15.4 Inner Classes

An ***inner class*** is a class defined within another class.

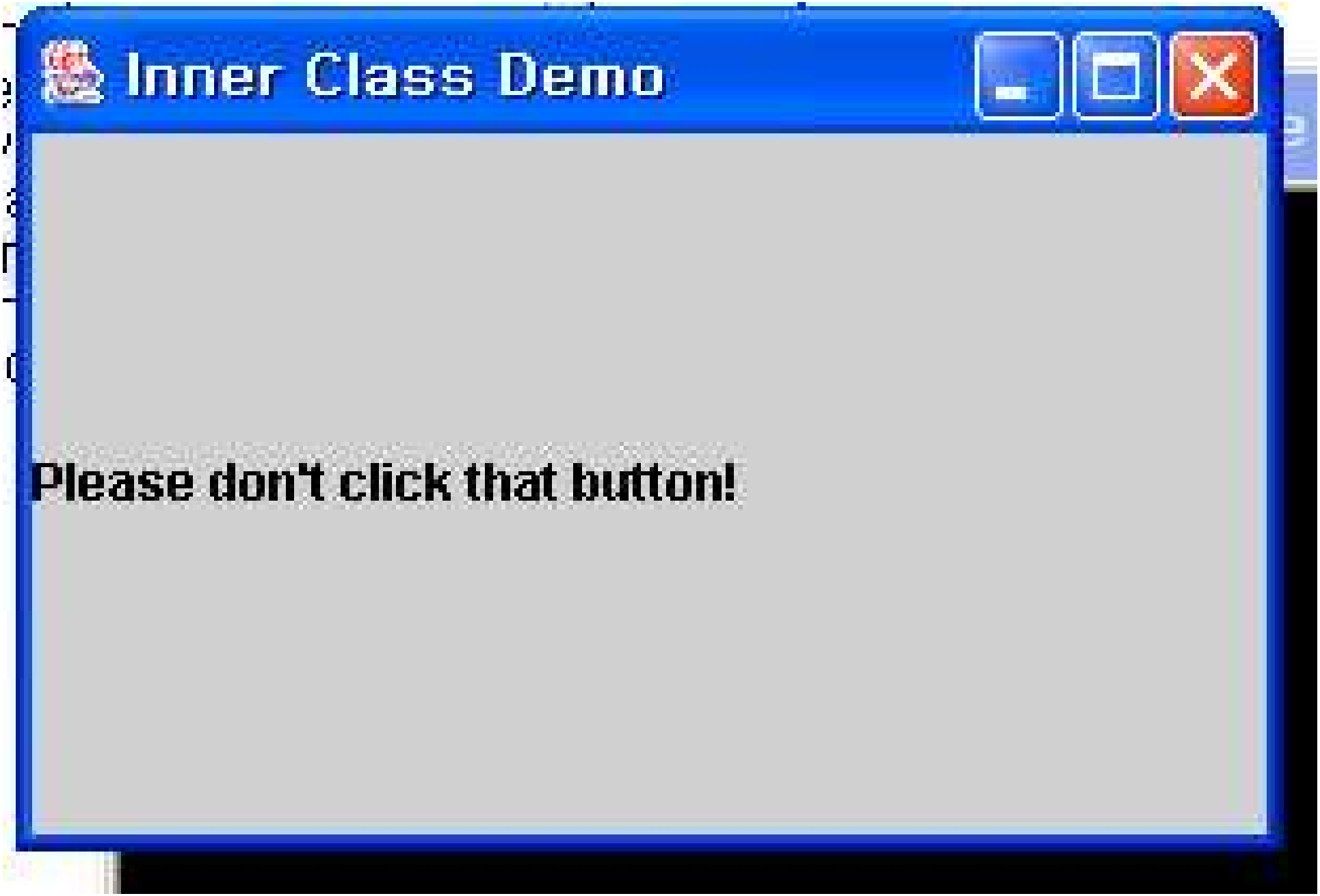
Advantages:

- They make the outer class more self contained.
  - » If `WindowDestroyer` is used, must make sure that class is available.
  - » If `InnerDestroyer` (inner class version of `WindowDestroyer`) is used, it will always be available.
- Inner class has access to all instance variables and methods of outer class, including private ones.
- Avoid name conflicts.
  - » You could have another (outer) class called `InnerDestroyer`
  - » Completely ignore the inner class of the same name.
  - » and there would not be a conflict.

