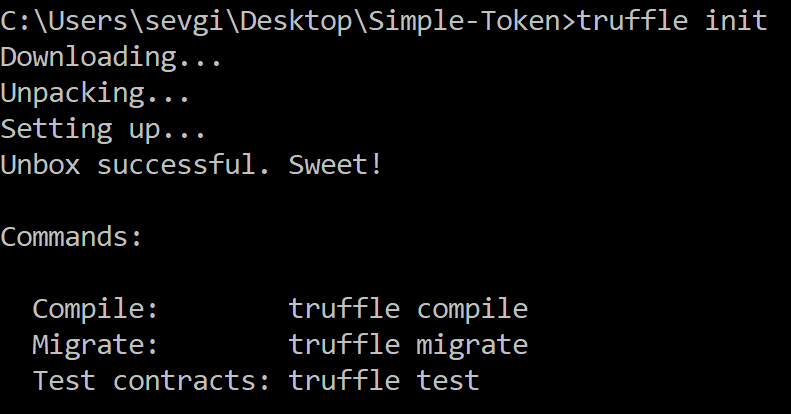
# Exercises: Simple Token Unit Testing

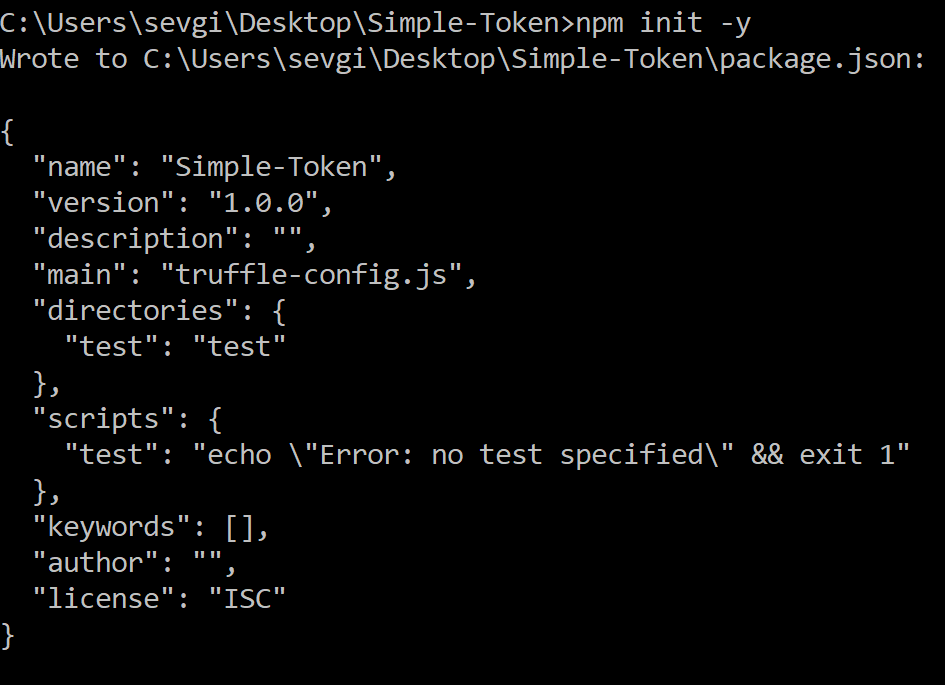
This document describes the **exercise assignments** for the ["Blockchain Academy" course @ Software University](https://softuni.bg/courses/programming-fundamentals). In this lesson, we learned the **basics of Solidity** programming language. The goal of this exercise is to get practical skills in writing simple smart contracts in Solidity, publishing and testing contracts in the Remix IDE.

## Setup project

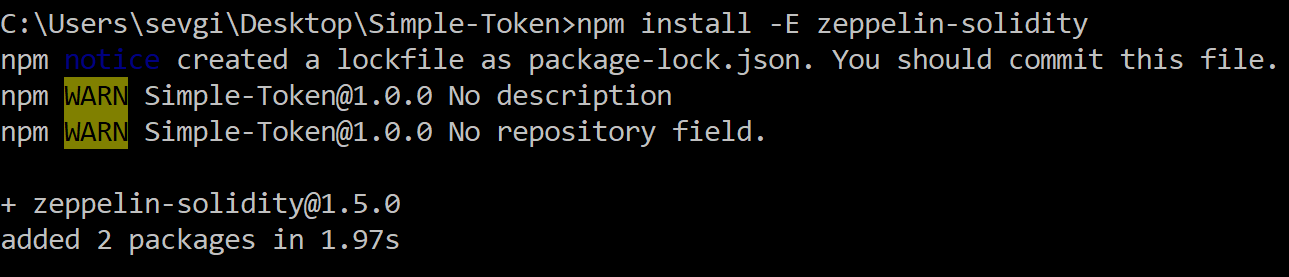
1. Initialize Project in the folder **Simple-Token**



