Java Programming 2
GUI programming with Swing

Mary Ellen Foster

MaryEllen.Foster@glasgow.ac.uk

Outline

GUI programming in Java

Introduction to Swing

Overview

Components

Events

Extended example

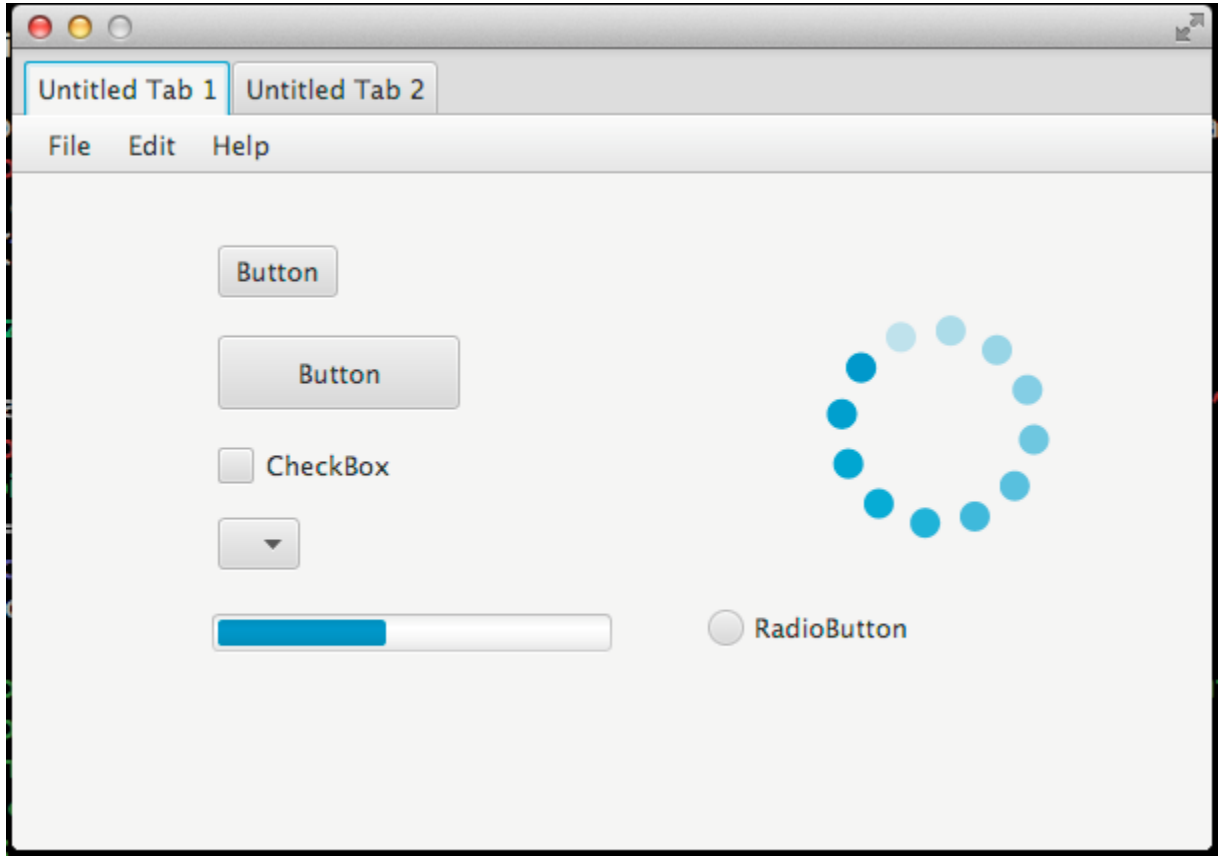


Image from https://stackoverflow.com/q/23862779

