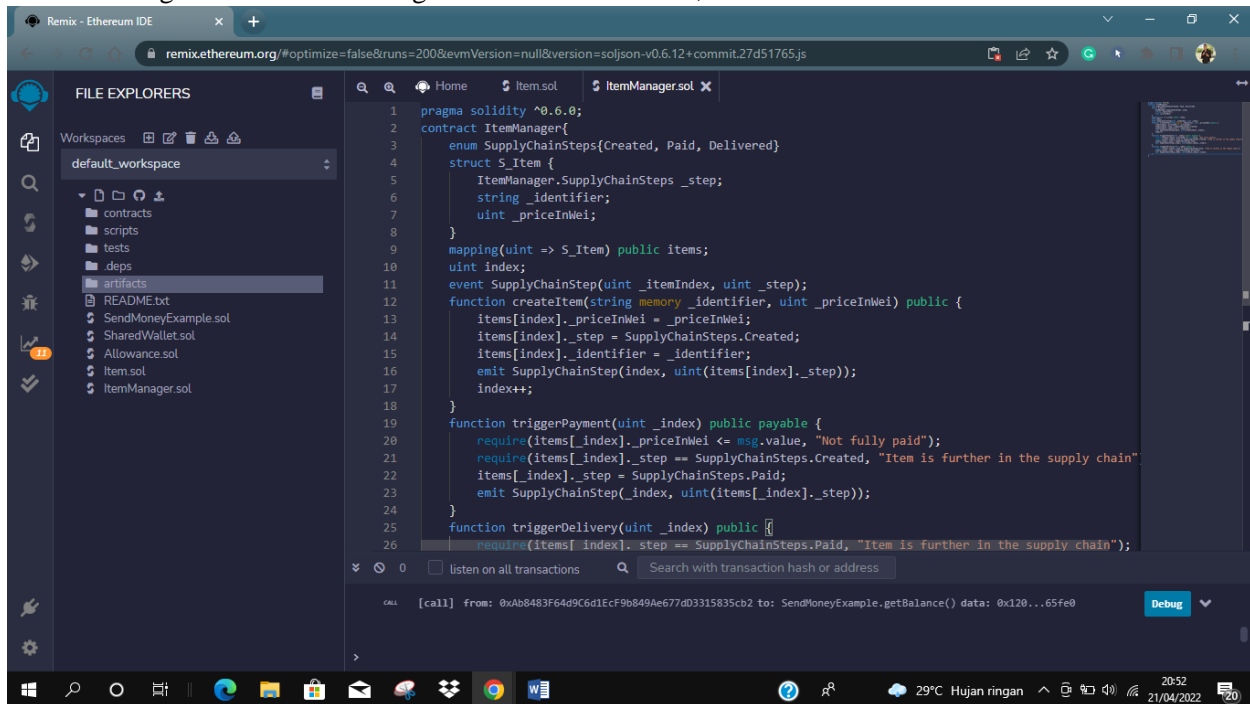


## The Item Manager Smart Contract

The first thing we need is a "Management" Smart Contract, where we can add items.