1. Create **package.json** file

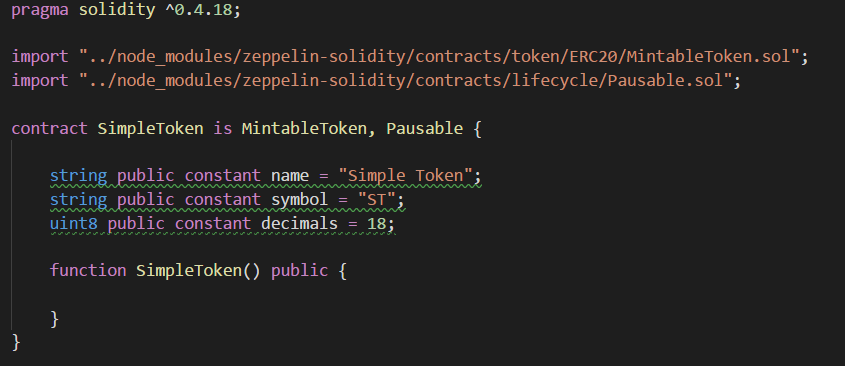


1. Then install **open-zeppelin** library



## Create Simple Token

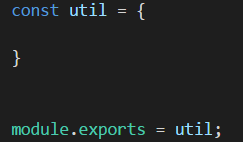
1. Create **SimpleToken.sol** contract in contracts folder
2. Our token will inherit the **MintableToken** and **Pausable** from open-zeppelin, we will just give it name, symbol and decimals



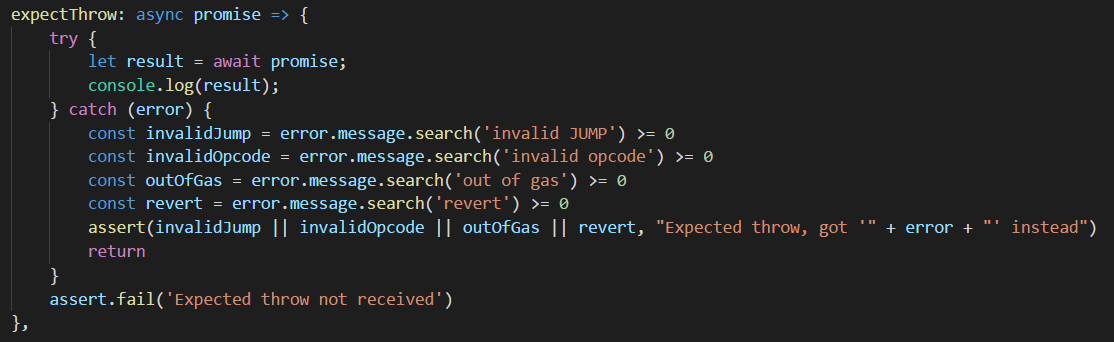
1. Before writing tests for our token, go and look the implementations of the inherited contracts **BasicToken**, **StandartToken**, **MintableToken** and **Pausable** lifecycle contract
   1. BasicToken – “../node\_modules/zeppelin-solidity/contracts/token/ERC20/BasicToken.sol”
   2. StandartToken – “../node\_modules/zeppelin-solidity/contracts/token/ERC20/StandartToken.sol”
   3. MintableToken – “../node\_modules/zeppelin-solidity/contracts/token/ERC20/MintableToken.sol”
   4. Pausable – “../node\_modules/zeppelin-solidity/contracts/lifecycle/Pausable.sol”
2. After understanding them go and look their unit tests and try to understand them too
   1. BasicToken Tests– “../node\_modules/zeppelin-solidity/test/token/BasicToken.test.js”
   2. StandartToken Tests– “../node\_modules/zeppelin-solidity/test/token/StandartToken.test.js”
   3. MintableToken Tests– “../node\_modules/zeppelin-solidity/test/token/MintableToken.test.js”
   4. Pausable Tests– “../node\_modules/zeppelin-solidity/test/token/Pausable.test.js”

## Test the Simple Token

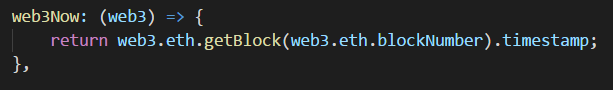
1. Now after looking and understanding the unit tests of the smart contracts we inherited we can create our own unit tests
2. Go to **/test** folder and create **SimpleToken. rtest.js**
3. Also create the file **util.js** and which will be our helper



