

Getting familiar with a Smart Contract:

A smart contract in Ethereum is just code. Unlike the "paper" contracts that you find elsewhere, this contract needs to make sense in a very precise manner. I will migrate a basic contract to a test blockchain, introduce some errors into it, and solve each one through the use of the built-in Truffle debugger. One of the most basic, non-trivial, types of smart contract is a *simple storage contract*.

Deploying the basic smart contract:

- I installed Truffle globally on my system:

```
npm install -g truffle
```

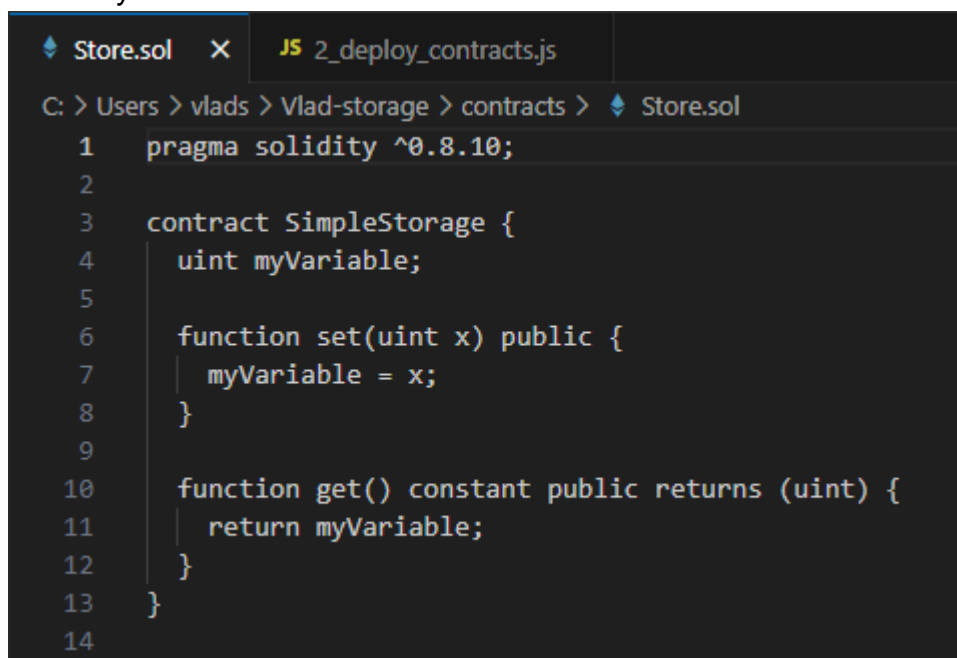
- I created a new directory for my Truffle project:

```
C:\Users\vlads>mkdir Vlad-storage  
C:\Users\vlads>cd Vlad-storage
```

- I initiated the Truffle project:

```
truffle init
```

- I started writing the Solidity smart contract code within a file. A basic Solidity contract typically begins with the "pragma solidity" statement and the contract definition. This is the contract that I'll be debugging. The contract is named SimpleStorage. I inserted the file named "Store.sol" inside the contracts/ directory.



```
Store.sol X JS 2_deploy_contracts.js  
C: > Users > vlads > Vlad-storage > contracts > Store.sol  
1  pragma solidity ^0.8.10;  
2  
3  contract SimpleStorage {  
4      uint myVariable;  
5  
6      function set(uint x) public {  
7          myVariable = x;  
8      }  
9  
10     function get() constant public returns (uint) {  
11         return myVariable;  
12     }  
13 }  
14
```

This PC > Local Disk (C:) > Users > vlads > Vlad-storage > contracts

Name	Date modified	Type	Size
.gitkeep	26.07.2023 14:38	GITKEEP File	0 KB
Store	26.07.2023 14:58	SOL File	1 KB

- Inside the migrations/ directory, I created a file called 2_deploy_contracts.js. This file is the directive that allows us to deploy the SimpleStorage contract to the blockchain.

```

Store.sol  JS 2_deploy_contracts.js X
C: > Users > vlads > Vlad-storage > migrations > JS 2_deploy_contracts.js > ...
1  var SimpleStorage = artifacts.require("SimpleStorage");
2
3  module.exports = function (deployer) {
4    deployer.deploy(SimpleStorage);
5  };
6

```

> This PC > Local Disk (C:) > Users > vlads > Vlad-storage > migrations

Name	Date modified	Type	Size
.gitkeep	26.07.2023 14:38	GITKEEP File	0 KB
2_deploy_contracts	26.07.2023 15:12	JavaScript File	1 KB

- I compiled the smart contract in the terminal.

```

truffle compile

> Artifacts written to C:\Users\vlads\Vlad-storage\build\contracts
> Compiled successfully using:
  - solc: 0.8.20+commit.a1b79de6.Emscripten.clang

```

- I opened a second terminal and run "truffle develop" to start a development blockchain built directly into Truffle that I can use to test my contract:

```

C:\Users\vlads\Vlad-storage>truffle develop
Truffle Develop started at http://127.0.0.1:9545/