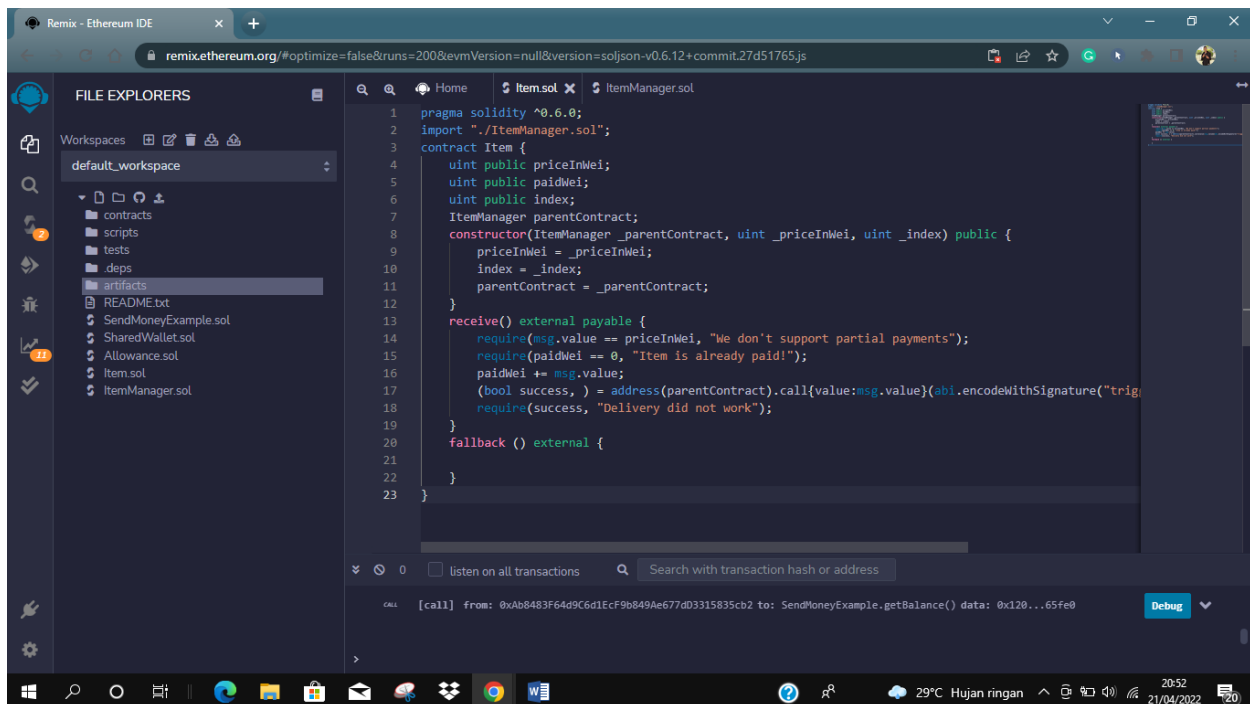


The screenshot shows the Remix IDE interface with the 'ItemManager.sol' file open. The code defines a smart contract for managing items in a supply chain. It includes an enumeration for supply chain steps, a struct for item details, and functions for creating items, triggering payments, and triggering deliveries. The IDE's file explorer on the left shows a project structure with folders for contracts, scripts, tests, and artifacts, and files for README.txt, SendMoneyExample.sol, SharedWallet.sol, Allowance.sol, Item.sol, and ItemManager.sol. The bottom status bar indicates a call from a transaction with data: 0x120...65Fe0.

```
1 pragma solidity ^0.6.0;
2 contract ItemManager{
3     enum SupplyChainSteps{Created, Paid, Delivered}
4     struct S_Item {
5         ItemManager.SupplyChainSteps _step;
6         string _identifier;
7         uint _priceInWei;
8     }
9     mapping(uint => S_Item) public items;
10    uint index;
11    event SupplyChainStep(uint _itemIndex, uint _step);
12    function createItem(string memory _identifier, uint _priceInWei) public {
13        items[index]._priceInWei = _priceInWei;
14        items[index]._step = SupplyChainSteps.Created;
15        items[index]._identifier = _identifier;
16        emit SupplyChainStep(index, uint(items[index]._step));
17        index++;
18    }
19    function triggerPayment(uint _index) public payable {
20        require(items[_index]._priceInWei <= msg.value, "Not fully paid");
21        require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain");
22        items[_index]._step = SupplyChainSteps.Paid;
23        emit SupplyChainStep(_index, uint(items[_index]._step));
24    }
25    function triggerDelivery(uint _index) public {
26        require(items[_index]._step == SupplyChainSteps.Paid, "Item is further in the supply chain");
```