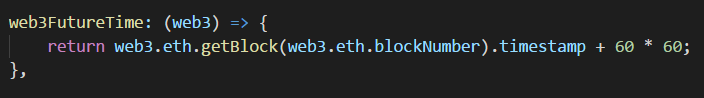
1. **expectThrow**



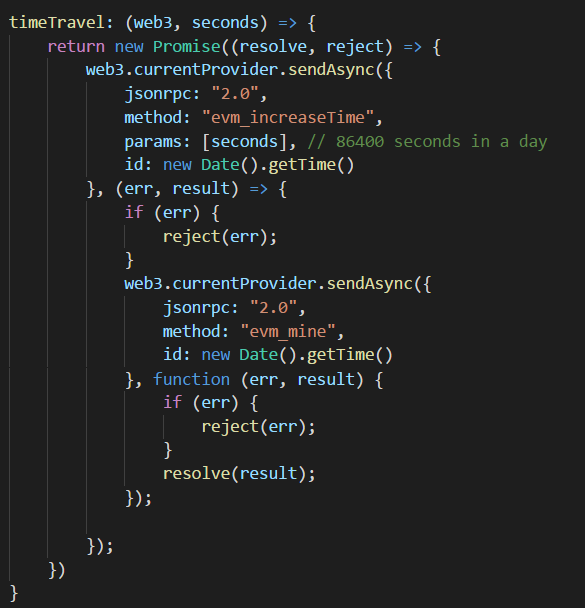
1. **web3Now**



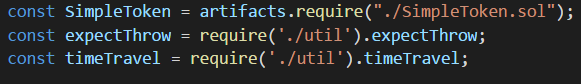
1. **web3FutureTime**



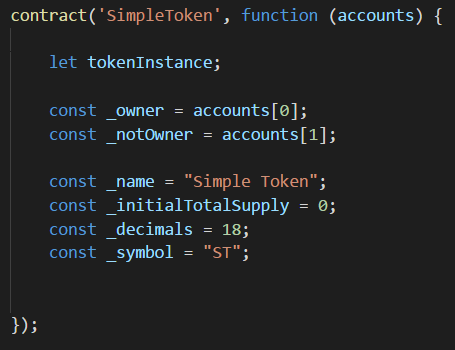
1. **timeTravel**



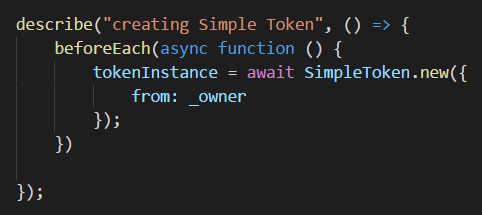
1. In **SimpleToken.test.js** import and the needed files



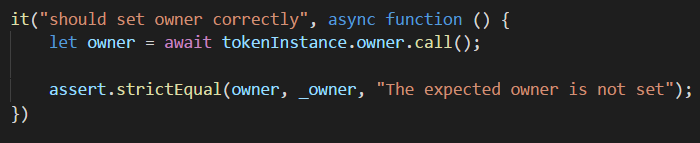
1. Then create the contract function and the parameters we will need for the tests



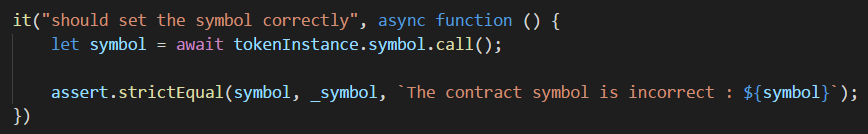
1. Describe and beforeEach:



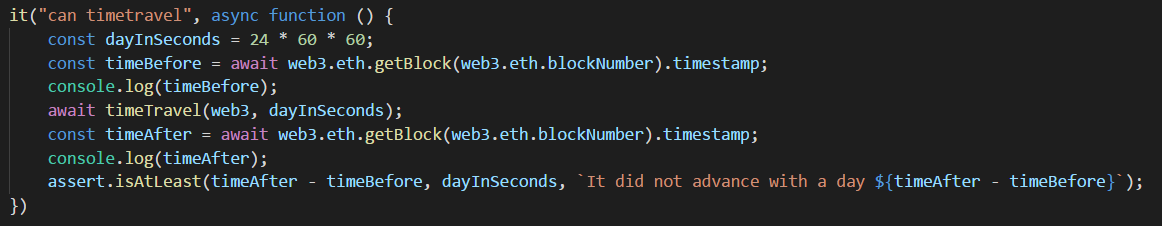
1. Now we can write some tests:
   1. Should set owner correctly



* 1. Should have no totalSupply
  2. Should set the name correctly
  3. Should set the symbol correctly

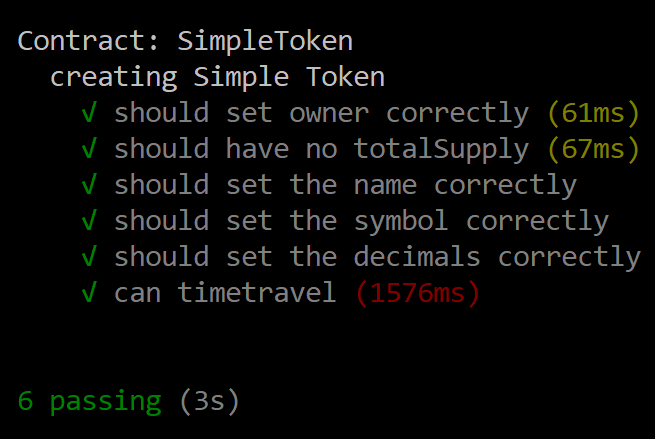


* 1. Should set the decimals correctly
  2. Can **timetravel**:



* 1. And more…

1. Finally open the console and go to the **/test** folder and run **truffle test**



# What to Submit?

Create a **zip file** (e.g. your-username-simple-token-unit-testing.zip) with the source code and images holding the results above.

Submit your **zip** file as **homework** at the course Web site.