BlockChain

Installation procedures :

Installing pipx

```
python3 -m pip install --user pipx
python3 -m pipx ensurepath
```

Installing brownie

```
pipx install eth-brownie
```

Issue wtih above: Unable to install other pacages

Updated version for installing brownie(05-032022):

```
pip install eth-brownie
```

Accounts

Checking all accounts

```
brownie accounts list
```

Adding accounts

```
brownie accounts new myAccount
```

Interacting with accounts programatically

Loading accounts

```
from brownie import accounts
account = accounts.load("testAccount")
```

Adding accounts via private key

```
from brownie import accounts
account = accounts.add(<private_key that you want to add.Can be found in Ganache>)
```

Loading smart contracts

Lets say that CrudOp.sol file has following code

```
pragma solidity ^0.4.22;
contract UserCrud {
  struct UserStruct {
.....
```

While importing we give UserCrud instead of CrudOp

```
from brownie import UserCrud
```

Opening Brownie console

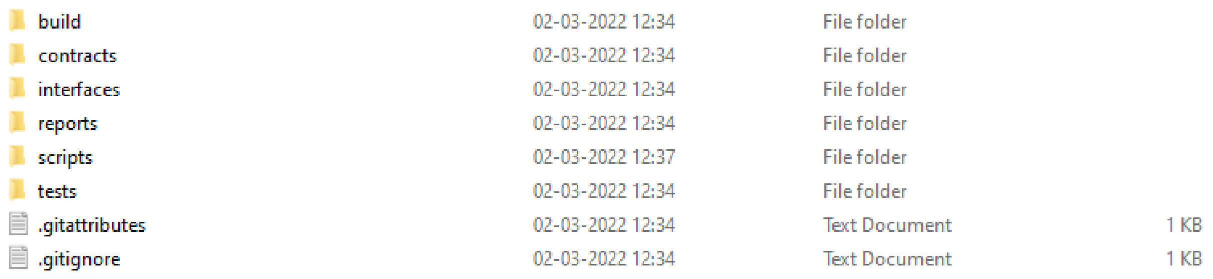
```
brownie console
```

Creating a new brownie project

1)Initiate a new project

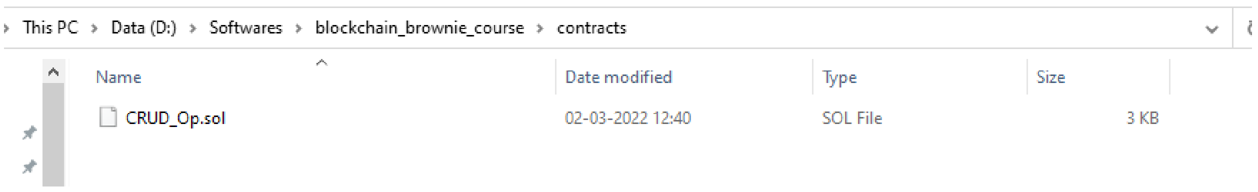
```
brownie init
```

2)Once the above command is executed you should see a file structure similar to follow



build	02-03-2022 12:34	File folder	
contracts	02-03-2022 12:34	File folder	
interfaces	02-03-2022 12:34	File folder	
reports	02-03-2022 12:34	File folder	
scripts	02-03-2022 12:37	File folder	
tests	02-03-2022 12:34	File folder	
.gitattributes	02-03-2022 12:34	Text Document	1 KB
.gitignore	02-03-2022 12:34	Text Document	1 KB

3)Within contract files , insert the smart contract file(.sol) that you want deployed



