Module Introduction

Event-Driven Programming

Introduction

- This module focuses on developing event-driven programs.
  - When an event occurs, the program responds to that event appropriately = key!
  - User events, system events, etc. trigger actions to be taken.
- We will use Java for this module.
  - Continues building your Java skills from other programming modules.
  - We will focus on building GUI-based desktop applications in particular as they provide an ideal vehicle for teaching and learning event-driven programming.
    - Interactions with GUI => events!

Assessment and Organisation

- Event-Driven Programming is a 100% continuous assessment (CA) module.
- The CA will consist of a <u>CA Portfolio</u> (circa 4 pieces of work) and a <u>CA Class Exam</u>
  - CA Portfolio ≈ 70-75%
  - CA Class Exam ≈ 25-30%
- Moodle will be used for CA submissions and for distributing materials (notes, examples, etc.)

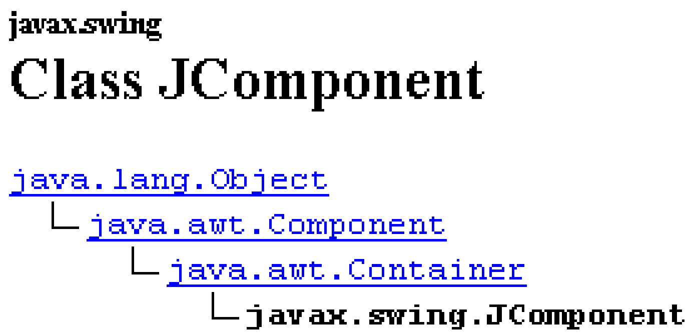
GUI Java Packages

- Abstract Windows Toolkit (AWT)
  - Original GUI package in Java.
  - "Component" tops the AWT hierarchy.
  - Many useful subclasses: Button, CheckBox, etc.
  - AWT components are "heavyweight" i.e. they make heavy use of the native systems resources and OS.
    - They will look like standard platform (e.g. Windows, Mac) components.

Continued…

- Swing
  - Newer GUI package than AWT.
  - Builds on the AWT (it doesn't replace it). Both are part of the Java Foundation Classes (JFC).
  - In general, "JComponent" tops the Swing hierarchy (except for top level containers).
  - Swing components are "lightweight" i.e. fully implemented in Java itself.
  - By far the most used GUI Java package.
  - Huge number of third-party Swing based components too.

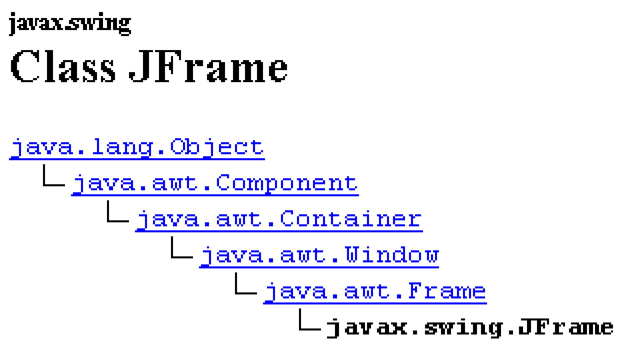
Examples of the Swing / AWT Hierarchies



javax.swing

Class JComponent

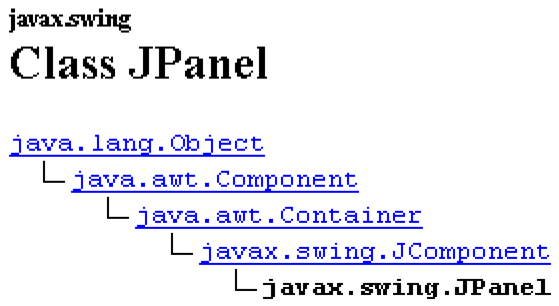
```
java.lang.Object
  └ java.awt.Component
      └ java.awt.Container
          └ javax.swing.JComponent
```



javax.swing

Class JFrame

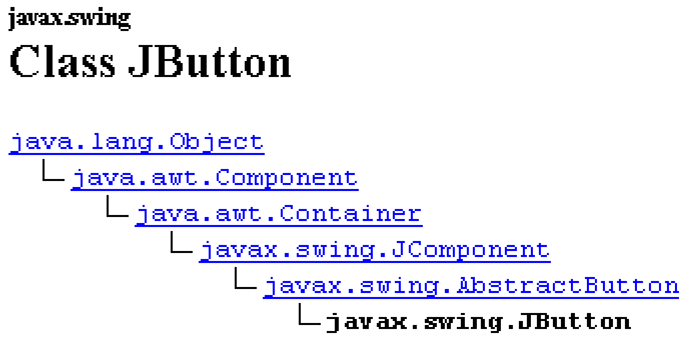
```
java.lang.Object
  └ java.awt.Component
      └ java.awt.Container
          └ java.awt.Window
              └ java.awt.Frame
                  └ javax.swing.JFrame
```



javax.swing

Class JPanel

```
java.lang.Object
  └ java.awt.Component
      └ java.awt.Container
          └ javax.swing.JComponent
              └ javax.swing.JPanel
```



javax.swing

Class JButton

```
java.lang.Object
  └ java.awt.Component
      └ java.awt.Container
          └ javax.swing.JComponent
              └ javax.swing.AbstractButton
                  └ javax.swing.JButton
```

Continued…

- Standard Widget Toolkit (SWT)
  - Third party alternative to standard Java AWT/Swing packages.
  - Uses native OS libraries through the Java Native Interface.
  - Maintained by the Eclipse Foundation.
- JavaFX
  - For developing rich internet application / client interfaces (using FXML for layouts, CSS, etc.).
  - Versions:
    - JavaFX version 1 used a scripting language (JavaFX Script), which was widely disliked by Java developers.
    - JavaFX version 2 on (circa late 2011) uses plain old Java classes and APIs again (JavaFX Script abandoned).
  - Still quite new, JavaFX might become a standard for building Java GUIs in the future. However:
    - SWT is still competing, huge number of Swing applications out there, other competing frameworks/technologies such as Adobe Flex, etc. may delay or scupper this somewhat.
    - Relatively little support at the moment compared to Swing, SWT, etc. (changing!)

Continued…

- Oracle are promoting JavaFX as the successor to Swing.
  - JavaFX is included in Java 7 update 6 onwards (previously a separate SDK, etc.).
- Swing will remain alongside it, though.
  - Still the dominant GUI Java package out there, even if a little long in the tooth (a.k.a. established).
- Support is provided for JavaFX / Swing interoperability.
  - SwingNode class for using a Swing component in a JavaFX application.
  - JFXPanel class for using a JavaFX component in a Swing application.

Continued…

- What will we use for this module?
  - We will focus mainly on using the newer JavaFX package.
    - “Shiny”.
  - However, I will also provide Swing examples, etc. as appropriate to highlight similarities, differences, etc. to JavaFX.
    - Practical and useful. Swing is very widely used/worth knowing.
  - This will provide the best of both worlds.
    - JavaFX/Swing interoperability is possible anyway, as noted.
  - Note that we will also cover other Java features that are not specific to JavaFX/Swing.

Tooling Up

- You will need to install/use the following:
  - Recent version of Eclipse IDE (Luna or later)
    - Can use NetBeans if preferred. Oracle tutorials/examples tend to use NetBeans.
  - Java JDK 8.
    - JavaFX is included in Java 8.
  - e(fx)clipse
    - Add on for Eclipse IDE for JavaFX support.
  - JavaFX Scene Builder
    - Visual editor for JavaFX GUIs.