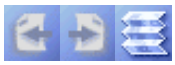


Table of Contents

Select one of the links below to reach the section in this Table of Contents for that trail.

- Copyright
- Getting Started
- Writing Java Programs
- Writing Applets
- Creating a User Interface
- The Java Development Environment
- Integrating Native Methods into Java Programs
- How the Java Language Differs from C and C++
- Troubleshooting

- Copyright
- Getting Started
  - The "Hello World" Application
  - The "Hello World" Applet
  - The "run:" Protocol Handler
  - The "text/plain:" Content Handler
- Writing Java Programs
  - Object–Oriented Programming Concepts: A Primer
    - What is an Object?
    - What are Messages?
    - What are Classes?
    - What is Inheritance?
    - Where Can I Get More Information?
  - The Anatomy of a Java Application
    - Comparison of DateApp and "Hello World"
    - Defining a Class
    - The main() Method
    - An Introduction to Java Objects
      - Declaring, Instantiating and Initializing an Object
      - Static vs. Instance
    - Saving, Compiling and Running an Application
  - The Nuts and Bolts of the Java Language
    - A Simple Class Definition
    - The main() Method
    - Introducing Exceptions
    - Variables and Data Types
    - Control Flow Statements

        - The Standard Input Stream
        - Operators
        - The Standard Output Stream
        - Character Data
        - Running the Application
    - The String and StringBuffer Classes
        - Why Two String Classes?
        - Creating Strings and StringBuffers
        - Accessor Methods
            - More Accessor Methods
        - Modifying StringBuffers
        - Converting Objects to Strings
        - Converting Strings to Numbers
        - Strings and the Java Compiler
    - Command Line Arguments
        - The Space Character Separates Command Line Arguments
        - Conventions
        - Parsing Command Line Arguments
    - Threads of Control
        - What Are Threads?
        - A Simple Thread Example
        - Thread Attributes
            - Thread Body
                - The Clock Applet
            - Thread State
            - Thread Priority
            - Daemon Threads
            - Thread Group
        - Multithreaded Programs
            - Synchronization
            - Deadlock
            - Re−entrant Monitors
        - Summary
    - Input and Output Streams
    - Handling Errors using Exceptions
        - What's an Exception and Why Do I Care?
        - Your First Encounter with Java Exceptions
        - Java's Catch or Declare Requirement
        - Dealing with Exceptions
            - The Example
            - Catching and Handling Exceptions
                - The try Block
                - The catch Block(s)
                - The finally Block
                - Putting It All Together
            - Declaring the Exceptions Thrown by a Method
        - How to Throw Exceptions
            - The throw Statement
            - The Throwable Class and Its Subclasses

        - Creating Your Own Exception Classes
    - Runtime Exceptions--The Controversy
- Writing Applets
    - The Anatomy of an Applet
        - The Life Cycle of an Applet
        - Methods for Milestones
        - Methods for Drawing and Event Handling
        - Using UI Components
        - Threads in Applets
            - Example
        - Saving, Compiling, and Running an Application
- Creating a User Interface
    - Overview of the Java UI
        - AWT Components
        - Other AWT Classes
        - The Anatomy of a GUI-Based Program
            - Classes in the Example Program
            - The Component Hierarchy
            - Drawing
            - Event Handling
- The Java Development Environment
    - Package Tour
        - The Java Language Package
        - The Java I/O Package
            - Input Streams
            - Output Streams
        - The Java Utility Package
    - Managing Your Programming Environment
        - The CLASSPATH Environment Variable
        - Creating and Using Packages
            - Roll Your Own Packages
            - Using Objects and Classes from Other Packages
            - Clarifying Object and Class Names
            - Directory Structure
    - System
        - Using the System Class
        - Standard Output and Error Streams
        - Using System-Dependent Resources
        - Miscellaneous System Methods
- Integrating Native Methods into Java Programs
    - Step By Step
        - Step 1: Write the Java Code
        - Step 2: Compile the Java Code
        - Step 3: Create the .h File
        - Step 4: Create a Stubs File
        - Step 5: Write the C Function
        - Step 6: Create a Dynamically Loadable Library
        - Step 7: Run the Program
    - A Comprehensive Example

- How the Java Language Differs from C and C++
  - C Data Types Not Supported By the Java Language
  - Operators
  - Java Command Line Arguments from C and C++
  - Java Strings are First–Class Objects
  - Miscellaneous Differences
- Troubleshooting
  - Troubleshooting Compiler Problems
  - Troubleshooting Interpreter Problems

Copyright: The Java Language Tutorial

All of the material in The Java Language Tutorial is copyright protected and may not be published in other works without express written permission from Sun Microsystems.

