# 第 1 章　Java 概述


- Java 的诞生
- Java 的特点
- Java 开发环境安装与配置
- 创建并运行一个简单的 Java 程序

Java 语言是当今计算机软件行业中最热门的网络编程语言，以 Java 为核心的芯片技术、编译技术、数据库连接技术，以及基于企业级应用的 J2EE 技术得到了迅猛的发展。Java 的应用已经深入到我们生活的每一个角落。

## 1.1　Java 的诞生

Java 的发展历史，可追溯到 1990 年。当时 Sun 公司为了发展消费性电子产品而进行了一个名为 Green 的项目计划。该计划负责人是 James Gosling。起初他以 C++来写一种嵌入式程序，可以放在烤面包机或 PAD 等小型电子消费设备的芯片上，使得机器更智能。但他发现 C++并不适合完成这类任务！因为 C++有个缺点，就是 C++只能针对特定的操作系统和 CPU 芯片进行编译。这样一来，一旦电子设备更换了操作系统或 CPU 芯片就不能保证原程序正确运行，可能程序员需要修改程序并针对新的操作系统或 CPU 芯片重新进行编译。

为了解决遇到的问题，Gosling 决定要发展一种新的语言，来解决 C++的潜在性危险问题，这个语言名叫 Oak（Green 小组成员公司楼外有一棵橡树）。Oak 是一种可移植性语言，也就是一种平台独立语言，能够在各种操作系统、各种 CPU 芯片上运行。

1994 年，Oak 技术日趋成熟，这时网络正开始蓬勃发展。Oak 研发小组发现 Oak 很适合作为一种网络程序语言。因此发展了一个能与 Oak 配合的浏览器——WebRunner，后更名为 HotJava，它证明了 Oak 是一种能在网络上发展的程序语言。由于已经有一种计算机语言的名字叫 Oak，且被注册。最后，工程师们便想到以自己常享用的咖啡（Java）来重新命名。“Java”是印度尼西亚一个盛产咖啡的岛屿，中文译名为爪哇。

## 1.2　Java 的特点

Java 是一种简单的、面向对象的、分布式的、解释的、健壮的、结构中立的、与平台无关的、优异的、多线程的动态语言，也是目前使用最为广泛的网络编程语言之一。

### 1.2.1 简单性

Java 语言的语法与 C 语言和 C++语言很接近，使得大多数程序员很容易学习和使用 Java。另一方面，Java 丢弃了 C++中很少使用的、很难理解的、令人迷惑的那些特性，如操作符重载、多继承、自动的强制类型转换。特别地，Java 语言不使用指针，并提供了自动的垃圾收集，使得程序员不必为内存管理而担忧。

### 1.2.2 面向对象

面向对象是当前软件开发的重要方法。它是在编程过程中采用封装、继承、多态的编程方法。面向对象的概念和应用已超越了程序设计和软件开发，扩展到很宽的范围。如数据库系统、交互式界面、应用结构、应用平台、分布式系统、网络管理结构、CAD 技术、人工智能等领域。

Java 语言提供类、接口和继承等原语，为了简单起见，只支持类之间的单继承，但支持接口之间的多继承，并支持类与接口之间的实现机制（关键字为 implements）。Java 语言全面支持动态绑定，而 C++语言只对虚函数使用动态绑定。总之，Java 语言是一个纯的面向对象程序设计语言。

本书将在后续章节详细、准确地讨论面向对象的基本概念。

### 1.2.3 分布式

Java 语言是分布式的。Java 语言支持 Internet 应用的开发，在基本的 Java 应用编程接口中有一个网络应用编程接口（java net），它提供了用于网络应用编程的类库，包括 URL、URLConnection、Socket、ServerSocket 等。Java 的 RMI（远程方法调用）机制也是开发分布式应用的重要手段。

### 1.2.4 健壮性

Java 的强类型机制、异常处理、垃圾的自动收集等是 Java 程序健壮性的重要保证。对指针的丢弃是 Java 的明智选择。Java 的安全检查机制使得 Java 更具健壮性。

### 1.2.5 安全性

Java 通常被用在网络环境中，为此，Java 提供了一个安全机制以防恶意代码的攻击。除了 Java 语言具有的许多安全特性以外，Java 对通过网络下载的类具有一个安全防范机制（类 ClassLoader），如分配不同的名字空间以防替代本地的同名类、字节代码检查，并提供安全管理机制（类 SecurityManager）让 Java 应用设置安全哨兵。

