CS 580 Client-Server Programming
Fall Semester, 2002
Doc 9 Threads & GUIs
Contents

References

http://java.sun.com/j2se/1.4.1/docs/api/javax/swing/SwingUtilities.html#invokeLater(java.lang.Runnable) for Swing

ForkedUI & ForkedUIExamples parcels, in other/parc directory in the VW7 installation, See ForkedUI parcel comment

Lecture Source Code

Java
Class cvs repository module guiThread

Smalltalk
Class store repository package guiThread

Threads & GUIs

Swing & VisualWorks GUIs maintain an event queues

Only the GUI thread should add events to the queue

Odd things may happen if

- A non-GUI thread adds an event to the event queue
- And the GUI thread is using the event queue

In simple examples it is hard to cause the problem

In complex examples is it really hard to debug the problem

Swing Solution

Use SwingUtilities.invokeLater to run a thread to change a GUI element

For more information see:

- javax.swing.SwingUtilities doc on method invokeLater

VisualWorks Solution

File in the ForkedUi parcel

In the others/parc directory of the VW 7 installation

Use Processor performUIBlock: to run a thread to change a GUI element

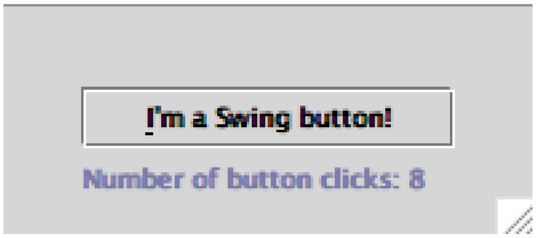
See ForkedUIExamples parcel for examples of use

Example

A window with:

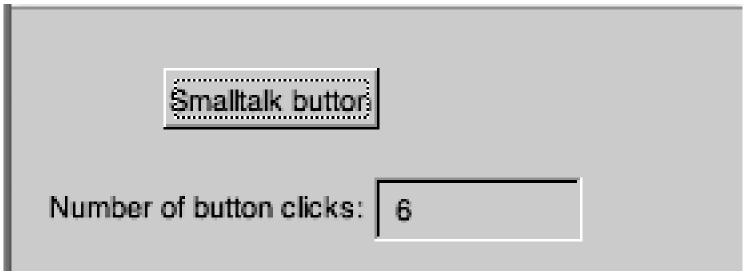
- A button
- A label with the number of button clicks
- A thread that increases the number of button clicks

Swing Window



Number of button clicks: 8

VisualWorks Example



Number of button clicks: 6

Swing Example

Swing code adapted from Sun on-line Swing Tutorial

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

public class SwingApplication {
   private static String labelPrefix = "Number of button clicks: ";
   private int numClicks = 0;
   private UnSafeGUIAccess clock;
   private JLabel label;

   public static void main(String[] args) {
      try {
         UIManager.setLookAndFeel(
            UIManager.getCrossPlatformLookAndFeelClassName());
      } catch (Exception e) { }

      //Create the top-level container and add contents to it.
      JFrame frame = new JFrame("SwingApplication");
      SwingApplication app = new SwingApplication();
      Component contents = app.createComponents();
      frame.getContentPane().add(contents, BorderLayout.CENTER);

      frame.addWindowListener(new WindowAdapter() {
         public void windowClosing(WindowEvent e) {
            System.exit(0);
         }
      });
      frame.pack();
      frame.setVisible(true);
   }
```

```
  public Component createComponents() {
      label = new JLabel(labelPrefix + "0    ");

      JButton button = new JButton("I'm a Swing button!");
      button.setMnemonic(KeyEvent.VK_I);
      button.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent e) {
            addClicks( 1);
         }
      });
      label.setLabelFor(button);
      clock = new UnSafeGUIAccess( this );
      clock.start();
      clock.secondsToSleep( 2);
      JPanel pane = new JPanel();
      pane.setBorder(BorderFactory.createEmptyBorder(
                              30, //top
                              30, //left
                              10, //bottom
                              30) //right
                              );
      pane.setLayout(new GridLayout(0, 1));
      pane.add(button);
      pane.add(label);

      return pane;
   }

   public  synchronized void addClicks(int clicksToAdd) {
      numClicks = numClicks + clicksToAdd;
      label.setText(labelPrefix + numClicks);
   }
}
```

Unsafe Thread

This thread directly modifies a GUI element

See gui.addClicks(1);

