Blockchain

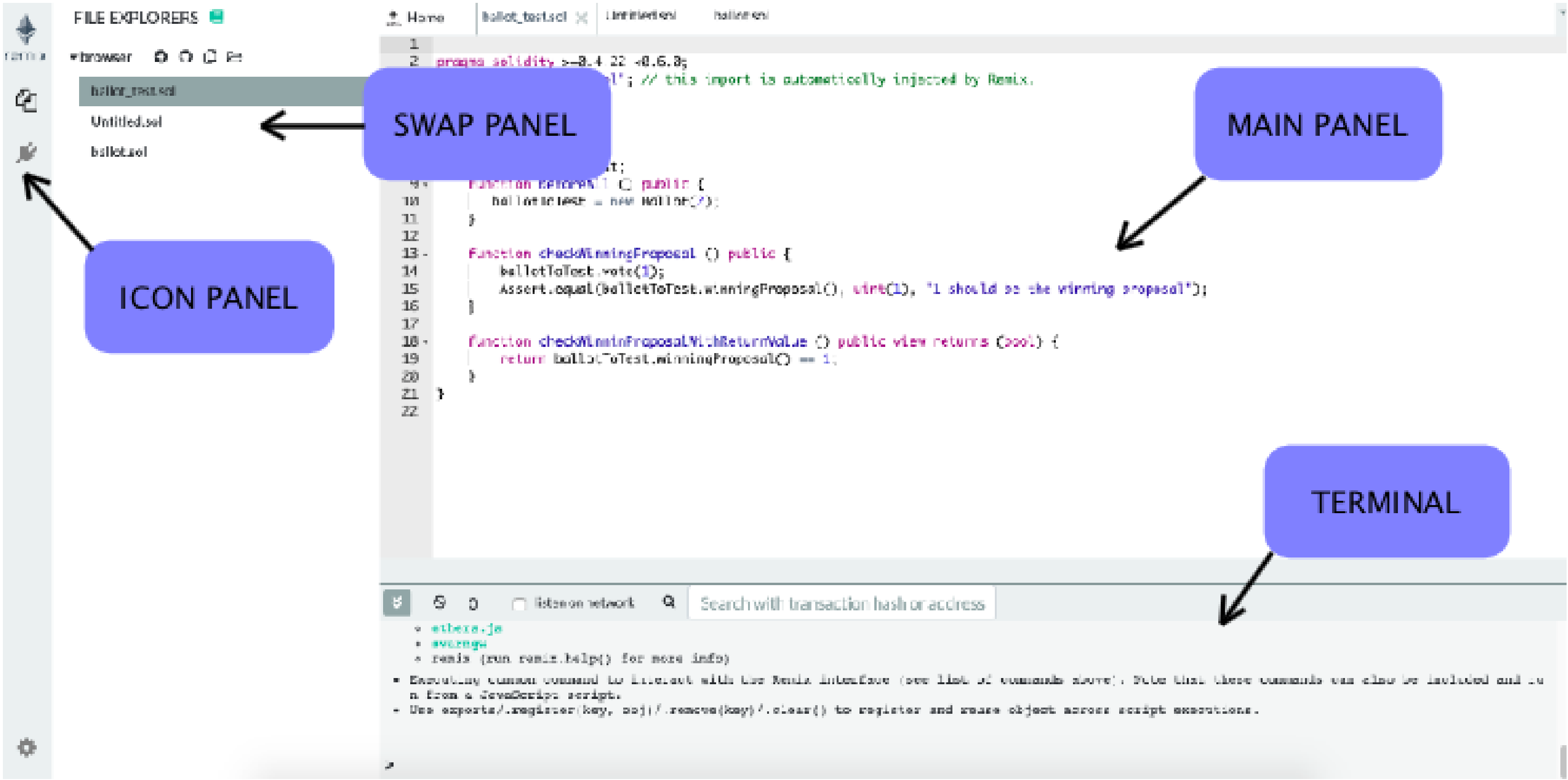
Programming with Solidity

Dr. Sarwan Singh

NIELIT Chandigarh

Agenda

- Solidity programming constructs
- Remix IDE
    - Compile, deploy…
- pragma directive
- Datatype
- Keywords
- Operators

