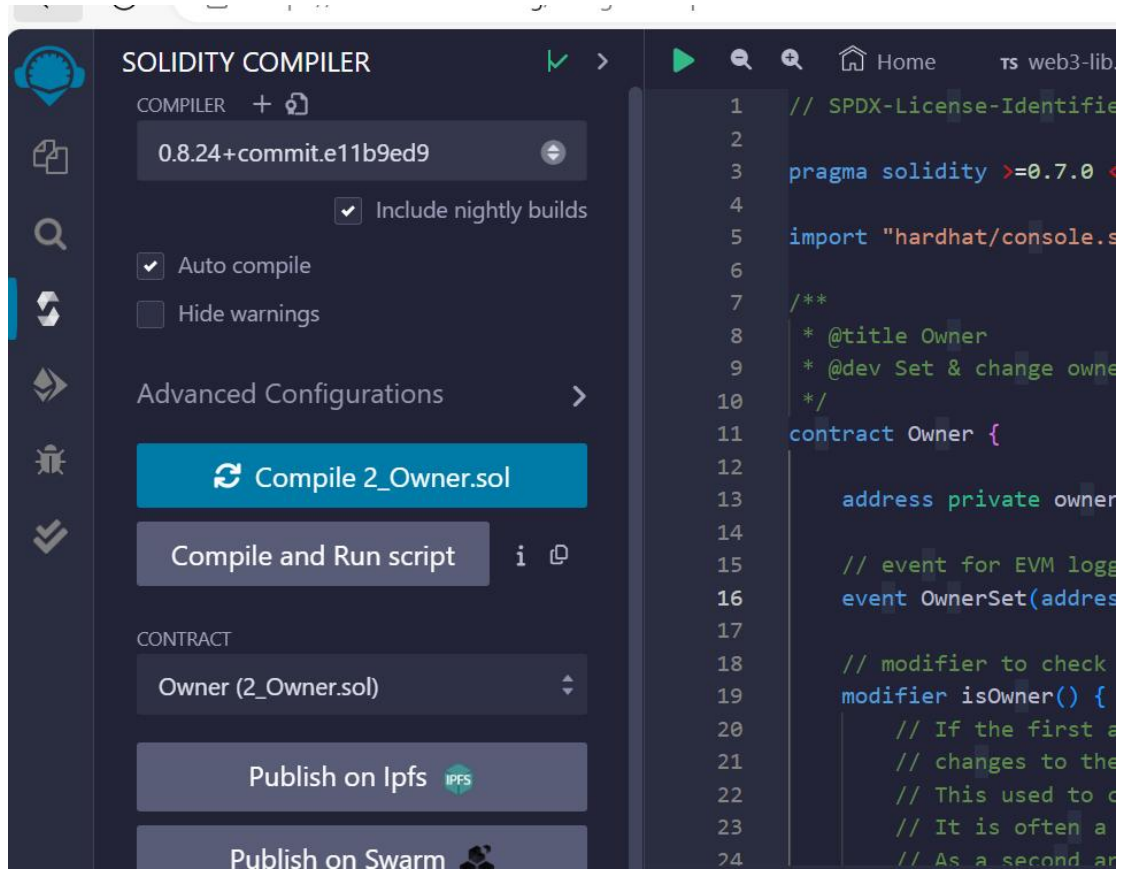












## TP2 : BLOCKCHAIN TIA ESTELLE





DEPLOY & RUN TRANSACTIONS ✓

Transactions recorded **1** ⓘ >

Deployed Contracts

▼ COUNTER AT 0XD91...39138 (MEM) ⓘ

Balance: 0. ETH

dec

inc

count

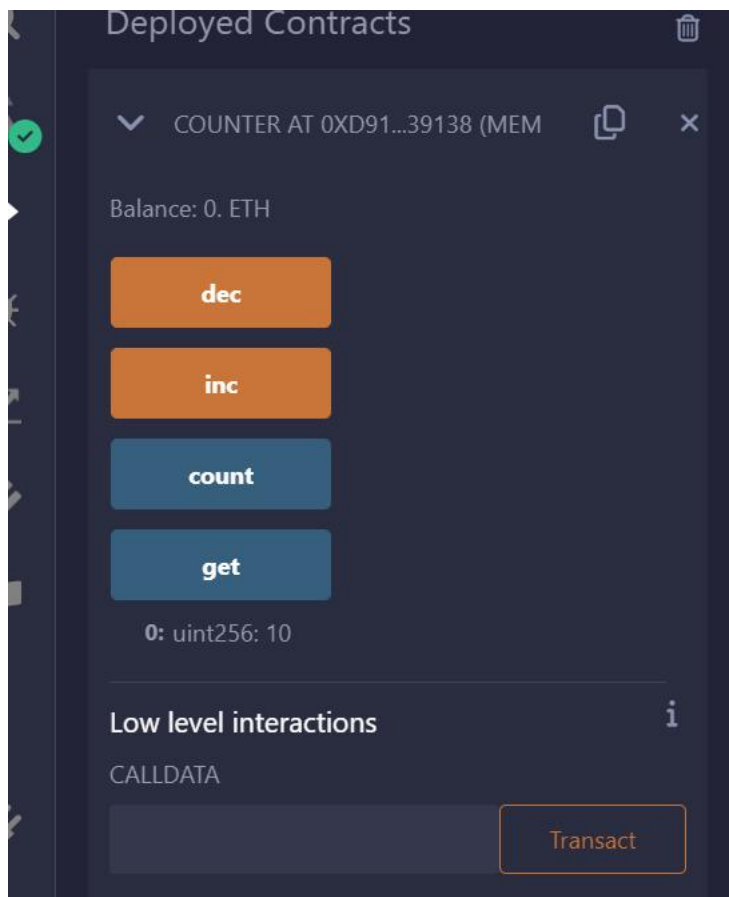
get

0: uint256: 0

Low level interactions

CALLDATA

Transact



```
1 // SPDX-License-Identifier: MIT
2 // compiler version must be greater than or equal to 0.8.3 and
3 pragma solidity ^0.8.3;
4
5 contract HelloWorld {
6     string public greet = "Hello World!";
7 }
```



contract.

Don't worry if you didn't understand some concepts like *visibility*, *data types*, or *state variables*. We will look into them in the following sections.

**Assignment**

1. Delete the HelloWorld contract and its content.
2. Create a new contract named "MyContract".

```
3 pragma solidity ^0.8.3;
4
5 contract MyContract {
6     string public name = "Alice";
7 }
```



< 2/19 >

following sections.

★

Assignment

1. Delete the HelloWorld contract and its content.

2. Create a new contract named "MyContract".

3. The contract should have a public state variable called "name" of the type string.

4. Assign the value "Alice" to your new variable.

Check Answer

Show answer

Next

Well done! No errors.

```
1 // SPDX-License-Identifier: MIT
2 // compiler version must be greater than 0.8.3
3 pragma solidity ^0.8.3;
4
5 contract MyContract {
6     string public name = "Alice";
7 }
```

LEARNETH

< 3/19 >

the available variable `addr`.

2. Create a `public` variable called `neg` that is a negative number, decide upon the type.

3. Create a new variable, `newU` that has the smallest `uint` size type and the smallest `uint` value and is `public`.

