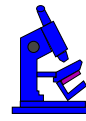


Chapter 7



Tcl/Tk - 3, Java



Bindings



Bind a Tcl/Tk script to an X event



Example 4



```
canvas .c -bd 3 -relief raised; pack .c
button .b -text BUTTON1; pack .b
bind .b <Enter> {.c config -cursor {clock}}
```



File I/O



```
open /etc/passwd r
set f file4
while {[gets $f line] >=0} {
    puts $line
}
close $f
```



File I/O



- ✓ flush fileID
- ✓ seek fileID offset [start/current/end]
- ✓ tell fileID
- ✓ read fileID numbytes



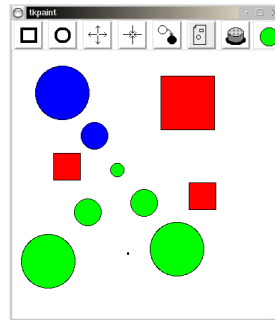
File I/O



- ✓ file dirname name
- ✓ file tail name



Demo tkpaint



Mainline



CODE LISTING

tkpaint1.tcl

```

#!/usr/local/bin/wish -f

set thistool rectangle
set thisop grow
set thiscolour black
button.exitbtn - bitmap @ exit.xbm - command exit
button.squarebtn - bitmap @ square.xbm - command setsquaretool
button.circlebtn - bitmap @ circle.xbm - command setcircletool
button.shrnkbtn - bitmap @ shrink.xbm - command "set thisop shrnk"
button.growbtn - bitmap @ grow.xbm - command "set thisop grow"
button.printbtn - bitmap @ print.xbm - command printit
button.colorbtn - bitmap @ newcolour.xbm - command setanewcolour
canvas.net - width 400 - height 400 - background white - relief sunken
canvas.status - width 40 - height 40 - background white - relief sunken
pack.net - side bottom
pack.status - side right
pack.squarebtn.circlebtn - side left - ipadx 1 m - ipady 1 m - expand 1
pack.exitbtn.printbtn - side right - ipadx 1 m - ipady 1 m - expand 1
pack.colorbtn.shrnkbtn.growbtn - side right - ipadx 1 m - ipady 1 m -
expand 1 bind.net < ButtonPress - 1 >
{
makenode % x % y}
. status create rectangle 10 10 37 37 - tag statusthingy - fill $thiscolour
set nodes 0;
set oldx 0;
set oldy 0;

```



Routines



CODE LISTING

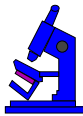
tkpaint4.tcl

```
proc beginmove {x y} {
    global oldx oldy
    set oldx $x; set oldy $y
}

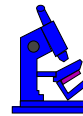
proc domove {item x y} {
    global oldx oldy
    .net move $item [expr "$x-$oldx"] [expr "$y-$oldy"]
    set oldx $x; set oldy $y
}

proc altersize {item x y z} {
    .net scale $item $x $y $z $z
}

proc printit {} {
    .net postscript -file "pic.ps"
}
```



Routines



CODE LISTING

tkpaint2.tcl

```
proc makenode {x y} {
    global nodes oldx oldy thistool thiscolor
    set nodes [expr "$nodes+1"]

    set x1 [expr "$x-20"]; set y1 [expr "$y-20"]
    set x2 [expr "$x+20"]; set y2 [expr "$y+20"]
    if {[string compare $thistool "oval"] == 0} {
        .net create oval $x1 $y1 $x2 $y2 -tag node$nodes -fill $thiscolor
    }
    if {[string compare $thistool "rectangle"] == 0} {
        .net create rectangle $x1 $y1 $x2 $y2 -tag node$nodes -fill $thiscolor
    }
    .net bind node$nodes <Enter> ".net itemconfigure node$nodes -width 5"
    .net bind node$nodes <Leave> ".net itemconfigure node$nodes -width 1"
    .net bind node$nodes <ButtonPress-3> "beginmove %x %y"
    .net bind node$nodes <B3-Motion> "domove node$nodes %x %y"
    .net bind node$nodes <ButtonPress-2> "dothisop node$nodes %x %y"
}

proc dothisop {item x y} {
    global thisop
    if {[string compare $thisop "shrink"] == 0} {
        altersize $item $x $y 0.5
    }
    if {[string compare $thisop "grow"] == 0} {
        altersize $item $x $y 2.0
    }
}
```