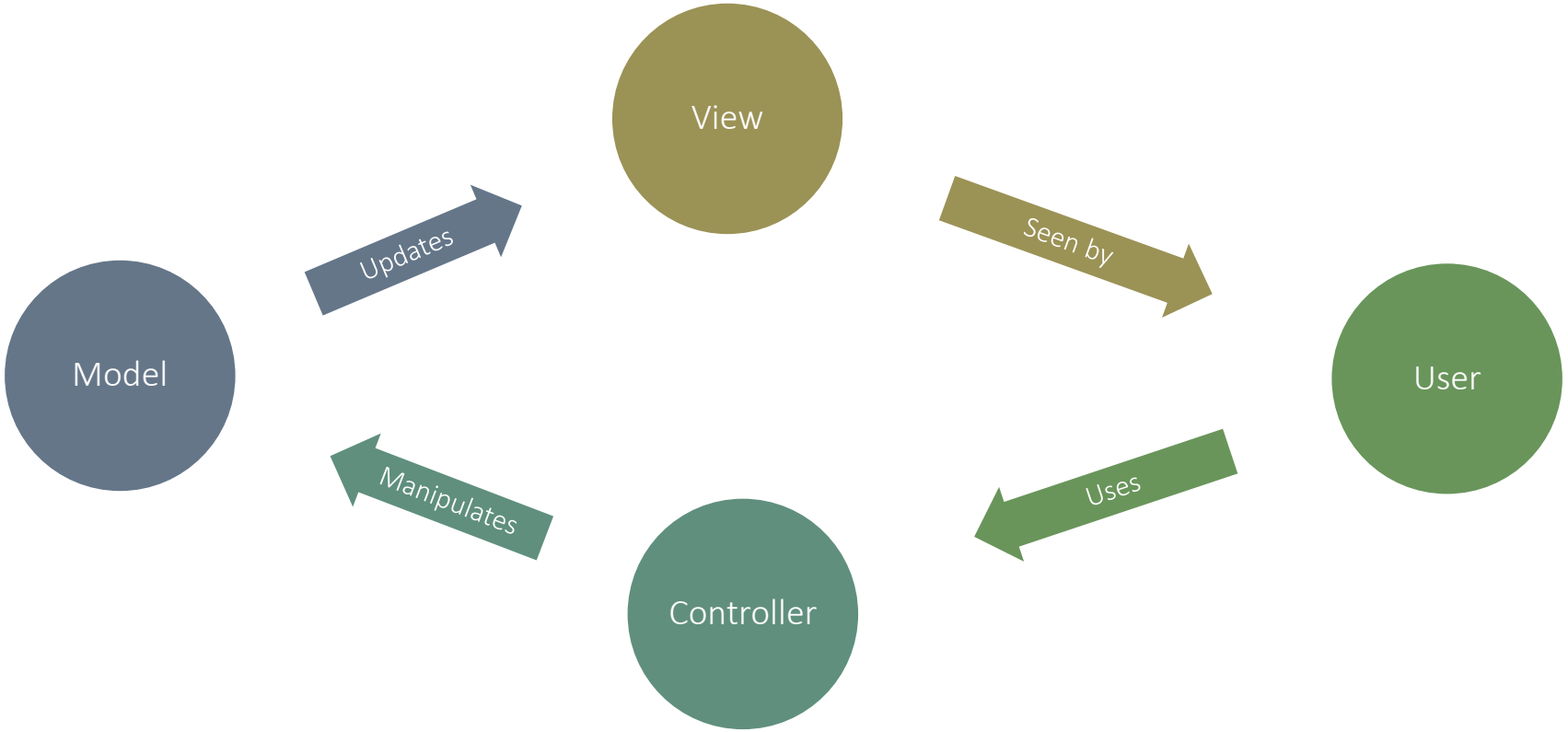
Java GUI toolkits: AWT, Swing, JavaFX

AWT ("Abstract Windowing Toolkit") – since the very beginning (January 1996)
- First Java GUI toolkit: set of "heavyweight" classes using native GUI widgets
- Still included in Java but not widely used

