

Documentation for private blockchain

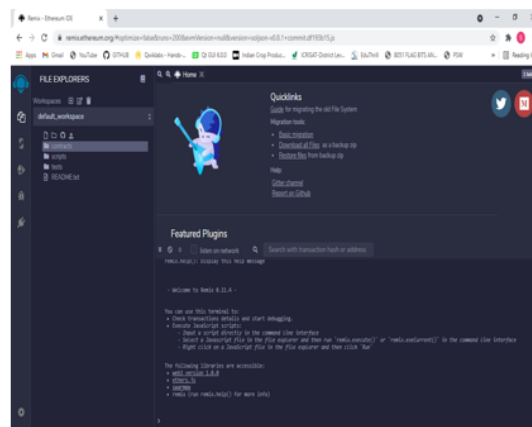
REMIXD

DEFAULT WORKSPACE MODE(1)

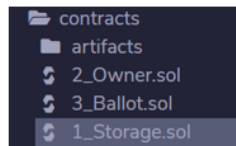
Step 1 : open the web Browser and visit the url which is mentioned below:

<https://remix.ethereum.org/>

(It will open the online remix ide into your web browser)

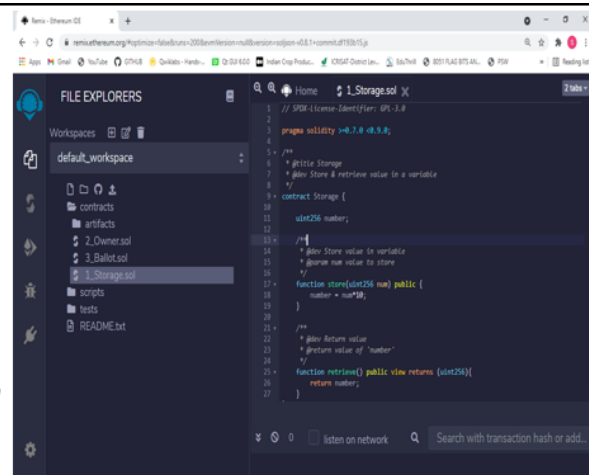


Step 2 : Here we have the contract section which has 3 types of Artifacts:

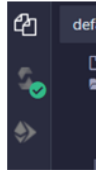


Choose any of the option as per the requirement.

Here I choose the Storage.sol file for my calculator program



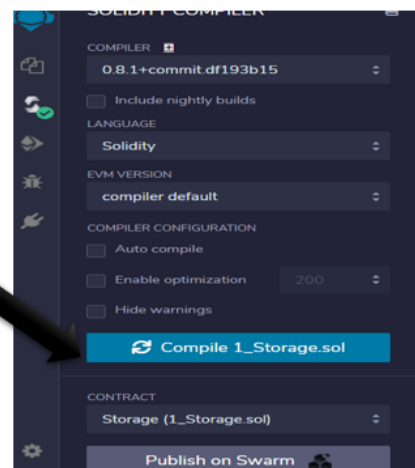
Step 3 : Now we have to create functions as per the requirement and save file using (ctrl+s)



If there is no error then it shows a green signal on the left side of the panel, it means you are now able to compile your program.

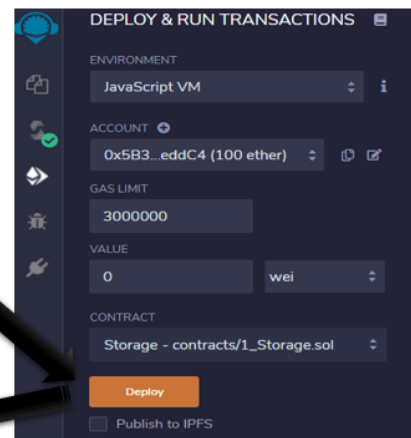
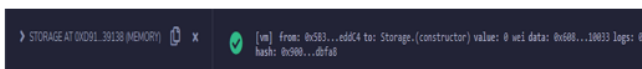
```
1 pragma solidity >=0.7.0 <0.9.0;
2
3 /**
4  * @title Storage
5  * @dev Store & retrieve value in a variable
6  */
7 contract Storage {
8
9     uint256 number;
10
11     /**
12      * @dev Store value in variable
13      * @param num value to store
14      */
15     function add(uint256 num ,uint256 num2) public {
16         number = num + num2;
17     }
18     function sub(uint256 num ,uint256 num2) public {
19         number = num - num2;
20     }
21     function mul(uint256 num ,uint256 num2) public {
22         number = num * num2;
23     }
24     function div(uint256 num ,uint256 num2) public {
25         number = num / num2;
26     }
27 }
28
29 /**
30  * @dev Return value
31  * @return value of 'number'
32  */
33 function retrieve() public view returns (uint256){
34     return number;
35 }
36 }
```

