Application Title (Splash) Screen with Borders

The due date for this assignment is on January 31st, 2021 and you'll demonstrate in the week of February 1st, 2021.

Objectives

The purpose of the lab is to give you a hand-on experience of how to create an application splash (title) screen and how to create a border for a GUI component. Splash screens are an intrinsic part of any modern GUI application. Their primary purpose is to inform the user that the application is starting up while at the same time it is loading all necessary resources.

The Nature of Things

Swing - Class JWindow

The Swing top-level container JWindow is an extension of java.awt.Window that similarly to JFrame uses a JRootPane as its single component. Other than this core distinction, JWindow does not change anything defined by the Window class. In AWT, one common reason for using the Window class was to create a popup menu. Since Swing explicitly provides a JPopupMenu class, there is no more need to use JWindow for this purpose. The only time you will use JWindow is if you have got something that needs to be displayed in its own window without the adornments added by JFrame. Since the close button of the JFrame is not part of JWindow, the window can only be moved or closed programmatically. One frequent use for JWindow is to display a splash (title) screen when an application is starting up. Many programs display such a screen, possibly containing copyright information, resource loading status, etc. See code examples for details: *SplashScreenDemo.java*

Component Borders

You can place a border around any Swing component that extends JComponent. There are eight border styles: Bevel, Soft Bevel, Empty, Etched, Line, Matte, Titled, and Compound. The MatteBorder border area can be filled with a solid color or an icon. The JComponent class contains a border property that is inherited by all Swing components. (Top-level containers that do not inherit from JComponent, like JFrame, JWindow, and JDialog, cannot have borders.) By default, the border property is null (no border), but you can access and modify it using the getBorder() and setBorder() methods. Once you have set a component's border, the component always paints itself using that border, and the insets of the border replace the component's default insets.

Borders are grouped into a separate package within the Swing hierarchy: javax.swing.border. All Swing borders directly or indirectly extend the AbstractBorder class, which in turn implements the more fundamental Border interface. Swing allows you to mix any number of border styles into a single border object. This gives Swing borders a useful compositional feature not often found in other windowing toolkits. See code examples for details: SwingBordersDemo.java.

Useful link: http://www.java2s.com/Code/Java/Swing-JFC/BorderDemo.htm

JavaFX

JavaFX does not have a container similar to JWindow but the Stage class could serve the same purpose (Stage actually extends Window in JavaFX). See code examples (SplashScreenDemoFX.java) for details (: In Java FX you can set border in two different ways. You can use CSS to style the borders around a node; or you can use the Border class introduced in Java 8. See code examples for details: BordersDemoFX.java.

Application Title (Splash) Screen with Borders

Starting Task

Download the CST8221_Lab2_code.zip file from Brightspace. Extract the contents. Four different code examples are provided for you. Two are Swing examples and the other two are similar FX examples. Compile and run the examples one by one and examine carefully the code. Consult the Java documentation to find out what the different classes and methods are implementing.
*Once you understand how the code works modify **SplashScreenDemo.java** or **SplashScreenDemoFX.java** so that when run it displays a splash screen with an image of your liking and your name at the bottom of the splash window.*

Note: To launch the program from the command line you must find the folder containing your class files (or jar file) and copy the image there. Then you can launch the program.

Another Note: Since Java 6 there it is possible to display a simple splash screen without writing any line of code. You can ask the virtual machine to display an image before even starting to load the application classes. There are also other ways to create splash scenes with Swing. For more details how to do that visit the following link:
http://docs.oracle.com/javase/tutorial/uiswing/misc/splashscreen.html

As you can see from the examples JavaFX does not have a container similar to *JWindow* but the ***Stage*** class serves the same purpose (Stage actually extends Window in JavaFX). Also JavaFX has introduced a special ***Preloader*** class which can be used to create complex splash screens. Visit the link below for more details.
http://docs.oracle.com/javafx/2/deployment/preloaders.htm

Deliverables

Demonstration to the lab instructor in your respective lab sessions. You may be asked some questions about the code.
Compress your modified java files and required images into a zip archive and submit this in the respective Brightspace submission folder.

Marks

1.0% towards the final Lab Assessments grade.
The lab exercise will be marked according to the following marking method:

```
public int markLab2(boolean demonstration, boolean program){
    int mark = 0;
    if(demonstration & program) mark = 1;
    return mark;
}
```

The purpose of borders:
“Good fences make good neighbors.”

But also
“Fences restrict the view”