address(this).balance)
        {
            payable(msg.sender).transfer(amountInWei);
        }
        else
        {
            emit Print("Error: Insuficcient funds");
        }
    }

    function getBalance()external view returns(uint)
    {
        return address(this).balance;
    }

}
```

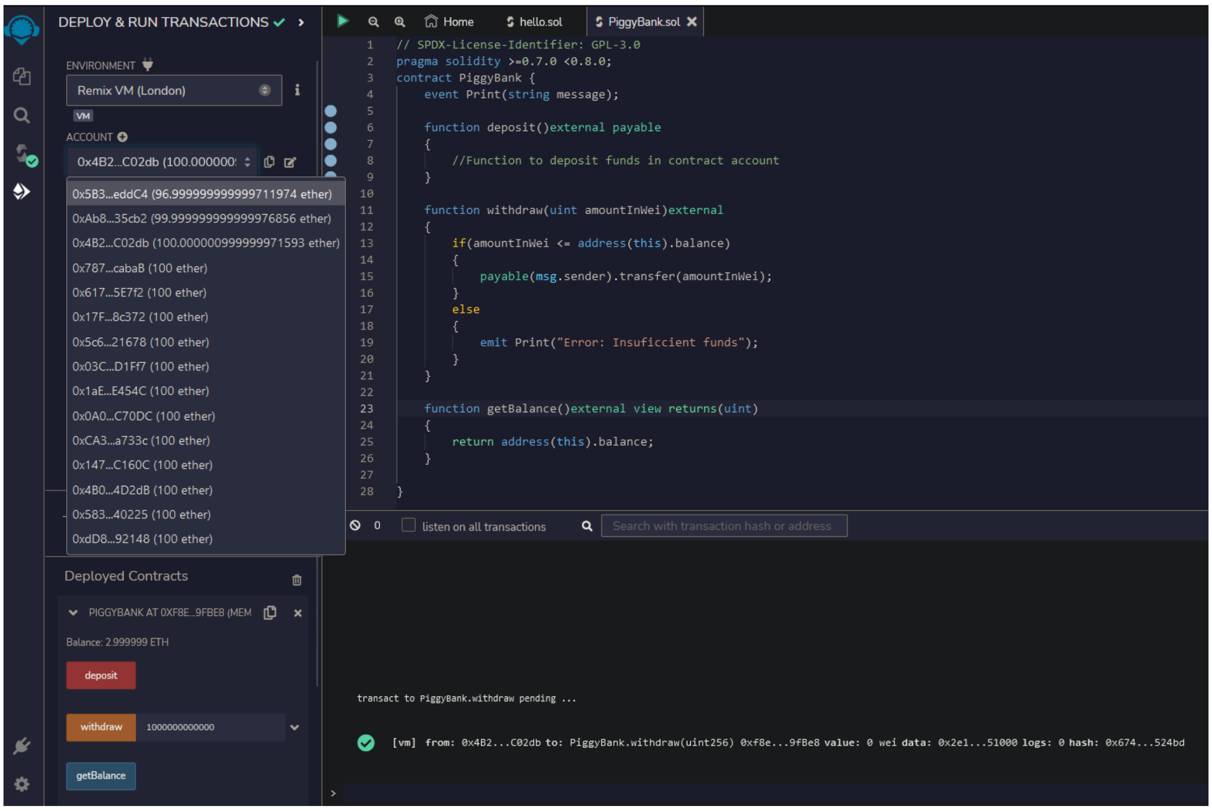
```
                    "uint256 amountInWei": "5000000000000000000"
                }
decoded output  {}
logs            [
                    {
                        "from": "0xf8e81D47203A594245E36C48e151709F0C19fBe8",
                        "topic": "0x241ba3bafc919fb4308284ce03a8f4867a8ec2f0401445d3cf41a468e7db4ae0",
                        "event": "Print",
                        "args": {
                            "0": "Error: Insuficcient funds",
                            "message": "Error: Insuficcient funds"
                        }
                    }
                ]
val             0 wei
```



withdraw 5000000000000000000