### 1.2.6 平台无关

无论哪种语言编写的应用程序的运行都需要通过操作系统和处理器来完成。与平台无关是指程序代码不因操作系统、处理器的变化导致发生无法运行或出现运行错误。

Java 能运行于不同的平台，这是因为 Java 引进虚拟机原理，并运行于虚拟机，在不同平台的 Java 接口之间实现。使用 Java 编写的程序能在世界范围内共享。Java 的数据类型与机器

无关，Java 虚拟机（Java Virtual Machine）是建立在硬件和操作系统之上，实现 Java 二进制代码的解释执行功能，提供于不同平台的接口的。

### 1.2.7　多线程

多线程机制能够使应用程序在同一时间并行执行多项任务，而且相应的同步机制可以保证不同线程能够正确地共享数据。使用多线程，可以带来更好的交互能力和实时行为。

Java 语言支持多个线程的同时执行，并提供多线程之间的同步机制。

## 1.3　Java 开发环境安装与配置

对于初学者要进行 Java 的开发，必须首先搭建 Java 开发环境（运行平台）。目前，Java 2 平台有三个版本

### 1.3.1　平台介绍

（1）Java SE（以前版本称为 J2SE，Java 2 Platform Standard Edition）

——Java 标准版或 Java 标准平台。Java SE 提供了标准的 Java Development Kit（JDK）。利用该平台可以开发 Java 桌面应用程序和低端的服务器应用程序，也可以开发 Java Applet 程序，Applet 是嵌入在 HTML 文件中的 Java 程序。当前最新的 JDK 版本为 JDK 6.0。

（2）Java EE（以前版本称为 J2EE，Java 2 Platform Enterprise Edition）

——Java 企业版或 Java 企业平台。使用 Java EE 可以构建企业级服务应用，Java EE 建立于 Java SE 基础之上，增加了附加类库，具有 Web 服务、组建模型，以及通信 API 等特性，这些为面向服务的架构（SOA）以及 Web 2.0 应用开发提供了支持。

（3）Java ME（以前版本称为 J2ME，Java 2 Platform Micro Edition）

——Java 微型版或 Java 小型平台。是一个技术和规范的集合，它为移动设备（消费类产品、嵌入式设备、高级移动设备等）提供了基于 Java 环境的开发与应用平台。Java ME 目前分为两类配置，一类是面向小型移动设备的 CLDC（Connected Limited Device Profile），一类是面向功能更强大的移动设备，如智能手机和机顶盒，称为 CDC（Connected Device Profile）。

有关 Java SE、Java EE 和 Java ME 的详细内容，可登录官方网站 http://java.sun.com 或 http://www.oracle.com/technetwork/java/index.html 进行了解。

### 1.3.2　用 JDK 管理 Java

1．JDK 简介

JDK 是 Java Development Kit（Java 开发工具包）的缩写，是 Sun Microsystems 针对 Java 开发的产品。它为 Java 应用程序提供了基本的开发环境。自从 Java 推出以来，JDK 已经成为使用最广泛的 Java 软件开发工具包。JDK 也是 Java SE 平台提供学习 Java 语言的最佳平台，只有掌握了 Java SE 的 JDK 才能进一步学习 Java EE 和 Java ME。

一般初学 Java 时都选用 JDK 作为开发环境，而目前很多优秀的 Java 集成开发环境（IDE），如 Eclipse，NetBeans，JCreator 等，都是在 JDK 基础上建立的，换句话说，如果没有 JDK，其他集成开发环境是无法工作的。

Sun 公司的 JDK 是免费的工具，可以到 Sun 公司网站（http://java.sun.com）或提供相关下载的网站下载。不同的版本适合不同的操作系统，用户可以根据自己所用的操作系统下载相应的 JDK 版本。本书以 Windows 操作系统平台为例进行开发环境的搭建，下载的版本为 jdk-6u24-windows-i586.exe（JDK6.0）。

2. 安装 JDK

JDK6.0 的下载地址：

http://www.oracle.com/technetwork/java/javase/downloads/index.html

下载完后双击 jdk-6u24-windows-i586.exe 文件就可以进行开发工具的安装，安装过程非常简单。当出现选择安装路径界面时，如图 1-1 所示，建议读者将默认的安装路径 C:\Program Files\Java\jdk1.6.0_24 改为 D:\jdk1.6.0_24，这样做是为了便于今后环境变量的设置。