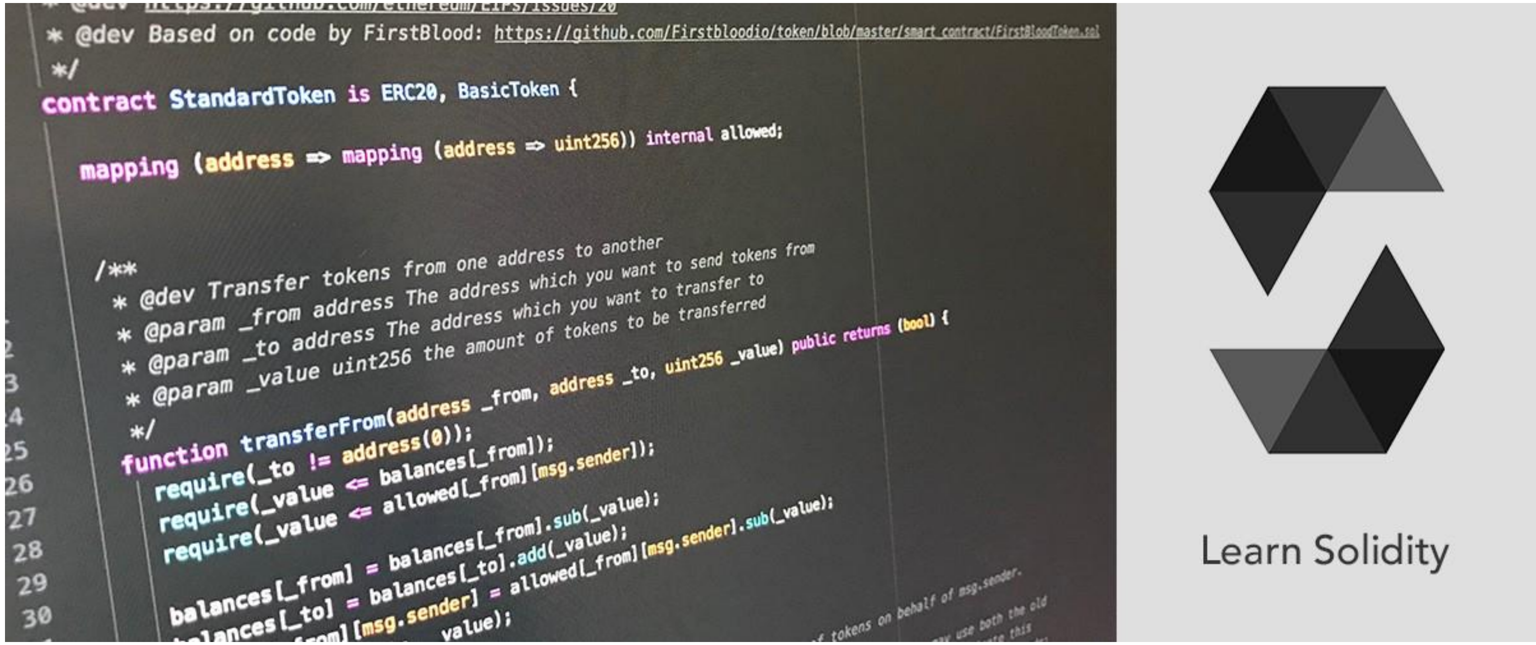


```
pragma solidity >=0.4.22 <0.6.0;
                  l'; // this import is automatically injected by Remix.

                  t;
    function beforeAll () public {
        ballotToTest = new Ballot(2);
    }

    function checkWinningProposal () public {
        ballotToTest.vote(1);
        Assert.equal(ballotToTest.winningProposal(), uint(1), "1 should be the winning proposal");
    }

    function checkWinninProposalWithReturnValue () public view returns (bool) {
        return ballotToTest.winningProposal() == 1;
    }
}
```

```
  • ethers.js
  • swarmgw
  • remix (run remix.help() for more info)
• Executing common command to interact with the Remix interface (see list of commands above). Note that these commands can also be included and ru
  n from a JavaScript script.
• Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script executions.
```

References

- Medium.com – Blockchain
- solidity.readthedocs.io
- tutorialspoint.com
- Dappuniversity.com
- Remix.readthedocs.io



```
 * @dev https://github.com/ethereum/EIPs/issues/20
 * @dev Based on code by FirstBlood: https://github.com/Firstbloodio/token/blob/master/smart_contract/FirstBloodToken.sol
 */
contract StandardToken is ERC20, BasicToken {

  mapping (address => mapping (address => uint256)) internal allowed;


  /**
   * @dev Transfer tokens from one address to another
   * @param _from address The address which you want to send tokens from
   * @param _to address The address which you want to transfer to
   * @param _value uint256 the amount of tokens to be transferred
   */
  function transferFrom(address _from, address _to, uint256 _value) public returns (bool) {
    require(_to != address(0));
    require(_value <= balances[_from]);
    require(_value <= allowed[_from][msg.sender]);

    balances[_from] = balances[_from].sub(_value);
    balances[_to] = balances[_to].add(_value);
    allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
```

Solidity – an Introduction

- Solidity is an object-oriented, high-level language for implementing smart contracts. Smart contracts are programs which govern the behavior of accounts within the Ethereum state.
- Solidity was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine (EVM).
- Solidity is statically typed, supports inheritance, libraries and complex user-defined types among other features.
- With Solidity you can create contracts for uses such as voting, crowdfunding, blind auctions, and multi-signature wallets.

Source : solidity.readthedocs.io

Solidity

- A Solidity source files can contain an any number of contract definitions, import directives and pragma directives.

```
pragma solidity >=0.4.0  <0.6.0;
contract SimpleStorage {
   uint  storedData;
   function set(uint x) public {
      storedData = x;
   }
   function get() public view returns (uint) {
      return storedData;
   }
}
```

Compile-Deploy… *first application*

- https://remix.ethereum.org/
- Step 1 – type/Copy the (given) code in Remix IDE Code Section.
- Step 2 – Under Compile Tab, click Start to Compile button.
- Step 3 – Under Run Tab, click Deploy button.
- Step 4 – Under Run Tab, Select Solidity Test at 0x... in drop-down.
- Step 5 – Click get *Button* to display the result.

