第1章 Java概述

Java是20世纪90年代初问世的一种纯面向对象的计算机语言，在短短的几年内风靡全球，从嵌入式系统到网络编程，得到了空前广泛的应用。本章将简单介绍Java技术，主要内容包括：

- Internet、Web与Java渊源；
- 程序设计语言的层次发展；
- C、C++与Java；
- Java的特征；
- Java程序的类型及其不同编程模式；
- Java开发过程及开发工具。

1.1 Internet、Web与Java渊源

1.1.1 Internet与Web

从20世纪80年代末期开始，因特网（Internet）的发展引人注目。Internet是当前全球最大的、开放的、由众多网络相互连接而成的计算机网络，它采用TCP/IP协议簇，其前身是1969年问世的美国ARPANET网。Internet的迅猛发展始于20世纪90年代，所以20世纪90年代又被称为Internet时代，也称网络时代。

万维网（World Wide Web，WWW，又称为Web）是Internet上一项发展最快的网络多媒体信息服务，它作为Internet上新一代用户界面，摒弃了以往纯文本方式的信息交互手段，采用超文本（HyperText）方式。万维网包括WWW服务器和WWW浏览器。1989年，欧洲粒子物理研究所的科学家蒂姆·伯纳斯·李用两年多的时间开发出了超文本服务器程序代码，超文本服务器是存储超文本标记语言（HTML）文件的计算机。在1993年，出现了轰动世界的WWW浏览器，浏览器实际上就是用于网上浏览的应用程序，程序的主要作用是显示网页和解释脚本。WWW浏览器具有精良的图形用户界面、方便的菜单和按钮等，可以让用户在全球网络上轻松浏览WWW服务器上的信息。万维网是基于Internet的，它被广泛应用于Internet之上，主要方便了广大非网络专业人员对网络的使用，从而使万维网的站点数目以指数级增长。

1.1.2 Java的崛起

1991年，Sun Microsystem公司在公司内部投资了一个名为Green的研究项目，研究解决

诸如电视机、电冰箱、电话等家用电器的通信和控制问题。开发小组最初构想以当时颇为流行的C++语言进行此项智能软件的开发。由于C++语言的编译过程与硬件密切相关，平台不能独立，而且C++语言本身太复杂，安全性也差，所以最后不得不定义一套新的语言系统。

开发小组开发了一种以C和C++语言为基础的语言。它的创造者James Gosling根据他在Sun公司办公室外的一棵橡树（Oak），将这种语言命名为Oak语言。申请注册时，发现命名冲突，后来有人提议将该语言命名为Java。

Java语言可以称得上是一种精巧而安全的语言了。可是，Sun公司遇到的第一个问题是“智能化家用电器”市场的萧条。市场对智能型电子装置需求的上升率并不像预期的那样快。同时，Sun公司以它投标一个自认为比较适合的交互式电视项目时，败北而归。就在Green项目几乎走入绝境之时，1993年万维网空前流行起来，Sun公司发现了用Java向Web页中添加“动态内容”的潜在需求。1994年，Sun公司决定将Java语言用在WWW开发中，并取得了设计上的成功。Java出现以前，万维网只是文本和静态图形；Java出现以后，Web页变得有了声音、动画和交互性，而且不久又出现了视频图像和三维图像，Web页变“活”了。因为Java的平台独立性，使Java程序适应了Internet上多样化的服务器站点环境，Java程序既可以在Windows平台，也可以在Unix、Linux等平台上运行，体现了Sun公司宣传的“Write Once，Run Anywhere”（一次编写，随处运行）的跨平台特征。

Java能实现基于Internet和Intranet的广泛应用，将成为世界上重要的具有通用目的的一种编程语言。Java是Sun公司1995年6月发布的网络编程语言，当年就被美国著名杂志《PC Magazine》评为十大优秀科技产品之一。

命名为Java语言有两种说法，其一是印度尼西亚有一个重要的岛屿——爪哇岛，盛产咖啡，开发人员起名Java寓意为世人端上一杯热腾腾的咖啡；其二为美洲俚语——咖啡之意。

