

TUGAS LAB 2 SHARED WALLET

Diajukan untuk memenuhi tugas pada mata kuliah Blockchain

oleh:

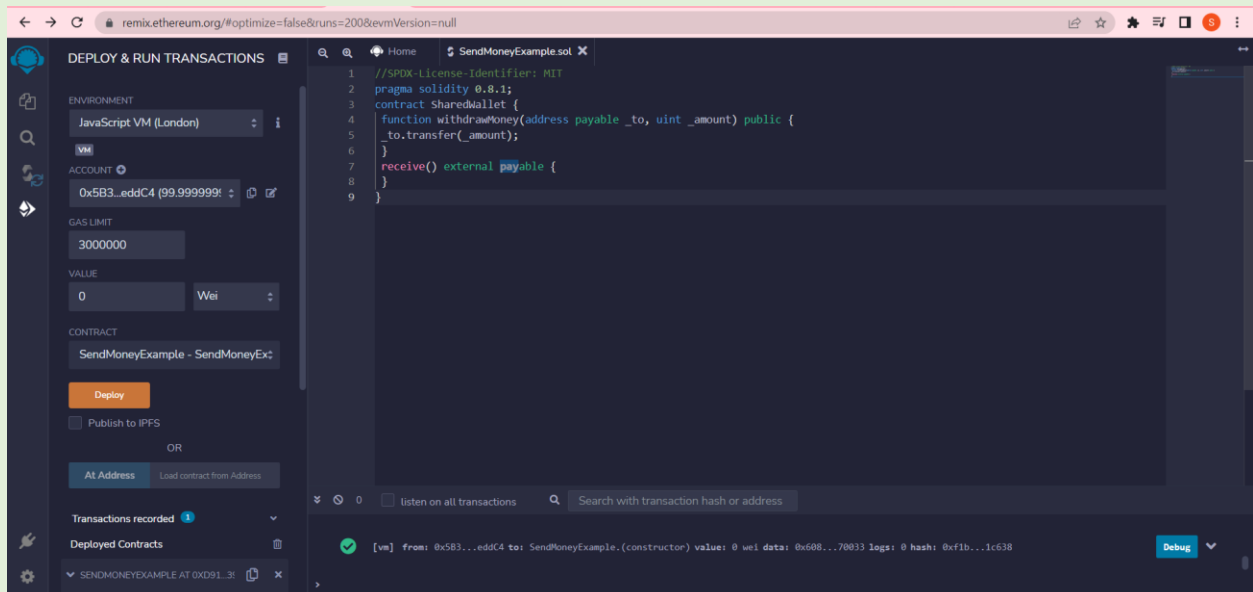
Nama : Shely Belinda Br Ginting (1103190009)