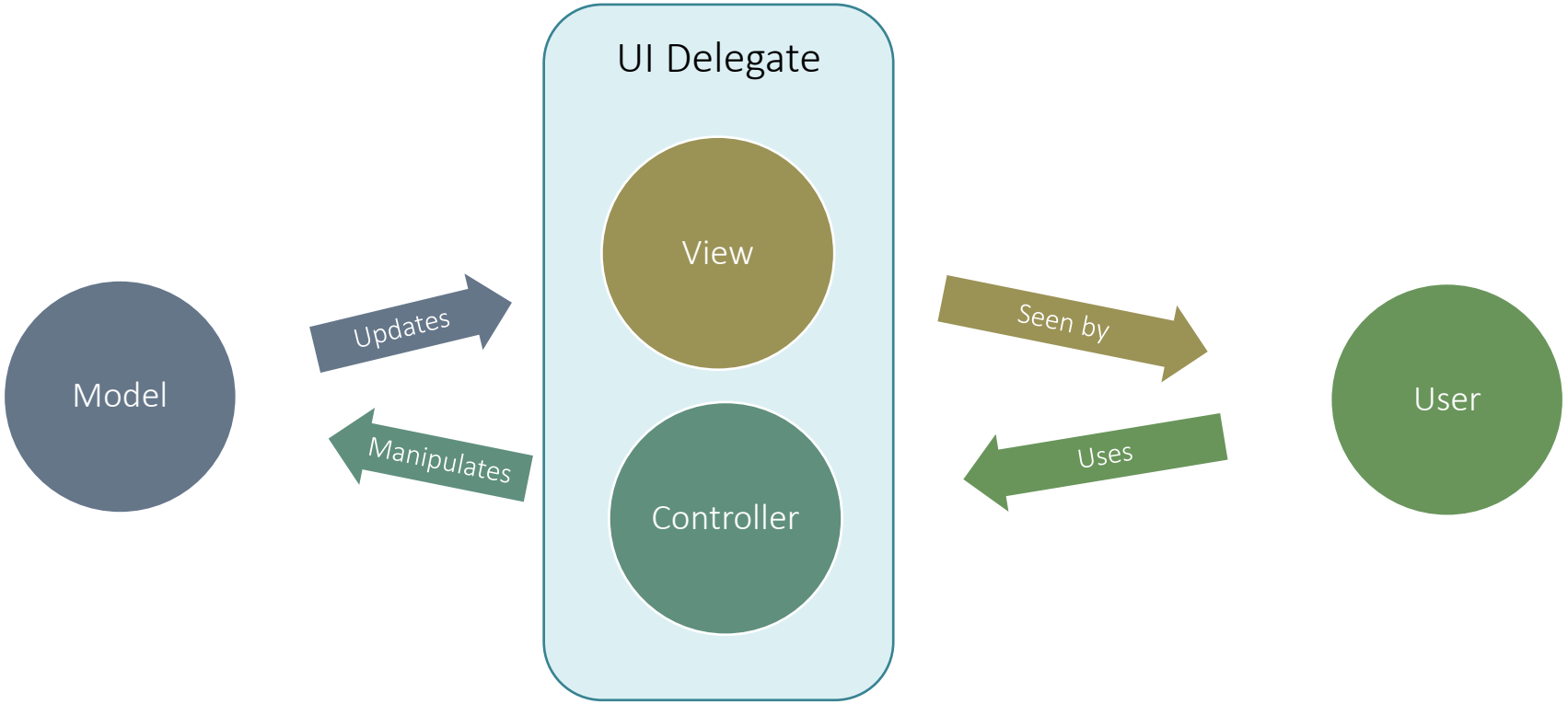
Swing – since Java 2 (December 1998)
- Cross-platform "lightweight" GUI components written entirely in Java
- More powerful and flexible components than AWT
- Still included in Java, widely used

JavaFX
- Introduced in 2007, bundled with Java as of Java 8 (March 2014)
- Un-bundled again as of Java 11; now separate open-source project at https://openjfx.io/
- Somewhat widely used – seen as more flexible than Swing, but never got traction, and now HTML5 appears to be taking over

Classic Model-View-Controller (MVC)



“Modified MVC” in Swing



More details at http://www.oracle.com/technetwork/articles/javase/index-142890.html

