21世纪高等学校计算机规划教材

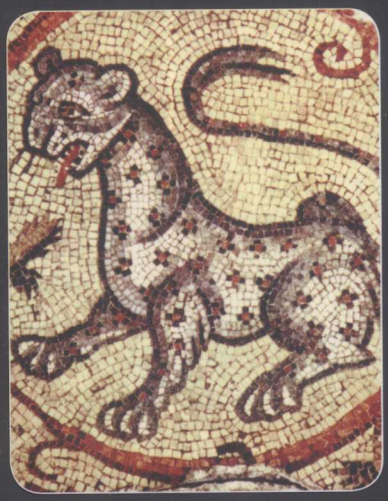
21st Century University Planned Textbooks of Computer Science

Java 语言程序设计

Java Programming

辛运帏 饶一梅 编著

- 注重讲解基础知识
- 深入阐述核心语法
- 精心设计教学示例



名家系列

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

Java 语言程序设计

Java Programming

辛运帏 饶一梅 编著

名家系列

人民邮电出版社

北京

图书在版编目（CIP）数据

Java语言程序设计 / 辛运帏，饶一梅编著. —北京：人民邮电出版社，2009.10
21世纪高等学校计算机规划教材
ISBN 978-7-115-20939-9

Ⅰ. J… Ⅱ. ①辛…②饶… Ⅲ. JAVA语言－程序设计－高等学校－教材 Ⅳ. TP312

中国版本图书馆CIP数据核字（2009）第108871号

内 容 提 要

本书是学习 Java 语言的入门教材。全书从 Java 语言的基本特点入手，详细介绍了 Java 语言的基本概念和编程方法，同时深入介绍了 Java 的高级特性。本书共分为 12 章，涉及 Java 中的基本数据类型、基本语法、类的概念及特性、异常处理、用户界面设计、小应用程序、输入/输出操作及线程等内容。

本书内容详尽，并配合大量示例，在每章的最后均列出若干习题，供读者参考。

本书可作为普通高等院校计算机专业本科生程序设计课程的教材，也可供程序设计人员参考使用。

21 世纪高等学校计算机规划教材

Java 语言程序设计

◆ 编　　著　辛运帏　饶一梅
责任编辑　滑　玉
执行编辑　武恩玉

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京顺义振华印刷厂印刷

◆ 开本：787×1092　1/16
印张：14.5
字数：383 千字　　2009 年 10 月第 1 版
印数：1－3 000 册　　2009 年 10 月北京第 1 次印刷

ISBN 978-7-115-20939-9/TP

定价：25.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

Java 语言经过近 20 年的发展完善，其功能日益强大，应用的领域越来越广。从手机等手持设备中的程序到各类企业级应用程序，都可以看到 Java 的身影，Java 已经成为世界级的编程语言，是编程人员开发时的首选工具。

目前国外的高等教育中，Java 语言已经列入计算机类本科生的教学计划。不仅如此，一些专业基础课也以 Java 语言为描述工具，如数据结构、计算方法等课程。国内的情况与此类似，很多重点院校相继为计算机专业的本科生开设了 Java 语言课程，目前这个趋势由重点院校逐渐扩展到一般院校，开设该课程的学校越来越多，很多学校已经将 Java 语言作为相关专业的第一门编程语言。

在 Java 语言逐步进入高校课堂的同时，我们也发现能够作为普通高等院校 Java 语言课程的教材相对缺乏。不可否认，市面上确实有很多 Java 语言的各类参考书籍，而且质量也相当高，但由于受众不同，其中相当一部分书不适合作为教材使用，特别不适合作为普通高等院校为本科生开设的第一门编程语言课程的教材。为此，我们决定编写本书，希望能为普通高等院校的本科教学提供帮助。另外，本书也可作为专业人员学习 Java 语言的参考书。愿本书能成为读者进入 Java 殿堂的铺路石。

本书共分 12 章。在简单介绍了 Java 语言的特点之后，第 2 章集中介绍了面向对象程序设计的基本概念和基本方法。同时考虑到有很多读者或许已经了解了 C++ 语言，所以将 C++与 Java 语言进行了简单的对比。面向对象的程序设计概念是独立于语言的，但在各种语言中的实现细节又有所区别，本章的内容旨在为没有接触过面向对象程序设计技术的读者提供帮助。

第 3 章和第 4 章介绍了 Java 语言的基本语法，包括标识符的定义、变量的说明、基本数据类型的使用、基本语句的格式等内容。这些知识几乎是每一种编程语言都要涉及的部分。