1.2 程序设计语言的层次发展

计算机程序设计的本质是把现实生活中遇到的问题，通过抽象后利用计算机语言转化为机器能够理解的层次，并最终利用机器来寻求问题的解。计算机程序设计语言的发展是一个不断演化的过程，从最开始的机器语言到汇编语言再到各种结构化高级语言，最后发展到支持面向对象技术的面向对象程序设计语言。

1. 机器语言

机器语言是第一代计算机语言，是最原始的编程语言，用二进制代码（0或1）书写，能被机器直接识别。机器语言是由计算机硬件设计定义的、与机器相关的，即某一种机器语言仅可用在某种特定型号的计算机上。由于机器语言是针对特定型号计算机的语言，故运算效率是所有语言中最高的。但机器语言远离人们习惯的思维方式，对于人类来说晦涩难懂，更难以记忆。在计算机发展初期，软件工程师们只能用机器语言来编写程序。这一阶段，人类的自然语言和计算机编程语言之间存在着巨大的鸿沟。

2. 汇编语言

汇编语言将一个特定指令的二进制串机器指令映射为简洁的英文助记符。例如，用“ADD”代表加法，用“MOV”代表数据传递等。汇编语言程序必须通过汇编语言的翻译软件（称为汇编程序）将程序员写的助记符直接转换为机器指令，才能在计算机中执行。它是

比机器语言“高层”的符号语言。

汇编语言同样十分依赖于机器硬件，移植性不好，但效率仍十分高，针对计算机特定硬件而编制的汇编语言程序，能准确发挥计算机硬件的功能和特长，程序精炼且质量高，所以至今仍是一种常用而强有力的软件开发工具。汇编语言与人类自然语言间的鸿沟略有缩小，但一般用户掌握和使用这种语言仍比较困难。

机器语言和汇编语言与机器的 CPU 有关，称为低级程序设计语言。

3. 高级语言

1954 年，第一个完全脱离机器硬件的高级语言——Fortran 语言问世。几十年来，共有几百种高级语言出现，有重要意义的有几十种。高级语言的发展也经历了从早期语言到结构化程序设计语言，从面向过程到面向对象程序语言的历程。1970 年，第一个结构化程序设计语言——Pascal 语言出现，标志着结构化程序设计时期的开始。从 20 世纪 80 年代初期开始，在软件设计思想上，又产生了一次革命，其成果就是面向对象的程序设计。

高级语言是一种面向用户的语言，使用一条简单的高级语言语句就可以完成由许多条汇编语句才能完成的任务。将高级语言转化为机器语言是通过编译器的编译程序实现的。高级语言是采用命令或语句的语言，屏蔽了机器的细节问题，提高了语言的抽象层次，比汇编语言更加接近人们的思维方式。这种语言易学、易用、易于理解。

1.3 Java 与 C、C++

随着 C、Pascal 和 Fortran 等结构化高级语言的诞生，程序员可以逐渐离开机器层次，在更抽象的层次上表达意图。随着程序规模的不断扩大，在 20 世纪 60 年代末期出现了软件危机，当时的程序设计方法都无法克服错误随着代码的扩大而级数般地扩大的问题，这个时候就出现了一种新的程序设计方法——面向对象程序设计。

1.3.1 Java 与 C++

Sun 公司的 Java 开发小组汲取了 C++的精华，并将其组合到 Java 中，舍弃了 C++中低效率和不便于程序员使用的特性。Java 开发小组也创造了一些新的特性，赋予 Java 在开发基于 Internet 的应用程序时所必须的动态性。

Java 的目的并不是改进 C++并最终取代 C++。C++和 Java 这两种语言是设计用来解决不同问题的。Java 是用来设计必须共存于不同机器的应用程序——常常是基于 Internet 的基础之上。相反，C++用来开发在一台特定机器上运行的程序，尽管 C++程序被重新编译后能够在其他机器上运行。

Java 语言的许多基本结构与 C++是相似的，有时甚至是相同的。例如，Java 是一种面向对象编程语言，它用类来创建对象的实例。类具有数据成员和方法成员，这和 C++中的类是相似的。

Java 没有指针，但是在 C/C++编程语言中指针是一个基石。在 C++中正确使用指针能使程序富有效率，但是指针难以掌握，如果使用不当会导致运行错误。

Java 带有自动的垃圾收集器，这是在 C/C++中没有的功能。垃圾收集器是一个常规程序，收集程序中不再使用的内存，程序员不必编写代码来释放之前使用的内存。

在不同的平台上使用 C/C++程序使系统会对每种数据类型依平台的不同分配不同的字节数。而在 Java 中，会为各种数据类型分配合理的固定位数，在每种平台上都不改变，这样便保证了 Java 的平台无关性。

