PROJECT NAME - JUnit5 Testing Module

Getting Started

Welcome to this JAVA TESTING FRAMEWORK workshop which aims at providing basics usage of most used frameworks = JUnit 5 and JMeter.

- JUnit 5 is a unit testing framework for the Java programming language.
- JUnit has been important in the raise of test-driven development, and is one of a family of unit testing frameworks which is collectively known as xUnit for various languages.
- Latest version of the framework, JUnit 5, introduces many add-ons no more compatible with previous ones. Thus, JUnit5 packages reside under org.junit.jupiter, while previous versions reside under org.junit.vintage.
- Offical web site - https://www.junit.org

Author

Cédric OBEJERO - cedric.obejero@tanooki.fr

Release

OpenJDK 17.0.1 JUnit5 Jupiter v5.8.2 JUnit5 Vintage v5.8.2 JUnit5 Platform v1.8.2

Powered by <shields.io>

Folder Structure

To begin with, you can create a Java project using `Ctrl + Shift + P` then `Java: Create Java Project ...`

The workspace contains three folders by default, where:

- `src`: the folder to maintain sources
- `lib`: the folder to maintain dependencies
- `bin`: the folder to maintain compiled CLASS or JAR files

> If you want to customize the folder structure, open `.vscode/settings.json` and update the related settings there. Compiling and debugging is based on WSL2 Unbuntu 20.04 LTS environnement.

Dependency Management

The `JAVA PROJECTS` view allows you to manage your dependencies. More details can be found here.
IMPORTANT It has to be completed by dependencies under WSL2 environment installing OpenJDK

Installing

- Under MS Windows platform ensure `WSL2` is intalled and `Remote WSL Extension` is installed under `VS Code` - https://code.visualstudio.com/blogs/2019/09/03/wsl2
- Restart and launch `VS Code` from WSL2 Ubuntu environment

- Install the following extensions under `VS Code` within WSL environment

```
# Extension Pack for Java
# Debbuger for Java
# Language Support for Java(TM) by Red Hat
# Project Manager for Java
# Test Runner for Java
```

- Under WSL2 Ubuntu environment, from the Terminal, install Java Development Tools

```
$ sudo apt update
$ sudo apt install openjdk-17-jdk
$ sudo apt isntall junit5
$ export JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64
$ export PATH=$PATH:$JAVA_HOME/bin
```

> WARNING - `export` command apply changes only for the current session, update `.bashrc` if needed

- Under WSL2 Ubuntu environment, from the Terminal, Test Java environment

```
$ java -version
openjdk version "17.0.1" 2021-10-19
OpenJDK Runtime Environment (build 17.0.1+12-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 17.0.1+12-Ubuntu-120.04, mixed mode, sharing)
```

JUnit5 - Development & Testing

Under `VS Code` do as follow

- `Ctrl + Shift + P` and select `Java: Create Java Project`
- Define name of the project as `junit5-testing` under `$HOME/java`
- From VS Code Terminal download inline command tool for JUnit as follow

```
$ cd $HOME/java/junit5-testing/bin
$ wget https://repo1.maven.org/maven2/org/junit/platform/junit-platform-console-standalone/1.8.2/junit-platform-console-standalone-1.8.2.jar
$ ln -s junit-platform-console-standalone-1.8.2.jar junit-platform-console-standalone-latest.jar
```

Useful command lines

```
# Build a specific Java source code
$ javac -d bin -cp /usr/share/java/*:bin src/CodeUnitTest.java
```

```
# Execute tests for a specific class
$ java -jar bin/junit-platform-console-standalone-latest.jar --class-path bin --
select-class CodeUnitTest
# Execute tests for all class available in a specific class path
$ java -jar bin/junit-platform-console-standalone-latest.jar --class-path bin --
scan-class-path
```

Your mission, M. Hunt - Code and Test ISBN Class

Ethan, your mission if you accept it, is to code and test ISBN Class in Java.

The world company has been recruited to develop a great new book management application for Oxford University Library. Before lunch time today, we need you to develop and test ISBN Class.

The ISBN (International Standard Book Number) is an international number which makes it possible to identify, unique way, each edition of each published book, whether its medium is digital or on paper. He is intended to simplify IT management for all those involved in the book chain (printer, publisher, bookseller, library, etc.).

Before 2007, the ISBN-10 number consisted of four segments, three segments of variable length and a fixed-length segment, the total length of the ISBN included 10 digits. Since January 1 2007 the length was extended to 13 digits by adding an initial group of 3 digits.

To facilitate their computer management, each book bears a barcode to the EAN 13 standard and a ISBN-13 code from which it is derived.

This code has 13 digits and is now mandatory (since January 2007), and must be used for all new ISBN codes instead of the 10-digit code. Indeed, the old numbering is reached saturation and would no longer allow the simple assignment of groups of specific codes to different publishers.

The conversion of an old ISBN-10 code into an ISBN-13 code compatible with the EAN standard is automatic (and mandatory for all electronic transactions from January 2007).

The ISBN class will therefore have an isbn attribute of type string to keep the code. The string type is the best choice because:

- ISBN codes are structured by segment and it will therefore be easier to process a character string that a number
- the ISBN codes separate the different segments with a dash: "2-266-11156-6" corresponds to the code 2266111566

The ISBN class will manage ISBN-10 codes and ISBN-13 codes.

ISBN codes include a checksum calculated from the previous 9 digits for a code ISBN-10 and the preceding 12 digits for an ISBN-13 code. The ISBN class will offer the following services:

- a method Normalize() which receives an isbn code of type string and which returns the isbn code without the hyphens (for an ISBN-10 code, "2-266-11156-6" −→ "2266111566")

- the getSize() methods which return the number of digits contained in a “cleaned up” isbn code of type string (for an ISBN-10 code, "2-266-11156-6" −→ 10)
- isValid() methods which return true for a valid isbn code (in size, in numbers and in control key) otherwise they return false: for an ISBN-10 code, "2-266-11156-6" −→ true

INFO If the ISBN is invalid (in size, digits or checksum), it should not be stored in the isbn attribute. In this case, an empty string will be assigned to the attribute. This behavior should be implemented for setISBN() constructor and setter. If it is valid, it will be kept in its initial form uncleaned ("2-266-11156-6" for example).

For more information regarding ISBN numbers validation, please refer to: http://fr.wikipedia.org/wiki/International_Standard_Book_Number

Hereunder, you will find the prototype of the ISBN Class expected

```
public class ISBN {

    private String isbnValue;
    private int isbnSize;

    private boolean isValidISBN10(String isbnToValidate);
    private boolean isValidISBN13(String isbnToValidate);

    public ISBN();
    public ISBN(String initialeISBN);
    public void setISBN(String inputISBN);
    public String getISBN();

    public int getSize();
    public int getSize(String inputISBN);
    public boolean isValid();
    public boolean isValid(String inputISBN);
    public String Normalize(String inputISBN);

}
```

To validate this mission, the unit tests provided has to cover each method implemented and validate a set of 1000 ISBN numbers test values. Moreover, for performance validation purpose, ISBN validation must reach the rate of 5000 per second.

At last but not least, if you or one of your team members does not provide the code with the tests, the school will deny being able to validate your diploma.

Good luck Ethan !