How to deploy the Smart Contract on Private Ethereum Network using GETH.

To deploy the smart contract using the geth (go-ethereum) client of ethereum. This guide is not the technical description of the ethereum.

1. **Installation of Geth and set up of private Blockchain on Ubuntu System**

<https://github.com/ethereum/go-ethereum/wiki/Installation-Instructions-for-Ubuntu>

**Installing NodeJS:**

Run the following commands:

curl -sL https://deb.nodesource.com/setup\_6.x | sudo -E bash -

sudo apt-get install -y nodejs

**Installing Ethereum Package:**

Run the following commands:

sudo apt-get install software-properties-common

sudo add-apt-repository -y ppa:ethereum/ethereum

sudo apt-get update

sudo apt-get install ethereum

1. **Write smart contract using Solidity Online IDE** (<https://remix.ethereum.org/> ).

Sample smart contract which will used in this example:

pragma solidity ^0.4.0;

contract sample {

string name;

function setName(string n) public {

name = n;

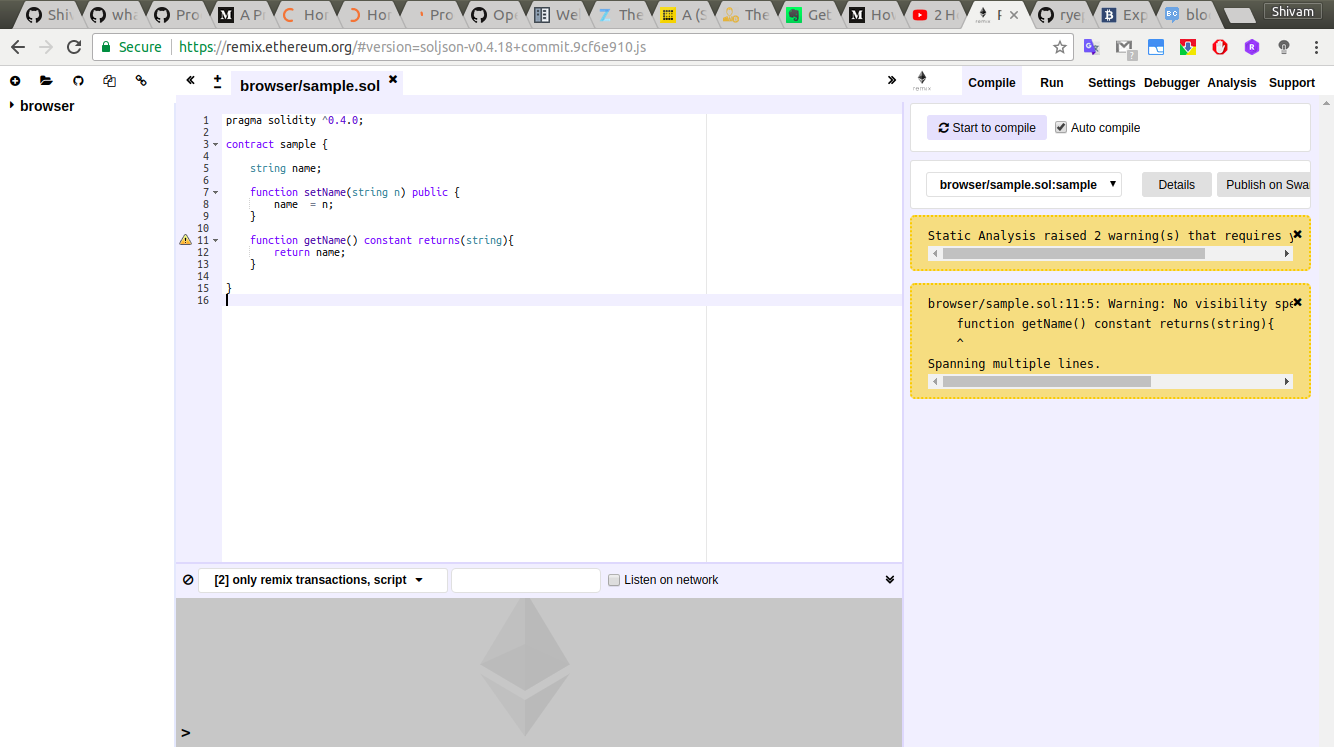
}

function getName() constant returns(string){

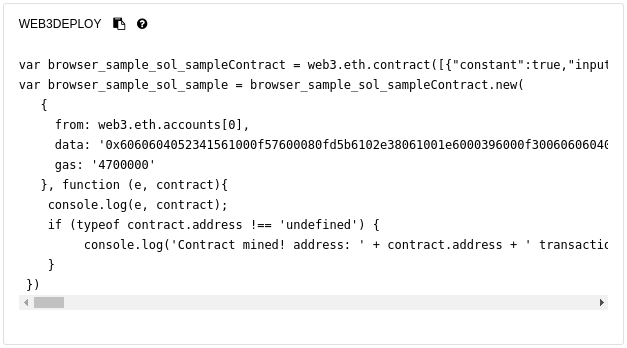
return name;