```

- With the develop console up and running, I can now deploy my contracts to the blockchain by running my migrations using "migrate" command:

```

Starting migrations...
=====
> Network name:      'develop'
> Network id:        5777
> Block gas limit: 6721975 (0x6691b7)

2_deploy_contracts.js
=====

    Deploying 'SimpleStorage'
    -----

    > transaction hash:      0xedc6f7f889160f6089255be4b518e129fc77578733
    > Blocks: 0              Seconds: 0
    > contract address:      0xe0fCb124ce9e298cD12c74e1439776d1726AB377
    > block number:          1
    > block timestamp:        1690377257
    > account:                0x23b6B8613111fdf6010481e66a0ccdc6769E358c
    > balance:                99.999584844625
    > gas used:                123009 (0x1e081)
    > gas price:               3.375 gwei
    > value sent:              0 ETH
    > total cost:              0.000415155375 ETH

    > Saving artifacts
    -----
    > Total cost:              0.000415155375 ETH

Summary
=====
> Total deployments:       1
> Final cost:               0.000415155375 ETH

```

Interacting with the basic smart contract:

The smart contract is now deployed to a test network via `truffle develop`, which launches a console against Ganache, a local development blockchain built right into Truffle. I want to interact with the smart contract to see how it works when working correctly. I'll interact using the `truffle develop` console.

In the console where `truffle develop` is running I introduced the command: `SimpleStorage.deployed()`


```

C:\Windows\system32\cmd.exe - "node "C:\Users\dash\AppData\Roaming\openmode\mode_modules\truffle\build\bundle.js" develop
setNetwork: [Function: bound setNetwork],
setNetworkType: [Function: bound setNetworkType],
setWallet: [Function: bound setWallet],
resetAddress: [Function: bound resetAddress],
link: [Function: bound link],
clone: [Function: bound clone],
addProp: [Function: bound addProp],
toJSON: [Function: bound toJSON],
decodeLogs: [Function: bound decodeLogs],
enums: {},
class_defaults: { from: '0x238680013111f0f001041e0d0a0c1d070001308c' },
interfaceAdapter: web3InterfaceAdapter { web3: [Web3Shim] },
web3: Web3Shim {
  currentProvider: [Getter/Setter],
  requestManager: [RequestManager],
  givenProvider: null,
  providers: [Object],
  _provider: [HttpProvider],
  setProvider: [Function (anonymous)],
  setRequestManager: [Function (anonymous)],
  batchRequest: [Function: bound batch],
  extend: [Function],
  version: '3.0.0',
  utils: [Object],
  eth: [Eth],
  sha: [Sha],
  bzz: [Bzz],
  networkType: 'ethereum'
},
currentProvider: HttpProvider {
  withCredentials: undefined,
  timeout: 0,
  headers: undefined,
  agent: undefined,
  connected: false,
  host: 'http://127.0.0.1:9545/',
  httpAgent: [Agent],
  send: [Function (anonymous)],
  request: [Function: bound modifiedRequest] AsyncFunction,
  alreadyWrapped: true
},
networkId: '5777',
disableConfirmationListener: undefined,
ens: { enabled: false, registryAddress: undefined }
},
methods: {
  sendRawTransaction: [Function (anonymous)] {
    call: [Function (anonymous)],
    sendTransaction: [Function (anonymous)],
    estimateGas: [Function (anonymous)],
    request: [Function (anonymous)]
  },
  set: [Function (anonymous)] {
    call: [Function (anonymous)],
    sendTransaction: [Function (anonymous)],
    estimateGas: [Function (anonymous)],
    request: [Function (anonymous)]
  }
},
abi: [
  {
    inputs: [Array],
    name: 'set',
  }
]
}

```

```

C:\Windows\system32\cmd.exe - "node "C:\Users\dash\AppData\Roaming\openmode\mode_modules\truffle\build\bundle.js" develop
{
  inputs: [Array],
  name: 'set',
  outputs: [],
  stateMutability: 'nonpayable',
  type: 'function',
  constant: undefined,
  payable: undefined,
  signature: '0x00f47b1'
},
{
  inputs: [],
  name: 'get',
  outputs: [Array],
  stateMutability: 'nonpayable',
  type: 'function',
  constant: undefined,
  payable: undefined,
  signature: '0x00f47b1'
},
address: '0xb0fC3124c9e2900D12c74e1439776d1726A8377',
transactionHash: undefined,
contract: Contract {
  setProvider: [Function (anonymous)],
  currentProvider: [Getter/Setter],
  _requestManager: RequestManager {
    provider: [HttpProvider],
    providers: [Object],
    subscriptions: Map(0) {}
  },
  givenProvider: null,
  providers: {
    WebsocketProvider: [Function: WebsocketProvider],
    IpcProvider: [Function: IpcProvider]
  },
  _provider: HttpProvider {
    withCredentials: undefined,
    timeout: 0,
    headers: undefined,
    agent: undefined,
    connected: false,
    host: 'http://127.0.0.1:9545/',
    httpAgent: [Agent],
    send: [Function (anonymous)],
    request: [Function: bound modifiedRequest] AsyncFunction,
    alreadyWrapped: true
  },
  setRequestManager: [Function (anonymous)],
  batchRequest: [Function: bound batch],
  extend: [Function: ex] {
    formatters: [Object],
    utils: [Object],
    Method: [Function: Method]
  },
  clearSubscriptions: [Function (anonymous)],
  options: { address: [Getter/Setter], jsonInterface: [Getter/Setter] },
  handleRevert: [Getter/Setter],
  defaultCommon: [Getter/Setter],
  defaultNetwork: [Getter/Setter],
  defaultChain: [Getter/Setter],
  transactionPollingTimeout: [Getter/Setter],
}
truffle(develop)