Pragma

```
pragma solidity >=0.4.0 <0.6.0;
```

- The first line is a pragma directive which tells that the source code is written for Solidity version 0.4.0 or anything newer that does not break functionality up to, but not including, version 0.6.0.
- A pragma directive is always local to a source file and if you import another file, the pragma from that file will not automatically apply to the importing file.

```
pragma solidity ^0.4.0
```

- pragma for a file which will not compile earlier than version 0.4.0 and it will also not work on a compiler starting from version 0.5.0

Contract

- A Solidity contract is a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain.
- The line uint storedData declares a state variable called storedData of type uint and the functions set and get can be used to modify or retrieve the value of the variable.

```
pragma solidity >=0.4.0  <0.6.0;
contract SimpleStorage {
    uint  storedData;
    function set(uint x) public {
        storedData = x;
    }
    function get() public view returns
 (uint) {
        return storedData;
    }
}
```

Comments

Solidity supports both C-style and C++-style comments, Thus –

- Any text between a // and the end of a line is treated as a comment and is ignored by Solidity Compiler.
- Any text between the characters /* and */ is treated as a comment. This may span multiple lines.

Import files

- Solidity supports import statements that are very similar to those available in JavaScript.
- The following statement imports all global symbols from "filename“.

```
import "filename";
```

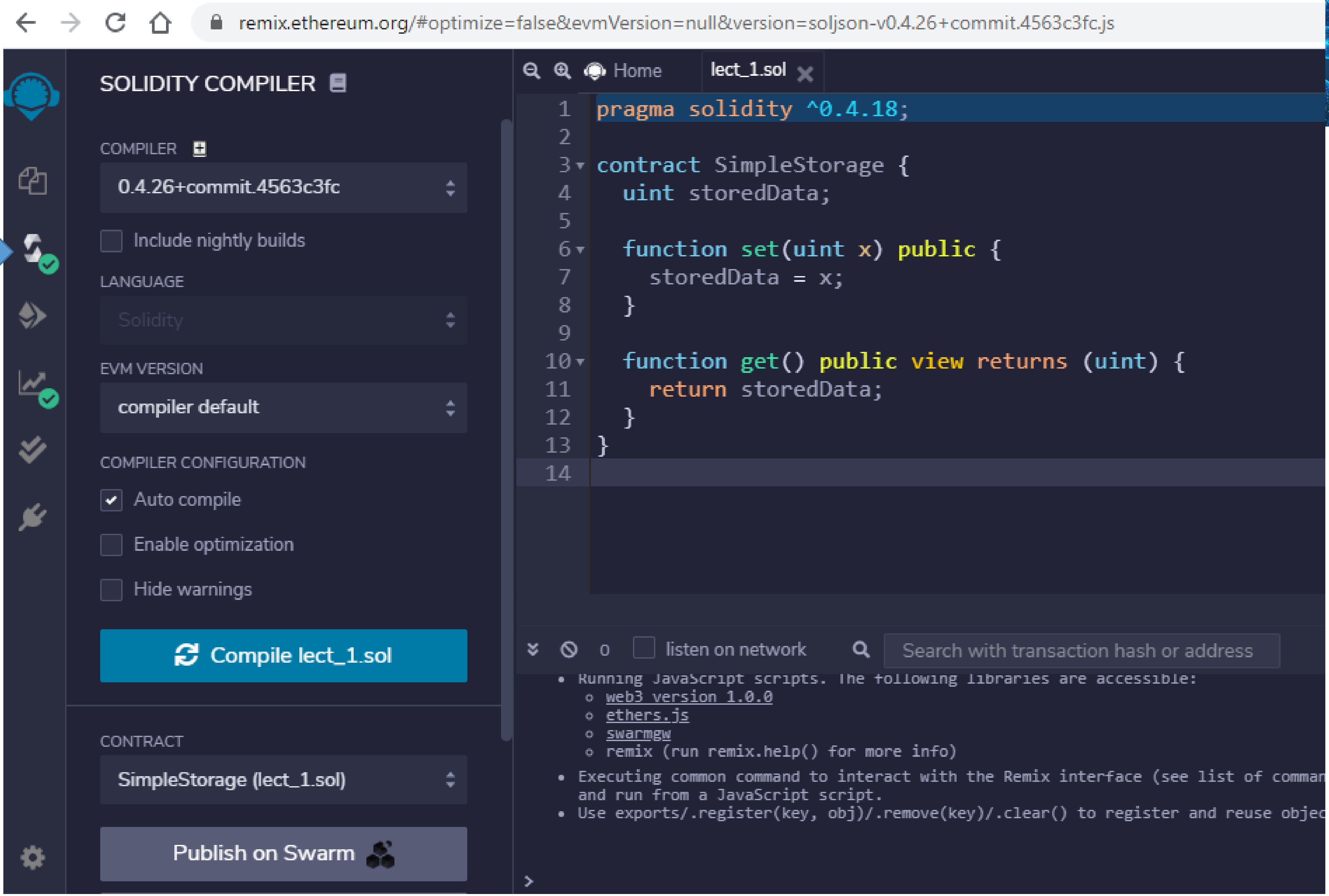
- creates a new global symbol symbolName whose members are all the global symbols from "filename".

```
import * as symbolName from "filename";
```

keywords

abstract	after	alias	apply
auto	case	catch	copyof
default	define	final	immutable
implements	in	inline	let
macro	match	mutable	null
of	override	partial	promise
reference	relocatable	sealed	sizeof
static	supports	switch	try
typedef	typeof	unchecked	

