Ch11: Swing overview

12 March 2007
CMPT167
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Review last time

- Polymorphism
  - Dynamic method binding
  - final keyword for classes and methods
  - Abstract and concrete classes
    - abstract keyword for classes and methods
- Interfaces
  - vs. abstract superclasses
- Type-wrapper classes for the primitive types

Quiz 3: 10 minutes

- Explain why main() has to be static [3]
- What does the final modifier mean for [6]
  - (a) variables, (b) methods, and (c) classes?
- Contrast abstract superclasses with interfaces [5]
- Given:

```
public class Ferrari extends Car {}
Car sentra = new Car();
Ferrari f430 = new Ferrari();
```

  which of the following are legal, and why? [6]

  (a) sentra = f430 (b) f430 = (Ferrari) sentra

  (c) sentra = f430; f430 = (Ferrari) sentra

Quiz 3: answers #1-2

- Explain why main() has to be static
  - The current class has not been instantiated yet; main() must be a class method accessible by VM
- What does the final modifier mean for
  - Variables:
    - Constant: can't change value
  - Methods:
    - Subclasses cannot override
  - Classes:
    - Cannot subclass

Quiz 3: answers #3-4

- Contrast abstract superclasses with interfaces
  - Superclass: identity; inherit variables/methods
    - No multiple inheritance in Java
  - Interface: capability; multiple interfaces okay
- sentra = f430
  - Okay: f430 is also a Car
- f430 = (Ferrari) sentra
  - Not okay: sentra can never be a Ferrari
- sentra = f430; f430 = (Ferrari) sentra
  - Okay: sentra refers to a Ferrari (downcast)

What's on for today

- More on JOptionPane
- Swing vs. AWT, lightweight vs. heavyweight
- Superclass structure of Swing
- Nested and inner classes
- Event handling
  - Delegate classes

JOptionPane

- import javax.swing.JOptionPane;
- showInputDialog(String prompt)
  - Prompt to the user, returns a string
- showMessageDialog(pos, msg, title, type)
  - Show dialog box to user
  - pos: null for centered in screen
    - Or pass a reference to widget
  - pype: JOptionPane.INFORMATION_MESSAGE
    - Or ERROR_MESSAGE, WARNING_MESSAGE, QUESTION_MESSAGE, PLAIN_MESSAGE

Swing vs. AWT, light vs. heavy

- A Java app can mix Swing and AWT features
- Swing is written in Java and is more portable
  - AWT relies on local platform's windowing system: varies across platforms
- Lightweight: not tied to local platform
- Heavyweight: depends on local platform
  - AWT widgets are heavyweight
  - Most Swing widgets are lightweight

Common superclasses of Swing

- Everything is an Object
- Component (java.awt): GUI, both Swing and AWT
- Container (java.awt): organizes Components
- JComponent (javax.swing):
  - Superclass of all lightweight Swing components
  - Pluggable look-and-feel, shortcut keys, tooltips, localization, etc.
- JLabel, JTextField, JButton, JCheckBox, JComboBox, JList, JPanel, etc.

Nested classes

- We've seen non-public helper classes defined in the same file as the primary public class:
  - public class Primary { ... }
  - class Helper1 { ... }
- We can also define classes nested in another:
  - public class Primary {
    - class Helper1 { ... }
  - }
- Inner classes: non-static nested classes
  - Can access even private items of top-level
  - Often used for event handlers

Event handling

- We've seen examples like this:

```
public class Histogram extends JPanel implements
  ActionListener {
    public Histogram() { ...
        widget.addActionListener( this ); ... };      // register
    public void actionPerformed() { ... };
    public static void createAndShowGUI() { ... };
    public static void main() { ... };
}
```

- One class does three functions:
  - main()/createAndShowGUI(): setup window
  - Constructor: create, layout widgets
  - actionPerformed(): event handler

Delegate classes

- Alternatively: use separate classes

```
public class Histogram extends JPanel {
    public Histogram() { ...
        InputHandler handler = new InputHandler();
        widget.addActionListener( handler ); ... };
    private class InputHandler implements ActionListener {
        public void actionPerformed() { ... };
    }
}
public class HistogramTest {  // in separate file
    public static void createAndShowGUI() { ... };
    public static void main() { ... };
}
```

- Uses inner class to define event handler

Summary of today

- More on JOptionPane
- Swing vs. AWT, lightweight vs. heavyweight
- Superclass structure of Swing
- Nested and inner classes
- Event handling
  - Delegate classes

TODO

- Lab4 due this Wed 14Mar
  - OO concepts (sets and vectors)