Step 4 : Now we compile our program using this compile option (Compile program_name)



Step 5 : In next Step we deploy our program by clicking this option (Deploy)

After deployment a contract is created



Test: Input

mul

15, 10

▼

sub

uint256 num, uint256 num2

▼

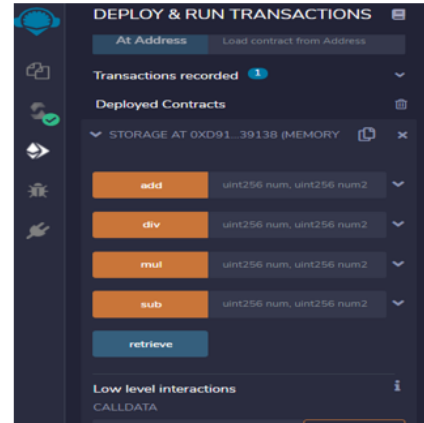
retrieve

Output

```

0: uint256: 150

```



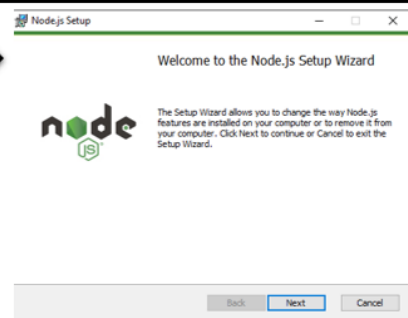
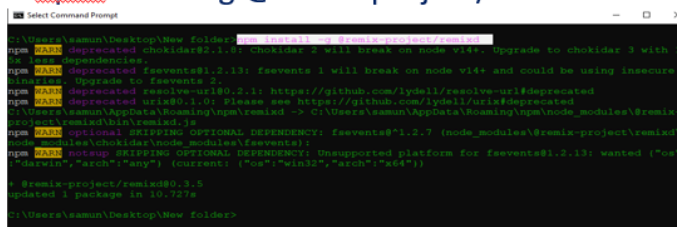
```
npm install -g @remix-project/remixd
```

LOCALHOST MODE(1)

Step 1 :install Nodejs in your PC

Step 2: Then install `remixd` using `npm install remixd` command in your folder:

```
npm install -g @remix-project/remixd
```



Warning: Do not use any space for the folder name otherwise it will create an error during **step 3 and 4**

Step 3 :run this command to share your folder with the remix ide.

`remixd -s <mention folder path> --remix-ide https://remix.ethereum.org`

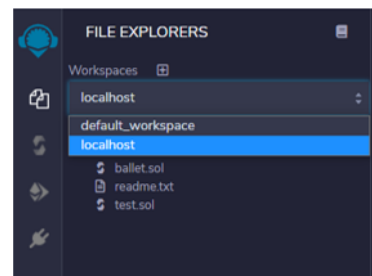
```
C:\Users\samun>remixd -s C:\Users\samun\Desktop\remixd_demo --remix-ide https://remix.ethereum.org
[WARN] You may now only use IDE at https://remix.ethereum.org to connect to that instance
[WARN] Any application that runs on your computer can potentially read from and write to all files in the directory.
[WARN] Symbolic links are not forwarded to Remix IDE
Sat Apr 24 2021 19:58:33 GMT+0530 (India Standard Time) remixd is listening on 127.0.0.1:65520
setup notifications for C:\Users\samun\Desktop\remixd_demo
```

`remixd -s <mention folder path> --remix-ide https://remix.ethereum.org`

Step 4 : Choose the localhost option from the work space.

Step 5: create a file then either write the command or copy the code skeleton from the default_workspace

Repeat **step 2 to step 6** from Default Workspace Mode in order to run program in localhost mode.