## Item Smart Contract

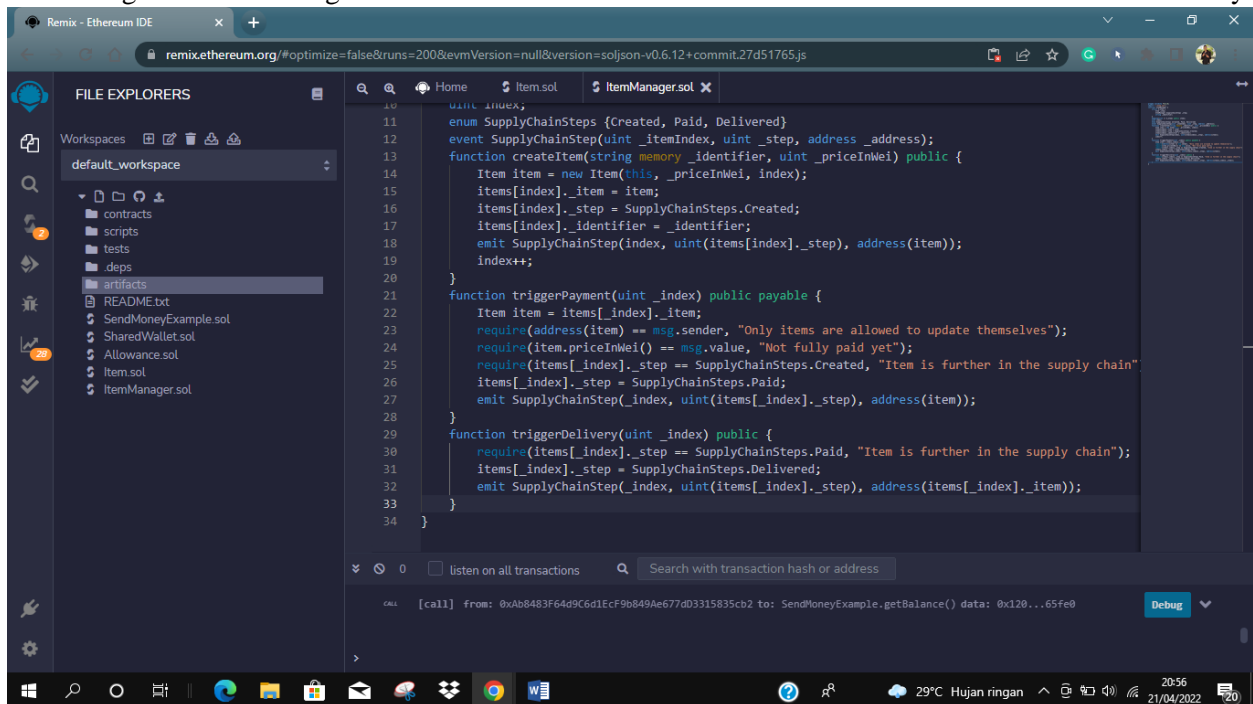
Let's add another smart contract:



The screenshot shows the Remix IDE interface with the 'Item.sol' file open. The code defines a smart contract for an item, which inherits from the 'ItemManager' contract. It includes a constructor to initialize the item's price, index, and parent contract, and a 'receive' function to handle payments. The IDE's file explorer on the left shows the same project structure as the previous screenshot. The bottom status bar indicates a call from a transaction with data: 0x120...65Fe0.

```
1 pragma solidity ^0.6.0;
2 import "../ItemManager.sol";
3 contract Item {
4     uint public priceInWei;
5     uint public paidWei;
6     uint public index;
7     ItemManager parentContract;
8     constructor(ItemManager _parentContract, uint _priceInWei, uint _index) public {
9         priceInWei = _priceInWei;
10        index = _index;
11        parentContract = _parentContract;
12    }
13    receive() external payable {
14        require(msg.value == priceInWei, "We don't support partial payments");
15        require(paidWei == 0, "Item is already paid!");
16        paidWei += msg.value;
17        (bool success, ) = address(parentContract).call{value:msg.value}(abi.encodeWithSignature("trig",
18        require(success, "Delivery did not work");
19    }
20    fallback () external {
21    }
22    }
23 }
```