第 5 章着重介绍了 Java 语言提供的面向对象的程序设计技术，包括类的定义、类的继承、方法的重载与重写、封装等。同时，还介绍了 Java 中特有的包概念、接口机制等。众所周知，Java 是纯面向对象的编程语言，编写 Java 程序的过程就是定义它的类的过程，所以类的定义及使用是 Java 语言中最核心的部分。

第 6 章介绍了介于基本数据类型及类类型之间的数组、表及字符串等类型。严格来说，这些类型属于类的范畴，但传统上它们的使用方式近似于基本数据类型，所以将它们列在一章内。

第 7 章介绍异常的处理，这是保证程序正常执行的一种手段。第 8 章是图形界面的编写技术，这是应用程序的外观，是人机交互的接口，一定程度上也反映了程序的功能。良好的人机交互界面能为使用人员起到良好的辅助功能，也能规范操作人员的操作流程。

第 9 章介绍 Java 中的小应用程序，这是需要浏览器支持的一类特殊程序。第 10 章介绍输入/输出操作，这是任何应用程序都不会缺少的部分。第 11 章和第 12

章分别介绍线程的概念及网络应用程序的编写。这些内容属于较高级的应用部分。

本书注重汲取国内外优秀教材的特点，结合编者在教学过程中的感受及学生们的反映，精选示例和程序，从 Java 语言的基本特点入手，循序渐进地向读者介绍 Java 语言，同时配合介绍编程的思想和方法。书中的程序均提供了较完整的代码，并在 JDK 6.0 下完成了调试。读者可以在自己的机器上尝试运行，得到执行结果。JDK 6.0 是目前最新的版本，随着 Java 的不断完善，会有更新的版本问世，读者可随时关注相关网站。

作为教材，在每章的最后列出若干习题，供读者练习使用。教师可根据各学校的具体课时安排，全部或选讲书中的内容。

在本书的编写过程中，编者得到了南开大学信息技术科学学院卢桂章教授、陈有祺教授、刘景教授、周玉龙教授、袁晓洁教授等的亲切关怀和悉心指导。感谢选择本书作为教材的教师们和同学们，也感谢读者您在众多的 Java 参考书中选中了本书。

本书由辛运帏、饶一梅等共同编写。其中，辛运帏编写了 1～7 章，饶一梅编写了 8～12 章，硕士生侯林峰、王春韶、于文平调试了书中大部分程序并制作了相关课件。最后由辛运帏对全书内容进行了审校。由于作者的水平有限，书中难免有错误和不妥之处，恳请广大读者特别是同行专家们批评指正，您的任何意见或建议，都是我们继续修改本书的动力。

编者

2009 年 4 月于南开园

目 录

第 1 章 Java 语言

程序设计语言是程序员编制程序完成某个任务的必备工具之一。自计算机问世以来，出现过很多编程语言，Java 属于高级程序设计语言范畴，是最具有代表性的语言之一。本章简要介绍 Java 语言的产生背景，并以一个简单程序为例，介绍程序执行的关键步骤。

1.1 Java 语言简介

Java 语言是目前主流的高级程序设计语言，下面先来了解它的历史。

1.1.1 Java 语言的问世

Java 语言产生于 1991 年。它是由美国 Sun Microsystems 公司（太阳微系统有限公司）的 Jame Gosling 领导的一个小组设计开发的，是专为在家用消费类电子产品上进行交互式操作而设计的。但是结果并不理想，没有引起业界太多的关注。

虽然 Java 语言最初的表现不尽如人意，但是设计人员及时调整了研制目标，因为他们意识到互联网的出现及广泛使用急需一种面向网络应用的语言，使用这种语言能够开发网络应用程序，开发的程序最好还能嵌入 HTML 页面中，从而能给 Web 用户带来生动的界面及交互方式。有了这样明确的需求，修改 Java 语言的设计就只是时间问题了。

1995 年，从内到外都已经重新设计的 Java 语言正式推出。Java 语言是第一种用于编写 Web 程序的高级编程语言，同时又是一种通用的程序设计语言，还是完全的面向对象的编程语言。

设计人员将这种语言的名字由原来的 Oak 改为现在的 Java，并配上一杯冒着热气的咖啡作为它的标志。至此，一种全新的语言诞生了。

从面世之后，Java 语言的设计人员一直在进行完善工作。Sun 公司不断丰富 Java 的类库，为开发人员提供功能越来越强大的软件开发工具包（Software Development Kit，SDK），语言的性能也日益提高，更加拓宽了语言的应用面。读者可以从 Sun 公司的网站（http://java.sun.com/javase）上查询当前最新的版本信息，并获取相关资料。

