In this appendix:

- The Results
- Test Program

C

Platform-Specific Event Handling

My life with Java began in September of 1995. I started on a Sun Sparc20 and have since used Java on Windows 95, Windows NT (3.51/4.0), a PowerMac, and an early version of a Java terminal. At the time I started using Java, it was in its alpha 3 release. Even before the beta release, the Internet crowd was hailing Java as the programming language for the next millennium, and people were lining up to take Sun's Java training classes.

Although Java has many important features, probably the most important is platform independence: you can compile a program once and run it anywhere. At least, that was the goal; and Java came impressively close to meeting that goal. However, there are some problems, particularly in the area of event handling. Java programs just do not act the same, from platform to platform, environment to environment. Even if you stay within Sun's Java Developer's Kit, you cannot take a program created on one platform, move it to another, and be guaranteed that it will react the same way to the user's interactions. To make matters worse, Netscape, the makers of the first run-time environment for beta API applets, Netscape, decided to take matters into its own hands with Navigator version 3.0; its version of AWT behaves slightly differently than the JDK's. On top of that, Navigator itself differs from platform to platform. Version 1.1 of the JDK introduces more idiosyncrasies, even as it resolves some others.

With more Java environments available, HotJava, Internet Explorer, and Java terminals to name a few, and new official extensions to AWT coming out, the differences are expanding, instead of contracting. Hopefully, there will be a day when this appendix can go away, completely. Until that time, I've tried to document the behavior of different run-time systems, on different platforms. If the platform is not included in this appendix, the source for a test program is. If you run the program on your platform and send the results to me at jaz@ora.com, they will be included in a future printing or provided online. The test program requires userinteraction, so please follow directions carefully. Between printings, the book's Web site will maintain the latest information at http://www.ora.com/catalog/javawt/. Only the results from using the latest releases of each platform are included in Table C-1.

C.1 The Results

Table C-1 shows the events delivered to each component on the major platforms in Java 1.0. An \checkmark in a particular entry means that the event is passed to Java from the component's peer; a dash means it is not.

Table C-1:	Combonent	Events	in	Iava	1.0
1404 0 1.	Component	LIUUIIII	in j	juou	1.0

Component/Events vs.	NN3.0	NN3.0	NN3.0	SDK	JDK	JDK	JDK	IE3.0	нј	нј
Run-time/Platform	NT/	Mac	Sun	NT/	NT/	Mac	Sun	NT/	NT/	Sun
	Win95			Win95	Win95		Win95	Win95		
			Butt	on		_				
KEY_PRESS	1		1	1	1	_	1	~	1	1
KEY_RELEASE	1	—	1	1	1	—	1	1	1	1
KEY_ACTION	1	—	_	1	1	_	1	1	1	1
KEY_ACTION_RELEASE	1	—	_	1	1	_	1	1	1	1
MOUSE_DOWN	1	_	_	_	_		_	—	_	_
MOUSE_UP	1	1	_	_	_		_	—	_	_
MOUSE_MOVE	1	1	_	_	_	1	_	—	_	—
MOUSE_ENTER	1	1	_	_	_	1	_	—	_	—
MOUSE_EXIT	1	1	_	_	_	1	_	—	_	_
MOUSE_DRAG	1	1	_	_	_	_	_	—	_	—
ACTION_EVENT	1	1	1	1	1	1	1	1	1	1
GOT_FOCUS	1	_	_	_	_	_	1	—	_	—
LOST_FOCUS	1	—		_	—		1	—	—	—
			Can	vas						
KEY_PRESS	1	1	1	1	1	1		1	1	_
KEY_RELEASE	1	1	1	1	1	1	_	1	1	—
KEY_ACTION	1	1	_	1	1	1	_	1	1	—
KEY_ACTION_RELEASE	1	1	_	1	1	_	_	1	1	_
MOUSE_DOWN	1	1	1	1	1	1	1	1	1	1
MOUSE_UP	1	1	1	1	1	1	1	1	1	1