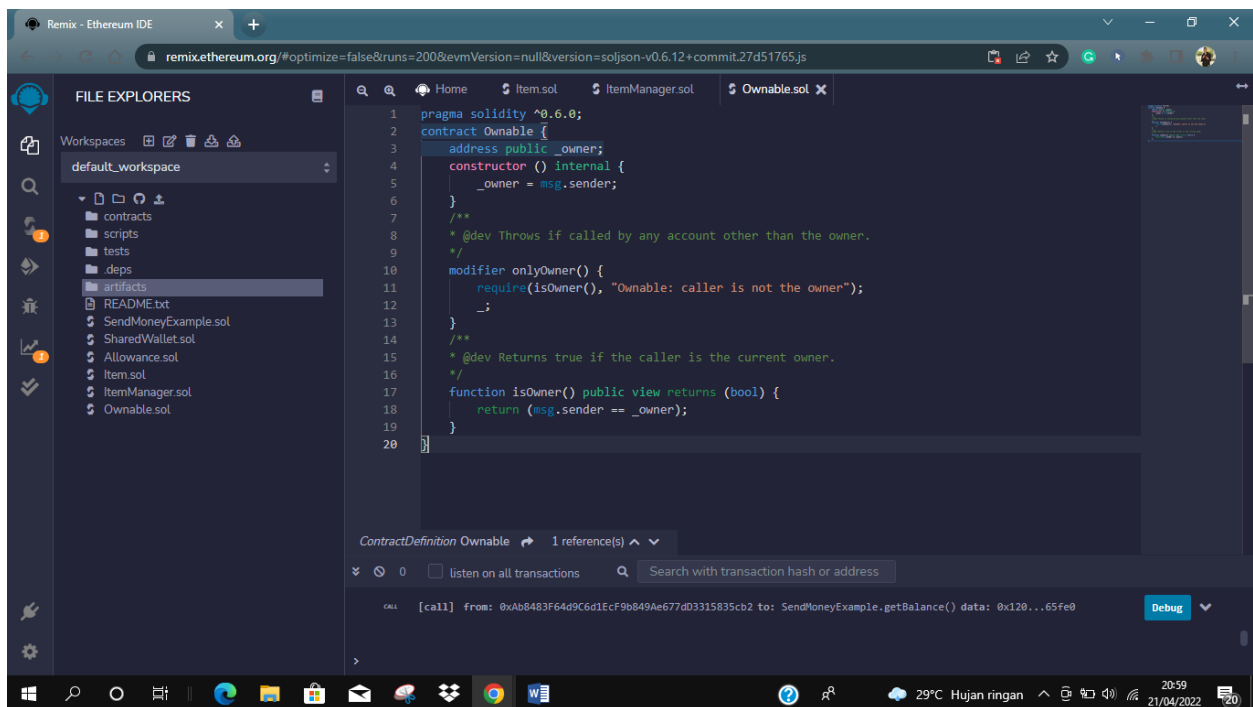
And change the ItemManager Smart Contract to use the Item Smart Contract instead of the Struct only:



```
10  uint _index;
11  enum SupplyChainSteps {Created, Paid, Delivered}
12  event SupplyChainStep(uint _itemIndex, uint _step, address _address);
13  function createItem(string memory _identifier, uint _priceInWei) public {
14      Item item = new Item(this, _priceInWei, _index);
15      items[_index]._item = item;
16      items[_index]._step = SupplyChainSteps.Created;
17      items[_index]._identifier = _identifier;
18      emit SupplyChainStep(_index, uint(items[_index]._step), address(item));
19      _index++;
20  }
21  function triggerPayment(uint _index) public payable {
22      Item item = items[_index]._item;
23      require(address(item) == msg.sender, "Only items are allowed to update themselves");
24      require(item.priceInWei() == msg.value, "Not fully paid yet");
25      require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain");
26      items[_index]._step = SupplyChainSteps.Paid;
27      emit SupplyChainStep(_index, uint(items[_index]._step), address(item));
28  }
29  function triggerDelivery(uint _index) public {
30      require(items[_index]._step == SupplyChainSteps.Paid, "Item is further in the supply chain");
31      items[_index]._step = SupplyChainSteps.Delivered;
32      emit SupplyChainStep(_index, uint(items[_index]._step), address(items[_index]._item));
33  }
34  }
```