C++中支持多重继承，一个类可以有多个父类，这种方式使 C++中的类可以使用多个父类的属性和方法，但结构特别混乱。而在 Java 中，一个类只能有一个父类，但是可以实现多个接口，这样既达到多重继承的目的，又保证了结构比多重继承更加清晰。

除此之外，Java 与 C++的不同还表现在，Java 中不支持结构和联合、不支持宏定义、不支持头文件、不支持友元，大大保证了 Java 程序的安全性。

1.3.2 Java 与 C

C 语言为面向过程的程序设计语言，C 语言中程序设计的单元是函数。C 编程人员着重于编写函数。执行同一任务的一系列动作构成函数，一系列函数再构成程序。这种语言的主要问题是程序中的数据和操作分离，不能够有效地组成与自然界中的具体事物紧密对应的程序成分。

Java 是纯面向对象的程序设计语言，Java 语言中程序设计的单元是类，从类中创建一个一个实例对象。Java 编程人员着重创建用户自定义的类。每一个类均可包含数据属性和若干操作数据的函数。一个类的函数部分称为方法。

Windows 下 C 语言开发过程如图 1-1 所示。

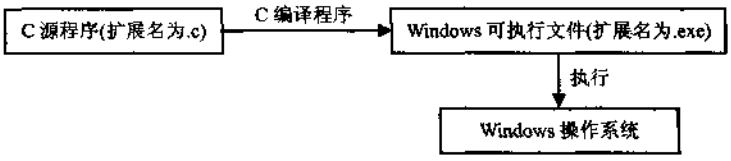

图 1-1 Windows 下 C 语言开发过程

Java 语言开发过程如图 1-2 所示。

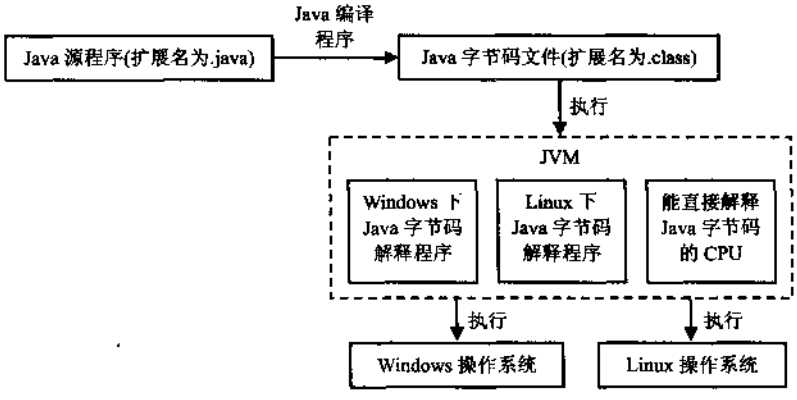

图 1-2 Java 语言开发过程

从图 1-1 与图 1-2 的比较中可以看出，Java 源程序编译后生成的字节码文件就相当于 C 源程序编译后 Windows 上的 exe 可执行文件，JVM（Java Virtual Machine，Java 虚拟机）的作用类似 Windows 操作系统。在 Windows 上运行的是 exe 文件，在 JVM 上运行的是 Java 字

节码文件，即编译后生成的后缀为.class 的文件。

Windows 执行 exe 可执行文件的过程，就是从 exe 文件中取出一条条的计算机指令，交给 CPU 去解释执行。JVM 执行 Java 字节码文件的过程，也是 JVM 从 Java 字节码文件中取出一条条的字节码指令交给“CPU”去执行。执行 Java 字节码的“CPU”可以是硬件，也可以是某个系统上运行的一个软件，这个软件称为 Java 字节码解释程序，也就是 Java 虚拟机。可见，只要能实现特定平台下的解释器程序，Java 字节码就能通过解释器程序在该平台上运行，这是 Java 跨平台的根本。

Java 兼顾解释性与编译性语言的特点，Java 源文件转换成 class 字节码文件过程是编译型的，class 字节码文件在操作系统上运行的过程则是解释型的，Java 虚拟机充当了解释器的作用。