图 1-1　JDK 安装路径的选择

表 1-1 显示了 JDK 安装成功后，在 D:\jdk1.6.0_24 目录下形成的目录结构。

表 1-1　JDK 主要目录内容

| 目录 | 描述 |
|---|---|
| bin 子目录 | 存放开发工具和实用程序，如编译器 javac.exe 和解释器 java.exe |
| demo 子目录 | 存放 Java 平台的编程示例，含源码 |
| include 子目录 | 存放支持使用 Java 本机界面、JVM 工具界面以及 Java 平台的其他功能进行本机代码编程的头文件 |
| jre 子目录 | 存放 Java 运行环境包括 Java 虚拟机、类库以及其他支持执行以 Java 语言编写的程序的文件 |
| lib 子目录 | 存放开发工具所需的其他类库和支持文件 |
| sample 子目录 | 存放 Java API 编程样例，含源码 |

3. 配置 Path 和 ClassPath 环境变量

JDK 安装结束后，为了方便 Java 程序的编译和运行，还需要对其运行环境进行配置，即 Path 和 ClassPath 的设置。Path 的设置主要是为了能够在 DOS 命令行下找到 Java 编译与运行

所用的程序，如 javac.exe 和 java.exe 等；而 ClassPath 的设置主要是为了让 Java 虚拟机能够找到需要的类库。不同的操作系统设置方法略有差异，下面以 Windows 7 为例分别讲解 Path 和 ClassPath 的设置步骤。

（1）系统环境 Path 的设置。

在 Windows 7 操作系统中选择【控制面板】→【系统】→【高级系统设置】→【高级】→【环境变量】命令就可以设置环境变量 Path，如图 1-2 所示的【环境变量】对话框，在系统变量中找到 Path 变量，点击【编辑】按钮，在图 1-3 所示的对话框中对 Path 变量进行编辑或修改。

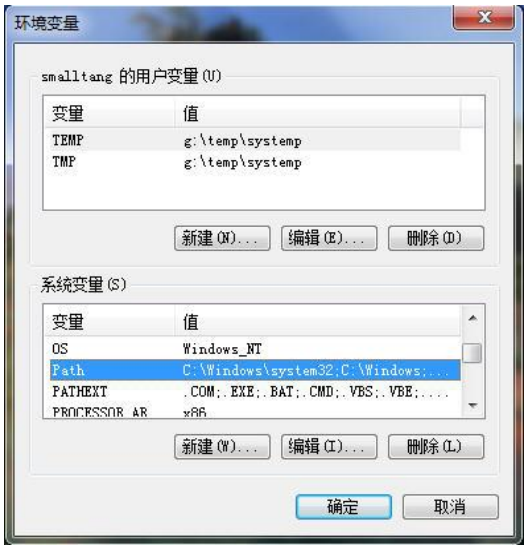


图 1-2　【环境变量】对话框


图 1-3　设置系统环境变量 Path

（2）系统环境 ClassPath 的设置。

在 Windows 7 操作系统中选择【控制面板】→【系统】→【高级系统设置】→【高级】→【环境变量】→【新建系统变量】命令就可以设置环境变量 ClassPath，如图 1-4 所示。


图 1-4　设置环境变量 ClassPath

# 1.4 创建并运行一个简单的 Java 程序

Java 程序分为三类，即 Application（Java 应用程序）、Java Applet （Java 小程序）和 Servlet（服务器端应用程序）。Java 应用程序可以在安装了 Java 标准平台的任何计算机上运行。Java Applet 不需要安装 Java 标准平台，只能嵌在 HTML 网页中由支持 Java 的浏览器直接运行。Servlet 是运行在服务器端的小程序，它可以处理客户传来的请求（request），然后对客户的请求给予响应（response）。本节以 Java 应用程序开发为例。

## 1.4.1 Java 程序开发步骤

开发一个 Java 应用程序分为三个步骤：

（1）创建一个带有文件扩展名*.java 的源文件。

1）使用编辑器（如记事本），输入如图 1-5 的例 1.1 中所示的 6 行文本。

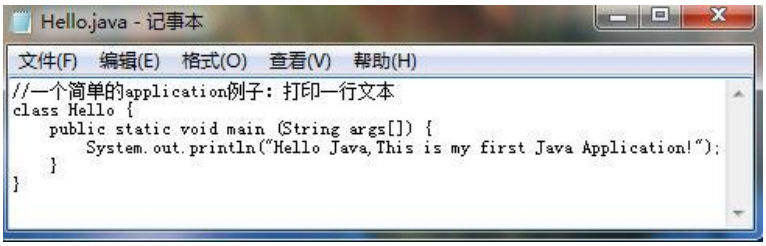


图 1-5 Hello.java 程序

2）把该文件命名为 Hello.java，并保存。为了方便起见，本章实例文件保存的目录为 E:\Java_pro。

（2）使用 Java 编译器 javac.exe 编译源文件生成一个带有文件扩展名*.class 的字节码文件。

1）打开 Windows 命令提示符。单击【开始】→【运行】命令，在文本框输入“cmd”命令后单击【确定】按钮进入 DOS 窗口。

2）使用 cd 命令将路径转入源文件“Hello.java”所在的路径（E:\Java_pro），输入“javac Hello.java”命令，并按回车键，如图 1-6 所示。