1.1.2 Java 语言的组成

Java 语言是一种全新的高级程序设计语言，它代表了一种新的计算模式。它既是开发环境，又是应用环境。Java 语言包含了 4 方面：面向对象的程序设计语言、Java 虚拟机（Java Virtual Machine，JVM）、与机器无关的二进制格式的类文件及完整的软件开发包。

1. 面向对象的程序设计语言

Java 是一种面向对象的程序设计语言。它所有的语句都包含在类的定义中。这样的设计方式代表了程序设计的主流趋势。它的语句格式非常类似于 C++语言风格，所以熟悉 C++语言的程序员可以很快地过渡到 Java 语言。另外，Java 又是非常严格的强类型语言，它去掉了 C++语言中容易出错的语法成分。例如，Java 中没有指针、结构和类型定义等概念，不再有全局变量，没有#include 和#define 等预处理器，也没有多重继承的机制。C++语言中经常使用的释放空间的语句在 Java 中也交给系统来处理。这些改变使设计 Java 程序时更加简单，对内存的控制也更加得心应手。Java 的系统对程序进行严格的类型检查，所以程序运行起来更加安全。Java 还有一个显著的特点，那就是它的解释器只占用很少的内存，适合在绝大多数类型的机器上运行。

使用 Java 语言便于开发网络应用，它内置了 TCP/IP、HTTP、FTP 等协议类库。Java 程序在语言定义阶段、字节码检查阶段及程序执行阶段进行的 3 级代码安全检查机制，对参数类型匹配、对象访问权限、内存回收、Java 小应用程序的正确使用等都进行了严格的检查和控制，可以有效地防止非法代码的侵入，阻止对内存的越权访问，能够避免病毒的侵害。

2. Java 虚拟机

使用高级程序设计语言编写的代码并不能直接在计算机上运行，代码需要经过编译过程、链接过程等的特殊处理，生成可执行文件，也就是机器能够识别的机器码，然后才能正常运行。在 Java 中，这个过程由 Java 虚拟机来担当。JVM 将 Java 源程序转为字节码（bytecode）文件。字节码文件又称为类文件，文件扩展名为.class。之后 JVM 读取类文件来执行。JVM 的实现称为 Java 运行时系统或运行时环境（Runtime Environment），简称为运行时。

Java 虚拟机是运行 Java 程序必不可少的机制。Java 虚拟机规范中给出了 JVM 的定义：JVM 是在一台真正的机器上用软件方式实现的一台假想机。JVM 的某些指令很像真正的 CPU 指令，包括算术运算、流控制和数组元素访问等。JVM 的具体实现包括指令集（等价于 CPU 的指令集）、寄存器组、类文件格式、栈、垃圾收集堆、内存区。JVM 既可以使用软件方式实现，也可以使用硬件方式实现。

可以将 JVM 看作是编译后的 Java 程序和硬件系统之间的接口，JVM 屏蔽了硬件平台的差异性，正因为有了这个机制，Java 做到了“一次编写，到处运行”，具备了平台无关性的特性。这才是 JVM 的奇妙之处。

3. 类文件格式

计算机所用的芯片不同，安装的操作系统也会有所不同，因此同一种软件产品往往只能安装并运行在某一类或某几种计算机上，这就是我们通常所说的平台依赖性。程序员在编写软件时，也会根据所运行机器的独特性，在代码一级进行特殊处理。所以程序移植时非常困难，这是困扰编程人员多年的大问题。Java 语言很好地解决了这个问题。

在 Java 中，编译后得到的文件称为类文件。由于有 JVM 的存在，这个类文件可以采取统一的格式而不依赖于硬件平台。同时，由于 Java 是强类型语言，所以程序中所有类型的长度都是固定的，不会依系统的不同而有所变化。凡是与机器相关的部分都交给 JVM 去处理，而在代码一级是完全不需要考虑机器平台差异的。所以，Java 程序是与平台无关的。

4. 丰富的 API 文档和类库

Java 为用户提供了详尽的应用程序编程接口（Application Programming Interface，API）说明文档。Java 开发工具包（Java Development Kit，JDK）中的类库包罗万象，程序员的开发工作可以在一个较高的层次上展开。这也正是 Java 受欢迎的重要原因之一。

