Software Development (cs2500)

Lecture 37: Layout Managers

M. R. C. van Dongen

15 February, 2014

Objectives

- We start by studying GUI *design*.
- We continue by studying *layout managers*.
- Based on [Lewis, and Loftus 2009, Chapter 6.10–6.12].

GUI Design

Know the user

- To the user the GUI *is* the program.
- It should serve the user well.

Prevent errors

- Choose components that prevent invalid input and inappropriate actions.

Optimise usability

- Not all users are the same.
  - Consider keyboard shortcuts.
  - Consider beginner and expert level menus.

Be consistent

- Different looks & feels may confuse the user.

Layout Managers

- *A layout manager* controls container component arrangement:
    - Size of of the components.
    - Position of the components.
    - Visibility of the components.
- Consulted when change to visual appearance may be needed.
    - This includes resizing the container and adding components.

Changing a Container's Layout Manager

- □ Every `Container` has a default layout manager.
- □ It is possible to replace the default layout manager.

Java

```
final JPanel panel = new JPanel( );
panel.setLayout( new ⟨Layout Manager⟩ );
```

Existing Layout Managers

FlowLayout Organises components from left to right.

- Creates new rows as necessary.

BorderLayout Organise components into five areas:

- North;
- East;
- South;
- West; and
- Centre.

GridLayout Organises components in grid of rows and columns.

BoxLayout Organises components in single row or column.

Tabbed Panes

- A *tabbed pane* is a container with one or more components.
- Only one component can be active (visible) at a time.
- Each component has a *tab*.
- All tabs are clickable.
- Clicking a tab activates its component.

Java

```
⟨tabbed pane⟩.addTab( ⟨label⟩, ⟨component⟩ );
```

Example

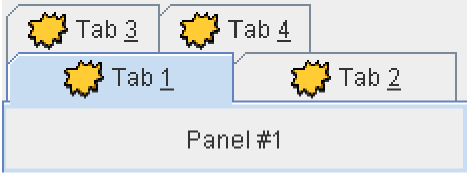

Figure : Example of a tabbed pane.

Tabbed Panes

Java

```
import java.swing.*;
public class LayoutDemo {
    public static void main( String[] args ) {
        final JFrame = new JFrame( "Demo" );
        ...
        final JTabbedPane tp = new JTabbedPane( );
        tp.addTab( "Intro",  new IntroPanel( ) );
        tp.addTab( "Flow",   new FlowPanel( ) );
        tp.addTab( "Border", new BorderPanel( ) );
        tp.addTab( "Grid",   new GridPanel( ) );
        tp.addTab( "Box",    new BoxPanel( ) );

        frame.getContentPane( ).add( tp );
        ...
    }
}
```

The Flow Layout Manager

Java

```
import java.awt.*;
import javax.swing.*;

public class FlowPanel extends JPanel {
    public FlowPanel( ) {
        setLayout( new FlowLayout( ) );
        setBackground( Color.green );

        final JButton bl = new JButton( "Button l" );
        ...

        add( bl );
        ...
    }
}
```

The Border Layout Manager

Java

```
public class BorderPanel extends JPanel {
    public BorderPanel( ) {
        setLayout( new BorderLayout( ) );
        setBackground( Color.green );

        final JButton b1 = new JButton( "Button 1" );
        ...

        add( b1, BorderLayout.CENTER );
        add( b2, BorderLayout.PAGE_START );
        add( b3, BorderLayout.PAGE_END );
        add( b4, BorderLayout.LINE_END );
        add( b5, BorderLayout.LINE_START );
    }
}
```

The Grid Layout Manager

Java

```
public class GridPanel extends JPanel {
    public GridPanel( ) {
        setLayout( new GridLayout( 2, 3 ) );
        setBackground( Color.green );

        final JButton b1 = new JButton( "Button 1" );
        ...

        add( b1 );
        ...
    }
}
```

The Box Layout Manager

Constructor Requires Component that Needs to be Laid Out

Java

```
public class BoxPanel extends JPanel {
    public BoxPanel( ) {
        final BoxLayout manager = new BoxLayout( this, BoxLayout.Y_AXIS );
        setLayout( manager );
        setBackground( Color.green );

        final JButton b1 = new JButton( "Button 1" );
        ...

        ⟨add the buttons⟩
    }
}
```

The Box Layout Manager

Java

```
add( b1 );
// add rigid dummy box of given dimension.
add( Box.createRigidArea( new Dimension( 0, 10 ) ) );
add( b2 );
// add dummy box with stretchable dimension.
add( Box.createVerticalGlue( ) );
add( b3 );
// add two dummy boxes with stretchable dimension.
add( Box.createVerticalGlue( ) );
add( Box.createVerticalGlue( ) );
add( b4 );
add( Box.createRigidArea( new Dimension( 0, 20 ) ) );
add( b5 );
```

For Friday

- Study the lecture notes.

Acknowledgements

- This lecture is based on:
  - [Lewis, and Loftus 2009, Chapter 6.10–6.12];
  - The online Java Tutorials.
- Some of the pictures are borrowed from the tutorials.

About this Document

- This document was created with `pdflatex`.
- The LaTeX document class is `beamer`.