```

```
timeout: 0,
headers: undefined,
agent: undefined,
connected: false,
host: 'http://127.0.0.1:9545/',
httpAgent: [Agent],
send: [Function (anonymous)],
request: [Function: bound modifiedRequest] AsyncFunction,
_alreadyWrapped: true
},
setRequestManager: [Function (anonymous)],
BatchRequest: [Function: bound Batch],
extend: [Function: extend] {
  formatters: [Object],
  utils: [Object],
  Method: [Function: Method]
},
clearSubscriptions: [Function (anonymous)],
options: { address: [Getter/Setter], jsonInterface: [Getter/Setter] },
handleRevert: [Getter/Setter],
defaultCommon: [Getter/Setter],
defaultGuardFor: [Getter/Setter],
defaultChain: [Getter/Setter],
transactionPollingTimeout: [Getter/Setter],
transactionPollingInterval: [Getter/Setter],
transactionConfirmationBlocks: [Getter/Setter],
transactionBlockTimeout: [Getter/Setter],
blockHeaderTimeout: [Getter/Setter],
defaultAccount: [Getter/Setter],
defaultBlock: [Getter/Setter],
methods: {
  set: [Function: bound .createObject],
  'set(address)': [Function: bound .createObject],
  'set(address)': [Function: bound .createObject],
  get: [Function: bound .createObject],
  'get(address)': [Function: bound .createObject],
  'get()': [Function: bound .createObject]
},
events: { allEvents: [Function: bound ] },
_address: '0x0000000000000000000000000000000000000000',
_jsonInterface: [ [Object], [Object] ]
},
set: [Function (anonymous)] {
  call: [Function (anonymous)],
  sendTransaction: [Function (anonymous)],
  estimateGas: [Function (anonymous)],
  request: [Function (anonymous)]
},
get: [Function (anonymous)] {
  call: [Function (anonymous)],
  sendTransaction: [Function (anonymous)],
  estimateGas: [Function (anonymous)],
  request: [Function (anonymous)]
},
sendTransaction: [Function (anonymous)],
estimateGas: [Function (anonymous)],
call: [Function (anonymous)],
send: [Function (anonymous)],
allEvents: [Function (anonymous)],
getPastEvents: [Function (anonymous)]
}
truffle(develop)>
```

- Then, I inserted the following sequence:

```
SimpleStorage.deployed()
  .then(function (instance) {
    return instance.get.call();
  })
  .then(function (value) {
    return value.toNumber();
  });
```

- This command looks at the SimpleStorage contract, and then calls the get() function as defined inside it. It then returns the output, which is usually rendered as a string, and converts it to a number: 0.

```
truffle(develop)> SimpleStorage.deployed().then(function(instance){return instance.get.call();
}).then(function(value){return value.toNumber();});
0
```

- This shows us that our variable is set to 0, even though we haven't set this variable to any value (yet). This is because variables with integer types are automatically populated with the value of zero in Solidity, unlike other languages where it might be NULL or undefined.

- Now I will run a transaction on my contract. I'll do this by running the `set()` function, where I can set my variable value to some other integer. I inserted the following sequence:

```
SimpleStorage.deployed().then(function (instance) {
  return instance.set(4);
});
```

- This sets the variable to 4. The output shows some information about the transaction, including the transaction ID (hash), transaction receipt, and any event logs that were triggered during the course of the transaction:

[illegible]

- To verify that the variable has changed values, I run the `get()` function again:

```
truffle(develop)> SimpleStorage.deployed().then(function(instance){return instance.get.call();
}).then(function(value){return value.toNumber();});
```

Testing the contract:

```
truffle(develop)> truffle test
Using network 'develop'.

Compiling your contracts...
=====
> Compiling .\contracts\Store.sol
> Compilation warnings encountered:

    Warning: SPDX license identifier not provided in source file. Before publishing,
    en-source code. Please see https://spdx.org for more information.
--> project:/contracts/Store.sol

,Warning: Function state mutability can be restricted to view
--> project:/contracts/Store.sol:10:3:
|
10 |     function get() public returns (uint) {
|       ^ (Relevant source part starts here and spans across multiple lines).

> Artifacts written to C:\Users\vlads\AppData\Local\Temp\test--6460-AEluzHrC6sHY
> Compiled successfully using:

    - solc: 0.8.20+commit.a1b79de6.Emscripten.clang

0 passing (0ms)
```

To accomplish this task, I followed this guide:

<https://trufflesuite.com/guides/debugging-an-example-smart-contract/>