STARTING PRIVATE BLOCKCHAIN USING GETH

```
geth --rpc --rpcport "8545" --datadir=./chaindata/TestChain --rpcapi  
admin,eth,net,web3,personal,miner,txpool --nodiscover --rpccorsdomain "*" --allow-insecure-unlock
```

```
PS C:\Users\Ashok\Workspace> cd .\blockchain\  
PS C:\Users\Ashok\Workspace\blockchain> geth --rpc --rpcport "8545" --datadir ./chaindata --rpcapi admin,eth,net,web3,personal,miner,txpool --rpccorsdo  
main "*" --allow-insecure-unlock  
INFO [04-28|16:07:01.514] Starting Geth on Ethereum mainnet...  
INFO [04-28|16:07:01.514] Bumping default cache on mainnet provided=1024 updated=4096  
INFO [04-28|16:07:01.527] Maximum peer count ETH=50 LES=0 total=50  
WARN [04-28|16:07:01.530] The flag --rpc is deprecated and will be removed June 2021, please use --http  
WARN [04-28|16:07:01.534] The flag --rpcport is deprecated and will be removed June 2021, please use --http.port  
WARN [04-28|16:07:01.538] The flag --rpccorsdomain is deprecated and will be removed June 2021, please use --http.corsdomain
```

INTERACTING WITH THIS GETH CONSOLE

```
geth attach http://localhost:8545
```

WEB3 .JS



The screenshot shows the web3.js documentation page. The header includes the web3.js logo and version 1.3.4. The sidebar on the left has a search bar and two main sections: 'USER DOCUMENTATION' with links for 'Getting Started', 'Callbacks Promises Events', and 'Glossary'; and 'API REFERENCE' with links for 'Web3' and 'web3.eth'. The main content area is titled 'web3.js - Ethereum JavaScript API' and includes a link to 'Edit on GitHub'. The text describes web3.js as a collection of libraries for interacting with Ethereum nodes via HTTP, IPC, or WebSocket. It also mentions that the documentation will guide users through installing and running web3.js, as well as providing an API reference with examples. A 'Contents' section is partially visible at the bottom.

To work with web3.js make sure you installed nodejs

To check node version : node -v

To check npm version : npm -v

```
PS C:\Users\Ashok\Workspace\web3-intro> ls
PS C:\Users\Ashok\Workspace\web3-intro> node -v
v14.16.1
PS C:\Users\Ashok\Workspace\web3-intro> npm -v
6.14.12
PS C:\Users\Ashok\Workspace\web3-intro>
```

START NEW PROJECT IN NODEJS :

npm init -y

```
PS C:\Users\Ashok\Workspace\web3-intro> npm init -y
Wrote to C:\Users\Ashok\Workspace\web3-intro\package.json:

{
  "name": "web3-intro",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

PS C:\Users\Ashok\Workspace\web3-intro> |
```

INSTALLING WEB3 MODULE :

Npm install web3 --save

```
PS C:\Users\Ashok\Workspace\web3-intro> npm install web3 --save
npm WARN deprecated mkdirp-promise@5.0.1: This package is broken and no longer maintained. 'mkdirp' itself supports promises now, please switch to that.
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
[.....] \ fetchMetadata: sill resolveWithNewModule core-util-is@1.0.2 checking installable status
```

This will install web3 dependencies of the project

INTERACTING WITH NODE :

Run node then inside node run following

Let Web3 = require('web3')

```
PS C:\Users\Ashok\Workspace\web3-intro> node
Welcome to Node.js v14.16.1.
Type ".help" for more information.
> let Web3 = require('web3')
undefined
>
```

We can check what are things available to this Web3 variable by typing **Web3** to console.

CONNECT WITH LOCAL BLOCKCHAIN :

Let web3 = new Web3("http://localhost:8545")

```
}  
}  
> let web3 = new Web3("http://localhost:8545")
```

web3.eth.getAccounts()

web3.eth.getAccounts().then(console.log)

This is function in eth module return accounts

```
> web3.eth.getAccounts()  
Promise { <pending> }  
> web3.eth.getAccounts().then(console.log)  
Promise { <pending> }  
> [  
  '0xA991a0a1Aa60494eaB796a0E3cB141240F27A11d',  
  '0xD53eF242f9df505FDA63f44c9b436452AEa1bB23',  
  '0xaC53fc0719c1e8157422442C4cbc9be6E2c45A83',  
  '0xb90dE1650396834c38a7677D54081Bda7c926A83',  
  '0x14DB9B75cd94e106e37ACF1e3a0ebcE99DbDbeDf'  
]
```