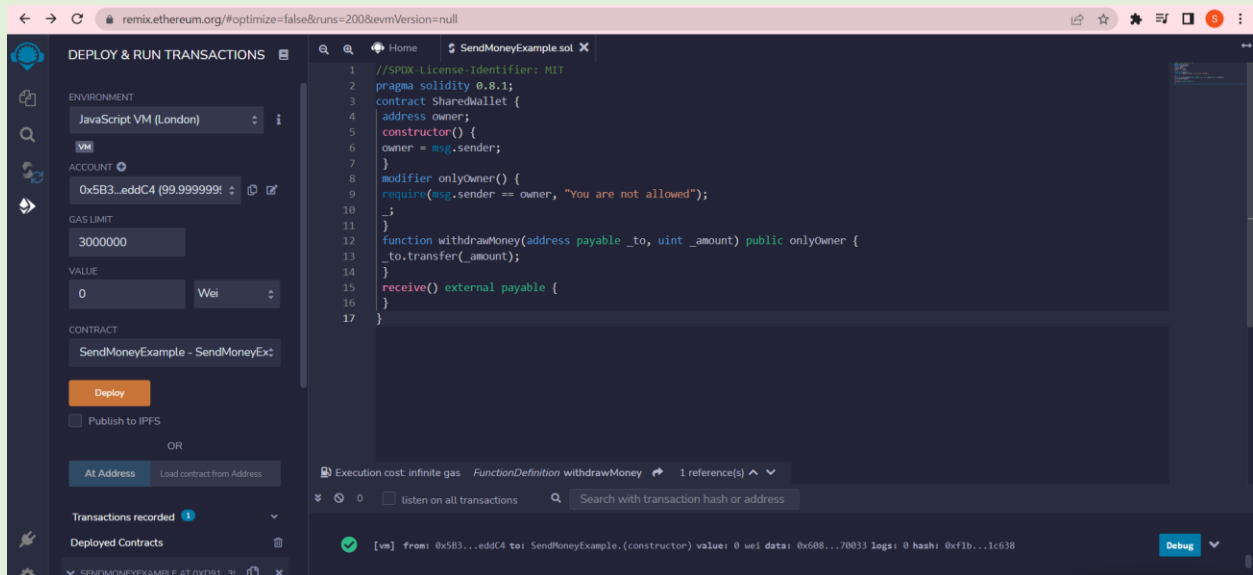
Universitas Telkom

**S1 TEKNIK KOMPUTER
FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
BANDUNG
2021**

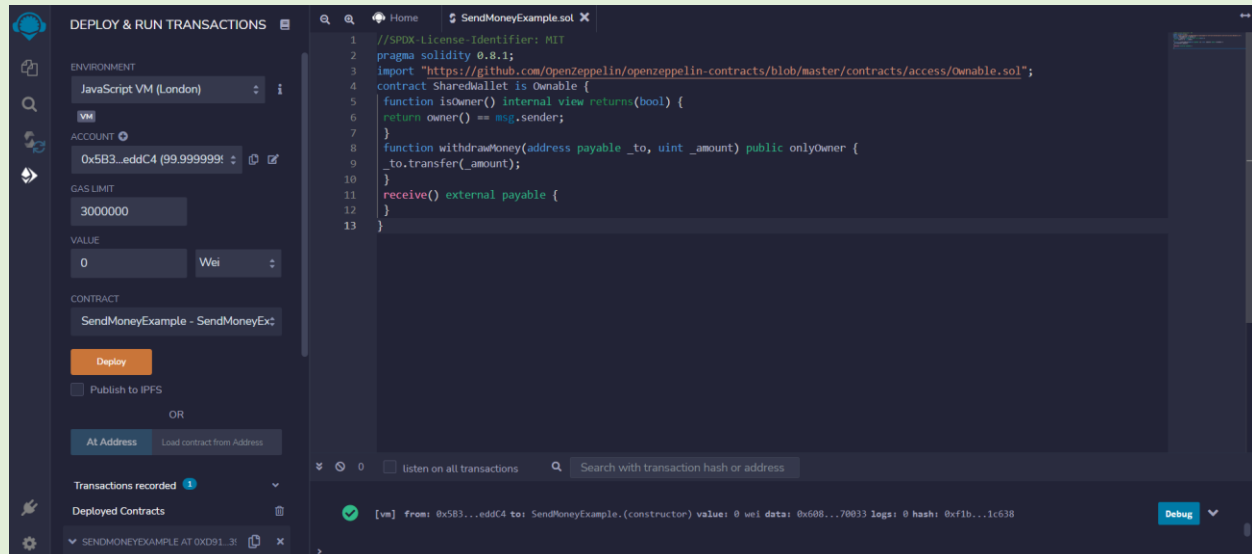
Pertama kita akan membuat file baru lalu membuat kontrak pintar yang dapat menerima Eter dan dimungkinkan untuk menarik Eter, tetapi secara keseluruhan, idak terlalu berguna cukup belum.