compile



remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.4.26+commit.4563c3fc.js

SOLIDITY COMPILER

COMPILER

0.4.26+commit.4563c3fc

☐ Include nightly builds

LANGUAGE

Solidity

EVM VERSION

compiler default

COMPILER CONFIGURATION

☑ Auto compile

☐ Enable optimization

☐ Hide warnings

Compile lect_1.sol

CONTRACT

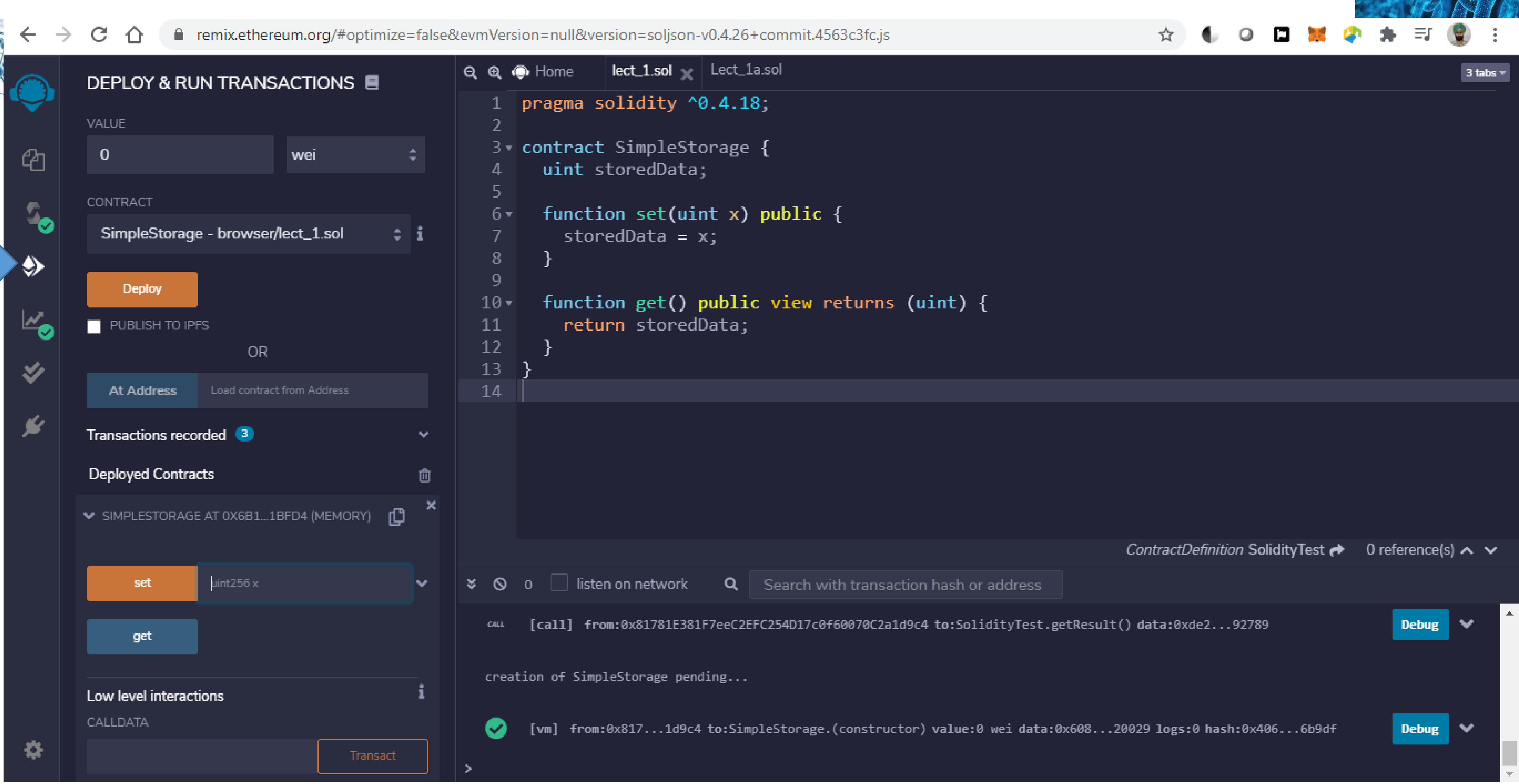
SimpleStorage (lect_1.sol)

Publish on Swarm

```
pragma solidity ^0.4.18;

contract SimpleStorage {
  uint storedData;

  function set(uint x) public {
    storedData = x;
  }

  function get() public view returns (uint) {
    return storedData;
  }
}
```

```
• Running JavaScript scripts. The following libraries are accessible:
  ○ web3 version 1.0.0
  ○ ethers.js
  ○ swarmgw
  ○ remix (run remix.help() for more info)
• Executing common command to interact with the Remix interface (see list of comman
  and run from a JavaScript script.
• Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse objec
>
```

Deploy



remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.4.26+commit.4563c3fc.js
DEPLOY & RUN TRANSACTIONS
Deploy
PUBLISH TO IPFS
OR
At Address
Load contract from Address
Transactions recorded 3
Deployed Contracts
SIMPLESTORAGE AT 0X6B1...1BFD4 (MEMORY)
set
uint256 x
get
Low level interactions
CALLDATA
Transact
Home
lect_1.sol
Lect_1a.sol
3 tabs
ContractDefinition SolidityTest
0 reference(s)
listen on network
Search with transaction hash or address
Debug