Component/Events vs. Run-time/Platform	NN3.0 NT/ Win95	NN3.0 Mac	NN3.0 Sun	SDK NT/ Win95	JDK NT/ Win95	JDK Mac	JDK Sun Win95	IE3.0 NT/ Win95	HJ NT/	HJ Sun
MOUSE_ENTER	1	1	1	1	1	1	1	1	1	<i>v</i>
MOUSE_EXIT	1	1	1	1	1	1	1	1	1	1
MOUSE_DRAG	1	1	1	1	1	1	1	1	1	1
ACTION_EVENT	_	_	_	_	_	_	_	_	_	_
GOT_FOCUS	1	1	_	1	1	1	_	1	1	_
LOST_FOCUS	1	1	_	1	1	1		1	1	
			Check	kbox						
KEY_PRESS	1		1	1	1	_	1	1	1	_
KEY_RELEASE	1	_	1	1	1		1	1	1	—
KEY_ACTION	1	_	_	1	1		1	1	1	—
KEY_ACTION_RELEASE	1	_	_	1	1	_	1	1	1	—
MOUSE_DOWN	1	_	_	_	_		_	_		_
MOUSE_UP	1	1	_	_	_		_	_		_
MOUSE_MOVE	1	1	_	_	_	1	_	_		_
MOUSE_ENTER	1	1	_	_	_	1	_	_		_
MOUSE_EXIT	1	1	_	_	_	1	_	_		_
MOUSE_DRAG	1	1	_	_	_		_	_		_
ACTION_EVENT	1	1	1	1	1	1	1	1	1	1
GOT_FOCUS	1	_	_	_	_		1	_		_
LOST_FOCUS	1	—	_				1			
			Cho	ice						
KEY_PRESS	1	_	_	1	1		—	1	1	—
KEY_RELEASE	1	—	_	1	1	_	_	1	1	_
KEY_ACTION	1	_	_	1	1		_	1	1	_
KEY_ACTION_RELEASE	1	_	_	1	1	_	_	1	1	_
MOUSE_DOWN	1	_	_	_	_		_	_		_
MOUSE_UP	1	_	1	_	_		_	_		_
MOUSE_MOVE	1	1	1	—	_	1	_			
MOUSE_ENTER	1	1	1	—	_	1	_			
MOUSE_EXIT	1	1	1	_	_	1	_			
MOUSE_DRAG	1	1	_	_	_		_	—		—
ACTION_EVENT	1	1	1	1	1	1	1	1	1	1

Table C-1: Component Events in Java 1.0 (continued)

Component/Events vs.	NN3.0	NN3.0	NN3.0	SDK	JDK	JDK	JDK	IE3.0	нј	нJ
Run-time/Platform	NT/	Mac	Sun	NT/	NT/	Mac	Sun	NT/	NT/	Sun
	Win95			Win95	Win95		Win95	Win95		
GOT_FOCUS	1	—	_	—	—		—	—	—	—
LOST_FOCUS	1	—	—	—	—	—	—	—	—	—
			Lab	el						
KEY_PRESS	1	_	1	_	_	-	1	_	_	
KEY_RELEASE	1	—	1	—	—	-	1	—	—	—
KEY_ACTION	1	—	_	—	—	-	1	—	—	—
KEY_ACTION_RELEASE	1	—	_	—	_	_	1	_	—	_
MOUSE_DOWN	1	—	_	—	_	_	_	—	—	_
MOUSE_UP	1	1	_	—	_		_	—	—	_
MOUSE_MOVE	1	1	_	—	_	1	_	—	—	_
MOUSE_ENTER	1	1	_	—	_	1	_	—	—	_
MOUSE_EXIT	1	1	_	—	_	1	_	—	—	_
MOUSE_DRAG	1	1	_	—	_	_	_	—	—	_
ACTION_EVENT	_	—	_	—	_	_	_	_	—	_
GOT_FOCUS	1	—	_	—	_		_	—	—	_
LOST_FOCUS	1	—		—			—	_	—	—
			Lis	st						
KEY_PRESS	1	—	_	1	1	_	1	1	1	_
KEY_RELEASE	1	—	_	1	1	-	1	1	1	—
KEY_ACTION	1	—	_	1	1	_	1	1	1	_
KEY_ACTION_RELEASE	1	—	_	1	1		1	1	1	_
MOUSE_DOWN	1	—	_	—	—	-	—	—	—	—
MOUSE_UP	1	1	_	—	—	-	—	—	—	—
MOUSE_MOVE	1	1	_	—	_	1	—	—	—	—
MOUSE_ENTER	1	1	_	—	_	1	—	—	—	—
MOUSE_EXIT	1	1	_	—	_	1	_	—	—	_
MOUSE_DRAG	1	1	_	—	_		_	—	—	_
LIST_SELECT	1	1	1	1	1	1	1	1	1	1
LIST_DESELECT	1	1	1	1	1	1	1	1	1	1
ACTION_EVENT	1	1	1	1	1	1	1	1	1	1
GOT_FOCUS	1		—		—	—	1		—	—
LOST_FOCUS	1						1		—	—

