

§17.3: Swing Widget Layout

8 Feb 2008

CMPT166

Dr. Sean Ho

Trinity Western University

Quiz 2

- In Java, what does **final** mean when applied to:
 - (a) **attributes**, (b) **methods**, (c) **classes**?
- Name and give a short description of each of the **eight** primitive Java **types**
- Write a short bit of Java code encoding these **relationships**:
 - “fido is a **Dog**.”
 - “A **Dog** is a **kind** of **Mammal**.”
 - “Every **Mammal** has a **Heart**.”
 - “Any **Dog** can bark().”

Quiz 2: answers #1

- In Java, what does **final** mean when applied to:
 - **Attributes**: constant (read-only)
 - **Methods**: subclasses cannot override
 - **Classes**: cannot be subclassed

Quiz 2: answers #2

- Name and give a short description of each of the **eight** primitive Java **types**
 - **boolean** (1 byte): true/false
 - **char** (2): Unicode character
 - **byte** (1), **short** (2), **int** (4), **long** (8): integers
 - **float** (4), **double** (8): real numbers

Quiz 2: answers #3

- Write a short bit of Java code encoding these relationships:
 - ◆ “fido is a Dog.”
 - ◆ “A Dog is a kind of Mammal.”
 - ◆ “Every Mammal has a heart.”
 - ◆ “Any Dog can bark().”

```
class Mammal { Heart h; }  
class Dog extends Mammal { void bark(); }  
Dog fido = new Dog();
```

Model-View-Controller

- **Design patterns:** reusable, generic concepts to help you design your programs
- **MVC** design pattern:
 - **Model:** stores data
 - ◆ Computation, methods to transform data
 - ◆ Data structure issues: arrays? Linked-lists? Classes?
 - **View:** display / output / read
 - ◆ println()? Swing? Web? JTextField?
 - **Controller:** manipulate / input / write
 - ◆ Command-line? Buttons? Mouse?

MVC in Swing

- Model:
 - Core **content**/functionality of program
 - Ideally, should be **independent** of Swing
- View:
 - **JFrame**, **JPanel**, **layout manager**, widgets
- Controller: Event handler:
 - implements **ActionListener**, **ItemListener** {
 - ◆ public void **actionPerformed**(**ActionEvent** e)
 - ◆ public void **itemStateChanged**(**ItemEvent** e)

Swing container classes

- Containers (`java.awt.Container`) hold other components
 - Swing container: `javax.swing.JComponent`
 - Both `JFrame` and `JPanel` are Swing containers
 - Every `JComponent` can have one layout manager: decides how to arrange widgets
- `JFrame`: Swing window
 - Can only have one layout manager
 - But can nest `JPanels`: each `JPanel` can have its own layout manager

Layout Managers

- **FlowLayout**: simple **left-to-right**
 - ◆ `setLayout(new FlowLayout());`
 - ◆ `JLabel label1 = new JLabel("Label One");`
 - ◆ `add(label1);`
- **BorderLayout**: **north, south, east, west, center**
 - ◆ `setLayout(new BorderLayout());`
 - ◆ `add(label1, BorderLayout.NORTH);`
- **GridLayout**: table of **equal-size** cells
 - ◆ `setLayout(new GridLayout(2, 3));`
 - ◆ `add(label1);`
 - Fills **left-to-right**, then **top-to-bottom**