Routines



CODE LISTING

tkpaint3.tcl

```
proc setcircletool {} {  
    global thistool thiscolor  
    set thistool oval  
    .status delete statusthingy  
    .status create oval 10 10 37 37 -tag statusthingy -fill $thiscolor  
}  
  
proc setsquaretool {} {  
    global thistool thiscolor  
    set thistool rectangle  
    .status delete statusthingy  
    .status create rectangle 10 10 37 37 -tag statusthingy -fill $thiscolor  
}  
  
proc setanewcolor {} {  
    global thiscolor  
  
    if {[string compare $thiscolor "black"] == 0} {  
        set thiscolor green  
    } { if {[string compare $thiscolor "green"] == 0} {  
        set thiscolor blue  
    } { if {[string compare $thiscolor "blue"] == 0} {  
        set thiscolor red  
    } { if {[string compare $thiscolor "red"] == 0} {  
        set thiscolor orange  
    } { set thiscolor black }  
    }  
    }  
    .status itemconfigure statusthingy -fill $thiscolor  
}
```



End of Tk intro



Next assignment will be a Tk programming one...

and now...



Java and JFC



- ✓ Java - applications across a network
- ✓ Native-code interpreter for Java code.
- ✓ Core functions called JFC (Java Foundation Classes).
- ✓ JFC for GUIs, accessibility, 2D drawing...



AWT



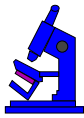
- ✓ AWT - the Abstract Windowing Toolkit.
- ✓ AWT provides buttons, frames, dialogs...
- ✓ Implemented in native code in the Java interpreter.



Swing



- ✓ Not implemented in native code
- ✓ Implemented in AWT.
- ✓ Swing and AWT coexist



Swing advantages



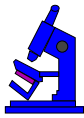
1. Consistent look-and-feel
2. Pluggable look-and-feel
3. High-level widgets



How not to...



- ✗ Watch interpreter overhead
- ✗ Internal thread scheduling



Getting started



<http://java.sun.com/j2se/1.4.2/download.html>

- **j2sdk1.4.2**
 - <http://www.comp.nus.edu.sg/~cs3283/ftp/Java/>
- **Netbeans+j2sdk**
 - <http://www.comp.nus.edu.sg/~cs3283/ftp/Java/>



Documentation



Sun's Java web site, and <http://www.netbeans.org>.

- <http://www.comp.nus.edu.sg/~cs3283/ftp/Java/jfcapi/>
- <http://www.comp.nus.edu.sg/~cs3283/ftp/Java/OpenAPIs/>
- <http://www.comp.nus.edu.sg/~cs3283/ftp/Java/JavaTutorial/>
- <http://www.comp.nus.edu.sg/~cs3283/ftp/Java/swingConnect/>



Demo



Find the file called `SwingSet2.jar`. Then try:

```
java -jar SwingSet2.jar
```

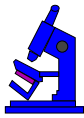


Swing programming



- ✓ Use same strategy as used in Tcl/Tk.
- ✓ A good book that covers this material in detail is

The JFC Swing Tutorial
by Kathy Walrath and Mary Campione.



Toplevel components



The toplevel components provided by Swing are:

1. **JApplet** - for applets within web pages
2. **JDialog** - for dialog boxes
3. **JFrame** - for building applications

All other Swing components derive from the **JComponent** class.



JComponent



- **Tool tips** - little windows with explanations
- **Pluggable look and feel** - as described
- **Layour management** - items within the component
- **Keyboard action management** - Hot keys and so on.
- And other facilities

Swing implements an MVC architecture.

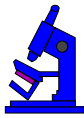
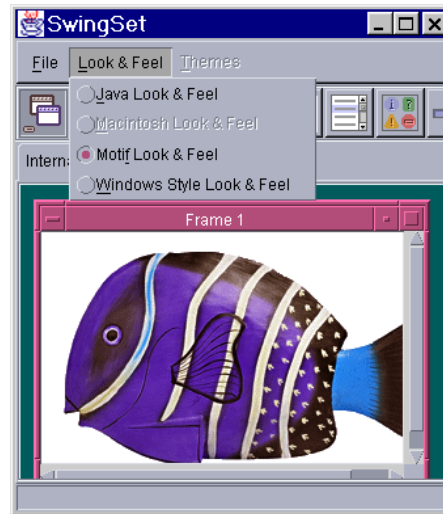