Listing 15.8 An Inner Class - InnerClassDemo.java

```
//  Listing 15.8 An Inner Class

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

public class InnerClassDemo extends JFrame
{
    public static final int WIDTH = 300;
    public static final int HEIGHT = 200;

    /**
     Creates and displays a window of the class InnerClassDemo.
    */
    public static void main(String[] args)
    {
        InnerClassDemo sampleGUI = new InnerClassDemo( );
        sampleGUI.setVisible(true);
    }
```

```
public InnerClassDemo( )
   {
      setSize(WIDTH, HEIGHT);
      setTitle("Inner Class Demo");
      Container contentPane = getContentPane( );
      contentPane.setLayout(new BorderLayout( ));

      JLabel label = new JLabel(
                     "Please don't click that button!");
      contentPane.add(label, BorderLayout.CENTER);

      addWindowListener(new InnerDestroyer( ));
   }

   //An inner class with the same functionality
   //as the class WindowDestroyer.
   private class InnerDestroyer extends WindowAdapter
   {
      public void windowClosing(WindowEvent e)
      {
         System.exit(0);
      }
   }
}
```


Inner Class Demo
Please don't click that button!

Invoking a Method of the Outer Class - innerClassDemo2.java

```
/**
 Demonstration of invoking a method of the outer
 class within an inner class.
*/
public class InnerClassDemo2 extends JFrame
{
   public static final int WIDTH = 300;
   public static final int HEIGHT = 200;

   public static void main(String[] args)
   {
      InnerClassDemo2 gui = new InnerClassDemo2();
      gui.setVisible(true);
   }
```