C/C++都是编译型的语言，运行速度较快。而 Java 刚出现的时候，执行速度要比 C 慢几十倍。尽管经过很多的改进和优化后，Java 的解释执行速度已经有了很大的提高，但和 C 仍无法同日而语。解释执行速度慢的问题是影响 Java 在更大范围内发展的致命问题。

1.4 Java 是什么

Java 是由 Sun 公司开发的新一代纯面向对象的网络编程语言。其目标是建立一种在任意一种机器、任意一种操作系统的网络环境中运行的软件，实现所谓的“程序写一次，到处运行”的目标。正因为如此，Java 已成为当今 Internet 上最流行、最受欢迎的一种程序开发语言。

Java 是产业。“Java 语言的出现，将会引起一场软件革命”。这是因为传统的软件往往都是与具体的实现环境有关，换一个环境就需要重新做一次变动，造成人力和财力的浪费，而 Java 语言在执行码（二进制码）上兼容。目前计算机产业的许多大公司购买了 Java 语言使用许可证，Java 语言已经得到工业界的广泛认可，其中包括 IBM、Apple、Adobe、Microsoft 等公司都拥有了 Java 的使用许可证。众多软件开发商开始支持 Java 语言的软件产品。今日世界是以网络为中心的计算机时代，不支持 HTML 和 Java 语言，应用程序的应用范围只能局限于同质的环境中。

Java 已经逐步从一种单纯的计算机高级语言发展成为一种重要的 Internet 平台，并进而引发、带动了 Java 产业的发展和壮大，成为当今计算机业界不可忽视的力量和重要的发展潮流与方向。

Java 开发小组把 Java 按特性分为基本版、移动版和企业版三个版本，每个版本有一个软件开发包（Software Development Kit，SDK）。Java 的基本版本叫 Java 2 标准版（Java 2 Standard Edition，J2SE）它包含建立 Java 应用程序或者 Applet 所需的应用程序编程接口（API）。Java 2 移动版（The Java 2 Mobile Edition，J2ME）包含创建无线 Java 应用程序的 API。还有 Java 2 企业版（The Java 2 Enterprise，J2EE）是 J2SE 的增强版本，包含建立多层架构应用程序 API。

1.5 Java 语言的特点

Java 语言是由 C++语言发展而来的，是一种彻底的面向对象的程序设计语言。作为一种纯面向对象的程序设计语言，它非常适合大型软件的开发。Java 语言去掉了 C++语言的一些

容易引起错误的特性。例如：Java 语言没有指针，避免了使用指针直接访问物理寄存器带来的风险；Java 取消了运算符重载；Java 语言用接口代替了 C++语言中容易引起混乱的多重继承机制等。Java 语言具有如下特点。

1. 面向对象

对象是程序的基本单元和构件。在面向对象的程序语言中，对象是类的实例，而类则是描述对象的模板。类是具有相同属性和服务的一组对象的抽象、一般描述。抽象是事物的泛化，抽象的目的是提取重要的特征而忽略不重要的细节。对象是现实世界中某一个实际存在的事物，软件对象是数据和方法的封装体。类与对象的关系，如同一个模具与用这个模具铸造出来的铸件之间的关系，如同自行车图纸和自行车的关系。

封装是面向对象的一个重要原则。它有两个含义，第一个含义是，把对象的全部属性和全部服务结合在一起，形成一个不可分割的独立单位（即对象）。第二个含义也称作“信息隐蔽”，即尽可能隐蔽对象的内部细节，对外形成一个边界（或者形成一道屏障），只保留有限的对外接口使之与外部发生联系。这主要是指对象的外部不能直接存取对象的属性，只能通过几个允许外部使用的服务（或称方法）与对象发生联系。