1.2 开发环境的安装

要编写一个 Java 应用程序，必须先安装开发环境。开发环境包括一个编辑软件及开发 Java 程序必需的 JDK 工具。编辑软件可以使用计算机上的任何一个文本编辑器，如 Notepad、TextPad、UltraEdit 等。读者也可以安装集成开发环境（Integrated Development Environment，IDE）。比较著名的 IDE 有 Java WorkShop、Jbuilder（Borland 公司）、Visual J++（MicroSoft 公司）、Visual Age for Java（IBM 公司）及 Eclipse 和 Eclipse 的加强版 WSAD。但建议读者在熟悉 JDK 之后再慢慢过渡到使用 IDE，毕竟 JDK 是基础。当编制大型程序时，使用 IDE 编程会更加方便快捷。

这里我们只介绍 Sun 公司提供的 JDK 的安装。

1. 文件下载

JDK 是 Sun 公司提供的软件开发工具包，其中含有编写和运行 Java 程序的所有工具，我们要用到的工具包括以下几种。

- javac：Java 语言编译器，用来将 Java 程序编译成字节码。它读入 Java 源程序，输出结果为 Java 字节码。
- java：Java 字节码解释器，执行 Java 编译器生成的字节码文件。这是面向 Java 程序的一个独立运行系统。
- javaAppletViewer：小应用程序浏览工具，用于测试并运行 Java 小应用程序。
- jdb：Java 调试器，用于调试 Java 程序。
- javap：反编译，将类文件还原为方法和变量。
- javadoc：文档生成器，创建 HTML 文件。

JDK 还包括 Java 类库（包括 I/O 类库、用户界面类库、网络类库等）。

运行 Java 程序时，一定要先安装 JDK，并正确设置系统的 PATH 和 CLASSPATH 环境变量，这样系统才能找到 javac（用于编译程序的命令）和 java（用于执行程序的命令）所在的目录。

我们以 Windows 环境为例，介绍安装过程。首先，登录到下列网址：

```
http://java.sun.com/javase/downloads/index.jsp
```

这是 Sun 公司提供的 Java 产品下载网页，从中找到最新版本的下载链接。这个页面中提供了多个版本的产品，读者可以根据实际使用的特定平台选择合适的那一个，目前最新的版本是 JDK 6 Update 11。

若选择下载 JDK 6，则下载的文件名是 jdk-6u11-windows-i586-p.exe，文件大小为 72.9MB，这是一个自展开文件。下载成功后，双击该文件即可开始安装 JDK，如图 1-1 所示，之后按照安装向导进行即可。

可以为 JDK 选择一个安装目录，默认设置下，JDK 安装在 C:\Program Files\ Java\ jdk1.6.0_11 目录下。安装 JDK 后产生的目录结构如图 1-2 所示。

其中，bin 目录下包括 Java 的开发工具，如 Java 编译器、解释器等；demo 目录中是一些实例程序，供读者参考；lib 目录中放置的是 Java 开发类库；jre 目录中放置的是 Java 运行环境，包括 Java 虚拟机、运行类库等。

2. 设置环境变量

安装 JDK 后，需要设置环境变量，以便系统能自动找到命令所在的目录。可以直接在系统的批处理文件“autoexec.bat”中添加环境变量的信息，也可以使用系统中提供的属性页来设置，后

者更加方便快捷。

下面仍以 Windows 系列环境为例，介绍使用属性页设置环境变量的方法。例如，使用 Windows NT/2000/XP。打开控制面板，选择“系统”→“高级”→“环境变量”，修改环境变量的值。

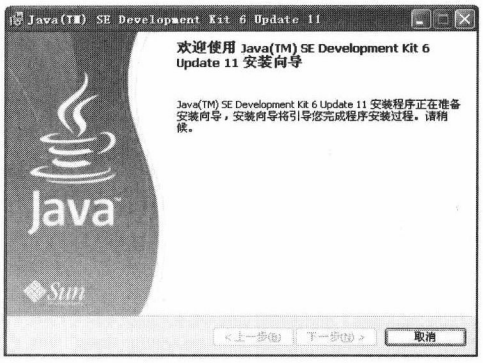


图 1-1　JDK 安装开始

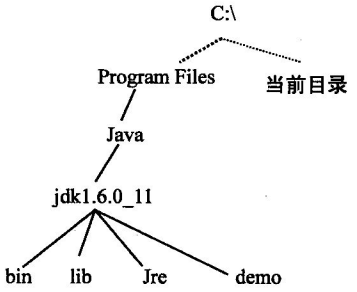


图 1-2　JDK 目录

为了简化环境变量的设置，可以先设定 Java 的安装目录。假设将 JDK 安装在 C:\Program Files\Java\jdk1.6.0_11 下，在打开的环境变量设置窗口中，新建一个系统变量，可以命名为 JAVA_HOME，并设置值为 C:\Program Files\Java\jdk1.6.0_11，也就是这里所指的安装目录。之后，修改环境变量 Path。一般情况下，这个值已经存在，只需要在现有值的最后添加以下内容。

```
;%JAVA_HOME%\bin;
```