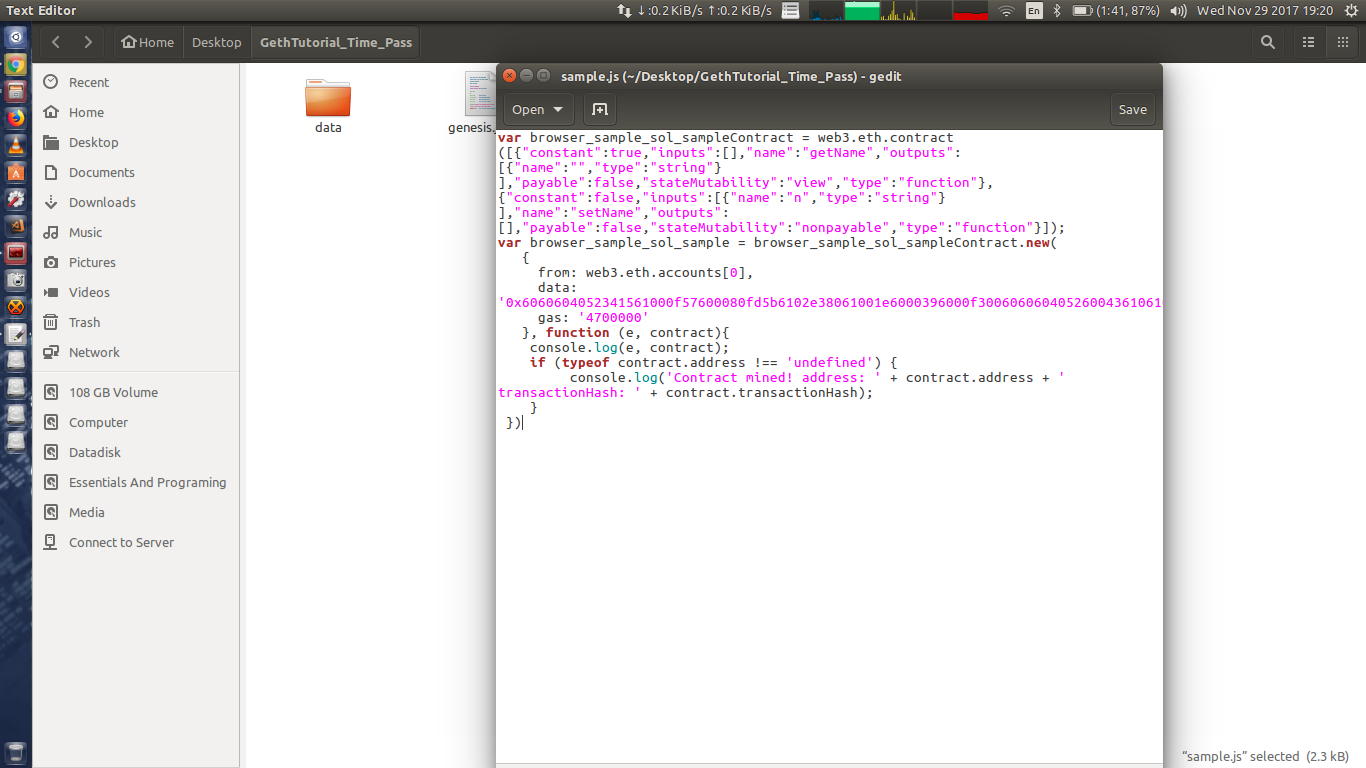
}

}



1. **Under “Compile” tab in remix IDE click on the button “Details” and copy the content of “Web3Deploy”. Now create a file with extension (.js), give name according to you and paste the content in the file.**For this tutorial name of file is sample.js. Used gedit here you can use whatever editor you like. You can copy the given smart contract and get this web3deploy from remix IDE.



****

**4.Now, Let’s initialize our private blockchain using geth.**

The genesis block is the start of the blockchain, and the genesis.json is the file that defines it. It is like the “settings” for your blockchain. For example, the chain configuration, level of difficulty to mine blocks, etc.

Below is the one you can use. Copy and save this in genesis.json file.

{

"nonce": "0x0000000000000042",

"mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",

"difficulty": "0x4000",

"alloc": {},

"coinbase": "0x0000000000000000000000000000000000000000",

"timestamp": "0x00",

"parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",