Table C-1: Component Events in Java 1.0 (continued)

Component/Events vs. Run-time/Platform	NN3.0 NT/ Win95	NN3.0 Mac	NN3.0 Sun	SDK NT/ Win95	JDK NT/ Win95	JDK Mac	JDK Sun Win95	IE3.0 NT/ Win95	HJ NT/	HJ Sun
			Scrol	lbar						
KEY_PRESS	_	_	1		_	_	_	_	_	_
KEY_RELEASE	_	_	1		_	_	_	_	_	_
KEY_ACTION	_	—	_	_	_		_	_	_	_
KEY_ACTION_RELEASE	_	_	_	_	_	_	—	_	—	
MOUSE_DOWN	1	_	_		_		_		_	_
MOUSE_UP	_	1	_		_		_		_	_
MOUSE_MOVE	1	1	_	_	_	1	_	_	_	_
MOUSE_ENTER	1	1	_	_	_	1	—	_	_	_
MOUSE_EXIT	1	1	_		_	1	_		_	_
MOUSE_DRAG	_	1	_	_	_		_	_	_	_
SCROLL_LINE_UP	1	1	1	1	1	1	1	1	1	1
SCROLL_LINE_DOWN	1	1	1	1	1	1	1	1	1	1
SCROLL_PAGE_UP	1	1	1	1	1	1	1	1	1	1
SCROLL_PAGE_DOWN	1	1	1	1	1	1	1	1	1	1
SCROLL_ABSOLUTE	1	1	1	1	1	1	1	1	1	1
ACTION_EVENT	_	_	_	_	_	_	_	_	_	_
GOT_FOCUS	_	_	_	_	_		1	_	_	_
LOST_FOCUS	_	_	_	_	_		1	_	_	_
			TextA	Irea	r					
KEY_PRESS	1	1		1	1	1	1	1	1	1
KEY_RELEASE	1	1	_	1	1	1	1	1	1	1
KEY_ACTION	1	_	_	1	1	1	1	1	1	1
KEY_ACTION_RELEASE	1	_	_	1	1	_	1	1	1	1
MOUSE_DOWN	1	_	_	_	_		_	_	_	_
MOUSE_UP	1	1	_	_	_		_	_	_	_
MOUSE_MOVE	1	1	_	_	_	1	_	_	_	_
MOUSE_ENTER	1	1	_	_	_	1	_	_	_	_
MOUSE_EXIT	1	1	_	_	_	1	_	_	_	_
MOUSE_DRAG	1	1		_	—		_	_		_
ACTION_EVENT				—	—		_	_		_
GOT_FOCUS	1	1	1	_	—	1	1	_		1
LOST_FOCUS	1	1	1			1	1	_	_	1

Table C-1: Component Events in Java 1.0 (continued)

Component/Events vs. Run-time/Platform	NN3.0 NT/ Win95	NN3.0 Mac	NN3.0 Sun	SDK NT/ Win95	JDK NT/ Win95	JDK Mac	JDK Sun Win95	IE3.0 NT/ Win95	HJ NT/	HJ Sun
			TextF	ïeld						
KEY_PRESS	1	1	1	1	1	1	1	1	1	1
KEY_RELEASE	1	1	1	1	1	1	1	1	1	1
KEY_ACTION	1	1	_	1	1	1	1	1	1	1
KEY_ACTION_RELEASE	1	1	_	1	1		1	1	1	1
MOUSE_DOWN	1	_	_	_	_		_	_	—	
MOUSE_UP	1	1	_	_	_		—	_	—	
MOUSE_MOVE	1	1	_	_	_	1	_	_	—	
MOUSE_ENTER	1	1	_	_	_	1	_	—	—	—
MOUSE_EXIT	1	1	_	_	_	1	_	_	—	
MOUSE_DRAG	1	1	_	_	_		_	_	—	
ACTION_EVENT	1	1	1	1	1	1	1	1	1	1
GOT_FOCUS	1	1	1	_	_	1	1	_	—	1
LOST_FOCUS	1	1	1	_	_	1	1	—	_	1