Creating new directory etherscan and creating project in it

Directory: C:\Users\Ashok\Workspace\etherscan

Mode	LastWriteTime		Length	Name
d----	28-04-2021	16:11		node_modules
d----	28-04-2021	16:10		public
d----	28-04-2021	16:10		src
-a----	28-04-2021	16:10	30	.browserslistrc
-a----	28-04-2021	16:10	351	.eslintrc.js
-a----	28-04-2021	16:10	231	.gitignore
-a----	28-04-2021	16:10	73	babel.config.js
-a----	28-04-2021	16:10	533540	package-lock.json
-a----	28-04-2021	16:10	652	package.json
-a----	28-04-2021	16:10	321	README.md

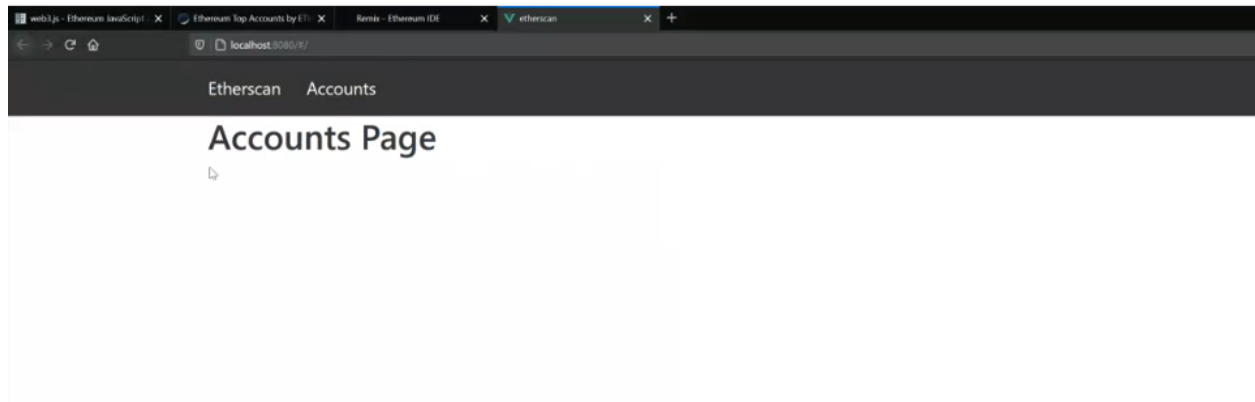
Starting server:

```
PS C:\Users\Ashok\Workspace\etherscan> npm run serve

> etherscan@0.1.0 serve C:\Users\Ashok\Workspace\etherscan
> vue-cli-service serve

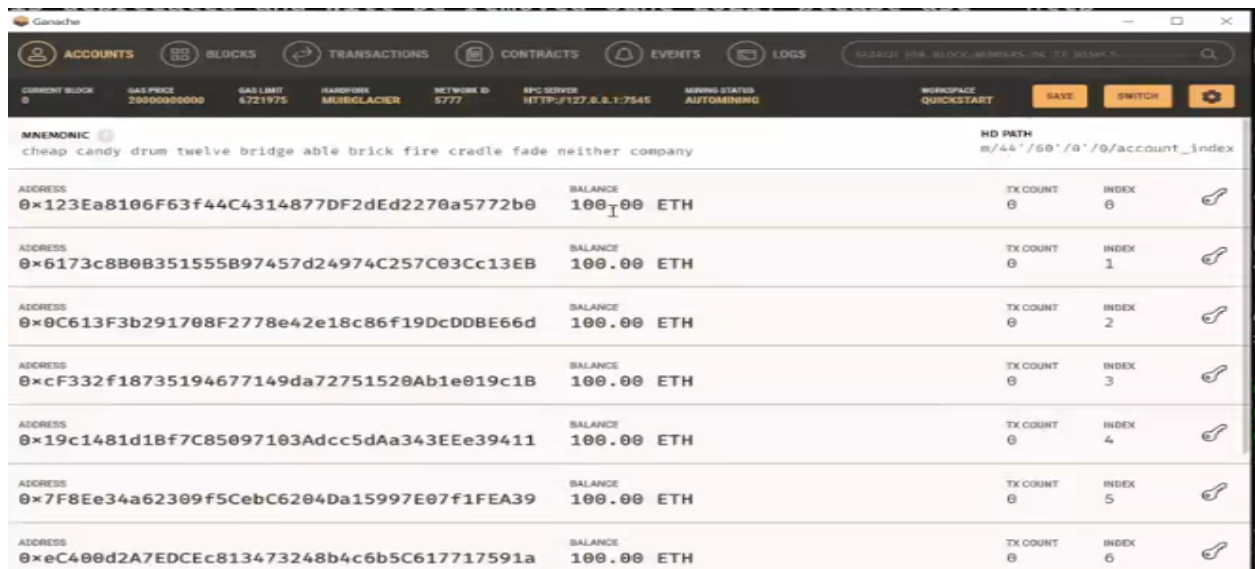
INFO Starting development server...
```

