CSF202 – Lab Class 7 – Wednesday 5/11/2020

This lab class involves getting some hands-on experience with coding an example of the singleton design pattern in Java.

□ Task 7.1

Program, in Java, a singleton class that models a settings manager. The settings manager will manage all global settings for an embedded device (running Java). The settings manager should have the following properties (i.e., instance variable with appropriate setters and getters):

- Device name: A String which the user assigns so that they can easily recognise the device.
- Back-light level: A floating point value between 0 and 1 which indicates the brightness of the back-light. 0 would be 0% power to the back-light, 1 would be 100% power to the back-light.
- Remember to make it a Singleton you will need:
  - A private constructor
  - A public static getInstance method
  - A private static instance

(You can imagine a real settings manager to have many more settings. Adding extra settings is straightforward).

Once you have implemented your class you will need to create a main class with a main method. There you should test your settings manager class.

Test that the singleton constructor works correctly. That is, that you cannot actually create an instance of the settings manager in the main class file using the settings manager constructor – it should be private after all.

Next, you should check the uniqueness of the instance, i.e., that only one settings manager instance is created. To this end, you might do the following:

- Set variable `sm1` to be the result of the `getInstance()` method of the settings manager class.
- Set variable `sm2` to be the result of the `getInstance()` method of the settings manager class.
- Set the device name via the variable `sm1`.
- Print the device name according to the settings manager via the variable `sm2`.
- The printed device name should be the same as what was set.