```
decoded output    {}
logs              [
                      {
                          "from": "0xf8e81D47203A594245E36C48e151709F0C19fBe8",
                          "topic": "0x241ba3bafc919fb4308284ce03a8f4867a8ec2f0401445d3cf41a468e7db4ae0",
                          "event": "Print",
                          "args": {
                              "0": "Error: Insuficcient funds",
                              "message": "Error: Insuficcient funds"
                          }
                      }
                  ]
val               0 wei
```

The “Insufficient funds” log is shown and nothing is sent. For the next example we will use another account who will try to withdraw 1 Szabo (10^{12} wei).


DEPLOY & RUN TRANSACTIONS
Home
hello.sol
PiggyBank.sol
listen on all transactions
Search with transaction hash or address
Deployed Contracts
PIGGYBANK AT 0XF8E...9FBE8 (MEM
deposit
withdraw
getBalance

ENVIRONMENT: Remix VM (London)

ACCOUNT: 0x4B2...C02db (100.000000

```
0x5B3...eddC4 (96.999999999999711974 ether)
0xAb8...35cb2 (99.999999999999976856 ether)
0x4B2...C02db (100.000000999999971593 ether)
0x787...cabaB (100 ether)
0x617...5E7f2 (100 ether)
0x17F...8c372 (100 ether)
0x5c6...21678 (100 ether)
0x03C...D1Ff7 (100 ether)
0x1aE...E454C (100 ether)
0x0A0...C70DC (100 ether)
0xCA3...a733c (100 ether)
0x147...C160C (100 ether)
0x4B0...4D2dB (100 ether)
0x583...40225 (100 ether)
0xdD8...92148 (100 ether)
```

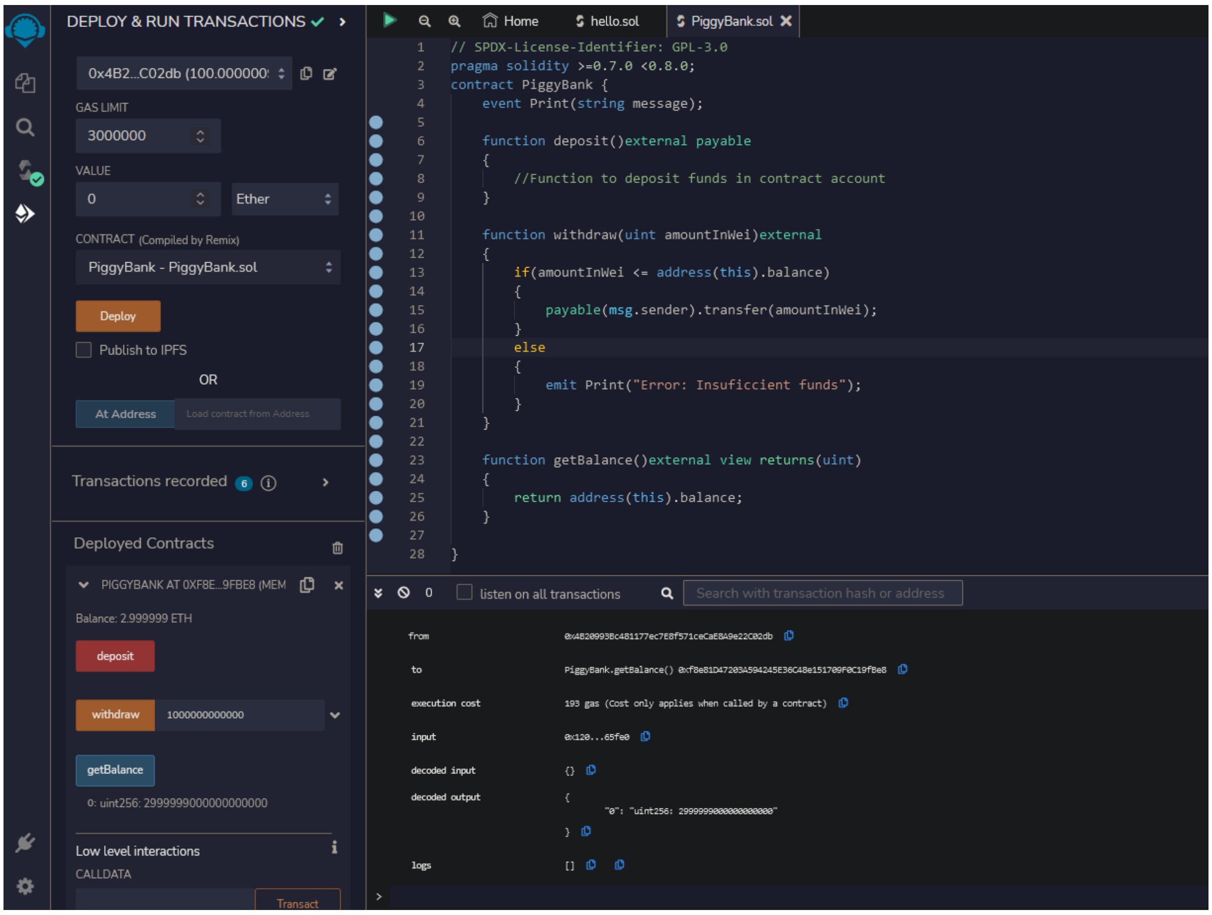
Balance: 2.999999 ETH

withdraw 1000000000000