We can see our application is running :



INSTALLING GANACHE TO START BLOCKCHAIN LOCALLY :

<https://www.trufflesuite.com/ganache>



IT COME WITH SOME ACCOUNTS

COMPILE THE SOLIDITY FILE FIRST :

The screenshot displays the Solidity Compiler web interface. On the left, the 'SOLIDITY COMPILER' sidebar contains settings for the compiler version (0.8.1+commit.df193b15), language (Solidity), and EVM version (compiler default). It also includes a 'COMPILER CONFIGURATION' section with checkboxes for 'Auto compile', 'Enable optimization' (set to 200), and 'Hide warnings'. A large blue button labeled 'Compile 1_Storage.sol' is prominently displayed. Below this, the 'CONTRACT' section shows 'Storage (1_Storage.sol)' selected, with buttons for 'Publish on Swarm', 'Publish on Ipfs', and 'Compilation Details'. At the bottom of the sidebar, there are links for 'ABI' and 'Bytecode'.

The main editor area on the right shows the Solidity code for '1_Storage.sol'. The code is as follows:

```
1 // SPDX-License-Identifier: GPL-3
2
3 pragma solidity >=0.7.0 <0.9.0;
4
5 /**
6  * @title Storage
7  * @dev Store & retrieve value in
8  */
9 contract Storage {
10
11     uint256 number;
12
13     /**
14      * @dev Store value in variable
15      * @param num value to store
16      */
17     function store(uint256 num) public {
18         number = num;
19     }
20
21     /**
22      * @dev Return value
23      * @return value of 'number'
24      */
25     function retrieve() public view {
26         return number;
27     }
28 }
```

[illegible]

INSTALLING TRUFFLE :

It ease the development of smart contracts

Npm install -g truffle



Once we have installed truffle create new project in new directory

```
PS C:\Users\Ashok\Workspace\etherscan> truffle.cmd^C
PS C:\Users\Ashok\Workspace\etherscan> cd ..
PS C:\Users\Ashok\Workspace> ls

Directory: C:\Users\Ashok\Workspace

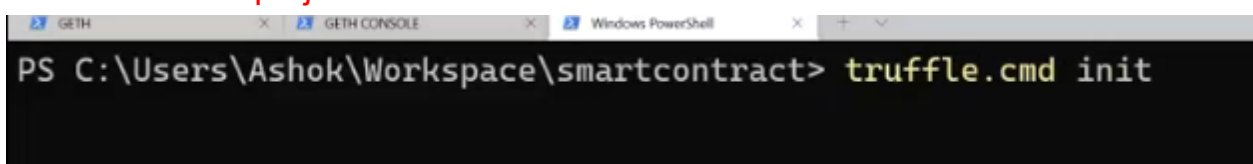
Mode                LastWriteTime         Length Name
----                -
d-----          26-04-2021    07:41             blockchain
d-----          28-04-2021    16:11             etherscan
d-----          28-04-2021    16:53             web3-intro

PS C:\Users\Ashok\Workspace> mkdir smartcontract

Directory: C:\Users\Ashok\Workspace

Mode                LastWriteTime         Length Name
----                -
d-----          28-04-2021    17:46             smartcontract
```

To initialize new project

A screenshot of a Windows PowerShell window. The title bar shows 'GETH', 'GETH CONSOLE', and 'Windows PowerShell'. The command 'truffle.cmd init' is entered in the console.

```
PS C:\Users\Ashok\Workspace\smartcontract> truffle.cmd init
```