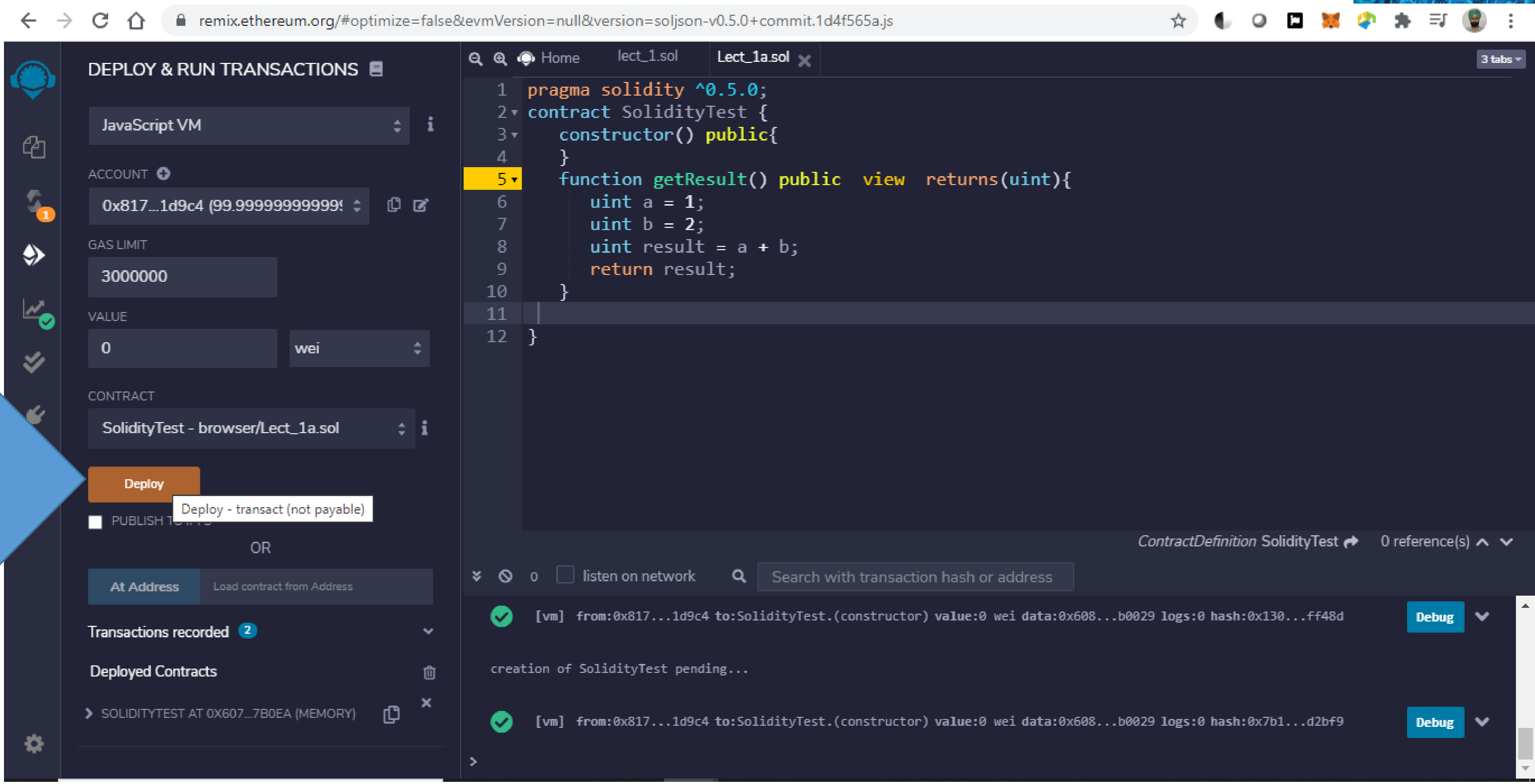
VALUE: 0 wei

CONTRACT: SimpleStorage - browser/lect_1.sol

```
pragma solidity ^0.4.18;

contract SimpleStorage {
  uint storedData;

  function set(uint x) public {
    storedData = x;
  }

  function get() public view returns (uint) {
    return storedData;
  }
}
```

```
CALL [call] from:0x81781E381F7eeC2EFC254D17c0f60070C2a1d9c4 to:SolidityTest.getResult() data:0xde2...92789

creation of SimpleStorage pending...

✓ [vm] from:0x817...1d9c4 to:SimpleStorage.(constructor) value:0 wei data:0x608...20029 logs:0 hash:0x406...6b9df
```

Click Deploy button, to deploy the contract


remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.5.0+commit.1d4f565a.js
DEPLOY & RUN TRANSACTIONS
Home
lect_1.sol
Lect_1a.sol
3 tabs
Deploy
Deploy - transact (not payable)
OR
At Address
Load contract from Address
Debug
Debug