Tip: Look at the other address in the contract or search the internet for an Ethereum address.

Check Answer Show answer

Next

```

21 Like uint, different ranges are available from int8 to int256
22 */
23 int8 public i8 = -1;
24 int public i256 = 456;
25 int public i = -123; // int is same as int256
26
27 address public addr = 0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c;
28
29 // Default values
30 // Unassigned variables have a default value
31 bool public defaultBool; // false
32 uint public defaultUint; // 0
33 int public defaultInt; // 0
34 address public defaultAddr; // 0x0000000000000000000000000000000000000000
35
36 address public newAddr = 0x0000000000000000000000000000000000000000;
37 int public neg = -12;
38 uint8 public newU = 0;
39 }

```

● 4/19

LEARNETH

< 4/19 >

Local Variables, and Global Variables.

★ Assignment

1. Create a new public state variable called `blockNumber`.

2. Inside the function `doSomething()`, assign the value of the current block number to the state variable `blockNumber`.

Tip: Look into the global variables section of the Solidity documentation to find out how to read the current block number.

Check Answer Show answer

Next

Well done! No errors.

```

8 /
9 uint public num = 123;
10 uint public blockNumber;
11
12 function doSomething() public { 22338 gas
13     // Local variables are not saved to the blockchain.
14     uint i = 456;
15
16     blockNumber = block.number;
17
18     // Here are some global variables
19     uint timestamp = block.timestamp; // Current block timestamp
20     address sender = msg.sender; // address of the caller
21 }

```

● 5/19

LEARNETH

< 5/19 >

We will explore the particularities of Solidity functions in more detail in the following sections.

Watch a video tutorial on Functions.