注意，添加的内容前面有一个分号。实际上，Path 中各个值之间以分号来分隔。

最后，增加 CLASSPATH 环境变量。一般情况下，这个变量不存在，所以需要新建。设置其值为

```
.;%JAVA_HOME%\lib\tools.jar
```

设置环境变量后，需要测试 JVM 是否能正常工作。打开 DOS 窗口，输入如下命令。

```
java -version
```

这个命令将显示本地计算机上安装的 JDK 的版本。如果能正确显示版本信息，表明 JDK 安装成功，且环境变量的设置也是正确的。类似的，还可以输入“javac”等命令。如果系统显示找不到相应的命令，就说明环境变量设置不正确，需要仔细检查。

1.3　一个简单的 Java 应用程序

本书主要介绍两种类型的 Java 程序：一类是 Java 应用程序（Java Application），通过 Java 解释器控制，在命令行下使用 javac/java 命令来运行；另一类是 Java 小应用程序（Java Applet），或称为 Java 小程序。Java 小程序不能独立运行，而是被嵌入 Web 页面中，由 Java 兼容浏览器控制执行。如果系统中安装了 IDE 环境，则这两种类型的程序都可以在 IDE 下运行。

本书的前几章先介绍 Java 应用程序，在第 9 章介绍小应用程序。除非特别指明，本书中“程序”一词是指应用程序。

1.3.1　Java 应用程序示例

我们先从一个小的程序示例开始，了解程序的编制、运行过程。这个程序非常简单，只是在

屏幕上显示字符串“Hello World!”，包含的是最基本的语句。

使用文本编辑器编写 Java 程序，然后保存文件，源文件的文件扩展名为.java。在命令行下使用 javac 命令编译这个程序，如果程序正确，就会生成类文件，类文件的文件扩展名为.class。执行程序时，使用 java 命令。程序从创建到执行的过程如图 1-3 所示。

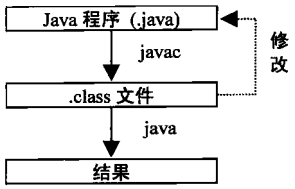


图 1-3　Java 程序的创建、执行过程

1. 程序的创建

可以使用文本编辑器来编写程序，如 Windows 系统中的记事本、写字板等。

在文本编辑器中输入程序 1-1 所示的内容（忽略第一列的数字序号），创建一个文件，命名为 HelloWorldApp.java。注意，保存文件时，不要带有任何排版信息。例如，使用记事本时，不要保存成文本文档的形式，因为这类文件的扩展名在默认情况下是.txt。

Java 是大小写敏感的语言，所以输入程序及为程序命名时都要注意字符的大小写。Java 是完全面向对象的语言，编写程序的过程就是定义类的过程。程序 1-1 中定义了一个公有类 HelloWorldApp，所以将文件命名为 HelloWorldApp.java，可以放到当前工作目录下。

程序 1-1　一个基本的 Java 应用程序。

```
1 //
2 // 简单的应用程序 HelloWorldApp
3 //
4 public class HelloWorldApp{
5   public static void main (String args[])
6   {  System.out.println ("Hello World!");
7   }
8 }
```

程序共有 8 行，前 3 行是注释行，对程序进行必要的解释和描述；后 5 行定义了一个公有类 HelloWorldApp，其中只有一个方法 main()。方法中只含有一条语句，功能是在屏幕上打印一个字符串。

程序很短，只包含了必要的语句。一个完整的程序大致包括以下内容。

- package 语句：语句个数不限，必须放在文件开始的地方。
- import 语句：语句个数不限，必须放在所有类定义之前。
- 类定义：类有公有类及非公有类。每个文件中公有类最多有一个，非公有类的个数没有限制。
- 接口定义：每个文件中包含的接口定义个数没有限制。

一个程序中必须至少有一个公有类，包含公有类的这个文件的文件名要与公有类的类名一致，所以每个文件中最多只能有一个公有类。如果一个程序中需要定义多个公有类，那么这些公有类必须分散放置在多个文件中。当然，一个文件中可以仅包含非公有类。

package 语句与 import 语句都是比较特殊的语句，必须放到所有类定义的前面，且 package 语句需要放到最前面。

例 1-1　语句序列示例。

正确的语句序列：

```
package Transportation;
import java.awt.Graphics;
import java.applet.Applet;
```