图 1-6 编译 Hello.java 源文件

（3）使用 Java 解释器 java.exe 运行字节码。

上一步中执行“javac Hello.java”命令后，如果没有任何提示信息，则表示编译成功，且生成一个字节码文件“Hello.class”。在当前路径下，输入“java Hello”命令，回车。运行结果如图 1-7 所示。


图 1-7　运行结果

### 1.4.2　一个简单 Java 程序的要素

【例 1.1】Hello.java

```
1    // 一个简单的 application 例子：打印一行文本
2    class Hello {
3        public static void main (String[] args) {
4       System.out.println("Hello Java,This is my first Java Application!");
5      }
6    }
```

第 1 行在程序中是注释行，编译器会忽略注释内容，注释可以加在程序中任何位置。注释的目的是为程序增加必要的解释，提高程序的可读性，适当给代码增加注释是一个良好的编程习惯。Java 支持三种注释方式：单行注释、多行注释和 Javadoc 文档注释。例 1.1 中的第 1 行使用符号“//”表示单行注释开始，该行从“//”后为注释内容。多行注释使用“/* … */”进行注释，以“/*”表示注释开始，“*/”表示注释结束。例如：

```
/*一个简单的 application 例子
功能：打印一行文本
*/
class Hello {    //声明类
        public static void main (String args[]) {
        System.out.println("Hello Java,This is my first Java Application!");
                //调用 println()方法，显示字符串
        }
}
```

Javadoc 文档注释以符号“/**”表示注释开始，“*/”表示注释结束。Javadoc 注释可以被 JDK 内置的 javadoc.exe 工具制作程序的开发文档。将例 1.1 代码进行如下修改，并保存。

```
/**
 *Simple Java Application
 *@author smalltang
```

```
 *@version 1.0 2011.2.1
 */
class Hello {
   public static void main (String args[]) {
    System.out.println("Hello Java,This is my first Java Application!");
   }
}
```

然后使用下列命令：

```
javadoc -d e:\html document hello.java
```

在 E:\html document 目录中得到 Hello 类的 HTML 文档，用浏览器打开 HTML 文档，效果如图 1-8 所示。

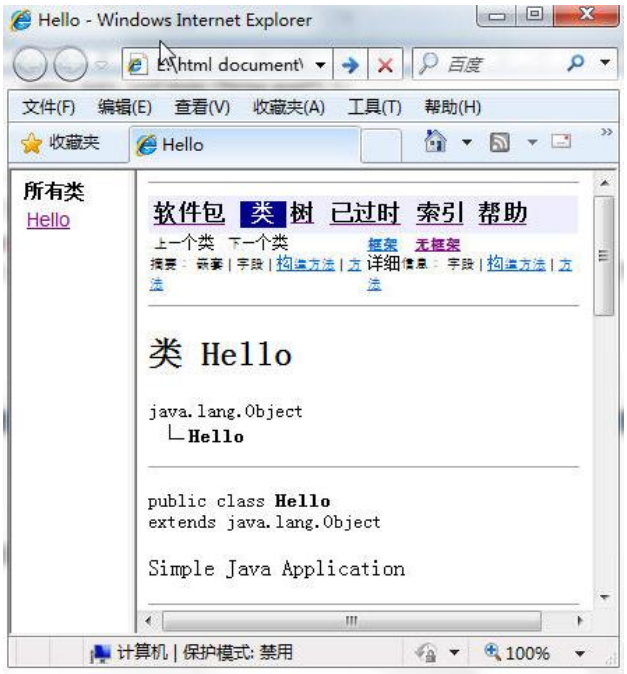


图 1-8　Hello 类的 Javadoc 网页

第 2 行声明类的名称 Hello。当源文件被编译时，根据该源文件中的类名会创建一个 Hello.class 文件。

第 3 行 main()方法是程序的执行的开始。

第 4 行程序调用 println()方法将字符串写入到标准输出流。

### 1.4.3　Java Applet 程序

Java Applet 就是用 Java 语言编写的一些小应用程序，它们可以直接嵌入到网页中，并能够产生特殊的效果。包含 Applet 的网页被称为 Java-Powered 页，可以称其为 Java 支持的网页。