★ Assignment

1. Create a public state variable called `b` that is of type `bool` and initialize it to `true`.

2. Create a public function called `get_b` that returns the value of `b`.

Check Answer Show answer

Next

Well done! No errors.

```

13 // You can read from a state variable without sending a transaction.
14 function get() public view returns (uint) { 2481 gas
15     return num;
16 }
17
18 function get_b() public view returns (bool) { 2545 gas
19     return b;
20 }
21 }

```

LearnEthereum is modifying .learneth/Solidity Beginner Course/5.1 Functions - Reading and Writing

● 6/19

and pure can save gas cost and make the code more readable and easier to maintain. Pure functions don't have any side effects and will always return the same result if you pass the same arguments.

Watch a video tutorial on View and Pure Functions.

★ **Assignment**

Create a function called `addToX2` that takes the parameter `y` and updates the state variable `x` with the sum of the parameter and the state variable `x`.

Check Answer Show answer

Next

Well done! No errors.

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.3;
3
4 contract ViewAndPure {
5     uint public x = 1;
6
7     // Promise not to modify the state.
8     function addToX(uint y) public view returns (uint) { infinite gas
9         return x + y;
10    }
11
12    function addToX2(uint y) public { infinite gas
13        x=x + y;
14    }
15
16    // Promise not to modify or read from the state.
17    function add(uint i, uint j) public pure returns (uint) { infinite gas
18        return i + j;
19    }
20 }
```

LearnEth is modifying .learneth/Solidity Beginner Course/5.2 Functions - View and Pure Functions

● 7/29

large arrays as inputs might fail when the gas costs are too high, a function using a smaller array might still be able to execute.

Watch a video tutorial on Function Outputs.

★ **Assignment**

Create a new function called `returnTwo` that returns the values `-2` and `true` without using a return statement.

Check Answer Show answer

Next

Well done! No errors.

```
77     return arr;
78 }
79
80 function returnTwo() 479 gas
81     public
82     pure
83     returns(
84         int i ,
85         bool a
86     )
87 {
88     i= -2;
89     a=true;
90 }
```

LearnEth is modifying .learneth/Solidity Beginner Course/5.4 Functions - Inputs and Outputs

8/19

large arrays as inputs might fail when the gas costs are too high, a function using a smaller array might still be able to execute.

Watch a video tutorial on Function Outputs.

★ **Assignment**

Create a new function called `returnTwo` that returns the values `-2` and `true` without using a return statement.

Check Answer Show answer

Next

Well done! No errors.

```
77     return arr;
78 }
79
80 function returnTwo() 479 gas
81     public
82     pure
83     returns(
84         int i ,
85         bool a
86     )
87 {
88     i= -2;
89     a=true;
90 }
```

LearnEth is modifying .learneth/Solidity Beginner Course/5.4 Functions - Inputs and Outputs

● 9/19



★ **Assignment**

1. Create a public `uint` state variable called `count` in the `Loop` contract.
2. At the end of the for loop, increment the count variable by 1.
3. Try to get the count variable to be equal to 9, but make sure you don't edit the `break` statement.

Check Answer Show answer

Next

Well done! No errors.

```

5  uint public count;
6  function loop() public {
7      // for loop
8      for (uint i = 0; i < 10; i++) {
9          if (i == 5) {
10             // Skip to next iteration with continue
11             continue;
12         }
13         if (i == 5) {
14             // Exit loop with break
15             break;
16         }
17         count ++;
18     }
19
20     // while loop

```

● 10/19

the `IFelse` contract:

- That takes in a `uint` as an argument.
- The function returns `true` if the argument is even, and `false` if the argument is odd.
- Use a ternary operator to return the result of the `evenCheck` function.

Tip: The modulo (%) operator produces the remainder of an integer division.

Check Answer Show answer

Next

Well done! No errors.

```

18     // }
19     // return 2;
20
21     // shorthand way to write if / else statement
22     return _x < 10 ? 1 : 2;
23 }
24 function evenCheck(uint y) public pure returns (bool){
25
26     if(y%2 == 0){
27         return true;
28     }
29     else{
30         return false;
31     }
32 }
33 }

```

● 12/19

★ **Assignment**

1. Initialize a public fixed-sized array called `arr3` with the values 0, 1, 2. Make the size as small as possible.
2. Change the `getArr()` function to return the value of `arr3`.

Check Answer Show answer

Next

Well done! No errors.

```

17     // But this function should be avoided for
18     // arrays that can grow indefinitely in length.
19     function getArr() public view returns (uint[3] memory) {
20         return arr3;
21     }
22
23     function push(uint i) public {
24         // Append to array
25         // This will increase the array length by 1.
26         arr.push(i);
27     }
28
29     function pop() public {
30         // Remove last element from array

```

● 13/19

remove to work with the mapping balances.

3. Change the function `set` to create a new entry to the balances mapping, where the key is the address of the parameter and the value is the balance associated with the address of the parameter.

Check Answer Show answer

Next

Well done! No errors.