This PC › Data (D:) › Softwares › blockchain_brownie_course › contracts

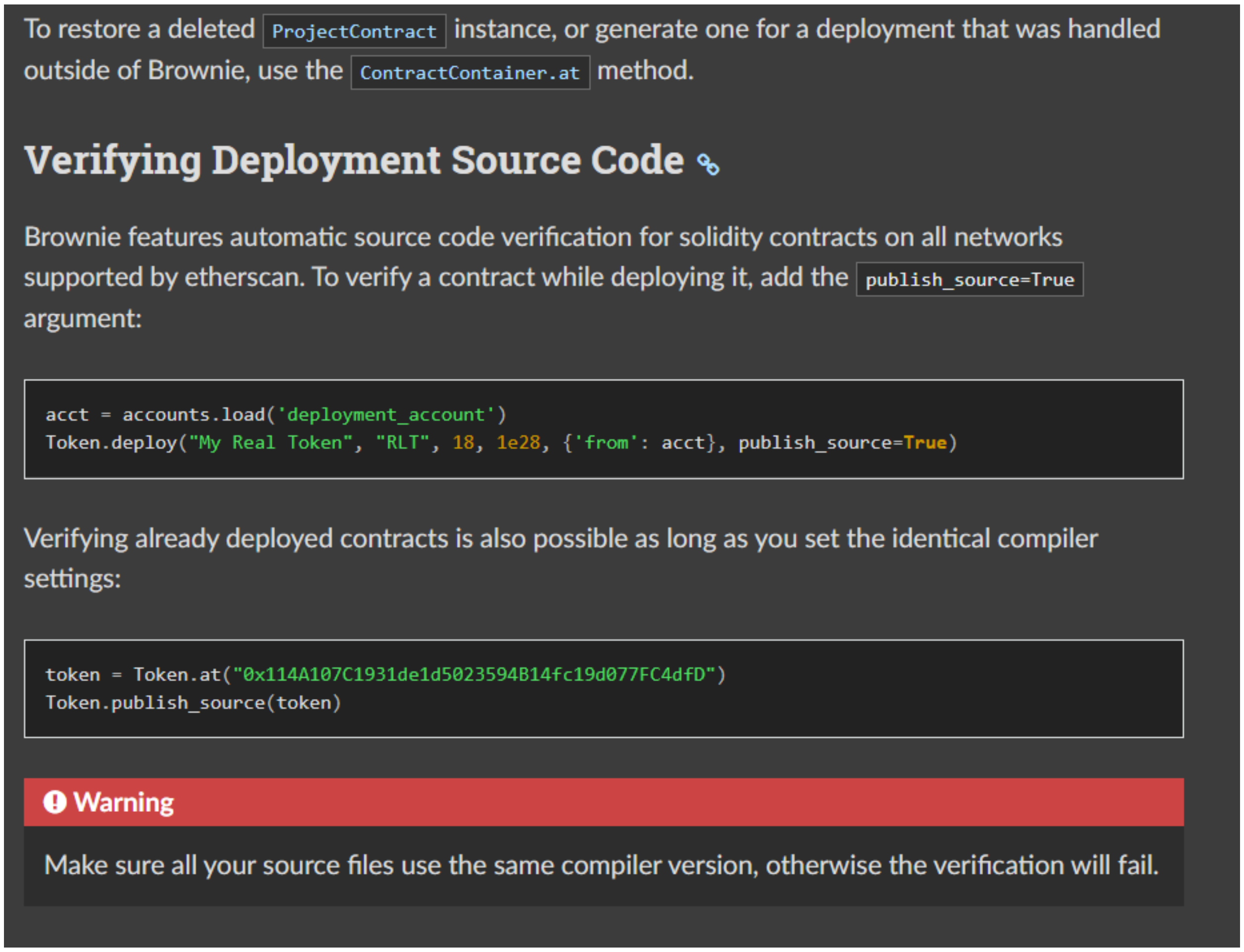
Name	Date modified	Type	Size
CRUD_Op.sol	02-03-2022 12:40	SOL File	3 KB

4)Within scripts create a new file named deploy with the following code .For testing

```
def main():
  print("Hello")
```

Fetching Smart contracts from servers

Link : https://eth-brownie.readthedocs.io/en/stable/deploy.html#interacting-with-deployed-contracts


To restore a deleted `ProjectContract` instance, or generate one for a deployment that was handled outside of Brownie, use the `ContractContainer.at` method.

Verifying Deployment Source Code

Brownie features automatic source code verification for solidity contracts on all networks supported by etherscan. To verify a contract while deploying it, add the `publish_source=True` argument:

```
acct = accounts.load('deployment_account')
Token.deploy("My Real Token", "RLT", 18, 1e28, {'from': acct}, publish_source=True)
```

Verifying already deployed contracts is also possible as long as you set the identical compiler settings:

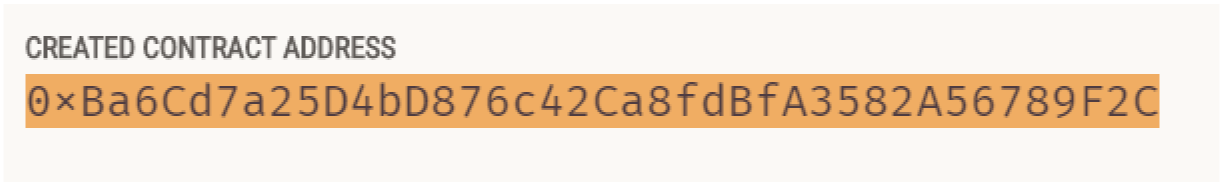
```
token = Token.at("0x114A107C1931de1d5023594B14fc19d077FC4dfD")
Token.publish_source(token)
```

Warning

Make sure all your source files use the same compiler version, otherwise the verification will fail.

```
from brownie import accounts,UserCrud,Contract
orgName = "GrayLogic"
regNo = 223
name = "Amos"
course = "Hindi"
certificate = "passed"
userAddressList = []
def fetchContract(address):
  contractData = UserCrud.at(address)
  print("Contract fetched : ",contractData)
def deploy_simple_storage():
  account = accounts[0]
  simpleStorage = UserCrud.deploy({"from":account})
  # simpleStorage.insertUser(account,orgName,regNo,name,course,certificate)
  print(type(simpleStorage))
  print(type(UserCrud))
  print(simpleStorage)
def addUser():
  # print(ac)
  accounts.add()
  account = accounts[-1]
  simpleStorage = UserCrud[-1]
  simpleStorage.insertUser(account,orgName,regNo,name,course,certificate)
  userAddressList.append(account)
def retrieveAllUsers():
  for i in userAddressList:
    retrieveUserDetails(i)
def retrieveUserDetails(address):
  # print("helo?")
  # account = accounts[0]
  print("Users got : ",UserCrud[-1].getUser(address))
def updateUserName():
  simpleStorage = UserCrud[-1]
#   simpleStorage.updateUserName(account,orgName,regNo,name,course,certificate)
def main():
  # deploy_simple_storage()
  # addUser()
  # addUser()
  # print("User Addresses : ",userAddressList)
  # retrieveAllUsers()
  fetchContract("0xBa6Cd7a25D4bD876c42Ca8fdBfA3582A56789F2C")
```

Remember to set contract address not fetching address



CREATED CONTRACT ADDRESS

0×Ba6Cd7a25D4bD876c42Ca8fdBfA3582A56789F2C