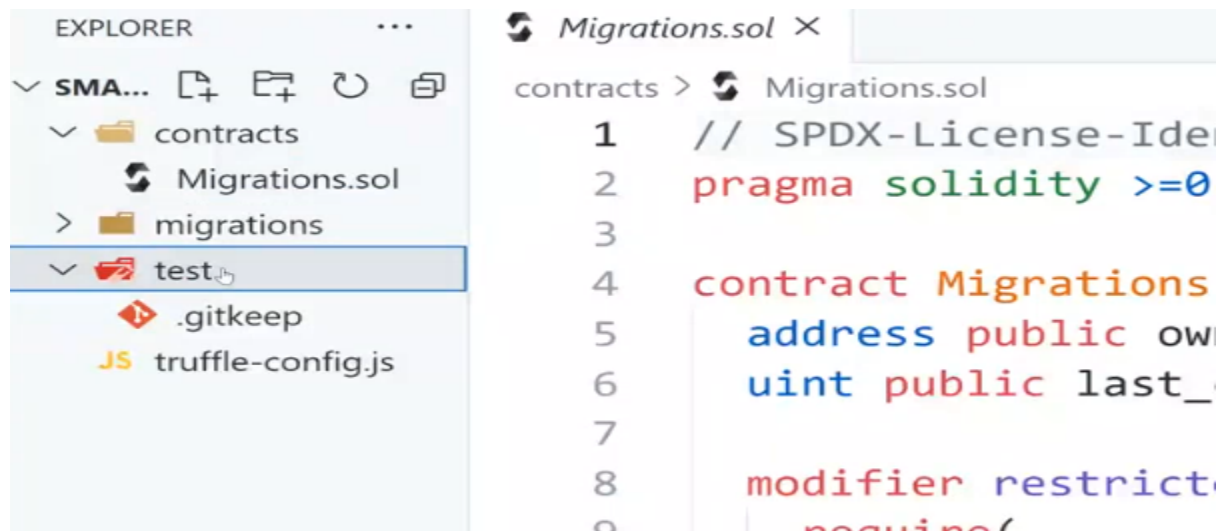
```
PS C:\Users\Ashok\Workspace\smartcontract> truffle.cmd init

Starting init...
=====

> Copying project files to C:\Users\Ashok\Workspace\smartcontract

Init successful, sweet!
```

It has three file :



- First one is main solidity file
- Truffle-config.js is configuration file for truffle

We can also compile solidity file without remixd with the help of solc :

We need to install solc

```
PS C:\Users\Ashok\Workspace\smartcontract\contracts> npm install solc
```



```
PS C:\Users\Ashok\Workspace\smartcontract\contracts> solcjs.cmd .\Migrations.sol --abi
PS C:\Users\Ashok\Workspace\smartcontract\contracts> ls
```

Directory: C:\Users\Ashok\Workspace\smartcontract\contracts

Mode	LastWriteTime		Length	Name
-a----	26-10-1985	13:45	419	Migrations.sol
-a----	28-04-2021	17:49	456	__Migrations_sol_Migrations.abi

```
PS C:\Users\Ashok\Workspace\smartcontract\contracts> |
```

We used command to compile sol file and it gave abi file

```
PS C:\Users\Ashok\Workspace\smartcontract\contracts> cat .\__Migrations_sol_Migrations.abi
[{"inputs":[],"name":"last_completed_migration","outputs":[{"internalType":"uint256","name":"","type":"uint256"}],"stateMutability":"view","type":"function"}, {"inputs":[],"name":"owner","outputs":[{"internalType":"address","name":"","type":"address"}],"stateMutability":"view","type":"function"}, {"inputs":[{"internalType":"uint256","name":"completed","type":"uint256"}],"name":"setCompleted","outputs":[],"stateMutability":"nonpayable","type":"function"}]
PS C:\Users\Ashok\Workspace\smartcontract\contracts>
```

This is what we have to pass in web3 js

```
PS C:\Users\Ashok\Workspace\smartcontract\contracts> solcjs.cmd .\Migrations.sol --bin
PS C:\Users\Ashok\Workspace\smartcontract\contracts>
```

This gives binary file which is used to deploy on our blockchain

Directory: C:\Users\Ashok\Workspace\smartcontract\contracts

Mode	LastWriteTime		Length	Name
-a----	26-10-1985	13:45	419	Migrations.sol
-a----	28-04-2021	17:49	456	__Migrations_sol_Migrations.abi
-a----	28-04-2021	17:51	1892	__Migrations_sol_Migrations.bin

We can see we have abi as well as bin file