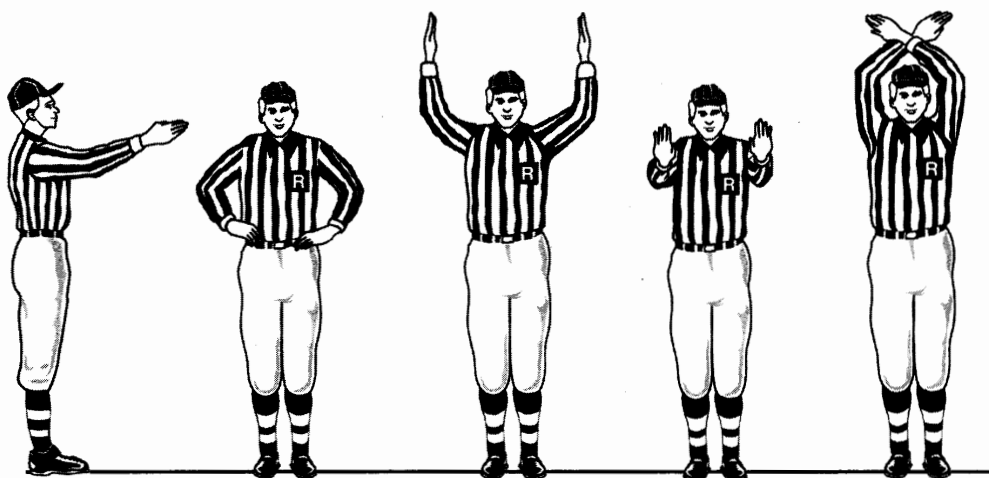




Java in a Nutshell

A Desktop Quick Reference for Java Programmers

David Flanagan

O'Reilly & Associates, Inc.
Bonn • Cambridge • Paris • Sebastopol • Tokyo

EG 7-1196

Java in a Nutshell

by David Flanagan

Copyright © 1996 O'Reilly & Associates, Inc. 103 Morris Street, Suite A, Sebastopol, CA 95472. All rights reserved. Printed in the United States of America.

Series Editor: Mike Loukides

Editor: Paula Ferguson

Production Editor: John Files

Printing History:

February 1996: First Edition

Nutshell Handbook and the Nutshell Handbook logo are registered trademarks of O'Reilly & Associates, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly & Associates, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.



This book is printed on acid-free paper with 85% recycled content, 15% post-consumer waste. O'Reilly & Associates is committed to using paper with the highest recycled content available consistent with high quality.

ISBN: 1-56592-183-6

[3/96]

An Applet Client

Example 7-7 shows another client for the server we developed. This one has a graphical user interface that sends input from a `TextField` component to the server and displays results from the server in a `TextArea` component. If you provide some kind of service from your host, you might want to include a simple client applet like this one on your home page so that people can try out the service. (Remember, though, that applets can only open network connections to the host that they came from—so don't try to create applet clients that connect to other machines.) Note that this applet uses a separate thread to read (and wait for) data from the server and display it in the text area.

Example 7-7: Another client

```
import java.applet.*;
import java.awt.*;
import java.io.*;
import java.net.*;

public class AppletClient extends Applet {
    public static final int PORT = 6789;
    Socket s;
    DataInputStream in;
    PrintStream out;
    TextField inputfield;
    TextArea outputarea;
    StreamListener listener;

    // Create a socket to communicate with a server on port 6789 of the
    // host that the applet's code is on. Create streams to use with
    // the socket. Then create a TextField for user input and a TextArea
    // for server output. Finally, create a thread to wait for and
    // display server output.
    public void init() {
        try {
            s = new Socket(this.getCodeBase().getHost(), PORT);
            in = new DataInputStream(s.getInputStream());
            out = new PrintStream(s.getOutputStream());

            inputfield = new TextField();
            outputarea = new TextArea();
            outputarea.setEditable(false);
            this.setLayout(new BorderLayout());
            this.add("North", inputfield);
            this.add("Center", outputarea);

            listener = new StreamListener(in, outputarea);

            this.showStatus("Connected to "
                + s.getInetAddress().getHostName()
                + ":" + s.getPort());
        }
        catch (IOException e) this.showStatus(e.toString());
    }
}
```

Example 7-7: Another client (continued)

```
// When the user types a line, send it to the server.
public boolean action(Event e, Object what) {
    if (e.target == inputfield) {
        out.println((String)e.arg);
        inputfield.setText("");
        return true;
    }
    return false;
}

// Wait for output from the server on the specified stream, and display
// it in the specified TextArea.
class StreamListener extends Thread {
    DataInputStream in;
    TextArea output;
    public StreamListener(DataInputStream in, TextArea output) {
        this.in = in;
        this.output = output;
        this.start();
    }
    public void run() {
        String line;
        try {
            for(;;) {
                line = in.readLine();
                if (line == null) break;
                output.setText(line);
            }
        } catch (IOException e) output.setText(e.toString());
        finally output.setText("Connection closed by server.");
    }
}
```