2. 跨平台

用 Java 语言写的程序，经过 Java 编译器编译后生成 Java 语言特有的字节码（Bytecode），而不生成特定的 CPU 机器代码。Java 字节代码运行在 Java 虚拟机（JVM）上。Java 语言借助 JVM，首先对 Java 编译后生成的字节码进行解释，虚拟机底层的运行系统把字节代码转化成实际的硬件调用，再执行它。JVM 是一种抽象机器，它附着在具体的操作系统之上，本身具有一套虚拟机器指令，并有自己的栈、寄存器等。JVM 类似一个小巧而高效的 CPU，是一个应用程序仿真的软件实现。JVM 通常不是在硬件上实现（目前，Sun 公司已经设计实现了 Java 芯片，主要使用在网络计算机上。另外，Java 芯片的出现也会使 Java 更容易嵌入到家用电器中）。JVM 与硬件、软件平台有关，它使 Java 语言程序在某一特定硬件、软件平台环境中直接运行目标代码指令。Java 编译出来的字节码与平台无关，这一点正是网络传输所需要的。

Java 主要靠 JVM 在目标代码级实现平台无关性。JVM 是 Java 平台无关的基础，Java 源代码先经过 Java 编译器生成 JVM 的字节码，再经过 Java 解释器将字节码转换成实际系统平台上的机器码，然后真正执行。任何一台机器只要配备了 Java 解释器，就可以运行字节码，而不管这种字节码是在何种平台上生成的。另外，Java 采用基于 IEEE 标准的数据类型。通过 JVM 保证数据类型的一致性，也就确保了 Java 的平台无关性。

3. 安全性

Java 将重点用于网络/分布式运算环境，确保建立无病毒且不会被病毒侵入的系统。内存分配及布局由 Java 运行系统决定，字节码验证，可以轻松构建出防病毒、防黑客的系统。

Java 最初的设计目的是应用于电子类消费产品，要求有较高的可靠性。Java 虽然源于 C++，但它消除了很多 C++中的不可靠因素，可以防止很多编程错误。首先，Java 是强类型的语言，要求显式的方法声明，保证了编译器可以发现方法调用错误，保证程序更加可靠；其次，Java 不支持指针，杜绝了内存的非法访问；第三，Java 的自动单元收集防止了内存丢失等动态内存分配的问题；第四，Java 解释器运行时实施检查，可以发现数组和字符串访问的越界；第五，Java 提供了异常处理机制，便于程序即时发现运行错误。

由于 Java 主要用于网络应用程序开发，因此对安全性有着较高的要求。如果没有安全保

证，用户从网络下载程序执行就非常危险。

4. 多线程

线程是操作系统的一种概念，被称为轻量级进程，是比传统进程更小的、可并发执行的单位。C 和 C++采用单线程体系结构，Java 提供了多线程支持。

一个线程是一个程序内部的顺序控制流。在 DOS 环境下只能同时运行一个程序，也就是程序只有一条顺序控制流；即一部分程序因为某种原因不能执行下去的时候，整个程序就停止在那里，其他的操作就不能执行。进程的特点是每个进程都有独立的代码和数据空间，进程切换的开销大。线程是轻量的进程，同一类线程共享代码和数据空间，每个线程有独立的运行栈和程序计数器，线程切换的开销小。通常，线程之间的切换是非常迅速的，使人们觉得好像所有的线程都是在同时执行似的。但是在系统内部来看，线程仍是串行执行的，只不过由于操作系统可以快速、自动地进行切换，从而给人一种并发执行的感觉。

多进程是指在操作系统中，能同时运行多个任务（程序）。多线程是指在同一应用程序中，有多个顺序流同时执行。如果多进程是指在操作系统中可以同时运行多个任务（程序）的话，那么，多线程就是指在同一应用程序（进程）中可以有多个任务（顺序流）同时执行。多线程的优点是具有更好的交互性以及实时行为。Java 提供了一个类 Thread，由它负责启动运行，终止线程，并可检查线程状态。

5. 图形功能强

Applet 是 Java 特有的一种小应用。Java 系统可以使 Applet 很方便地加入到 Internet 的网页之中，从而使 Internet 网页增加了各种动态的多媒体图形效果，增强了可视化的互动对话，对计算机图形学、计算机多媒体通信提供了良好的支持。

1.6 Java 程序的类型及其不同的编程模式

用 Java 书写的程序有两种类型：Java 应用程序（Java Application）和 Java 小应用程序（Java Applet）。

Java 应用程序必须得到 Java 虚拟机的支持才能够运行。Java 小应用程序则需要客户端浏览器的支持。Java 小应用程序运行之前必须先将其嵌入 HTML 文件的<applet> 和</applet>标记中。当用户浏览该 HTML 页面时，Java 小应用程序将从服务器端下载到客户端，进而被执行。