```
public class UnSafeGUIAccess extends Thread {
   private SwingApplication gui;
   int sleepMillseconds;
   public UnSafeGUIAccess(SwingApplication owner) {
      gui = owner;
      secondsToSleep( 1);
   }

   public void run() {
      try {
         while (true) {
         sleep( sleepMillseconds);
         gui.addClicks( 1);
         }
      } catch (InterruptedException interrupt) {
         //On interrupt end thread
      }
   }

   public void secondsToSleep(int seconds ) {
      sleepMillseconds = seconds * 1000;
   }
}
```

Safe Thread

Use SwingUtilities.invokeLater to modifies a GUI element

```
import javax.swing.SwingUtilities;

public class SafeGUIAccess extends Thread {
   private SwingApplication gui;
   int sleepMillseconds;
   public SafeGUIAccess(SwingApplication owner) {
      gui = owner;
      secondsToSleep( 1);
   }

   public void run() {
      try {
         while (true) {
         sleep( sleepMillseconds);
         Runnable click = new Runnable() {
               public void run() {
                  gui.addClicks( 1);
               }
            };
         SwingUtilities.invokeLater(click);
         }
      } catch (InterruptedException interrupt) {
         //On interrupt end thread
      }
   }

   public void secondsToSleep(int seconds ) {
      sleepMillseconds = seconds * 1000;
   }
}
```

VisualWorks Unsafe Example

```
Smalltalk defineClass: #VisualWorksApplication
    superclass: #{UI.ApplicationModel}
    indexedType: #none
    private: false
    instanceVariableNames: 'clicks sleepSeconds '
    classInstanceVariableNames: ''
    imports: ''
    category: 'Examples-cs580'!

!VisualWorksApplication class methodsFor: 'interface specs'!

windowSpec
    "UIPainter new openOnClass: self andSelector: #windowSpec"
    <resource: #canvas>
    ^#(#{UI.FullSpec}
        #window:
        #(#{UI.WindowSpec}
            #label: 'VW Application'
            #bounds: #(#{Graphics.Rectangle} 682 378 956 576 ) )
        #component:
        #(#{UI.SpecCollection}
            #collection: #(
                #(#{UI.ActionButtonSpec}
                    #layout: #(#{Graphics.Rectangle} 57 22 138 45 )
                    #name: #ActionButton1
                    #model: #buttonPressed
                    #label: 'Smalltalk button'
                    #defaultable: true )
                #(#{UI.LabelSpec}
                    #layout: #(#{Core.Point} 12 63 )
                    #name: #Label1
                    #label: 'Number of button clicks:' )
                #(#{UI.InputFieldSpec}
                    #layout: #(#{Graphics.Rectangle} 136 63 214 87 )
                    #name: #InputField1
                    #model: #clicks
                    #isReadOnly: true
                    #type: #number ) ) ) )
```

Unsafe Example Instance methods

```
buttonPressed
   ^self addClicks: 1

clicks
   ^clicks

initialize
   super initialize.
   clicks := 0 asValue.
   sleepSeconds :=2.
   self startClock

addClicks: anInteger
   clicks value: (clicks value + anInteger)

processPriority
   ^60

startClock
   clock :=
         [
         [(Delay forSeconds: sleepSeconds) wait.
         self addClicks: 1] repeat]
               forkAt: self processPriority
```

VisualWorks Safe Example

Change the previous startClock method to the following:

```
startClock
  clock :=
      [
      [(Delay forSeconds: sleepSeconds) wait.
      Processor performUIBlock: [self addClicks: 1]]
          repeat]
          forkAt: self processPriority
```