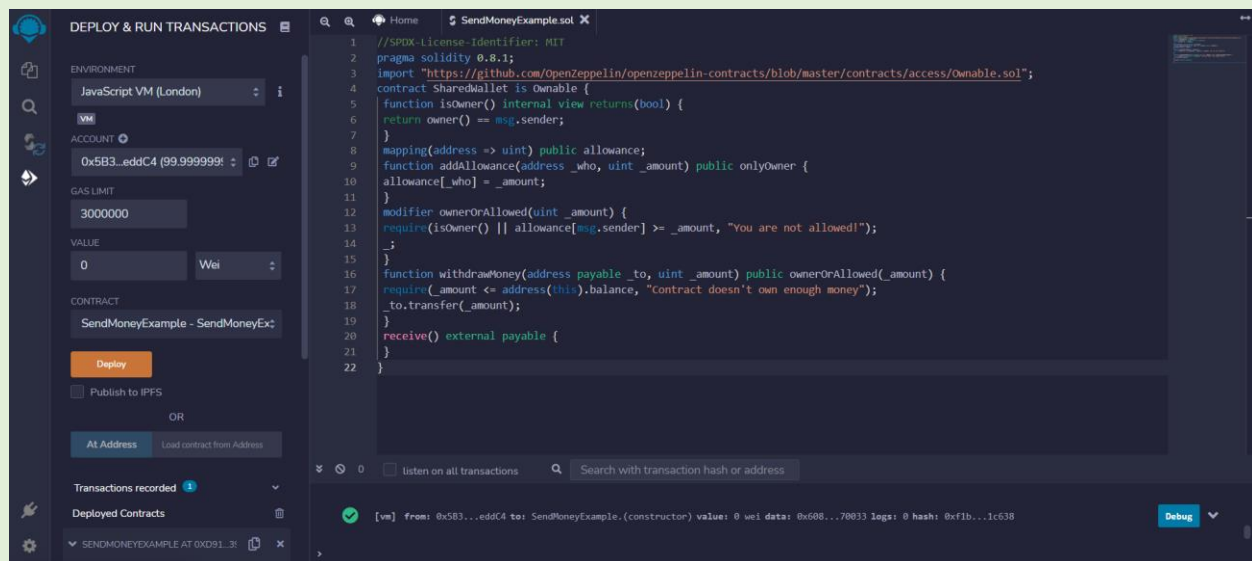
Pada langkah ini kami membatasi penarikan kepada pemilik dompet. Dimana pengguna yang menyebarkan kontrak pintar. selain itu kita juga menambahkan pengubah "onlyOwner" ke fungsi withdrawMoney!



Memiliki logika pemilik secara langsung dalam satu kontrak pintar tidak mudah untuk diaudit. Kita akan kembali kontrak pintar yang telah diaudit dari OpenZeppelin untuk itu. Kontrak OpenZeppelin terbaru tidak memiliki `isOwner()` fungsi lagi, jadi kita harus membuat sendiri. tanda `()` adalah fungsi dari kontrak `Ownable.sol`



Pada langkah ini kita menambahkan pemetaan sehingga kita dapat menyimpan alamat => jumlah uint. selain itu juga kita akan menambahkan modifikator yang memeriksa: Apakah pemiliknya sendiri atau hanya seseorang dengan uang saku?



Agar tidak mengurangi tunjangan penarikan seseorang dan agar mereka dapat terus menerus menarik jumlah yang sama berulang-ulang. maka kita harus mengurangi tunjangan untuk semua orang selain pemilik. Dari semua program yang telah kita jalankan maka kita dapat mengetahui fungsi yang mendasar, kita dapat menyusun kontrak pintar secara berbeda. Untuk membuatnya lebih mudah dibaca, kita bisa istirahat fungsionalitas menjadi dua kontrak pintar yang berbeda. Dengan catatan Perhatikan bahwa karena Allowance adalah Milik, dan SharedWallet adalah Allowance, oleh karena itu berdasarkan properti komutatif, SharedWallet juga Dapat Dimiliki. Kedua kontrak masih dalam file yang sama, jadi kami tidak memiliki impor (belum).

```

1 //SPDX-License-Identifier: MIT
2 pragma solidity 0.8.1;
3 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";
4 contract Allowance is Ownable {
5     function isOwner() internal view returns(bool) {
6         return owner() == msg.sender;
7     }
8     mapping(address => uint) public allowance;
9     function setAllowance(address _who, uint _amount) public onlyOwner {
10         allowance[_who] = _amount;
11     }
12     modifier ownerOrAllowed(uint _amount) {
13         require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
14         _;
15     }
16     function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
17         allowance[_who] -= _amount;
18     }
19 }
20 contract SharedWallet is Allowance {
21     function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
22         require(_amount <= address(this).balance, "Contract doesn't own enough money");
23         if(!isOwner()) {
24             reduceAllowance(msg.sender, _amount);
25         }
26         _to.transfer(_amount);
27     }
28 }

