1495 UNIVERSITY OF ABERDEEN

JC2002 Java Programming

Lecture 20: Using top-level containers in Swing

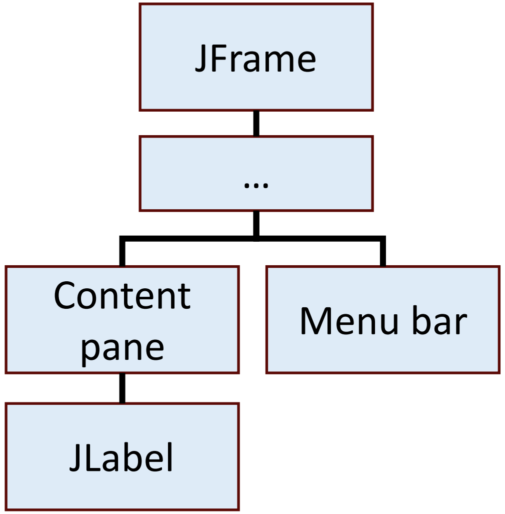
Top-level container classes

- Swing provides two generally useful top-level container classes: `JFrame` and `JDialog`
  - Each program that uses Swing components has at least one top-level container that is the root of a *containment hierarchy*
  - To appear onscreen, every GUI component must be part of a containment hierarchy
  - Each GUI component can be contained only once; if a component is already in a container and you try to add it to another container, the component will be removed from the first container and then added to the second

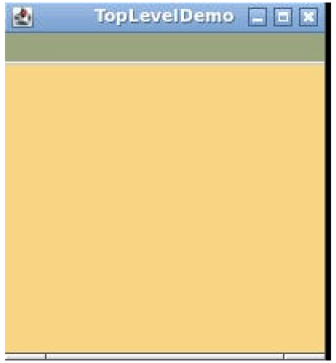
Content pane of top-level container classes

- Each top-level container has a *content pane* that contains (directly or indirectly) the visible components in that top-level container's GUI
- You can optionally add a *menu bar* to a top-level container
  - The menu bar is by convention positioned within the top-level container, but outside the content pane
  - Some look-and-feels, such as the Mac OS, give you the option of placing the menu bar in another place more appropriate for the look-and-feel, such as at the top of the screen

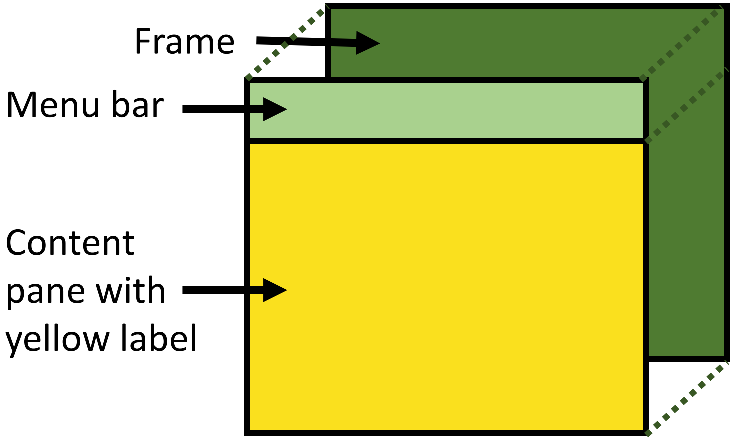
JFrame top-level container



Containment hierarchy



Window on screen



Container structure

Example of top-level container (1)

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

public class TopLevelDemo {
  private static void createAndShowGUI() {
    JFrame frame = new JFrame("TopLevelDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JMenuBar greenMenuBar = new JMenuBar();
    greenMenuBar.setOpaque(true);
    greenMenuBar.setBackground(new Color(154, 165, 127));
    greenMenuBar.setPreferredSize(new Dimension(200, 20));
    JLabel yellowLabel = new JLabel();
    yellowLabel.setOpaque(true);
    yellowLabel.setBackground(new Color(248, 213, 131));
    yellowLabel.setPreferredSize(new Dimension(200, 180));
    frame.setJMenuBar(greenMenuBar);
    frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
    frame.pack();
    frame.setVisible(true);
  }
```