Table C–1:	Component	Events in	ı Java	1.0	(continued)
	1				\ /

Key:

IE Microsoft's Internet Explorer

HJ Sun's Hot Java Prebeta 1

JDK

Java Developer's Kit 1.0.2 (appletviewer/Java)

NN

Netscape Navigator

SDK

Microsoft SDK

Sun

Solaris 2.x (UNIX/Motif)

Yes, things changed again with the 1.1 release. Table C-2 shows which Java 1.0 events are generated for each component in Java 1.1. Fortunately, there is one clear improvement: the Java 1.1 event model promises much more uniform event processing, since it's largely under your control. For example, you can attach a MouseListener to a Label and receive mouse events that would not be generated with the 1.0 event model.

Component/Events vs. Run-time/Platform	HJ/JDK WinNT/95	HJ/JDK Sun
Butt	on	
KEY_PRESS	1	1
KEY_RELEASE	1	1
KEY_ACTION	1	1
KEY_ACTION_RELEASE	1	1
MOUSE_DOWN		
MOUSE_UP		
MOUSE_MOVE		
MOUSE_ENTER	_	
MOUSE_EXIT		
MOUSE_DRAG	_	_
ACTION_EVENT	1	1
GOT_FOCUS	1	1
LOST_FOCUS	1	1
Can	vas	
KEY_PRESS	_	_
KEY_RELEASE	_	
KEY_ACTION	_	
KEY_ACTION_RELEASE	_	
MOUSE_DOWN	1	1
MOUSE_UP	1	1
MOUSE_MOVE	1	1
MOUSE_ENTER	1	1
MOUSE_EXIT	1	1
MOUSE_DRAG	1	1
ACTION_EVENT	_	_
GOT_FOCUS		
LOST_FOCUS	_	_
Check	kbox	
KEY_PRESS	1	1
KEY_RELEASE	1	1
KEY_ACTION	✓	
KEY_ACTION_RELEASE	1	
MOUSE_DOWN	_	—

Table C-2: Java 1.0 Component Events in Java 1.1

Component/Events vs. Run-time/Platform	HJ/JDK WinNT/95	HJ/JDK Sun
MOUSE_UP		
MOUSE_MOVE	—	—
MOUSE_ENTER	—	—
MOUSE_EXIT	—	—
MOUSE_DRAG	—	—
ACTION_EVENT	1	1
GOT_FOCUS	1	1
LOST_FOCUS	1	1
Cho	ice	
KEY_PRESS	1	—
KEY_RELEASE	\checkmark	—
KEY_ACTION	1	—
KEY_ACTION_RELEASE	1	—
MOUSE_DOWN	—	—
MOUSE_UP	—	—
MOUSE_MOVE		—
MOUSE_ENTER		—
MOUSE_EXIT		—
MOUSE_DRAG		—
ACTION_EVENT	\checkmark	1
GOT_FOCUS	1	—
LOST_FOCUS	✓	—
Lab	el	
KEY_PRESS	—	—
KEY_RELEASE	—	—
KEY_ACTION		—
KEY_ACTION_RELEASE	—	—
MOUSE_DOWN	—	—
MOUSE_UP		—
MOUSE_MOVE	—	—
MOUSE_ENTER	—	—
MOUSE_EXIT	—	—
MOUSE_DRAG	—	_
ACTION_EVENT	_	—

Table C-2: Java 1.0 Component Events in Java 1.1 (continued)