```solidity
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract PiggyBank {
    event Print(string message);

    function deposit()external payable
    {
        //Function to deposit funds in contract account
    }

    function withdraw(uint amountInWei)external
    {
        if(amountInWei <= address(this).balance)
        {
            payable(msg.sender).transfer(amountInWei);
        }
        else
        {
            emit Print("Error: Insuficcient funds");
        }
    }

    function getBalance()external view returns(uint)
    {
        return address(this).balance;
    }
}
```

```
transact to PiggyBank.withdraw pending ...

[vm] from: 0x4B2...C02db to: PiggyBank.withdraw(uint256) 0xf8e...9fBe8 value: 0 wei data: 0x2e1...51000 logs: 0 hash: 0x674...524bd
```

As seen in the picture above the first account has 100 - (3 ETH + gas costs) ether, the second one didn't get to withdraw the contract’s balance and was only charged with the execution cost. The last one was also charged with the total gas costs but managed to withdraw 1 szabo from the account.



0x4B2...C02db (100.000000

GAS LIMIT
3000000

VALUE
0 Ether

CONTRACT (Compiled by Remix)
PiggyBank - PiggyBank.sol

Deploy
Publish to IPFS
OR
At Address Load contract from Address

Transactions recorded 6

Deployed Contracts

PIGGYBANK AT 0XF8E...9FBE8 (MEM

Balance: 2.999999 ETH

deposit

withdraw 1000000000000

getBalance

0: uint256: 2999999000000000000

Low level interactions
CALLDATA
Transact

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.8.0;
contract PiggyBank {
    event Print(string message);

    function deposit()external payable
    {
        //Function to deposit funds in contract account
    }

    function withdraw(uint amountInWei)external
    {
        if(amountInWei <= address(this).balance)
        {
            payable(msg.sender).transfer(amountInWei);
        }
        else
        {
            emit Print("Error: Insuficcient funds");
        }
    }

    function getBalance()external view returns(uint)
    {
        return address(this).balance;
    }

}
```

0 listen on all transactions Search with transaction hash or address

```
from              0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
to                PiggyBank.getBalance() 0xf8e81D47203A594245E36C48e151709F0C19fBe8
execution cost    193 gas (Cost only applies when called by a contract)
input             0x120...65fe0
decoded input     {}
decoded output    {
                        "0": "uint256: 2999999000000000000"
                  }
logs              []
```

We can check the contract's balance by calling the function getBalance() and notice that it now has less than 3 ETH below the getBalance button or in the terminal's "decoded output" section.