Application 的基本编程模式：

```
class 用户自定义的类名  // 定义类
{
    public static void main(String args[ ])    //定义 main( )方法
     {
        方法体
     }
}
```

Applet 的基本编程模式：

```
import java.awt.Graphics;                    //引入 java.awt 系统包中的 Graphics 类
import java.applet.Applet;                   //引入 java.applet 系统包中的 Applet 类
class  用户自定义的类名  extends Applet    //定义类
```

```
{
  public void paint(Graphics g)                    //调用 Applet 类的 paint( ) 方法
    {
      方法体
    }
}
```

Applet 需要的 HTML 文件的最小集的格式为：

```
<HTML>
  <applet code=类名.class    width=宽度    height=高度>
  </applet>
</THML>
```

注意：HTML 标记包含在尖括号内，并且总是成对出现，前面加斜杠表明标记结束。<HTML>和</THML>来标记 HTML 文件的开始和结束，用<applet>和</applet>标记 applet 的开始和结束。必须把以.class 结尾的字节码文件名嵌入到 HTML 文件中。HTML 文件应和字节码文件放在同一目录下。另外，HTML 对字符大小写是不敏感的。参数值可加引号也可不加。

综上所述，Applet 和 Application 是 Java 程序的两种基本类型，从源代码的角度来看，Applet 和 Application 有以下两个基本的不同点。

① 一个 Applet 类必须定义一个从 Applet 类派生的类；Application 则没有这个必要。

② 一个 Application 必须定义一个包含 main 的方法，以控制它的执行，即程序的入口；Applet 不会用到 main 方法，它的执行是由 Applet 类中的几个系统方法来控制的。

两者共同之处是：编程语法是完全一样的。

1.7 Java 程序开发过程

1.7.1 开发过程简介

要编写和运行第一个 Java 程序，需要有文本编辑器和 Java 开发平台。文本编辑器可以使用 DOS 操作系统提供的 Edit 或 Windows 环境中的记事本，用 J2SDK（Java 2 Software Development Kit）作为开发平台，J2SDK 往往习惯称为 JDK。也可以在安装好 JDK 之后，下载 TextPad 作为编辑和运行平台。通常初学者使用 Windows 环境中的记事本作为创建源文件的文件编辑器。

要创建一个 Java 需要三个基本步骤。

（1）创建带有文件扩展名.java 的源文件。

（2）利用 Java 编译器生成文件扩展名为.class 的字节码文件。

（3）Application 程序利用 Java 解释器运行该字节码文件，Applet 利用 Java 自带查看器或浏览器运行嵌入字节码文件的 HTML 文件。

注意：保存文件时一定要用 Public 的类名作为文件名，记事本默认的扩展名是 txt，所以源文件的文件扩展名必须修改为.java，并在文件名的开始和扩展名的结尾处加上一对双引号后保存；或者不加双引号，保存类型选择 all files。

Java 编译器是 JDK 中的 javac.exe，将 Java 源程序编译成字节码文件。使用语法：javac 类名.java 按回车键即可。如果源程序没有错误，则屏幕上没有输出，否则将显示出错信息。

Java 解释器是 JDK 中的 java.exe，解释和执行 Java 应用程序。使用语法： java 类名 按回车即可。Java 的平台无关性就是因为每一种计算机上都安装了一个合适的解释器，将不同计算机上的系统差别隐藏起来，使字节码面对一个相同的运行环境，实现了“程序写一次，到处运行”的目标。Application 程序脱离浏览器，通过解释器单独来运行。

对于 Applet 程序来说，需要 HTML 文件的配合。

使用语法： appletviewer HTML 文件名.html 按回车即可。字节码文件嵌入 HTML 文件中，appletviewer 为 Applet 查看器（JDK 中的 appletviewer.exe），含有内置 Java 解释器。appletviewer 又称小浏览器，它仅显示相关 Applet 的属性，初学者使用方便。

1.7.2 创建 Java Application 程序示例

编写一个 Java Application 程序，过程简要描述如下：

第一，用户需要下载和安装 J2SDK（JDK）。以 J2SDK1.3.1_01 版本为例。暂且把源文件放置在 J2SDK 的 bin 目录之下自己创建的 code 文件夹中。