错误的语句序列：

```
import java.awt.Graphics;
import java.applet.Applet;
package Transportation;          //在包说明语句之前含有其他语句
```

错误的语句序列：

```
package Transportation;
package House;
import java.applet.Applet;       //含有两个包说明语句
```

2. 程序的编译

源文件保存后，使用命令行命令对程序进行编译。Java 的编译程序是 javac.exe。编译程序 1-1 的命令如下。

```
javac HelloWorldApp.java
```

如果编译器没有返回任何错误信息，则表示编译成功，源文件正确。编译后，在工作目录下会生成一个新文件 HelloWorldApp.class。这是二进制格式的字节码文件。

3. 程序的运行

编译后生成的类文件就可以执行了。Java 采用的是解释执行方式，它的解释器是 java，JVM 通过解释器解释执行类文件。

执行程序 1-1 的命令如下。

```
java HelloWorldApp
```

执行命令后，会在屏幕上看到一行信息，即“Hello World!”。

4. 程序的解释

下面逐行解释一下程序 1-1 的代码。

程序的前 3 行是注释行，注释是程序附加的内容，对程序的运行不起任何作用，只用来对程序进行解释，帮助读程序的人理解程序。

从第 4 行开始至第 8 行结束是程序的执行语句行，这几行定义了一个完整的公有类，类的名字是 HelloWorldApp。类的内容写在一对花括号中。

类中定义了一个方法 main()，从第 5 行开始到第 7 行结束。这个方法称为程序的主函数，是执行程序时的主入口，地位非常特殊。凡是应用程序都需要一个 main()方法。main()方法前面有 3 个修饰符，不能缺少，但次序可以有小的变化。例如，可以写成如下形式。

```
static public void
```

这 3 个修饰符各有独特的含义。public 表示方法 main()是公有方法，可以被任何方法访问。static 表示 main()方法是静态的，可用在类 HelloWorldApp 中，不需要通过该类的实例来调用。如果方法不是静态的，则必须先创建类的实例，然后才能调用实例的方法。有关类和实例的内容可以参看本书后面相关章节。void 指明 main()方法不返回任何值。这很重要，因为 Java 要进行严格的类型检查，包括对调用方法所返回的类型和方法说明的类型之间的检查。如果方法没有返回值，必须说明为 void，不可省略。如果方法有返回值，则以返回值类型替换 void。

main()方法中还有一个重要的量，即参数列表 args[]。它用来接受命令行参数，即动态传给程序的参数。运行程序时，命令行参数列在程序名之后，以空格为分界。系统将这些参数依次放到 args 数组中。该数组各元素是 String 类型的，它的名字可由程序员自己定义，如 myargs。数组的下标从 0 开始计数。

下面我们把程序 1-1 稍作修改，让它接受一个命令行参数，并将参数的内容打印出来。

程序 1-2 处理命令行参数。

```
//
// 简单的应用程序 HelloWorldApp
//
public class HelloWorldApp{
     public static void main (String args[]){
          System.out.println ("Hello World!");
          if (args.length!=0) System.out.println("Hello " + args[0]+ "!");
     }
}
```

执行程序 1-2 时，我们给定一个参数，相应的命令如下。

```
java HelloWorldApp earthman
```

屏幕上显示的结果如下。

```
Hello World!
Hello earthman!
```

这里我们只带了一个命令行参数，系统将这个参数放到 args[0]中。args.length 表示命令行参数的个数，运行程序时，系统自动判定命令行参数的个数，并进行赋值。本例中，args.length 的值为 1。

程序 1-1 中 main()方法内唯一的可执行语句是输出语句，将一个字符串显示在屏幕上。它调用的是系统标准输出流中的方法。其中，System 是系统包 java.lang 中的一个类，该类中有成员变量 out，这是标准输出流，用来显示信息；println 方法接受一个字符串参数，并将其输出到标准输出流中。这行语句反映了类名、对象名和方法调用之间的关系。

5. 程序的调试

程序编写后，难免会存在错误，往往都需要一个调试过程。如果编译时出现错误，则需要按照错误内容提示进一步修改程序，并重新进行编译。找出并修正程序中的错误，重新编译、运行，直到得到正确的结果。