Pluggable look and feel

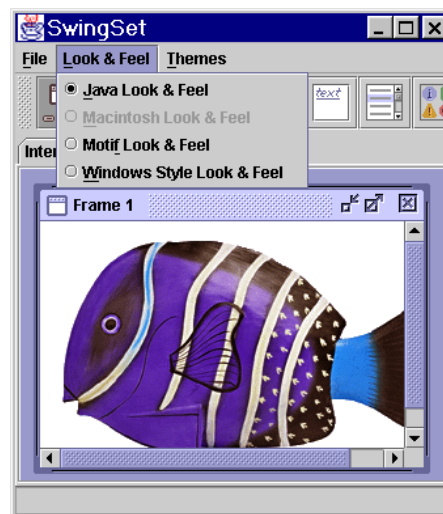




Pluggable look and feel



Pluggable look and feel



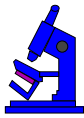


Pluggable look and feel



If you wished to use the WinXX look-and-feel, in the main of your application, you can make the following call:

```
UIManager.setLookAndFeel( "com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
```



Example application



CODE LISTING	t2.java
<pre>public class t2 extends javax.swing.JFrame { public t2() { initComponents(); } private void initComponents() { jLabel2 = new javax.swing.JLabel(); addWindowListener(new java.awt.event.WindowAdapter() { public void windowClosing(java.awt.event.WindowEvent evt) { exitForm(evt); } }); jLabel2.setText("Hello Singapore!"); jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER); getContentPane().add(jLabel2, java.awt.BorderLayout.CENTER); pack(); } private void exitForm(java.awt.event.WindowEvent evt) { System.exit(0); } public static void main(String args[]) { new t2().show(); } private javax.swing.JLabel jLabel2; }</pre>	



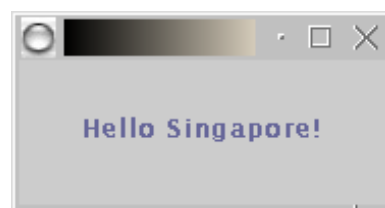
Example application



- ✓ Instantiates a `JLabel`, and sets the text field.
- ✓ Call to `getContentPane` returns the `contentPane` object for the frame - this is a generic AWT container for components associated with each `JFrame`.
- ✓ The `addWindowListener` call is from `java.awt.Window`.



Example application





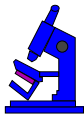
Example applet



CODE LISTING

HelloWorldApp.java

```
public class HelloWorldApp extends javax.swing.JApplet {
    public HelloWorldApp() {
        initComponents();
    }
    private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
        jLabel1.setText("Hello Singapore!");
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        getContentPane().add(jLabel1, java.awt.BorderLayout.CENTER);
    }
    private javax.swing.JLabel jLabel1;
}
```



Example applet



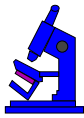
- ✓ This code follows the same structure,
- ✓ Class extends a **JApplet** instead of a **JFrame**.
- ✓ Compile to get a **HelloWorldApp.class** file,
- ✓ Has to be referenced in a web page



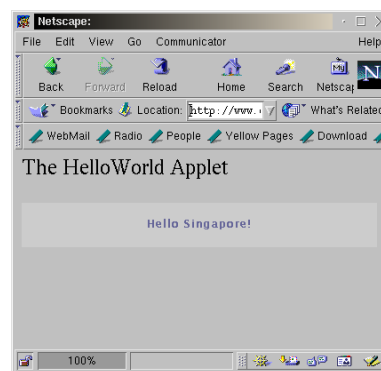
Example applet



CODE LISTING	HelloWorldApp.txt
<pre><BASE HREF="http://www.comp.nus.edu.sg/~hugh/swing/"> <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN"> <html> The HelloWorld Applet <p> <EMBED type = "application/x-java-applet;version=1.1.2" java_CODE = "HelloWorldApp.class" java_ARCHIVE = "applets.jar" WIDTH = 400 HEIGHT = 50 ></EMBED> </HTML></pre>	



Example applet





Netbeans IDE



```
NetBeans IDE 3.2.1, Release Candidate 2 (Build 89) [Project Default]
File Edit View Project Build Debug Versioning Tools Window Help
Selection Mode
Editing GUI Editing Browsing Running Debugging Finished initComponents

Source Editor [12]
18 /** This method is called from within the constructor to
19  * initialize the form.
20  * WARNING: Do NOT modify this code. The content of this method
21  * always regenerated by the Form Editor.
22  */
23 private void initComponents() {
24     jLabel2 = new javax.swing.JLabel();
25
26     addWindowListener(new java.awt.event.WindowAdapter() {
27         public void windowClosing(java.awt.event.WindowEvent
28             evt) {
29             exitForm(evt);
30         }
31     });
32     jLabel2.setText("Hello Singapore!");
33     jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CE
34     getContentPane().add(jLabel2, java.awt.BorderLayout.CE
35
36     pack();
37 }
```



Summary of topics



In this module, we introduced the following topics:

- Tool sets for Java/Swing
- The relationship between JFC, Java and Swing.
- Simple first programs