```java
  public static void main(String[] args) {
    javax.swing.SwingUtilities.invokeLater(new Runnable() {
      public void run() { createAndShowGUI(); }
    });
  }
}
```

This part is standard, and we will not show it in the following slides

Example of top-level container (2)

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

public class TopLevelDemo {
  private static void createAndShowGUI() {
    JFrame frame = new JFrame("TopLevelDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JMenuBar greenMenuBar = new JMenuBar();
    greenMenuBar.setOpaque(true);
    greenMenuBar.setBackground(new Color(154, 165, 127));
    greenMenuBar.setPreferredSize(new Dimension(200, 20));
    JLabel yellowLabel = new JLabel();
    yellowLabel.setOpaque(true);
    yellowLabel.setBackground(new Color(248, 213, 131));
    yellowLabel.setPreferredSize(new Dimension(200, 180));
    frame.setJMenuBar(greenMenuBar);
    frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
    frame.pack();
    frame.setVisible(true);
  }
```

Essential imports

Create and set up the window (JFrame)

Example of top-level container (3)

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

public class TopLevelDemo {
  private static void createAndShowGUI() {
    JFrame frame = new JFrame("TopLevelDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JMenuBar greenMenuBar = new JMenuBar();
    greenMenuBar.setOpaque(true);
    greenMenuBar.setBackground(new Color(154, 165, 127));
    greenMenuBar.setPreferredSize(new Dimension(200, 20));
    JLabel yellowLabel = new JLabel();
    yellowLabel.setOpaque(true);
    yellowLabel.setBackground(new Color(248, 213, 131));
    yellowLabel.setPreferredSize(new Dimension(200, 180));
    frame.setJMenuBar(greenMenuBar);
    frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
    frame.pack();
    frame.setVisible(true);
  }
```

Create a menu bar with size (200,20) and green background

Example of top-level container (4)

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

public class TopLevelDemo {
  private static void createAndShowGUI() {
    JFrame frame = new JFrame("TopLevelDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JMenuBar greenMenuBar = new JMenuBar();
    greenMenuBar.setOpaque(true);
    greenMenuBar.setBackground(new Color(154, 165, 127));
    greenMenuBar.setPreferredSize(new Dimension(200, 20));
    JLabel yellowLabel = new JLabel();
    yellowLabel.setOpaque(true);
    yellowLabel.setBackground(new Color(248, 213, 131));
    yellowLabel.setPreferredSize(new Dimension(200, 180));
    frame.setJMenuBar(greenMenuBar);
    frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
    frame.pack();
    frame.setVisible(true);
  }
```

Create a label with yellow background

Example of top-level container (5)