Component/Events vs. Run-time/Platform	HJ/JDK WinNT/95	HJ/JDK Sun
GOT_FOCUS	_	
LOST_FOCUS	_	_
Lis	st	
KEY_PRESS	1	1
KEY_RELEASE	1	1
KEY_ACTION	1	1
KEY_ACTION_RELEASE	1	1
MOUSE_DOWN		_
MOUSE_UP		
MOUSE_MOVE		
MOUSE_ENTER		
MOUSE_EXIT		
MOUSE_DRAG	_	_
LIST_SELECT	1	1
LIST_DESELECT	1	1
ACTION_EVENT	1	1
GOT_FOCUS	1	1
LOST_FOCUS	1	1
Scrol	lBar	
KEY_PRESS	_	
KEY_RELEASE		
KEY_ACTION		
KEY_ACTION_RELEASE	_	_
MOUSE_DOWN	_	_
MOUSE_UP	_	_
MOUSE_MOVE		_
MOUSE_ENTER		_
MOUSE_EXIT		_
MOUSE_DRAG		_
SCROLL_LINE_UP	1	1
SCROLL_LINE_DOWN	1	1
SCROLL_PAGE_UP	1	1
SCROLL_PAGE_DOWN	1	1
SCROLL_ABSOLUTE	1	1

Table C-2: Java 1.0 Component Events in Java 1.1 (continued)

Component/Events vs. Run-time/Platform	HJ/JDK WinNT/95	HJ/JDK Sun
ACTION_EVENT	_	
GOT_FOCUS	_	1
LOST_FOCUS	_	1
TextA	area	
KEY_PRESS	1	1
KEY_RELEASE	1	1
KEY_ACTION	1	1
KEY_ACTION_RELEASE	1	1
MOUSE_DOWN	_	—
MOUSE_UP		_
MOUSE_MOVE		_
MOUSE_ENTER	_	_
MOUSE_EXIT	_	—
MOUSE_DRAG		—
ACTION_EVENT		—
GOT_FOCUS	1	1
LOST_FOCUS	1	1
TextF	ield	
KEY_PRESS	1	1
KEY_RELEASE	1	1
KEY_ACTION	1	1
KEY_ACTION_RELEASE	1	1
MOUSE_DOWN		—
MOUSE_UP		—
MOUSE_MOVE		—
MOUSE_ENTER		—
MOUSE_EXIT	_	—
MOUSE_DRAG		—
ACTION_EVENT	1	1
GOT_FOCUS	1	1
LOST_FOCUS	1	1

Table C-2: Java 1.0 Component Events in Java 1.1 (continued)

Key:

HJ Sun's Hot Java Prebeta 2

JDK

Java Developer's Kit 1.1 (appletviewer/Java)

Sun

Solaris 2.x (UNIX/Motif)

C.2 Test Program

The test program, compList, listed in Section C.2.2 shows the events peers pass along to the Java run-time system. You can then examine the output to see how the run-time system reacts to the different events. When you run compList, the screen looks something like the one in Figure C-1.

👹 Applet Viewer: compList.class	
Applet	
Label Button List 2 List 3	▲ ▼
Multi 1 Multi 2 Multi 3	•
Checkbox TextField	
TextArea	Done
Applet started.	

Figure C-1: Test program

C.2.1 How to Use the Program

Java does not have an automated record and playback feature, so the work is left for you to do. The program displays 10 components: Label, Button, Scrollbar, List, multiselection List, Choice, Checkbox, TextField, TextArea, and Canvas (the black box in Figure C-1). Basically, you must manually trigger every event for every component. For every component on the screen (except Done), do the following:

With the mouse

Move the cursor over the object, press the mouse button and release, and drag the cursor over the object.

With the keyboard

Press and release an alphabetic key, press and release the Home and End keys, arrow keys, and function keys. Do this for every component, even for components like Button and Label that have no logical reason for using keyboard events.

For items with choices

Select and deselect a few choices; double-click and single-click selections.

For the scrollbar

Click on each arrow, drag the slider, and click in the paging area (the space between each arrow and the slider).

For the text field

Press Enter.

When finished

Press the Done button, and analyze the results. Run the program again (without exiting), and check the results again. Try to trigger any specific events that you expect but didn't appear in the output from the first pass. Generating some events requires a little work. For example, on a Macintosh, in order to get the MOUSE_UP and MOUSE_DRAG events, you must do a MOUSE_DOWN off the component; otherwise, the MOUSE_DOWN/MOUSE_UP combination turns into an ACTION_EVENT, if that component can generate it.