【例 1.2】HelloApplet.java

```
import java.applet.*;    //引入 java.applet 包中的系统类
import java.awt.*;       //引入 java.awt 包中的系统类
```

```
public class HelloApplet extends Applet{
    public void paint(Graphics g){
        g.setColor(Color.red);              //设置字体颜色
        g.drawString("Hello Java!",2,20);   //在坐标(2,20)处输出字符串 Hello,java!
   }
}
```

由于 Applet 没有 main()方法作为程序的入口，它是由浏览器来运行，因此，需要编写一个 HTML 文件将该 Applet 的字节码文件嵌入其中，然后用支持 Java 的浏览器或 appletview 来运行。

在文本编辑器中，编写如下 HTML 代码：

```
<html>
    <head>Java Applet</head>
    <body>
          <applet>
               Code = "HelloApplet.class"
               Width = "500"
               Height = "300"
          </applet>
    </body>
</html>
```

把上述的 HTML 代码保存为“HelloApplet.html”文件，并和“HelloApplet.class”文件保存在同一目录下。否则，需要在 Applet 中增加 codebase 属性，来指定 Applet 的字节码文件所在的目录。操作命令如图 1-9 所示。运行结果如图 1-10 所示。

```
E:\Java_pro>javac HelloApplet.java

E:\Java_pro>appletviewer HelloApplet.html
```

图 1-9　运行 Applet


图 1-10　运行结果

## 1.5　Java 开发工具

当前，基于 Java 的程序开发工具比较常见的有几种：JDK+文本编辑器（Editplus）、JCreator、JBuilder 和 Eclipse。

1. JDK+文本编辑器

JDK+Editplus 的组合适合编写一些规模小的，独立的 Java 程序，通常不适合于中大型项目的开发，但作为 Java 的初学者应该了解这种模式，有助于了解 Java 程序运行的过程及其相关命令的使用。

2. JCreator

JCreator 是 Xinox 公司开发的一个可视化的 Java 集成开发环境（IDE）。JCreator 为用户提供了相当强大的功能，例如项目管理功能，项目模块功能，可以个性化设置语法高亮属性、行数、类浏览器、标签文档、多功能编译器，向导功能以及完全可自定义的用户界面。通过 JCreator 可以不用激活主文档而直接编译或运行 Java 程序。

JCreator 的设计接近于 Windows 界面风格，用户对它的界面比较熟悉。其最大特点是与所装的 JDK 完美结合，是其他任何一款 IDE 所不能比拟的。它是一种初学者很容易掌握的 Java 开发工具，缺点是只能进行简单的程序开发，不能进行企业 J2EE 的开发应用。

3. JBuilder

JBuilder 是 Borland 公司开发的 Java 集成开发环境（IDE）。它在 Eclipse 和 NetBeans 出现之前是非常流行的。一方面它的功能强大，另一方面 JBuilder 是当时唯一能够真正称得上 IDE 的产品。

4. Eclipse

Eclipse 是一个非常成功的开源项目。在 2001 年，IBM 为了对抗微软日益强大的垄断地位，将价值 4000 万美元的源代码 Eclipse 捐给开源组织，组建了 Eclipse 联盟，负责该工具的后续开发。在软件功能上，Eclipse 接近于 JBuilder，但其具有非常杰出的可扩展性。经过几年的发展，Eclipse 已经成为当前最流行的 Java IDE 和开发 Java 项目的首选 IDE。

## 1.6 小结

- Java 语言是当今流行的网络编程语言，特别适合于开发网络上的应用程序，具有面向对象、简单、平台无关、多线程等优秀特性。
- 面向对象编程的三大特性是封装、继承和多态。
- Java 程序主要包括 Application（Java 应用程序）、Java Applet（Java 小程序）和 Servlet（服务器端应用程序）三种。Java 应用程序的开发必须经过编写、编译、运行三个步骤。使用记事本等文本编辑工具进行程序代码的编写，使用 Java 开发工具集 JDK 提供的编译器进行编译，最后使用 Java 解释器解释运行。Java 虚拟机使 Java 应用程序实现了跨平台运行。


1．简述 Java 语言的特点。
2．J2SE、J2ME 与 J2EE 如何区别？
3．在计算机程序设计中，“可移植性”意味着什么？
4．编写运行 Java 程序需要经过哪些主要步骤？
5．编写并运行一个 Java 应用程序，在屏幕上输出“欢迎学习 Java！”。
6．编写并运行一个 Java Applet 程序，在浏览器中能显示“欢迎学习 Java！”。