编译阶段发现的错误称为编译时错误，运行阶段发现的错误称为运行时错误。一般来讲，编译时错误很多都是语法方面的问题，比如某个标识符拼写错误。而运行时错误更多的是逻辑方面的问题，比如除法运算时除数为零。当程序中有语法错误时，语法检查就可以提示程序员错误的位置及错误的类型，所以这类错误比较容易发现并修正。运行时错误不是简单的语法问题，在语法检查阶段往往发现不了，必须等到运行时才能发现。

1.3.2 使用 Java 核心 API 文档

JDK 文档中有许多 HTML 文件，这些是 JDK 配套的 API 文档，可使用浏览器查看。API 是 Sun 公司提供的使用 Java 语言开发的类集合，用来帮助程序员开发自己的类、Applet 和应用程序。程序员使用最多的应该是 Java 核心 API，此外还有 Java 商业 API、Java 服务器 API、Java 媒体 API、Java 管理 API、Java 嵌入式 API。

核心 API 文档是按层设计的，以主页方式提供给用户。主页按照链接列出包的所有内容。如果选定了一个具体的包，则页面中将列出作为包成员的所有内容。每个类对应为一个链接，选择这个链接将提供该类的信息页。每个类文档有相同的格式，不过，根据具体类的不同，有些内容项可能没有。Java 核心 API 中共有 32 个包，每个包中都有个数不等的类和接口，类及接口中又含有若干属性。API 文档中依次列出各类的相应内容。

Java 提供的内容非常多，读者不可能在读了一两本 Java 教科书之后就全部掌握。在实际编程

时，API 是不可缺少的工具。实际上，Java 正是因其丰富的 API 才获得了如此巨大的成功。

当要查看文档时，必须先从 Sun 公司的网站上下载文档文件，假设文件存放在 jdk 目录下，则在浏览器地址框中输入“jdk\docs\index.html”，即可查看 JDK 文档。