"extraData": "0x",

"gasLimit": "0xffffffff",

"config": {}

}

geth --rpc --rpcport “8085” --rpcapi=”db,eth,web3,personal,web3” --datadir data --networkid 123 --nodiscover --maxpeers 0 init genesis.json

Let’s go through each options we have used to initialize our private blockchain.

**1. --rpc :**  Enable the HTTP-RPC server

**2. --rpcport value**: HTTP-RPC server listening port (default: 8545)

**3. --rpcapi value :** API's offered over the HTTP-RPC interface

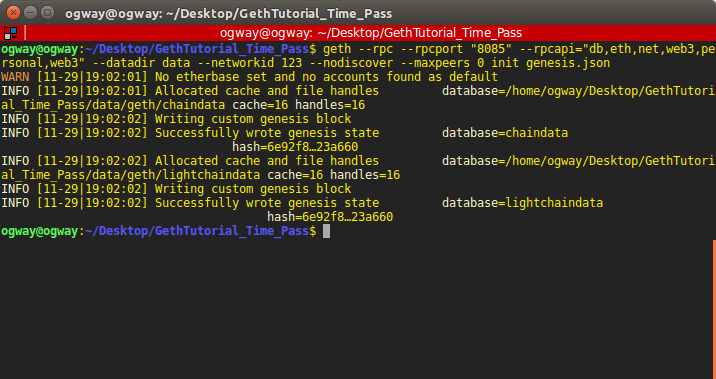
**4. --datadir data :** Data directory for the databases and keystore

**5. --networkid value:** the main network has id 1 (the default). So if you supply your own custom network ID which is different than the main network your nodes will not connect to other nodes and form a private network.

**6. --nodeiscover :** Disables the peer discovery mechanism (manual peer addition)

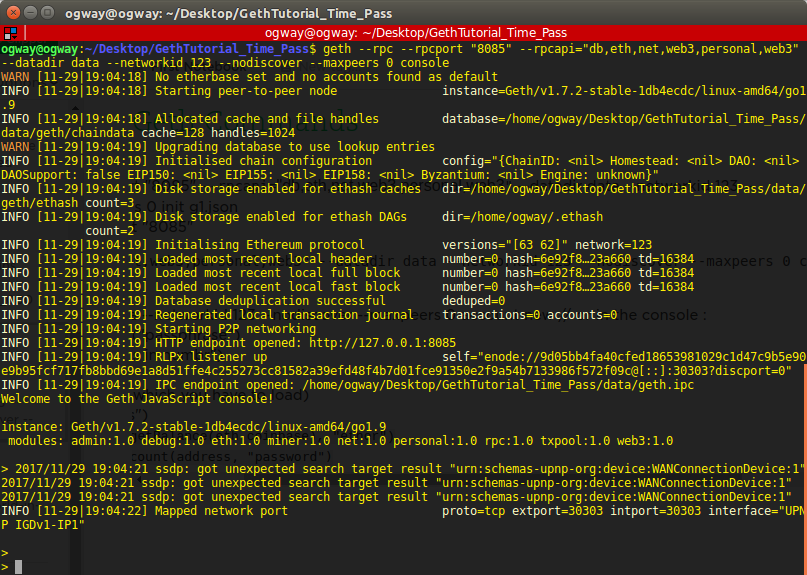
**7. --nodiscover value :** Maximum number of network peers (network disabled if set to 0) (default: 25)