ENVIRONMENT: JavaScript VM

ACCOUNT: 0x817...1d9c4 (99.99999999999

GAS LIMIT: 3000000

VALUE: 0 wei

CONTRACT: SolidityTest - browser/Lect_1a.sol

□ PUBLISH TO IPFS

Transactions recorded 2

Deployed Contracts

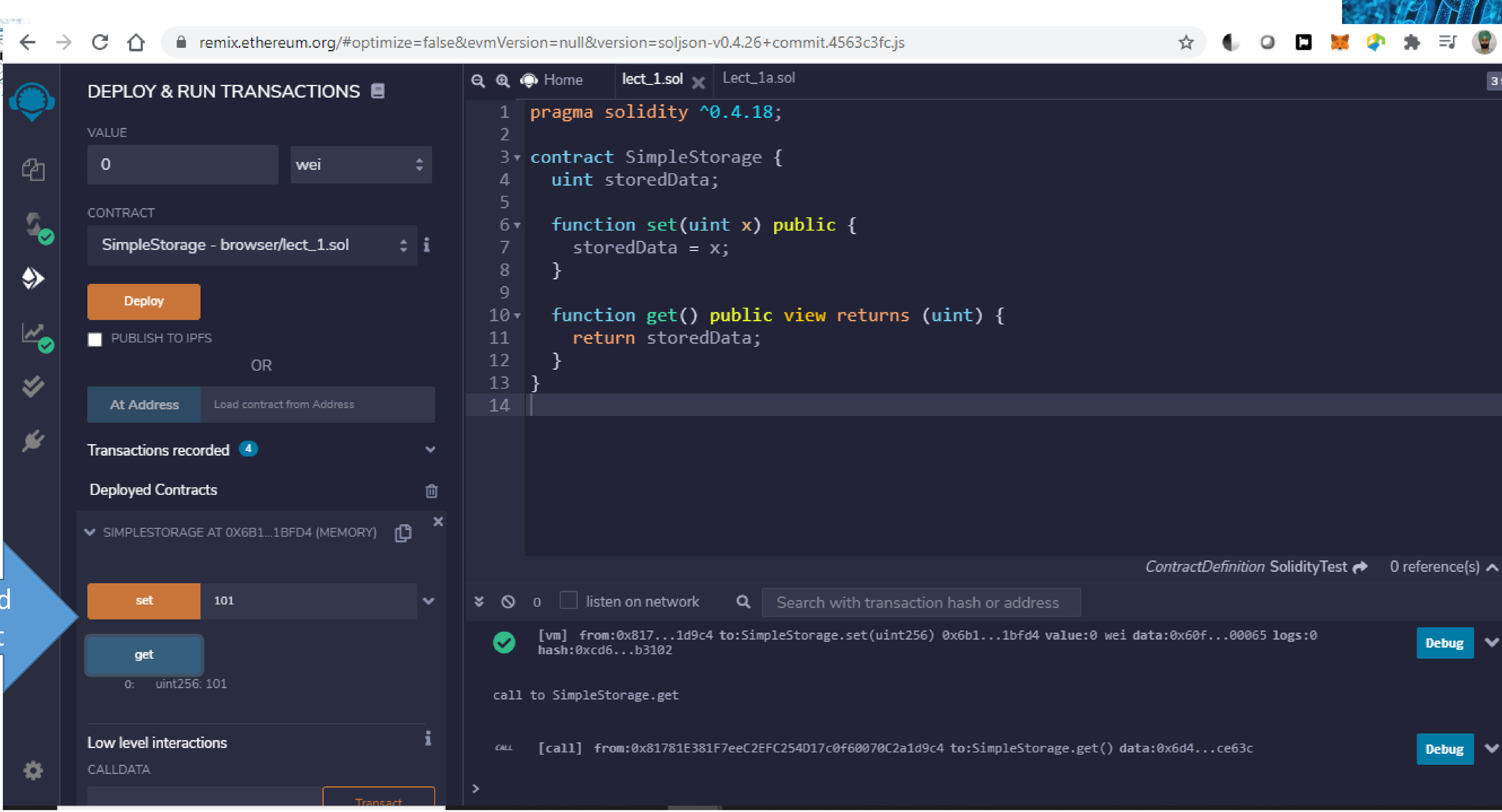
SOLIDITYTEST AT 0X607...7B0EA (MEMORY)

```
pragma solidity ^0.5.0;
contract SolidityTest {
   constructor() public{
   }
   function getResult() public  view  returns(uint){
      uint a = 1;
      uint b = 2;
      uint result = a + b;
      return result;
   }

}
```

ContractDefinition SolidityTest   0 reference(s)

0   □ listen on network   Search with transaction hash or address

```
[vm] from:0x817...1d9c4 to:SolidityTest.(constructor) value:0 wei data:0x608...b0029 logs:0 hash:0x130...ff48d

creation of SolidityTest pending...

[vm] from:0x817...1d9c4 to:SolidityTest.(constructor) value:0 wei data:0x608...b0029 logs:0 hash:0x7b1...d2bf9
```


remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.4.26+commit.4563c3fc.js
DEPLOY & RUN TRANSACTIONS
Home
lect_1.sol
Lect_1a.sol
Deploy
PUBLISH TO IPFS
OR
At Address
Load contract from Address
Transactions recorded 4
Deployed Contracts
set
get
Low level interactions
CALLDATA
Transact
ContractDefinition SolidityTest
0 reference(s)
listen on network
Search with transaction hash or address
Debug
Debug