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

public class TopLevelDemo {
  private static void createAndShowGUI() {
    JFrame frame = new JFrame("TopLevelDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JMenuBar greenMenuBar = new JMenuBar();
    greenMenuBar.setOpaque(true);
    greenMenuBar.setBackground(new Color(154, 165, 127));
    greenMenuBar.setPreferredSize(new Dimension(200, 20));
    JLabel yellowLabel = new JLabel();
    yellowLabel.setOpaque(true);
    yellowLabel.setBackground(new Color(248, 213, 131));
    yellowLabel.setPreferredSize(new Dimension(200, 180));
    frame.setJMenuBar(greenMenuBar);
    frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
    frame.pack();
    frame.setVisible(true);
  }
```

Add the menu bar and the label to the frame

Example of top-level container (6)

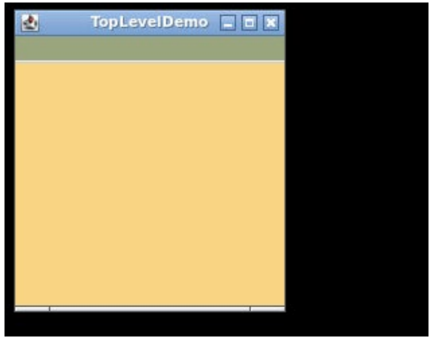
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class TopLevelDemo {
  private static void createAndShowGUI() {
    JFrame frame = new JFrame("TopLevelDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JMenuBar greenMenuBar = new JMenuBar();
    greenMenuBar.setOpaque(true);
    greenMenuBar.setBackground(new Color(154, 165, 127));
    greenMenuBar.setPreferredSize(new Dimension(200, 20));
    JLabel yellowLabel = new JLabel();
    yellowLabel.setOpaque(true);
    yellowLabel.setBackground(new Color(248, 213, 131));
    yellowLabel.setPreferredSize(new Dimension(200, 180));
    frame.setJMenuBar(greenMenuBar);
    frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
    frame.pack();
    frame.setVisible(true);
  }
```

Console:

```
$ java TopLevelDemo
█
```

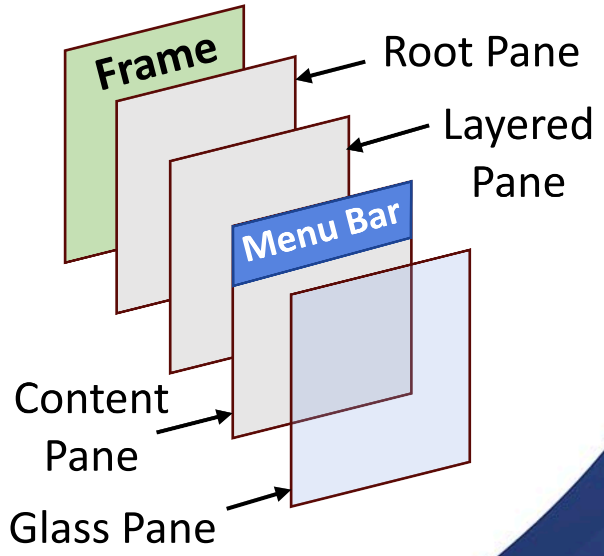
Virtual Desktop:



The root pane

- Each top-level container relies on a reclusive intermediate container called the *root pane*
  - The root pane manages the content pane and the menu bar, along with a couple of other containers, such as content and glass pane
  - The layered pane contains the menu bar and content pane
  - The glass pane is often used to intercept input events over the top-level container, and it can also be used to paint over other components

A list of the components that a root pane provides to a frame

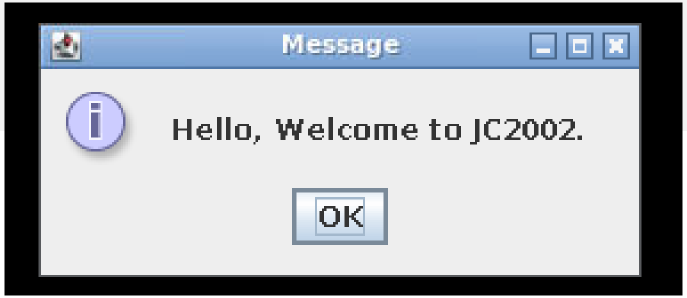


Dialogs

- A *dialog* is an independent sub window (or a *pop-up window*) carrying notifications apart from the main application window
  - Most dialogs present an error message or warning to a user, but dialogs can also present images, directory trees, etc.
  - A dialog is *modal* if it blocks user input to all the other windows in the program until it is closed
- In Swing, class **`JDialog`** is used to instantiate top-level containers for dialogs
  - Class **`JOptionPane`** provides simple standard modal dialog boxes, but to create a *non-modal* dialog, you must use `JDialog` class directly

Simple example dialog via JOptionPane

```java
import javax.swing.*;
public class OptionPaneExample {
  JFrame f;
  OptionPaneExample(){
    f=new JFrame();
    JOptionPane.showMessageDialog(f,"Hello, Welcome to JC2002.");
  }
  public static void main(String[] args) {
    new OptionPaneExample();
  }
}
```



Hello, Welcome to JC2002.

