CISC 275: Introduction to Software Engineering

Lab 3:
Creating GUIs with Java Swing

Charlie Greenbacker
University of Delaware
Fall 2011
Overview

• Java Swing at a glance
• Simple “Hello World” example
• MVC review
• Intermediate example
• Lab exercise
Java Swing at a glance

- Java toolkit for graphical user interfaces (GUIs)
- Provides native “look & feel” based on host OS
  - More sophisticated than earlier AWT toolkit
  - Also supports customized “look & feel”
- Lightweight; doesn't use host OS GUI API
- Makes for natural use of MVC pattern
“Hello World” example

• We'll create a small window with just a single label containing a “Hello World” message

• 3 easy steps to get a GUI window on screen:
  • Set up the window
  • Add a label
  • Show the window

• All of the following code lives inside a simple class with just a main() method
“Hello World” example

• Set up the window:
  • JFrame acts as window component
  • Assign action to close button/operation

```java
JFrame frame = new JFrame();
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```
“Hello World” example

• Add a label:
  • JLabel contains our message
  • Add label to frame's content pane

```java
JLabel label = new JLabel("Hello World!");
frame.getContentPane().add(label);
```
“Hello World” example

- Show the window:
- Size frame to fit layout of components (pack)
- Make frame visible

```java
frame.pack();
frame.setVisible(true);
```
“Hello World” example

• Run the program & you should see a window like this (according to your OS of choice):

• Example code is available for review at: HelloWorldGUI.java
MVC review

- Model-View-Controller design pattern
- Isolate data model from user interface from application logic
  - Model: data model classes
  - View: user interface (i.e. GUI, console, etc.)
  - Controller: interacts w/ View, manipulates Model
- Next example will demonstrate elements of MVC
Intermediate example

- Uses part of MVC... the VC part, that is
  - Really no “model” to speak of
- View is a window that extends JFrame class
- Code looks a bit different than HelloWorldGUI
  - Self-referencing; manipulates self (an extension of JFrame) rather than a JFrame it created
- Controller launches GUI & performs conversion
- Example program is a widget for converting from Celsius to Fahrenheit temperatures
Intermediate example

- Begin by defining the class & attributes

```java
public class CelsiusConverterGUI extends JFrame {

    // variables used throughout class
    private static JLabel celsiusLabel;
    private static JButton convertButton;
    private static JLabel fahrenheitLabel;
    private static JTextField inputTextField;
```
Intermediate example

- Initialize GUI window by adding components
- Start by setting up the window

```java
private void initComponents() {
    // set up the window
    setTitle("Celsius Converter");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setPreferredSize(new Dimension(250, 100));
}
```
Intermediate example

- Initialize GUI window by adding components
- Next, set up the individual components

```java
inputTextField = new JTextField(10);
celsiusLabel = new JLabel("Celsius");
convertButton = new JButton("Convert");
fahrenheitLabel = new JLabel("Fahrenheit");
```
Intermediate example

• Initialize GUI window by adding components
• Then, initialize window layout & add components

```java
setLayout(new FlowLayout());
getContentPane().add(inputTextField);
getContentPane().add(celsiusLabel);
getContentPane().add(convertButton);
getContentPane().add(fahrenheitLabel);
```
Intermediate example

- Initialize GUI window by adding components
- Finally, create & assign action listener for button

```java
convertButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        convertButtonActionPerformed(evt);
    }
});
```

// end of initComponents() method
Intermediate example

• Create & display Celsius conversion window

```java
public CelsiusConverterGUI() {
    initComponents(); // the method we just made

    // show the window
    pack();

    setVisible(true);
}
```
Intermediate example

- Implement event handler for button action listener

```java
private void convertButtonActionPerformed(ActionEvent evt) {
    // parse Celsius value as Double, convert to
    // Fahrenheit, cast as int

    double tempFahr =
        CelsiusController.celsiusToFahrenheit(
            Double.parseDouble(inputTextField.getText()));

    // change text of Fahrenheit label to reflect
    // converted value

    fahrenheitLabel.setText(tempFahr + " Fahrenheit");
}
} // end of CelsiusConverterGUI class
```
Intermediate example

• Controller class is extremely simple

```java
public class CelsiusController {
    public static void main(String[] args) {
        new CelsiusConverterGUI();
    }

    public static double celsiusToFahrenheit(double celsius) {
        return celsius * 1.8 + 32;
    }
}
```
Intermediate example

- Run the program & you should see a window like this (according to your OS of choice):

  ![Celsius Converter](image)

- Example code is available for review at: [CelsiusConverterGUI.java](#) & [CelsiusController.java](#)
Lab Exercise

• On your own or in pairs, add new functionality to the Celsius Converter
  • Add a drop-down list offering multiple conversion options (e.g., meters to inches, kgs to lbs) in GUI
  • Add new conversion methods to controller
  • Program should make appropriate conversion based on option selected in drop-down list
• Email your code (GUI & controller files) to charlieg@cis.udel.edu by Tuesday
Lab Exercise

- Remember: you must follow MVC pattern
- No conversion code in GUI class
- No GUI code in controller class
- You will need to use some Java libraries and Swing features not covered in these lab slides
- Consult the skeleton code for ideas
- The Swing Tutorial from Sun is very helpful
- Google is your friend, too