```

24
25 contract NestedMapping {
26     // Nested mapping (mapping from address to another mapping)
27     mapping(address => mapping(uint => bool)) public nested;
28
29     function get(address _addr1, uint _i) public view returns (bool) {
30         // You can get values from a nested mapping
31         // even when it is not initialized
32         return nested[_addr1][_i];
33     }
34
35     function set(
36         address _addr1,
37         uint _i,
38         bool _val

```

● 14/19

To access a member of a struct we can use the dot operator (line 33).

### Updating structs

To update a structs' member we also use the dot operator and assign it a new value (lines 39 and 45).

[Watch a video tutorial on Structs.](#)

★ **Assignment**

Create a function `remove` that takes a `uint` as a parameter and deletes a struct member with the given index in the `todos` mapping.

Check Answer
Show answer

Next

Well done! No errors.

```

35
36
37 // update text
38 function update(uint _index, string memory _text) public {
39     Todo storage todo = todos[_index];
40     todo.text = _text;
41 }
42
43 // update completed
44 function toggleCompleted(uint _index) public {
45     Todo storage todo = todos[_index];
46     todo.completed = !todo.completed;
47 }
48 function remove(uint _index) public {
49     delete todos[_index];
50 }

```

● 15/19

the default value to 0.

[Watch a video tutorial on Enums.](#)

★ **Assignment**

- Define an enum type called `Size` with the members `S`, `M`, and `L`.
- Initialize the variable `sizes` of the enum type `Size`.
- Create a getter function `getSize()` that returns the value of the variable `sizes`.

Check Answer
Show answer

Next

Well done! No errors.

```

29 // Canceled - 4
30 function getSize() public view returns(Size){
31     return sizes;
32 }
33 function get() public view returns (Status) {
34     return status;
35 }
36
37 // Update status by passing uint into input
38 function set(Status _status) public {
39     status = _status;
40 }
41
42 // You can update to a specific enum like this
43 function cancel() public {
44     status = Status.Canceled;
45 }

```

● 16/19

- Create a new struct `myMemStruct3` with the data location `memory` inside the function `f` and assign it the value of `myStruct`. Change the value of the `myMemStruct3` member `foo` to 3.
- Let the function `f` return `myStruct`, `myMemStruct2`, and `myMemStruct3`.

Tip: Make sure to create the correct return types for the function `f`.

Check Answer
Show answer

Next

Well done! No errors.

```

20 MyStruct memory myMemStruct = MyStruct(0);
21 MyStruct memory myMemStruct2 = myMemStruct;
22 myMemStruct2.foo = 1;
23
24 MyStruct memory myMemStruct3 = myStruct;
25 myMemStruct3.foo = 3;
26 return (myStruct, myMemStruct2, myMemStruct3);
27 }
28
29 function _f(
30     uint[] storage _arr,
31     mapping(uint => address) storage _map,
32     MyStruct storage _myStruct
33 ) internal {
34     // do something with storage variables
35 }

```

● 17/19



Watch a video tutorial on Ether and Wei.

★ **Assignment**

1. Create a public uint called `oneGwei` and set it to 1 `gwei`.
2. Create a public bool called `isOneGwei` and set it to the result of a comparison operation between 1 `gwei` and `10^9`.

Tip: Look at how this is written for `gwei` and `ether` in the contract.

Check Answer Show answer

Next

Well done! No errors.

```
4 contract EtherUnits {
5     uint public oneWei = 1 wei;
6     // 1 wei is equal to 1
7     bool public isOneWei = 1 wei == 1;
8
9     uint public oneEther = 1 ether;
10    // 1 ether is equal to 10^18 wei
11    bool public isOneEther = 1 ether == 1e18;
12
13    uint public oneGwei = 1 gwei;
14    bool public isOneGwei = 1 gwei == 1e9;
15 }
```

0 ☐ listen on all transactions Search with transaction hash or address

Type the library name to see available commands.

● 19/19

parameters or function code.

4. Create a withdraw function that is public and sends the total balance of the contract to the `owner` address.

Tip: Test your contract by deploying it from one account and then sending Ether to it from another account. Then execute the withdraw function.

Check Answer Show answer

Next

Well done! No errors.

```
55     owner = msg.sender;
56 }
57 function donate() public payable{ 142 gas
58 }
59 }
60 function withdraw() public{ infinite gas
61     uint amount = address(this).balance;
62     (bool sent, bytes memory data) = owner.call{
63         value: amount
64     }("");
65     require(sent, "erreur lors de l'envoi d'ETH");
66 }
67 }
```

0 ☐ listen on all transactions Search with transaction hash or address

Type the library name to see available commands.