Non-modal example dialog via JDialog (1)

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
  private static JDialog d;
  DialogExample() {
    JFrame f= new JFrame();
    d = new JDialog(f, "Dialog Example", true);
    d.setLayout(new FlowLayout());
    JButton b = new JButton ("OK");
    b.addActionListener (new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        DialogExample.d.setVisible(false);
      }
    });
    d.add(new JLabel ("Click button to continue."));
    d.add(b);
    d.setSize(300,300);
    d.setVisible(true);
  }
```

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

Non-modal example dialog via JDialog (2)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
  private static JDialog d;
  DialogExample() {
    JFrame f= new JFrame();
    d = new JDialog(f, "Dialog Example", true);
    d.setLayout(new FlowLayout());
    JButton b = new JButton ("OK");
    b.addActionListener (new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        DialogExample.d.setVisible(false);
      }
    });
    d.add(new JLabel ("Click button to continue."));
    d.add(b);
    d.setSize(300,300);
    d.setVisible(true);
  }
```

```
  public static void main(String args[]) {
    new DialogExample();
  }
}
```

Create JFrame object to contain the dialog

Create JDialog object

Create JButton object

Non-modal example dialog via JDialog (3)

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
  private static JDialog d;
  DialogExample() {
    JFrame f= new JFrame();
    d = new JDialog(f, "Dialog Example", true);
    d.setLayout(new FlowLayout());
    JButton b = new JButton ("OK");
    b.addActionListener (new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        DialogExample.d.setVisible(false);
      }
    });
    d.add(new JLabel ("Click button to continue."));
    d.add(b);
    d.setSize(300,300);
    d.setVisible(true);
  }
```

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

We will discuss about layouts and action listeners later

UNIVERSITY OF ABERDEEN

Non-modal example dialog via JDialog (4)

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
  private static JDialog d;
  DialogExample() {
    JFrame f= new JFrame();
    d = new JDialog(f, "Dialog Example", true);
    d.setLayout(new FlowLayout());
    JButton b = new JButton ("OK");
    b.addActionListener (new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        DialogExample.d.setVisible(false);
      }
    });
    d.add(new JLabel ("Click button to continue."));
    d.add(b);
    d.setSize(300,300);
    d.setVisible(true);
  }
```

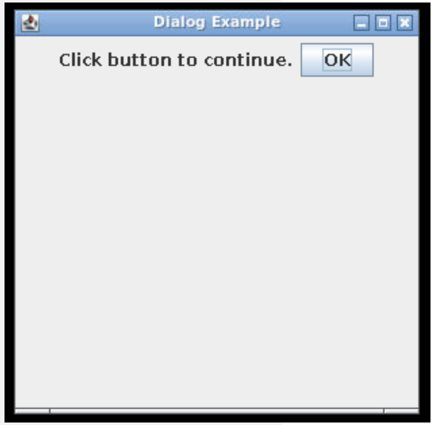
```java
  public static void main(String args[]) {
    new DialogExample();
  }
}
```

Add a label, the button, and make the dialog visible

Non-modal example dialog via JDialog

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
  private static JDialog d;
  DialogExample() {
    JFrame f= new JFrame();
    d = new JDialog(f, "Dialog Example", true);
    d.setLayout(new FlowLayout());
    JButton b = new JButton ("OK");
    b.addActionListener (new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        DialogExample.d.setVisible(false);
      }
    });
    d.add(new JLabel ("Click button to continue."));
    d.add(b);
    d.setSize(300,300);
    d.setVisible(true);
  }
```

```
$ java DialogExample
█
```



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

Questions, comments?