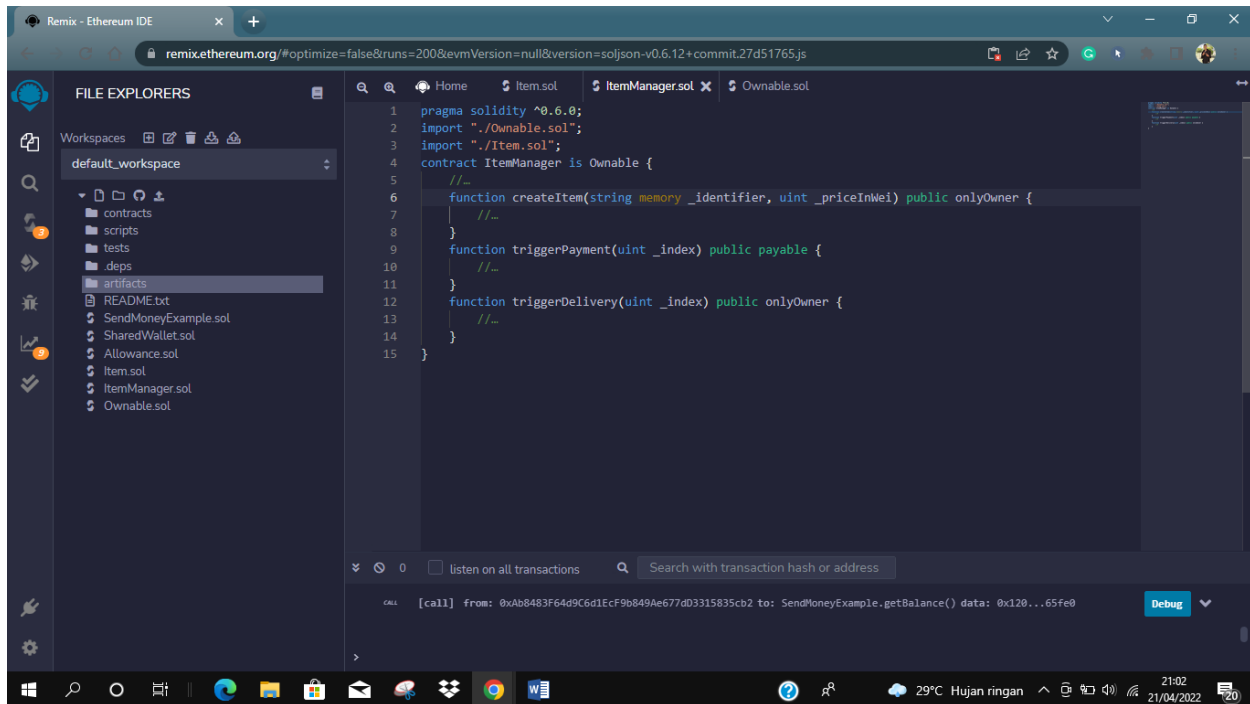
## Owable Functionality

Normally we would add the OpenZeppelin Smart Contracts with the Owable Functionality. But at the time of writing this document they are not updated to solidity 0.6 yet. So, instead we will add our own Owable functionality very much like the one from OpenZeppelin:



```
1  pragma solidity ^0.6.0;
2  contract Owable {
3      address public _owner;
4      constructor () internal {
5          _owner = msg.sender;
6      }
7      /**
8       * @dev Throws if called by any account other than the owner.
9       */
10     modifier onlyOwner() {
11         require(isOwner(), "Owable: caller is not the owner");
12         _;
13     }
14     /**
15      * @dev Returns true if the caller is the current owner.
16      */
17     function isOwner() public view returns (bool) {
18         return (msg.sender == _owner);
19     }
20 }
```

Then modify the ItemManager so that all functions, that should be executable by the "owner only" have the correct modifier:



## Install Truffle

To install truffle open a terminal (Mac/Linux) or a PowerShell (Windows 10)

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User> npm install -g truffle@5.1.8
npm WARN deprecated mkdirp@0.5.1: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises in 1.x.)
changed 27 packages, and audited 28 packages in 15s
3 critical severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
PS C:\Users\User>
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User> npm install -g truffle@5.1.8
npm WARN deprecated mkdirp@0.5.1: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises in 1.x.)
changed 27 packages, and audited 28 packages in 15s
3 critical severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
PS C:\Users\User> mkdir s06-eventtrigger

Directory: C:\Users\User

Mode                LastWriteTime         Length Name
----                -
d-----          4/21/2022   9:28 PM             s06-eventtrigger

PS C:\Users\User> cd s06-eventtrigger
PS C:\Users\User\s06-eventtrigger> ls
PS C:\Users\User\s06-eventtrigger>
```