```

Deployed Contracts: SENDMONEYEXAMPLE AT 0xD91...31

Transaction: [vm] from: 0x5B3...eddC4 to: SendMoneyExample.(constructor) value: 0 wei data: 0x608...70033 logs: 0 hash: 0xf1b...1c638

Tambahkan kode program dibawah ini kedalam program yang telah di buat:

[event AllowanceChanged(address indexed _forWho, address indexed _byWhom, uint _oldAmount, uint _newAmount);]

[emit AllowanceChanged(_who, msg.sender, allowance[_who], _amount);]

[emit AllowanceChanged(_who, msg.sender, allowance[_who], allowance[_who] - _amount);]

```

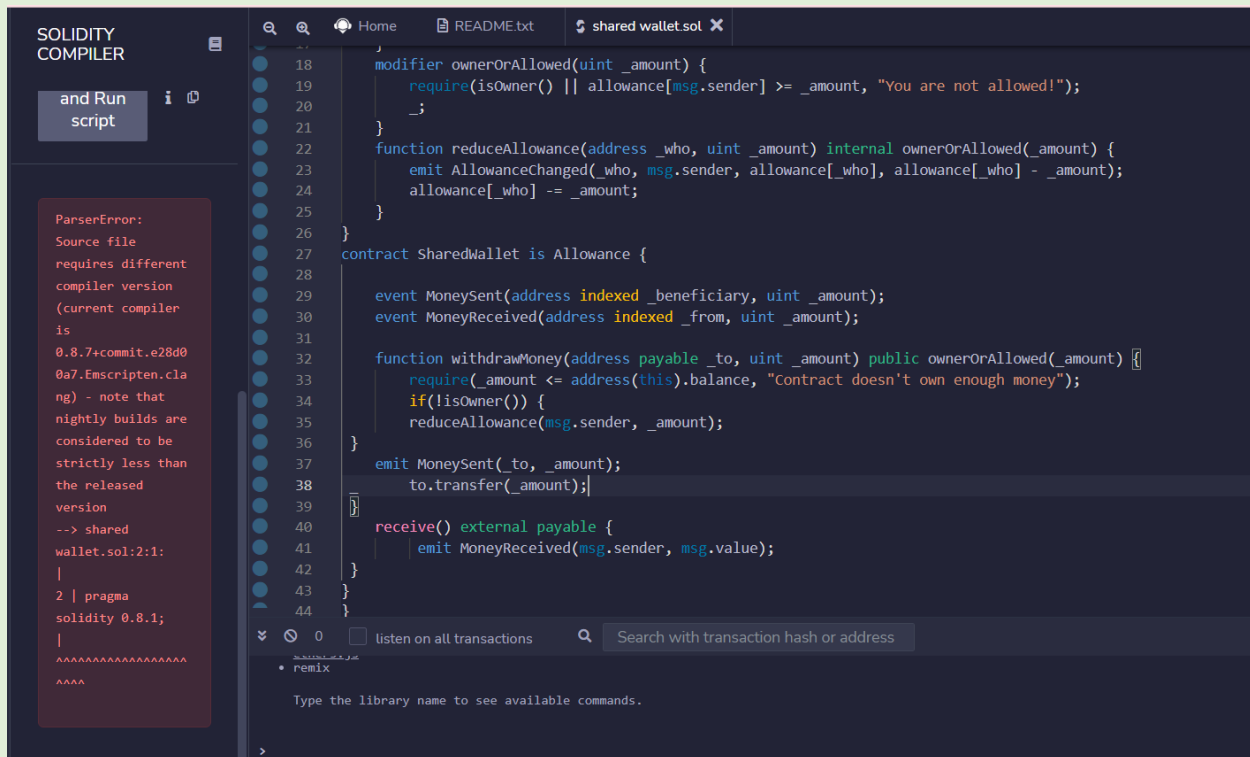
1 //SPDX-License-Identifier: MIT
2 pragma solidity 0.8.1;
3 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";
4 contract Allowance is Ownable {
5     event AllowanceChanged(address indexed _forWho, address indexed _byWhom, uint _oldAmount, uint _newAmount);
6     mapping(address => uint) public allowance;
7     function isOwner() internal view returns(bool) {
8         return owner() == msg.sender;
9     }
10    function setAllowance(address _who, uint _amount) public onlyOwner {
11        emit AllowanceChanged(_who, msg.sender, allowance[_who], _amount);
12        allowance[_who] = _amount;
13    }
14    modifier ownerOrAllowed(uint _amount) {
15        require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
16        _;
17    }
18    function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
19        emit AllowanceChanged(_who, msg.sender, allowance[_who], allowance[_who] - _amount);
20        allowance[_who] -= _amount;
21    }
22 }
23 contract SharedWallet is Allowance {
24     //...
25 }