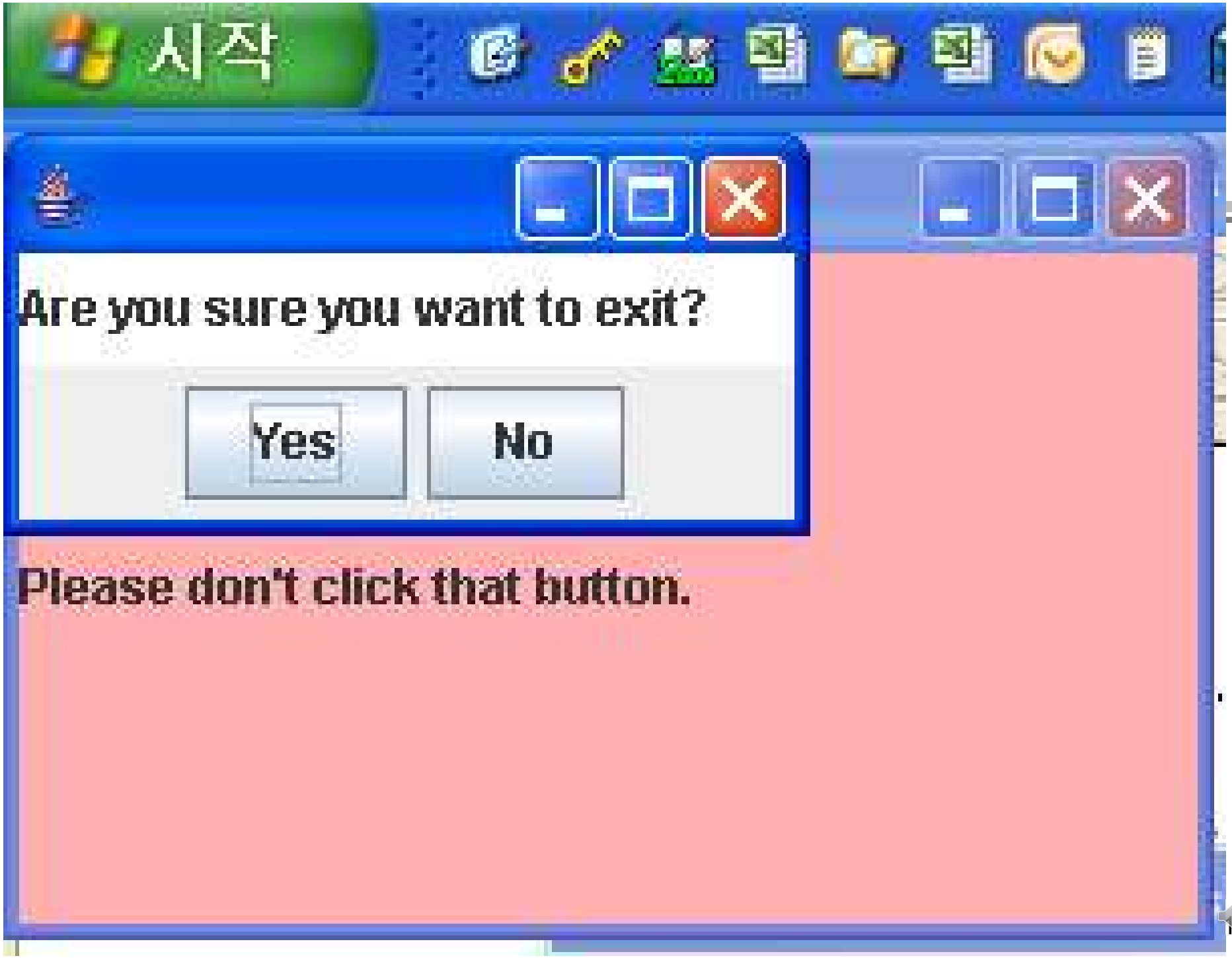
```
public InnerClassDemo2()
  {
     setSize(WIDTH, HEIGHT);
     setDefaultCloseOperation(
           WindowConstants.DO_NOTHING_ON_CLOSE);
     addWindowListener(new InnerDestroyer());
     setTitle("Close Window Demo");
     Container contentPane = getContentPane();
     contentPane.setLayout(new BorderLayout());

     JLabel message = new JLabel(
                  "Please don't click that button.");
     contentPane.add(message, BorderLayout.CENTER);
  }

  public void blushMainWindow()   // private ???
  {
     Container contentPane = getContentPane();
     contentPane.setBackground(Color.PINK);
     repaint();
  }
```

```
public void unBlushMainWindow()
  {
     Container contentPane = getContentPane();
     contentPane.setBackground(Color.WHITE);
     repaint();
  }

  //Displays a window that checks if the user wants to exit.
  private class InnerDestroyer extends WindowAdapter
  {
     public void windowClosing(WindowEvent e)
     {
        ConfirmWindow askWindow = new ConfirmWindow();
        askWindow.setVisible(true);
     }
  }
```

```
//Designed to be used with the inner class InnerDestroyer in
  //the class CloseWindowDemo. Checks if the user wants to exit.
  private class ConfirmWindow extends JFrame
                        implements ActionListener
  {
     public static final int WIDTH = 200;
     public static final int HEIGHT = 100;

     public ConfirmWindow()
     {
        blushMainWindow();    ///
        setSize(WIDTH, HEIGHT);
        Container confirmContent = getContentPane();
        confirmContent.setBackground(Color.WHITE);
        confirmContent.setLayout(new BorderLayout());

        JLabel msgLabel = new JLabel(
                    "Are you sure you want to exit?");
        confirmContent.add(msgLabel, BorderLayout.CENTER);

        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout());

        JButton exitButton = new JButton("Yes");
        exitButton.addActionListener(this);
        buttonPanel.add(exitButton);
```

```
JButton cancelButton = new JButton("No");
            cancelButton.addActionListener(this);
            buttonPanel.add(cancelButton);

            confirmContent.add(buttonPanel, BorderLayout.SOUTH);
        }
 public void actionPerformed(ActionEvent e)
        {
            if (e.getActionCommand().equals("Yes"))
                System.exit(0);
            else if (e.getActionCommand().equals("No"))
            {
                unBlushMainWindow();
                dispose();//Destroys only the ConfirmWindow.
            }
            else
                System.out.println("Error in Confirm Window.");
        }
    }
}
```


Close Window Demo
Please don't click that button.


시작
Are you sure you want to exit?
Yes
No
Please don't click that button.