© 1995 Sun Microsystems, Inc. All rights reserved.
2550 Garcia Avenue, Mountain View, California 94043–1100 U.S.A.

This release and related documentation are protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or related documentation may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Portions of this product may be derived from the UNIX® and Berkeley 4.3 BSD systems, licensed from UNIX System Laboratories, Inc., a wholly owned subsidiary of Novell, Inc., and the University of California, respectively. Third–party font software in this product is protected by copyright and licensed from Sun's font suppliers.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the United States Government is subject to the restrictions set forth in DFARS 252.227–7013 (c)(1)(ii) and FAR 52.227–19.

The release described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

TRADEMARKS
Sun, the Sun logo, Sun Microsystems, Solaris, HotJava, and Java are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and certain other countries. The "Duke" character is a trademark of Sun Microsystems, Inc., and Copyright (c) 1992–1995 Sun Microsystems, Inc. All Rights Reserved. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd. OPEN LOOK is a registered trademark of Novell, Inc. All other product names mentioned herein are the trademarks of their respective owners.

All SPARC trademarks, including the SCD Compliant Logo, are trademarks or registered trademarks of SPARC International, Inc. SPARCstation, SPARCserver, SPARCengine, SPARCstorage, SPARCware, SPARCcenter, SPARCclassic, SPARCcluster, SPARCdesign, SPARC811, SPARCprinter, UltraSPARC, microSPARC, SPARCworks, and SPARCompiler are licensed exclusively to Sun Microsystems, Inc. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK® and Sun\ Graphical User Interfaces were developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non−exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

X Window System is a trademark of the X Consortium.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON−INFRINGEMENT.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.

About the Java Language Tutorial

This document was last updated: 12 Dec 95

The Java Language Tutorial: Object−Oriented Programming for the Internet is a practical, on−line guide to writing programs in the Java language.

Note: This document reflects the Java Developer's Kit (Version 1.0 Beta2) release.

- This is a DRAFT!
- Copyright
- Where is it?
- What's New?

- About the Applet References in the Tutorial
- Downloading the Java Language Tutorial

This is a DRAFT!

This tutorial is a **DRAFT**: it's incomplete, buggy, and may contain bad links. We're making this tutorial available to you for two reasons:

1. So you can learn from it −− some of the information in this tutorial is not documented anywhere else.
2. So we can get feedback on this tutorial.

To help us with #2, please let us know what's confusing in these lessons, what seems unnecessary, and whether the lessons helped you at all. Write us at java@java.sun.com or Submit a Bug Report.

Copyright

All of the material in The Java Language Tutorial is copyright protected and may not be published in other works without express written permission from Sun Microsystems. You can use source code from the examples in this work as a basis for your own non−commercial Java programs.

Where is it?

The Java Language Tutorial lives at our website along with all of our other documentation. You can read it "over−the−wire" by clicking on this link to the tutorial's Trail Map or you can download the tutorial in HTML or PostScript formats.

What's ?

12 Dec 95 −− Updated the Second DRAFT

- Made the few changes necessary to reflect Beta2 instead of Beta.
- Fixed various typos, bugs and broken links.
- Jan Kärrman from Uppsala University, Sweden, gave us a new version of the Perl script, html2ps, that we use to generate PostScript files from HTML. This new version supports images, page breaks and page numbers. So, now there is only one PostScript file per lesson. You can download the PostScript files or the HTML versions.
- Added a new lesson, Handling Errors using Exceptions, in the Writing Java Programs trail.