NOTE The SunTest business unit of Sun Microsystems has an early version of a record and playback Java GUI testing tool called JavaSTAR. Information about it is available at http://www.suntest.com/JavaS-TAR/JavaSTAR.html. In the future, it may be possible to use JavaS-TAR to help automate this process.

C.2.2 Source Code

The following is the source code for the test program:

```
import java.awt.*;
import java.util.*;
import java.applet.*;
public class compList extends Applet {
  Button done = new Button ("Done");
  Hashtable values = new Hashtable();
```

```
public void init () {
    add (new Label ("Label"));
    add (new Button ("Button"));
    add (new Scrollbar (Scrollbar.HORIZONTAL, 50, 25, 0, 255));
    List 11 = new List (3, false);
    l1.addItem ("List 1");
    l1.addItem ("List 2");
    l1.addItem ("List 3");
    l1.addItem ("List 4");
    l1.addItem ("List 5");
    add (11);
    List 12 = new List (3, true);
    12.addItem ("Multi 1");
    12.addItem ("Multi 2");
    12.addItem ("Multi 3");
    12.addItem ("Multi 4");
    12.addItem ("Multi 5");
    add (12);
    Choice c = new Choice ();
    c.addItem ("Choice 1");
    c.addItem ("Choice 2");
    c.addItem ("Choice 3");
    c.addItem ("Choice 4");
    c.addItem ("Choice 5");
    add (c);
    add (new Checkbox ("Checkbox"));
    add (new TextField ("TextField", 10));
    add (new TextArea ("TextArea", 3, 20));
    Canvas c1 = new Canvas ();
    cl.resize (50, 50);
    cl.setBackground (Color.blue);
    add (c1);
    add (done);
3
public boolean handleEvent (Event e) {
    if (e.target == done) {
        if (e.id == Event.ACTION_EVENT) {
            System.out.println (System.getProperty ("java.vendor"));
            System.out.println (System.getProperty ("java.version"));
            System.out.println (System.getProperty ("java.class.version"));
            System.out.println (System.getProperty ("os.name"));
            System.out.println (values);
        }
    }else {
        Vector v;
        Class c = e.target.getClass();
        v = (Vector)values.get(c);
        if (v == null)
            v = new Vector();
        Integer i = new Integer (e.id);
        if (!v.contains (i)) {
            v.addElement (i);
            values.put (c, v);
        }
```

```
}
return super.handleEvent (e);
}
```

An HTML document to display the applet in a browser should look something like the following:

```
<APPLET code="compList.class" height=300 width=300> </APPLET>
```

C.2.3 Examining Results

The results of the program are sent to standard output when you click on the Done button. What happens to the output depends on the platform. It may be sent to a log file (Internet Explorer), the Java Console (Netscape Navigator), or the command line (appletviewer). The following is sample output from Internet Explorer 3.0 on a Windows 95 platform.

```
Microsoft Corp.
1.0.2
45.3
Windows 95
{class java.awt.Canvas=[504, 503, 1004, 501, 506, 502, 505, 1005,
401, 402, 403, 404], class java.awt.Choice=[1001, 401, 402, 403,
404], class java.awt.Checkbox=[1001, 402, 401, 403, 404], class
compList=[504, 503, 501, 506, 502, 505, 1004, 1005], class java.
awt.TextField=[401, 402, 403, 404], class java.awt.List=[701,
1001, 401, 402, 403, 404, 702], class java.awt.Scrollbar=[602,
605, 604, 603, 601], class java.awt.TextArea=[401, 402, 403, 404],
class java.awt.Button=[1001, 401, 402, 403, 404]}
```

In addition to some identifying information about the run-time environment, the program displays a list of classes and the events they passed. The integers represent the event constants of the Event class; for example, Canvas received events with identifiers 504, 503, etc. The events are not sorted, so you can see the order in which they were sent. Unfortunately, you have to look up these constants in the source code yourself. The class listed as compList is the applet itself and shows you the events that the Applet class receives.