2 Graphical User Interfaces (cont.)

BK Chap. 11.

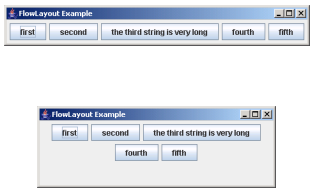
Overview

- GUI layout
- Nested containers
- Borders
- Spacing
- Dialogues
- ImageViewer ver. 2.0, 3.0, 3.05, 3.1

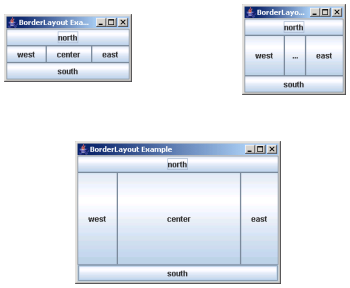
Layout managers

- Manage limited space for competing components.
  - `FlowLayout`, `BorderLayout`, `GridLayout`, `BoxLayout`, `GridBagLayout`.
- Manage `Container` objects, e.g. a content pane.
- Each imposes its own style.

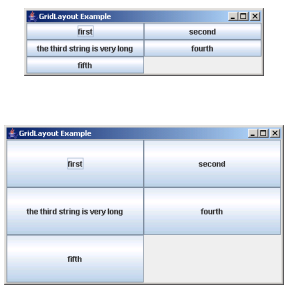
FlowLayout



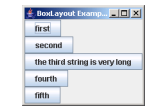
BorderLayout

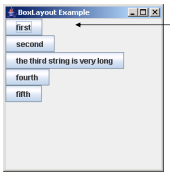


GridLayout



BoxLayout





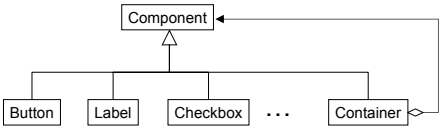
Note: no component resizing.

Nested containers

- Sophisticated layouts can be obtained by nesting containers.
  - Use `JPanel` as a basic container.
- Each container will have its own layout manager.

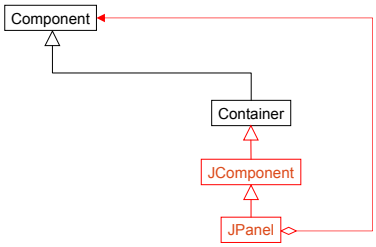
AWT Components and Containers

- Components have graphical representation
- Containers have components
- Containers are components



Swing Components and Containers

- Use JPanel as container for Swing components



Buttons and nested layouts

A GridLayout inside a FlowLayout inside a BorderLayout.

Borders

- Used to add decoration around components.
- Defined in `javax.swing.border`
  - `BevelBorder`, `CompoundBorder`, `EmptyBorder`, `EtchedBorder`, `TitledBorder`.

Adding spacing

```
JPanel contentPane = (JPanel)frame.getContentPane();
contentPane.setBorder(new EmptyBorder(6, 6, 6, 6));

// Specify the layout manager with nice spacing
contentPane.setLayout(new BorderLayout(6, 6));

imagePanel = new ImagePanel();
imagePanel.setBorder(new EtchedBorder());
contentPane.add(imagePanel, BorderLayout.CENTER);
```


Struts and Glue

- Invisible components used as spacing.
- Available from the `Box` class.
- Strut: fixed size.
  - `Component createHorizontalStrut(int width)`
  - `Component createVerticalStrut(int height)`
- Glue: fills available space.
  - `Component createHorizontalGlue()`
  - `Component createVerticalGlue()`

Other components

- Slider
- Spinner
- Tabbed pane
- Scroll pane

Dialogs

- Modal dialogs block all other interaction.
  - Forces a response from the user.
- Non-modal dialogs allow other interaction.
  - This is sometimes desirable.
  - May be difficult to avoid inconsistencies.

JOptionPane standard dialogs

- Message dialog
  - Message text plus an OK button.
- Confirm dialog
  - Yes, No, Cancel options.
- Input dialog
  - Message text and an input field.
- Variations are possible.

A message dialog

```
private void showAbout()
{
    JOptionPane.showMessageDialog(frame,
                "ImageViewer\n" + VERSION,
                "About ImageViewer",
                JOptionPane.INFORMATION_MESSAGE);
}
```

Image filters

- Functions applied to the whole image.

```
int height = getHeight();
int width = getWidth();
for(int y = 0; y < height; y++) {
    for(int x = 0; x < width; x++) {
        Color pixel = getPixel(x, y);
        alter the pixel's color value;
        setPixel(x, y, pixel);
    }
}
```


Adding further filters

```
private void makeLighter()
{
    if(currentImage != null) {
        currentImage.lighter();
        frame.repaint();
        showStatus("Applied: lighter");
    }
    else {
        showStatus("No image loaded.");
    }
}
```

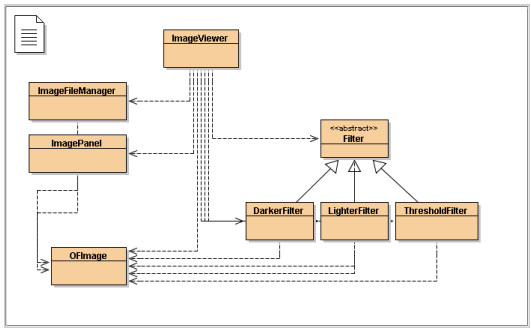
```
private void threshold()
{
    if(currentImage != null) {
        currentImage.threshold();
        frame.repaint();
        showStatus("Applied: threshold");
    }
    else {
        showStatus("No image loaded.");
    }
}
```

Code duplication? Refactor!

Adding further filters

- Define a `Filter` superclass (abstract).
- Create function-specific subclasses.
- Create a collection of subclass instances in `ImageViewer`.
- Define a generic `applyFilter` method.
- See *imageviewer2-0*.

imageviewer2-0



Review

- Aim for cohesive application structures.
  - Endeavor to keep GUI elements separate from application functionality.
- Pre-defined components simplify creation of sophisticated GUIs.
- Layout managers handle component juxtaposition.
  - Nest containers for further control.

Review

- Many components recognize user interactions with them.
- Reactive components deliver events to listeners.
- Anonymous inner classes are commonly used to implement listeners.