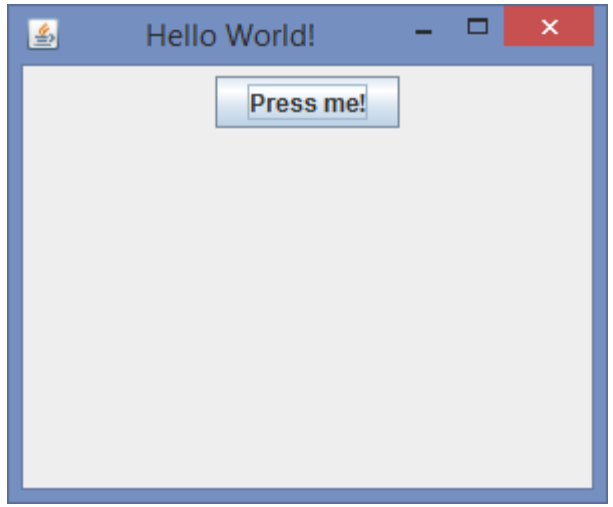
First Swing program

```
public class HelloWorld {
    private static void createAndShowGUI() {
        JFrame frame = new JFrame("Hello World!");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());
        frame.setSize(300, 250);

        JButton button = new JButton("Press me!");
        frame.getContentPane().add(button);

        frame.setVisible(true);
    }

    public static void main(String[] args) {
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```



javax.swing.JFrame

Represents a top level window

Relevant operations

- `setSize(width, height)` – sets default dimensions
- `setLocationByPlatform(boolean)` – if true, lets the OS decide where to place the window
- `setVisible(boolean)` – shows/hides the window
- `setDefaultCloseOperation()` – what should the program do when the window closes
  - *EXIT_ON_CLOSE – entire program exits when window closes*

Adding components

`JFrame` is a **container** – it can hold other components inside it

To access the main container in a `JFrame`, use `getContentPane()`

(Just adding directly to the `JFrame` will often also work, but this is the official method)

Then we added a **`JButton`** – a button that can be pressed