第二，确定文本编辑器。在本例中，使用记事本。以 Windows 2000 为例，单击“开始”→“程序”→“附件”→“记事本”，打开“记事本”文本编辑器。当然，用户也可以选择其他文本编辑器。

【例 1.1】 实现第一个简单的应用程序：打印一行文字。

（1）在“记事本”中编写如下源程序：

```
// 文件名: Welcome.java
public class Welcome {
  public static void main( String args[] )
  {
     System.out.println( "Welcome to Java Programming!");
  }                  //结束 main 方法的定义
}                    //结束类 Welcome 的定义
```

（2）语法说明

程序中的“//”是单行注释符，只对当前行有效，表示该行后面文字是注释行。程序人员在程序中加入注释，用于提高程序的可读性，使程序便于阅读和理解。在执行程序时不会执行注释行。注释会被 Java 编译器所忽略。多行注释用“/*”开始，以“*/”结束。

Java 程序是由类或类的定义组成。类构成了 Java 程序的基本程序单位。创建一个类是 Java 程序的首要工作，而且类名要和所使用的文件名完全一样，包括字母的大小写。Java 用关键字 class 标志一个类定义的开始，class 前面的 public 关键字代表该类的访问属性是公共的，表示这个类在所有场合中可使用。一个程序文件中可以声明多个类，但仅允许有一个公共的类，程序文件名要与公共类的名称相同。class 后面是该类的类名，在本例中是 Welcome。

Application 中有一个显著标记就是必须定义一个 main()主方法，而且应该按照例 1.1 中所示的那样来定义其修饰符和命令行参数，用关键字说明它是 public，静态的 static，无返回值的 void，主方法的参数是字符串类型 String 的数组 args[]。一个类中可以声明多个方法，Java

应用程序自动从 main 主方法开始运行，通过主方法再调用其他的方法。Java 语言的每条语句都必须用分号结束。

System.out 是标准输出对象。它用于在 Java 应用程序执行的过程中向命令窗口显示字符串和其他类型的信息。方法 System.out.println 在命令窗口中显示一行文字后，会自动将光标位置移到下一行（与在文本编辑器中按 Enter 键类似）。

（3）编译运行程序

打开一个“命令提示符”窗口如图 1-3 所示。

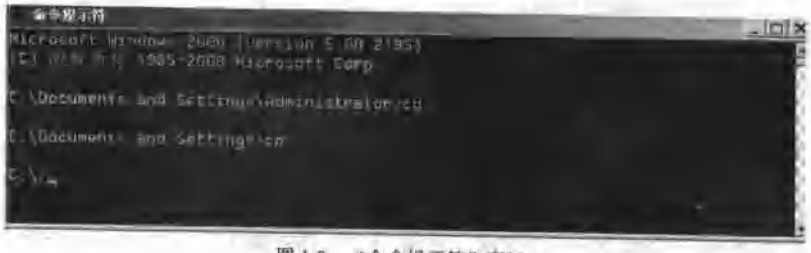

图 1-3 “命令提示符”窗口

接着，进入程序所存储的目录（假定应用程序存放在 c:\jdk1.3.1_01\bin\code 之下），在“命令提示窗口”中键入 dir 命令，显示文件如图 1-4 所示。