```

Deployed Contracts: SENDMONEYEXAMPLE AT 0xD91...31

Transaction: [vm] from: 0x5B3...eddC4 to: SendMoneyExample.(constructor) value: 0 wei data: 0x608...70033 logs: 0 hash: 0xf1b...1c638

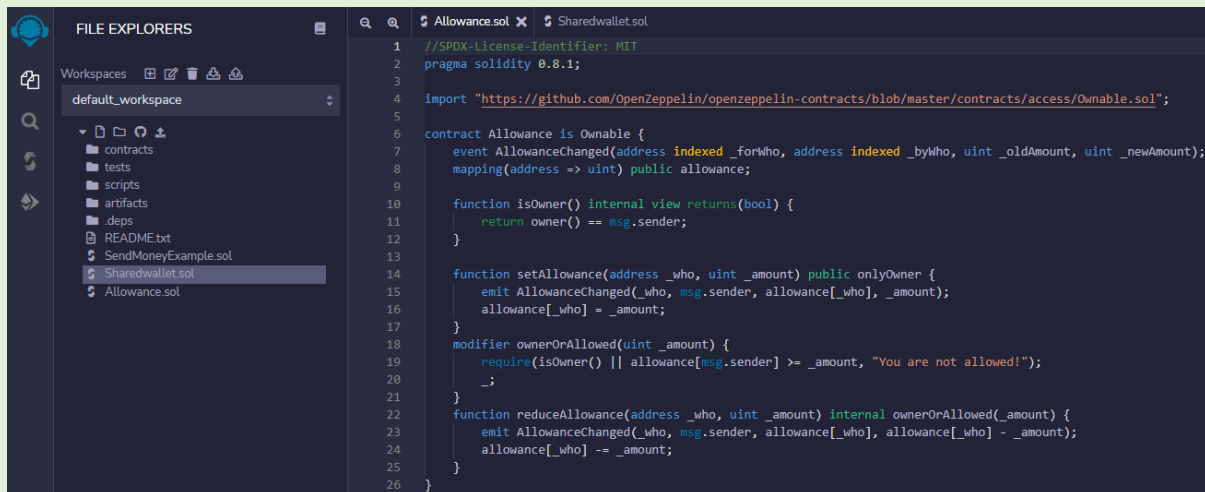
Selanjutnya kita akan menambahkan events kedalam sharedwallet smart kontak



The screenshot shows the Solidity Compiler interface. On the left, a red error message box states: "ParserError: Source file requires different compiler version (current compiler is 0.8.7+commit.e28d00a7.Emscripten.clang) - note that nightly builds are considered to be strictly less than the released version --> sharedwallet.sol:2:1: | 2 | pragma solidity 0.8.1; | ^^^^^^^^^^^^^^^^^^^^^^ ^^^^^". The main editor displays the code for the SharedWallet contract, which includes events for MoneySent and MoneyReceived, and functions for ownerOrAllowed, reduceAllowance, and withdrawMoney. The contract is named SharedWallet and inherits from Allowance.

```
18 modifier ownerOrAllowed(uint _amount) {
19     require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
20 }
21
22 function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
23     emit AllowanceChanged(_who, msg.sender, allowance[_who], allowance[_who] - _amount);
24     allowance[_who] -= _amount;
25 }
26
27 contract SharedWallet is Allowance {
28
29     event MoneySent(address indexed _beneficiary, uint _amount);
30     event MoneyReceived(address indexed _from, uint _amount);
31
32     function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
33         require(_amount <= address(this).balance, "Contract doesn't own enough money");
34         if(!isOwner()) {
35             reduceAllowance(msg.sender, _amount);
36         }
37         emit MoneySent(_to, _amount);
38         _to.transfer(_amount);
39     }
40
41     receive() external payable {
42         emit MoneyReceived(msg.sender, msg.value);
43     }
44 }
```