`JButton` constructor sets the button label

Not discussed today: layout managers
https://docs.oracle.com/javase/tutorial/uiswing/layout/using.html

Other useful Swing components

JLabel: a textual label

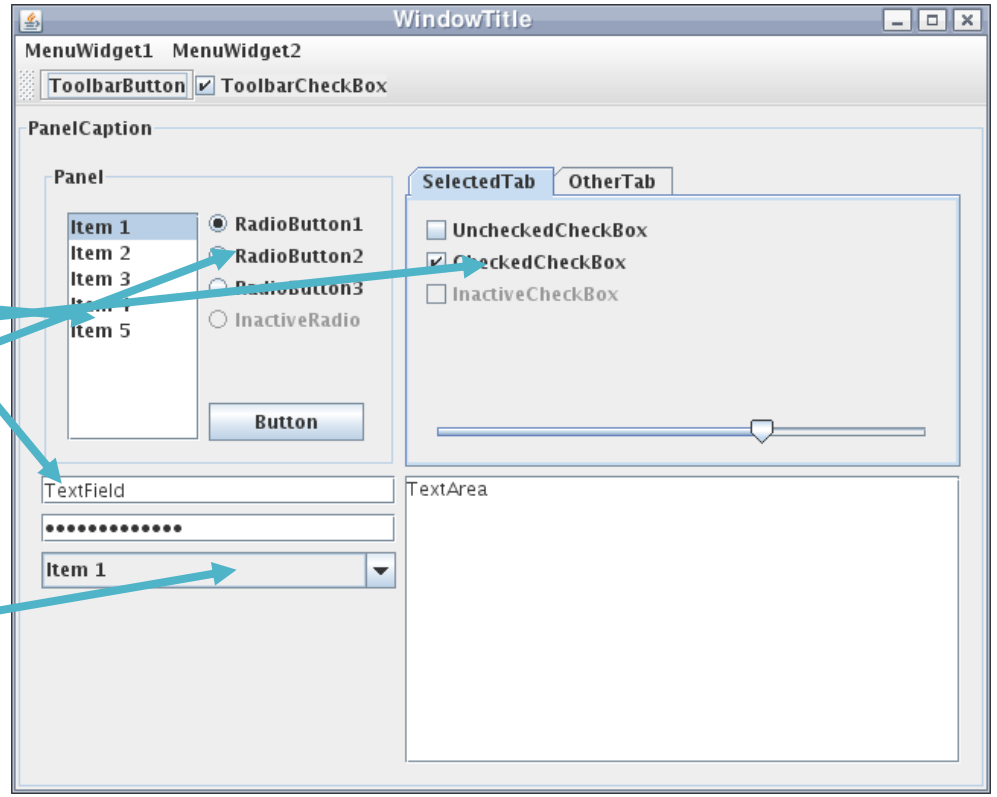
JTextField: a field for entering text

JList: a list box

JCheckbox: a box that can be checked or not

JRadioButton: one of a set of option buttons

JComboBox: a drop-down list



JList in Modified MVC

JList is the UI delegate (i.e., it is the view and the controller)

It has an associated **ListModel** which provides the model (usually you can use a **DefaultListModel**)

ListModel is almost the same as **java.util.List** – methods include **addElement()**, **remove()**, **insertElementAt()**, **getSize()**

When the ListModel is changed, the information displayed in the Jlist is changed as well

Similar link between JTable and TableModel

More on Swing Components

Parent class of all components (except top-level windows like `JFrame`, `JDialog`, `JWindow`): `javax.swing.JComponent`

All have a `setEnabled(boolean)` method

- When true: component is active and can be used
- When false: component is “greyed out”

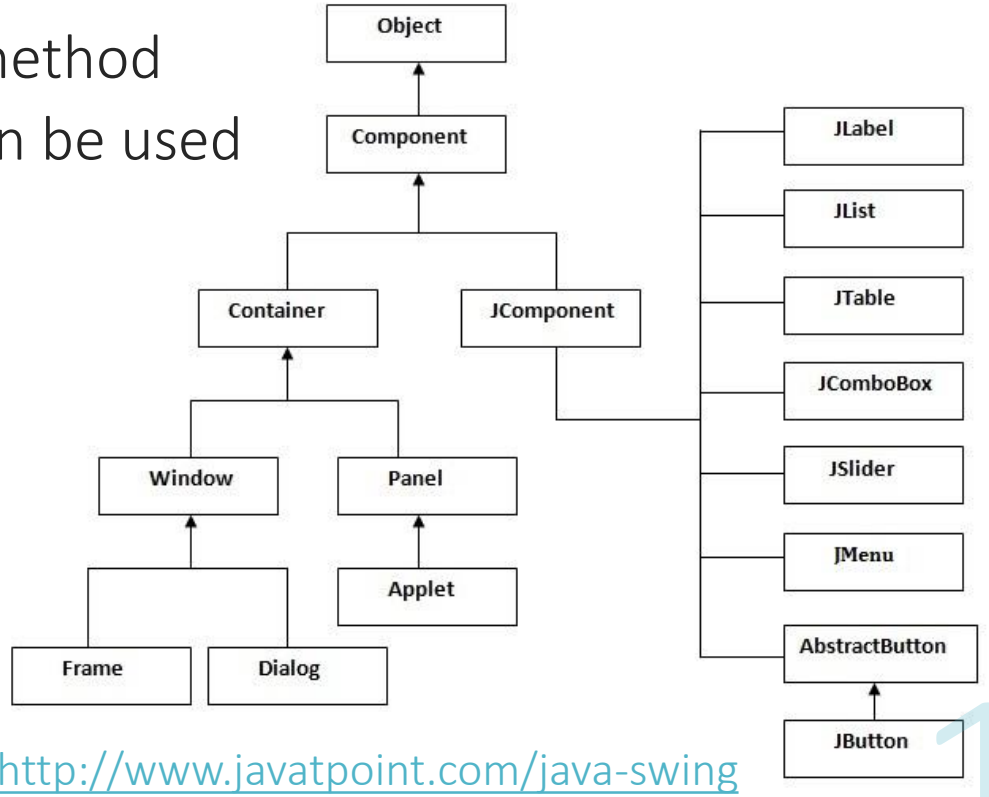


Image from http://www.javatpoint.com/java-swing