13 Nov 95 −− Updated the Second DRAFT

- Made the few changes necessary to reflect Beta instead of Pre−Beta.
- Changed the name of this document from "The Java Programmer's Guide" to "The Java Language Tutorial: Object−Oriented Programming for the Internet." Why the long name? This document is going to be published as a book, and we wanted to make sure the title was as descriptive as possible, without requiring much prior knowledge of the potential buyer.
- Added ALT text to our link graphics, so that people using non−graphical browsers can understand the information the graphic was conveying.

- Added a new lesson, Using Layout Managers, in the Creating a User Interface trail.

2 Oct 95 –– Released the Second DRAFT

- Everything was updated to reflect the new APIs (except for *The "run:" Protocol Handler* and *The "text/plain" Content Handler* in the Getting Started trail).
- We fixed many typos, clarified many obfuscations, fixed broken links and miscommunications.
- We got a face–lift with new icons and a new page design.
- And, we added these trails and lessons:
    - The Writing Applets trail has a new lesson: The Anatomy of an Applet which describes how applets work and how you use the Applet class to create an applet.
    - The new Creating a User Interface trail has two new lessons: Overview of UI Elements which introduces you to the objects that the Java development environment provides for building UIs, and Arranging Components within a Container which tells you how to use each of the Components provided in the AWT.
    - We've added a trail The Java Development Environment that provides a tour of the packages provided with the development environment, talks about the different tools including the new prototype debugger. This trail includes a lesson on Using System Resources.
    - We've also added the Integrating Native Methods into Java Programs trail that shows you how to integrate native methods into your Java programs. Currently, this trail contains two lessons. The first, Step By Step, walks you through an example showing you each step you take to implement a native method. The second, A Comprehensive Example, which is largely unfinished, is an example that implements a "character–replacement" program using native methods.
    - And finally, two new lessons have been added to the Writing Java Programs trail: Threads of Control and Object–Oriented Programming Concepts: A Primer.

18 May 95 –– Released the First DRAFT

About the Applet References in the Tutorial

Applet references in these pages actually contain references to two different versions of the applet:

- a 1.0alpha3 compatible version which you can view these using the 1.0alpha3 version of the HotJava browser
- a Beta–compatible version which you can view these using the appletviewer or other Beta–compatible browser

Note to 1.0alpha3 Developers: This programmer's tutorial doesn't contain any information on how to upgrade your 1.0alpha3 Java applets to the 1.0 Beta release. For more information on how to do this please refer to Converting Applets.

Downloading the Java Language Tutorial

UNIX

UNIX users can download a compressed tar file containing the tutorial in HTML format or in PostScript format. Note that our PostScript files are generated from HTML using Jan Kärrman's wonderful html2ps script.

HTML format

Approximate size when compressed: 900K

Approximate size when uncompressed: 1,840K

PostScript format

Approximate size when compressed: 940K

Approximate size when uncompressed: 5M

Windows/DOS

Windows/DOS users can download a zip file containing the tutorial in HTML format or in PostScript format.

HTML format

Approximate size when compressed: 825K

Approximate size when uncompressed: 1,840K

PostScript format

Approximate size when compressed: 655K

Approximate size when uncompressed: 5M

Table of Contents

Trail Map: The Java Language Tutorial

This document was last updated: 12 Dec 95

Welcome to The Java Language Tutorial: Object–Oriented Programming for the Internet
The Java Language Tutorial is a practical, on–line guide to writing programs in the Java language. For information about the structure and organization of this tutorial please refer to About this Tutorial.

Note: This draft reflects the API distributed in the 1.0 Beta2 Java Developers Kit (JDK). For information on how the API has changed and how to get the JDK, see the JDK home page. For the change history of the tutorial, how to download the tutorial in PostScript and HTML formats, and other information about this document, see About the Java Language Tutorial.

Copyright

All of the material in The Java Language Tutorial is copyright protected and may not be published in other works without express written permission from Sun Microsystems. You can use source code from the examples in this work as a basis for your own non–commercial Java programs.