**8. init :** Bootstrap and initialize a new genesis block



**5. Open the geth console for passing the command.**

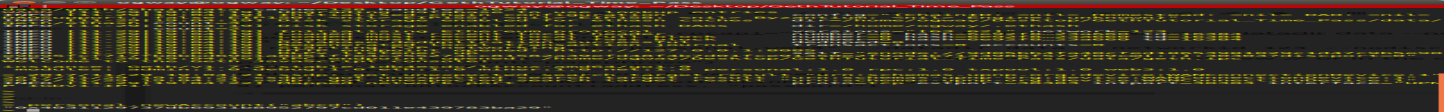
geth --rpc --rpcport “8085” --rpcapi=”db,eth,web3,personal,web3” --datadir data --networkid 123 --nodiscover --maxpeers 0 console



**6.Now, When your geth console is running following needs to be done to deploy the smart contract in ethereum private network. Execute the commands in the geth console.**

1. **Create a account** : personal.newAccount("<passphrase>")  
    Passphrase is the password for the account.   
    Note: If you forgot the password then there is no way to recover it.

You will get return the account address. Note that address it will be used in next step.



B. **Unlock the Account** : personal.unlockAccount(“<Account Address>”, <Passphare>)

To unlock an account in geth console you will need to pass three arguments,

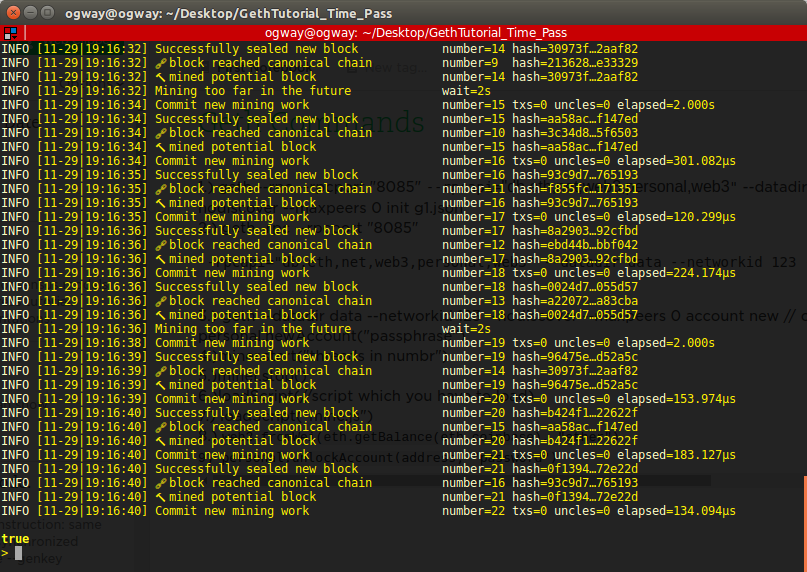
* + 1. Account Address which you want to unlock.
    2. Password to unlock that account.
    3. (Optional) Time period in seconds for which the account is unlocked.  
       Note: If you do not provide, Account is unlocked until geth console is running.

C. **Mining**: When you start up your ethereum node with geth it is not mining by default. You can also start and stop CPU mining at runtime using the console. miner.start takes an optional parameter for the number of miner threads.