VALUE: 0 wei

CONTRACT: SimpleStorage - browser/lect_1.sol

SIMPLESTORAGE AT 0X6B1...1BFD4 (MEMORY)

set 101

get

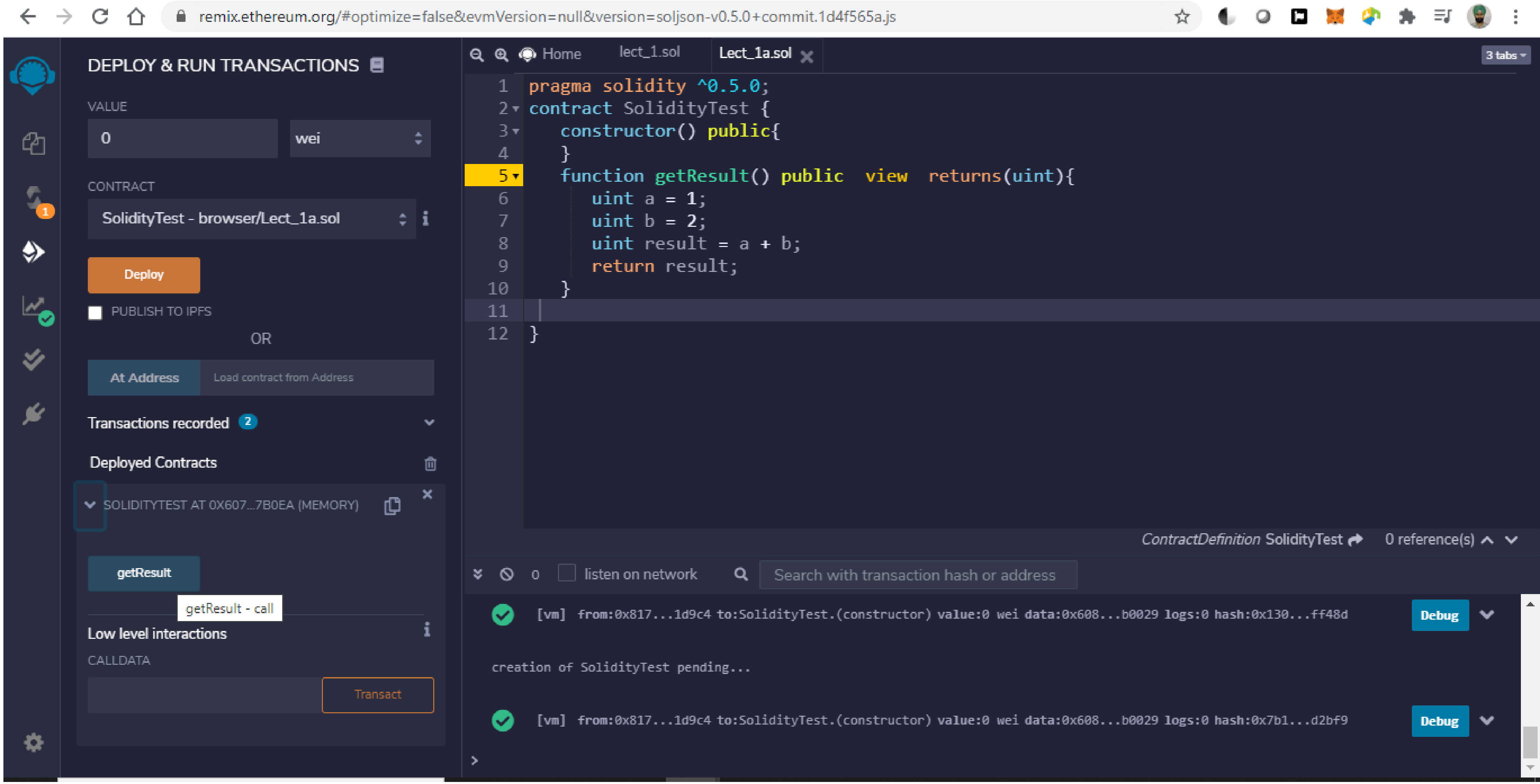
0: uint256: 101

```
pragma solidity ^0.4.18;

contract SimpleStorage {
  uint storedData;

  function set(uint x) public {
    storedData = x;
  }

  function get() public view returns (uint) {
    return storedData;
  }
}
```

```
[vm] from:0x817...1d9c4 to:SimpleStorage.set(uint256) 0x6b1...1bfd4 value:0 wei data:0x60f...00065 logs:0
hash:0xcd6...b3102

call to SimpleStorage.get

[call] from:0x81781E381F7eeC2EFC254D17c0f60070C2a1d9c4 to:SimpleStorage.get() data:0x6d4...ce63c
```