Getting Started

The most common Java programs are applications and applets. *Applications* are stand–alone programs, such as the HotJava browser. *Applets* are similar to applications, but they don't run on their own. Instead, applets adhere to a set of conventions that lets them run within a Java–compatible browser. If you're using a browser that can view applets, then you should see an animation just below this paragraph –– that's an applet embedded in this HTML page.

The "Hello World" Application –– Start here if you want to create stand–alone Java applications.

The "Hello World" Applet –– Start here if you are interested in writing applets only.

Besides applications and applets, you can also write two types of *handlers* –– programs that the HotJava browser loads when it encounters a protocol or content type that it hasn't seen yet. Each *protocol handler* understands a particular protocol (such as http: or doc:) and translates it into a form that your Java–compatible browser can display. Similarly, each *content handler* understands

and translates a particular MIME type/subtype combination, such as text/plain or image/jpeg.

The "run:" Protocol Handler –– A step–by–step example of writing a protocol handler.

The "text/plain" Content Handler –– A step–by–step example of writing a content handler.

What Next?

Now that you've seen how to write a Java program, you can proceed with any of the topics below.

Writing Java Programs –– These lessons discuss techniques and concepts that any Java programmer can use.

Writing Applets –– These lessons discuss techniques and concepts that are specific to writing Java applets.

Creating a User Interface –– Here's the information you need to create the user interface of your applet or application.

The Java Development Environment –– Take a look here for information about the Java development environment, tools, how to structure and manage your programming environment, debugging techniques and memory management techniques.

Integrating Native Methods into Java Programs –– These lessons show you everything you need to know to write native methods in the Java language. A native method is a Java method whose implementation is provided in another programming language such as C. This lesson includes step–by–step instructions and a comprehensive example program.

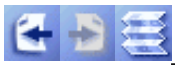
Networking These lessons will have the information you'll need if your program deals with the network.

Extending the HotJava Browser −− These lessons will have information on writing protocol handlers and content handlers −− loadable modules that extend the capabilities of the HotJava browser.

How the Java Language Differs from C and C++ −− This trail points out the differences between the Java language and C and C++.

Troubleshooting −− Follow this trail to find all of the pages that discuss troubleshooting.

Table of Contents

 *Table of Contents*

About this Tutorial

The Java Language Tutorial is a practical, on–line guide to writing programs in the Java language. Java programs can be standalone applications, or they can execute within a Java–enabled browser such as HotJava.

Documentation Trails

This tutorial consists of a series of intersecting *trails*––programming lessons grouped together based on topic. To learn about a certain area of Java programming, go to the *Trail Map* and decide which trail is of interest to you and follow the link to the *trail head* for that trail. For example, if you are interested in information about writing applets, select the link to the *Writing Applets* trail, marked with this signpost.

The trail head is the first page on a trail and it lists, describes, and provides links to all of the lessons on the trail. Selecting a link from the trail head takes you to the *lesson head* for the lesson that you chose. The lesson head is the first page in the lesson and introduces the lesson, and typically provides an outline or overview of the lesson. Usually, several *lesson pages* comprise the entire lesson.

The Control Panel

Each page in **The Java Language Tutorial** has a control panel at the top with the following five elements:

The "Previous" link which takes you to the previous page in the lesson.

The "Next" link which takes you to the next page in the lesson.

This link transports you to the top of the tutorial: to the *Trail Map*.

This is a trail marker and is a link to the trail head of the current trail. Each trail has its own

identifying *trail marker*––the symbol shown here is represents the *Creating a User Interface* trail.

Lesson Title And finally, this right–aligned (supposedly) link is the title of the current lesson and is linked to the lesson head (the first page in the lesson) of the current lesson.

In–Line Links

Links within the text of the tutorial have one of the following forms:

- Links to pages within the current lesson are text links.
- Links to pages within a different lesson, but on the same trail as the current lesson are also text links but they have an icon that indicates which trail the lesson is on. Both the text and the icon are links.
- Links to pages in a different trail are also text links but they have an icon that indicates which trail the lesson is on. Both the text and the icon are links.

Table of Contents