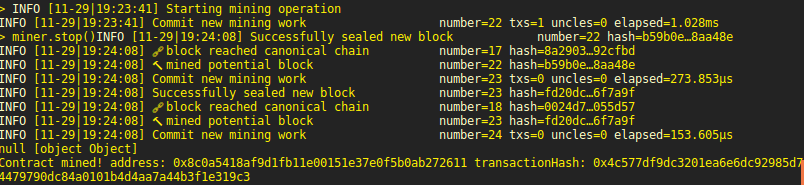
Basically, This command will collect certain amount of gas value so you can execute your smart contract methods and perform transactions.

miner.start(number of threads)

Miner.stop()

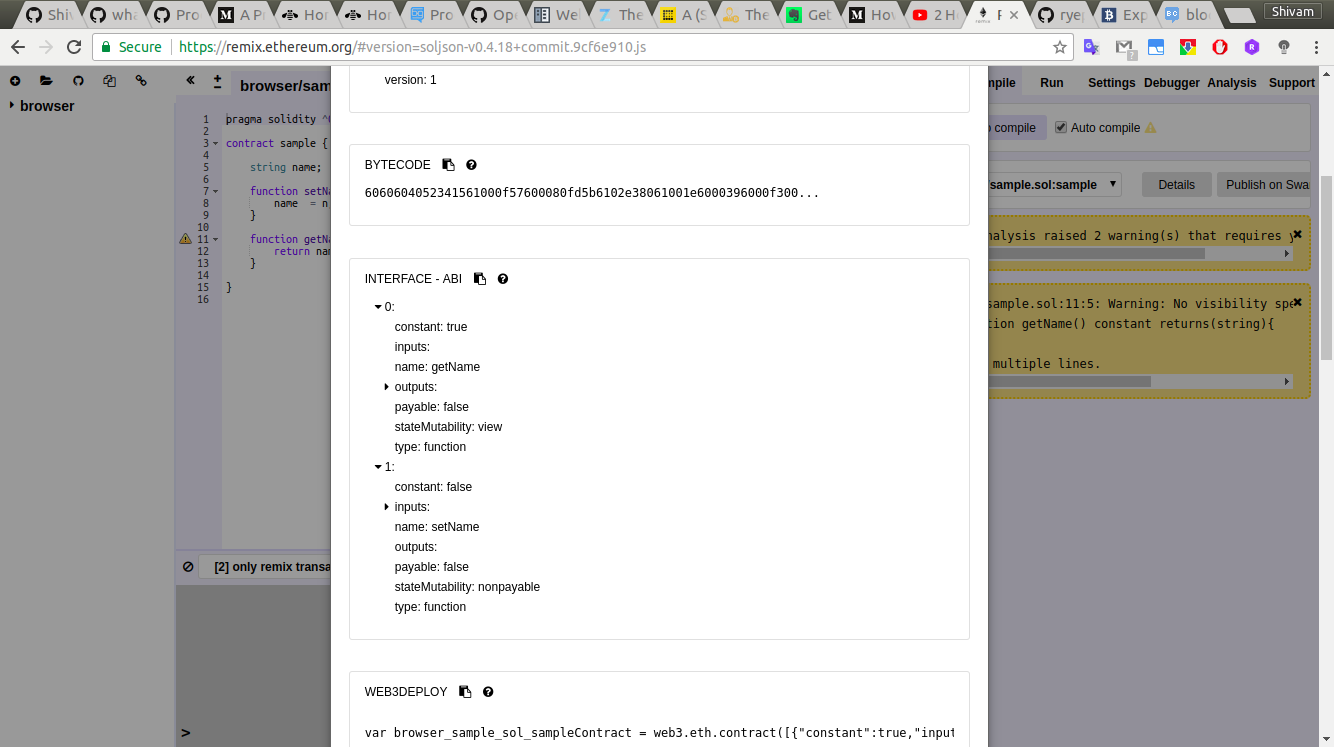


**D. Now load the contract (Deploy):** loadScript(“<File\_Name>”)  
  
 This will create a transaction and a contract will be mined. In return you will get contract address which will usefull in creating APIs.



**7.Now copy the ABI Interface from the remix IDE under the same details section.**

ABI is Application Binary Interface which is signature for the smart contract.

****

**8.Now you can start creating API using any server side programming language to call smart contract methods and access data from blockchain.**

NOTE: For the purpose of this tutorial I have used node.js for creating API. The common settings which are used for nodejs are given below in the picture.   