Step yang terakhir kita akan memindahkan smart kontrak ke separate files and use import functionality



The screenshot shows the VS Code interface. The File Explorer on the left shows a workspace named 'default_workspace' with a folder structure including contracts, tests, scripts, artifacts, and deps. The main editor displays the code for the Allowance.sol contract, which includes a pragma statement for solidity 0.8.1, an import statement for Ownable, and the contract definition for Allowance. The contract includes events for AllowanceChanged and functions for isOwner, setAllowance, and reduceAllowance.

```
1 //SPDX-License-Identifier: MIT
2 pragma solidity 0.8.1;
3
4 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";
5
6 contract Allowance is Ownable {
7     event AllowanceChanged(address indexed _forWho, address indexed _byWho, uint _oldAmount, uint _newAmount);
8     mapping(address => uint) public allowance;
9
10    function isOwner() internal view returns(bool) {
11        return owner() == msg.sender;
12    }
13
14    function setAllowance(address _who, uint _amount) public onlyOwner {
15        emit AllowanceChanged(_who, msg.sender, allowance[_who], _amount);
16        allowance[_who] = _amount;
17    }
18
19    modifier ownerOrAllowed(uint _amount) {
20        require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
21    }
22
23    function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
24        emit AllowanceChanged(_who, msg.sender, allowance[_who], allowance[_who] - _amount);
25        allowance[_who] -= _amount;
26    }
27 }
```

```
1 //SPDX-License-Identifier: MIT
2 pragma solidity 0.8.1;
3
4 import "./Allowance.sol";
5
6 contract Sharedwallet is Allowance {
7     event MoneySent(address indexed _beneficiary, uint _amount);
8     event MoneyReceived(address indexed _from, uint _amount);
9
10    function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
11        require(_amount <= address(this).balance, "Contract doesn't own enough money");
12        if(!isOwner()) {
13            reduceAllowance(msg.sender, _amount);
14        }
15        emit MoneySent(_to, _amount);
16        _to.transfer(_amount);
17    }
18    receive() external payable {
19        emit MoneyReceived(msg.sender, msg.value);
20    }
21 }
```

Dan setelah itu kita deploy

ENVIRONMENT
JavaScript VM (London)

ACCOUNT
0x5B3...eddC4 (99.999999%)

GAS LIMIT
3000000

VALUE
0 Wei

CONTRACT
Allowance - Allowance.sol

Deploy

Publish to IPFS

OR

At Address Load contract from Address

Transactions recorded 1

Deployed Contracts

ALLOWANCE AT 0xD91...39138 (ME)

renounceOwn...

setAllowance address _who, uint256 _an

transferOwner... address newOwner

allowance address

owner

Allowance.sol Sharedwallet.sol

```
1 //SPDX-License-Identifier: MIT
2 pragma solidity 0.8.1;
3
4 import "./Allowance.sol";
5
6 contract SharedWallet is Allowance {
7
8     event MoneySent(address indexed _beneficiary, uint _amount);
9     event MoneyReceived(address indexed _from, uint _amount);
10
11    function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
12        require(_amount <= address(this).balance, "Contract doesn't own enough money");
13        if(!isOwner()) {
14            reduceAllowance(msg.sender, _amount);
15        }
16        emit MoneySent(_to, _amount);
17        _to.transfer(_amount);
18    }
19    receive() external payable {
20        emit MoneyReceived(msg.sender, msg.value);
21    }
22 }
```

listen on all transactions Search with transaction hash or address

Type the library name to see available commands.

creation of Allowance pending...

[va] from: 0x5B3...eddC4 to: Allowance.(constructor) value: 0 wei data: 0x608...10033 logs: 1 hash: 0xa37...9afed