Java 核心 API 文档的初始页面如图 1-4 所示。

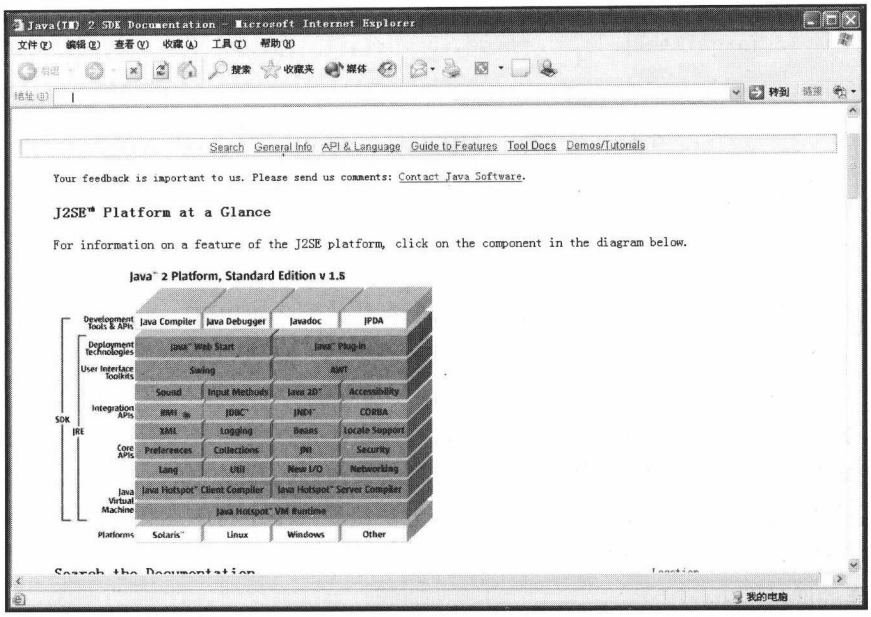


图 1-4　Java 核心 API 文档页面

在浏览文档的页面中，单击“API & Language”，进入 Overview 窗口，显示类文档的信息，如图 1-5 所示。

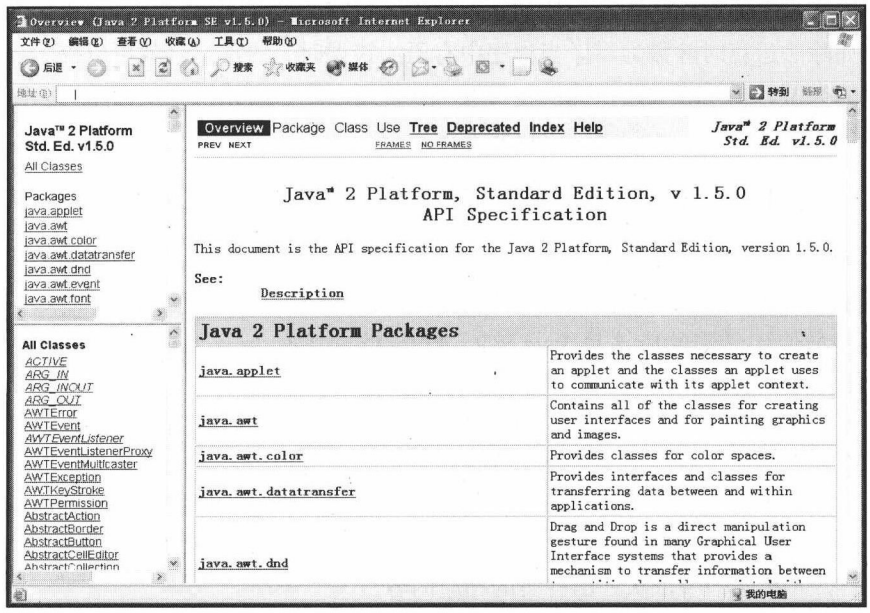


图 1-5　API 类文档

类文档中主要包括类层次结构、类及其一般目的的说明、成员变量表、构造函数表、方法表、变量详细说明表及每一个变量使用目的的详细描述、构造方法的详细说明及进一步的描述、方法的详细说明及进一步的描述。

读者必须要了解，有些类的许多工具不在该类的文档页中讲述，而是在其“父”类的文档中描述。例如，按钮（Button）派生于组件（Component）就是在类层次结构中说明的。要找到 Button 的 setColor()方法，必须查看 Component 类的文档。

类文档窗口分为 3 部分：左上部分显示 Java JDK 中提供的所有包的信息，选中某个包后，将在左下部分显示这个包中所有接口及类的信息。例如，选择查看 java.lang 包，左下窗口内将显示与这个包相关的内容。

如果想进一步查看包中的信息，如 Integer 类，选中 Integer，右侧窗口部分将显示 java.lang 中 Integer 类的所有接口及类的内容，向下拉动滚卷条，定位到所需的位置，就可以看到需要的内容了。

一般地，包页面中都会列有接口索引（Interface Index）、类索引（Class Index）、异常索引（Exception Index）、错误索引（Error Index）等。因各包内容不同，有的包中可能只列有其中的若干项。例如，java.applet 包中只列有接口索引和类索引，java.math 中只有类索引。如果想查看包中任一项的内容，则直接选中该链接，进入相应的页面即可。

总的来说，一个类中的信息包括以下几部分。

- Field Summary。
- Constructor Summary。
- Method Summary。
- Field Detail。
- Constructor Detail。
- Method Detail。

Field Summary 中列出类中成员变量的信息，包括名字、类型及含义。在 Field Detail 中将详细介绍这些成员变量。

Constructor Summary 中列出类构造方法的信息，包括参数列表并解释所创建的实例。构造方法的详细信息显示在 Constructor Detail 部分中。

在 Method Summary 中可以查找到要使用的方法名，在 Method Detail 中将详细介绍该方法的使用方法，包括调用参数表及返回值情况。

在 API 文档每个页面的最上一行有几个常用链接。其中，All Packages 将列出所有包的名字，Class Hierarchy 列出所有类之间的层次关系，Index 则将所有的变量和方法按字母顺序排列。使用变量和方法名的第一个字母可建立一级索引，该一级索引列在页面的顶端，点中一个字母后，将列出以该字母打头的所有变量和方法，可以按名查找。

习　题

1.1　概述 Java 语言的特点。

1.2　什么是 Java 虚拟机？它包括哪几部分？

1.3　简述 Java 开发环境的安装过程，并在自己的计算机上安装。

1.4 简述环境变量的设置方法，并在自己的计算机上正确设置。

1.5 简述 Java API 文档的安装过程，并在自己的计算机上安装。

1.6 练习使用浏览器查看 Java API 文档，以 java.lang 为例。

1.7 查阅 API 文档，列出 java.lang.Math 类的常用方法。

1.8 在自己的计算机上调试程序 1-1，直到正确为止。

1.9 解释如何从命令行中获取传递给主函数的值？

1.10 在自己的计算机上调试程序 1-2，直到正确为止。试着输入不同的命令行参数。

1.11 命令行参数的作用是什么？多个参数之间的分隔符是什么？

1.12 运行 Java 程序需要几步？各步的命令格式是什么？

1.13 设程序主函数的定义为

```
public static void main (String args[]){}
```

输入的命令行如下。

```
java myProg1 one two three
```

则参数 three 将在数组 args 的哪个元素中？