Here we will show a simple application of multicasting. We will have a frame that can spawn multiple windows with the New button. And, it can close all windows with the Close all button—see Figure 8–10.

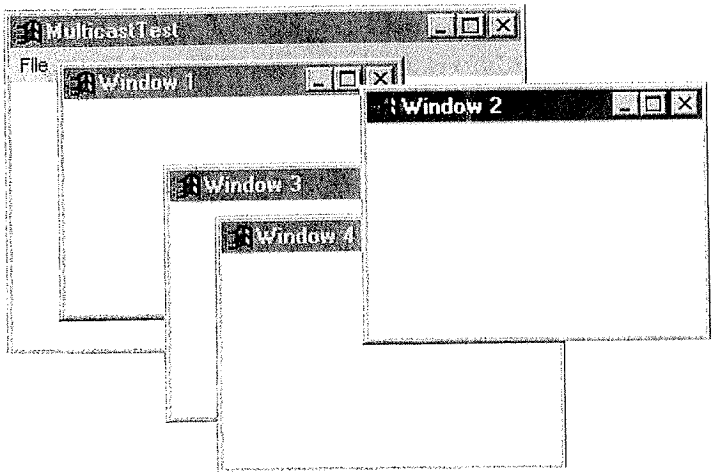

Figure 8–10: All windows listen to the Close all command

The listener to the New button is the panel containing the buttons—it makes the new child windows. But the Close all menu item has *multiple listeners*—each child window is added to the set of listeners. When that button is clicked, all windows are notified and close themselves. Example 8–6 shows the source code.

Example 8–6: MulticastTest.java

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

class MulticastPanel extends JPanel
   implements ActionListener
{  public MulticastPanel()
   {  JButton newButton = new JButton("New");
      add(newButton);
      newButton.addActionListener(this);

      closeAllButton = new JButton("Close all");
      add(closeAllButton);
   }

   public void actionPerformed(ActionEvent evt)
   {  // handles New button
      SimpleFrame f = new SimpleFrame();
      counter++;
      f.setTitle("Window " + counter);
```

```
        f.setSize(200, 150);
        f.setLocation(30 * counter, 30 * counter);
        f.show();
        closeAllButton.addActionListener(f);
    }

    private int counter = 0;
    private JButton closeAllButton;
}

class MulticastFrame extends JFrame
{   public MulticastFrame()
    {   setTitle("MulticastTest");
        setSize(300, 200);
        addWindowListener(new WindowAdapter()
            {   public void windowClosing(WindowEvent e)
                {   System.exit(0);
                }
            } );

        Container contentPane = getContentPane();
        contentPane.add(new MulticastPanel());
    }
}

public class MulticastTest
{   public static void main(String[] args)
    {   JFrame frame = new MulticastFrame();
        frame.show();
    }
}

class SimpleFrame extends JFrame
    implements ActionListener
{   public void actionPerformed(ActionEvent evt)
    {   // handles Close all button
        dispose();
    }
}
```

Advanced Event Handling

In this section, we show you various advanced event handling techniques that bypass or augment the regular event handling mechanism. You'll see techniques for:

- Consuming events
- Implementing secondary event loops
- Adding custom events