Deployed contract

Another Example



remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.5.0+commit.1d4f565a.js

DEPLOY & RUN TRANSACTIONS

VALUE: 0 wei

CONTRACT: SolidityTest - browser/Lect_1a.sol

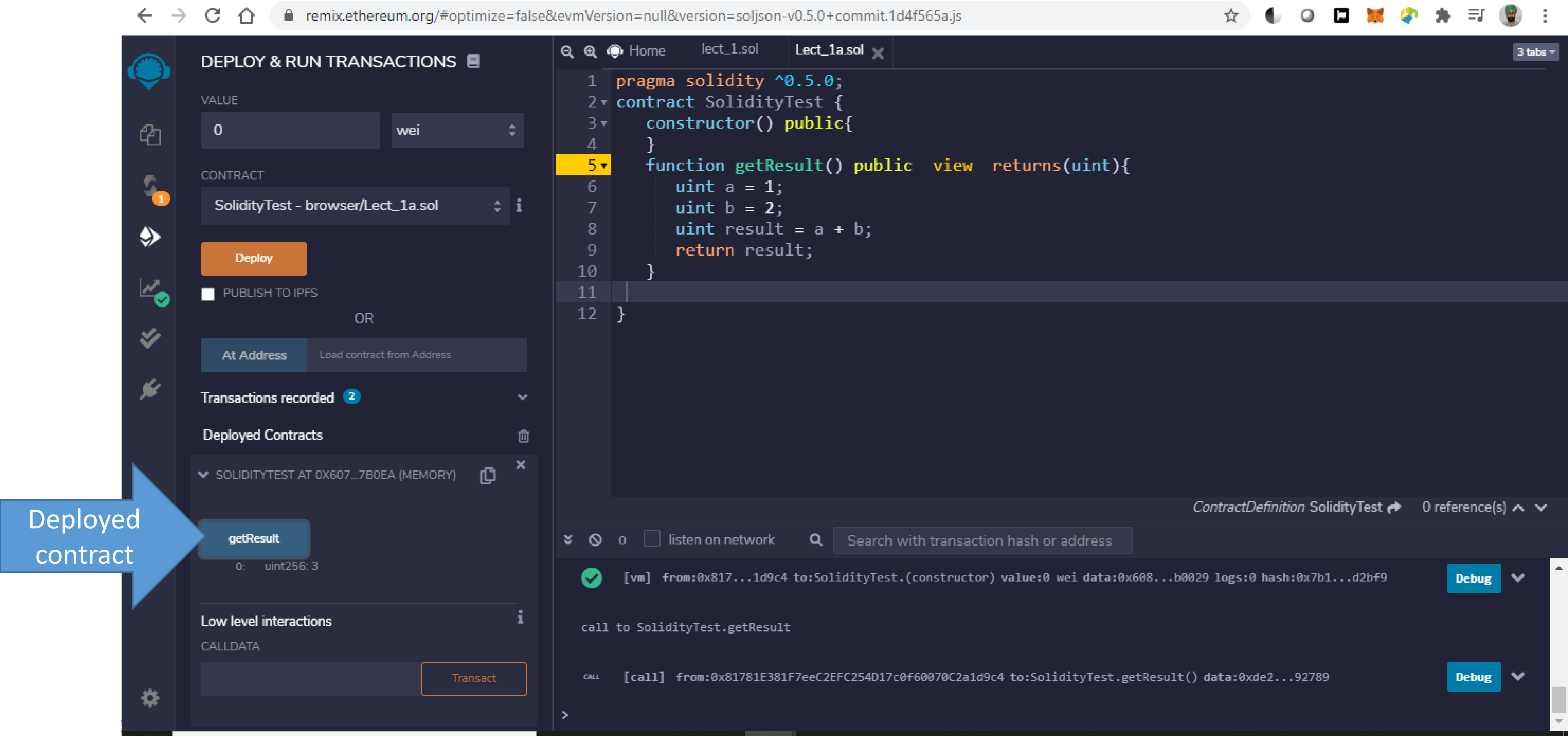
Transactions recorded 2

Deployed Contracts

SOLIDITYTEST AT 0X607...7B0EA (MEMORY)

```
pragma solidity ^0.5.0;
contract SolidityTest {
   constructor() public{
   }
   function getResult() public  view  returns(uint){
      uint a = 1;
      uint b = 2;
      uint result = a + b;
      return result;
   }

}
```

```
[vm] from:0x817...1d9c4 to:SolidityTest.(constructor) value:0 wei data:0x608...b0029 logs:0 hash:0x130...ff48d

creation of SolidityTest pending...

[vm] from:0x817...1d9c4 to:SolidityTest.(constructor) value:0 wei data:0x608...b0029 logs:0 hash:0x7b1...d2bf9
```



remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.5.0+commit.1d4f565a.js
DEPLOY & RUN TRANSACTIONS
Home
lect_1.sol
Lect_1a.sol
3 tabs
Deploy
At Address
Load contract from Address
Transact
Debug
Search with transaction hash or address

VALUE: 0 wei

CONTRACT: SolidityTest - browser/Lect_1a.sol

□ PUBLISH TO IPFS

OR

Transactions recorded 2

Deployed Contracts

SOLIDITYTEST AT 0X607...7B0EA (MEMORY)

getResult

0: uint256: 3

Low level interactions

CALLDATA

```
pragma solidity ^0.5.0;
contract SolidityTest {
   constructor() public{
   }
   function getResult() public  view  returns(uint){
      uint a = 1;
      uint b = 2;
      uint result = a + b;
      return result;
   }

}
```

ContractDefinition SolidityTest   0 reference(s)

0 □ listen on network

```
[vm] from:0x817...1d9c4 to:SolidityTest.(constructor) value:0 wei data:0x608...b0029 logs:0 hash:0x7b1...d2bf9

call to SolidityTest.getResult

CALL [call] from:0x81781E381F7eeC2EFC254D17c0f60070C2a1d9c4 to:SolidityTest.getResult() data:0xde2...92789
```

Deployed contract