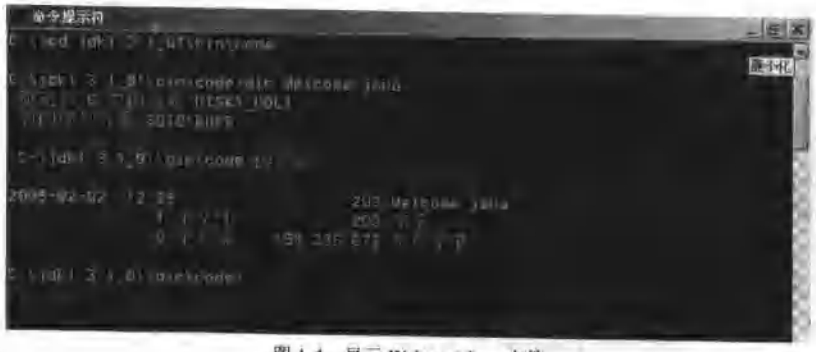

图 1-4 显示 Welcome.java 文件

再在“命令提示符”窗口中键入 javac Welcome .java，如图 1-5 所示。

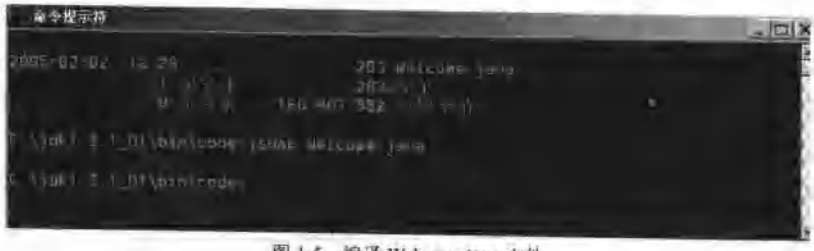

图 1-5 编译 Welcome.java 文件

如果此程序不含语法错误提示，那么，将生成一个 Welcome.class 文件，自动保存在源文件同级目录下，此文件含有表示该程序的 Java 字节代码。

再在“命令提示符”中键入 dir 命令，就可以看到编译后生成的.class 文件，表明程序编译成功，如图 1-6 所示。

图 1-6　显示编译后的情况

运行字节码文件程序，键入：java　Welcome

此命令启动 Java 解释器，载入“.class”文件，字节代码被 Java 解释器解释执行。解释命令省略了.class 文件扩展名，否则解释器不执行。解释器自动调用方法 main，然后通过 System.out.println 方法显示 “Welcome to Java Programming!”，如图 1-7 所示。

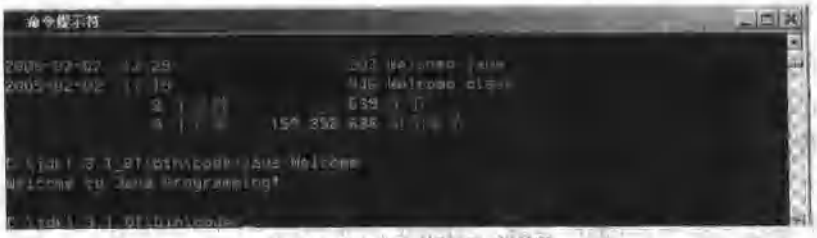

图 1-7　运行程序并显示运行结果

注意：如果在安装 Java 时没有另外指定 JDK 安装目录，则 javac.exe 和 java.exe 被存放在 c:\jdk1.3.1_01\bin 目录之下（以 JDK 1.3.1_01 版本为例）。如果想在任何目录下都能使用编译器和解释器，应在 dos 提示符下运行命令：C:\>path c:\jdk1.3.1_01\bin 或将 path c:\jdk1.3.1_01\bin 放到 autoexe.bat 文件中。 path 环境变量指定操作系统应到什么地方查找 Java 工具。

1.7.3　创建 Java Applet 程序示例

Java 程序分为 Application 应用程序 和 Applet 小应用程序。Java Applet 作为实现动态的、交互式网页功能的编程工具，在当今网络世界中扮演着重要的角色。Applet 是一种嵌入到 HTML 文件当中的 Java 程序，可以通过网络下载来运行。HTML 是超文本标记语言，它采用一整套标记来定义 Web 页。一个 HTML 文件可定义一个 Web 页，文件扩展名为.html 或 .htm。与从命令窗口执行 Java 应用程序不同，Applet 通过 JDK 的查看器 appletvierer 或 WWW 浏览器运行。

【例 1.2】　显示一行字符串的简单 Java Applet。

（1）在记事本中编写源代码

```
// 文件名: WelcomeApplet.java
// A first applet in Java
import javax.swing.JApplet;          // 加载系统类 JApplet
import java.awt.Graphics;            // 加载系统类 Graphics

public class WelcomeApplet extends JApplet {
    public void paint( Graphics g )
    {
```

```
        g.drawString( "Welcome to Java Programming!", 25, 25 );
    }                                   //结束 paint 方法的定义
}                                       //结束类 WelcomeApplet 的定义
```

（2）语法说明

"//"表示单行注释。Java 含有许多预定义的类或数据类型，这些类被归入 Java API（Java 应用程序编程接口，Java 类库）的各个包中。程序中使用 import 语句引入系统预定义类。程序中两行加载语句告诉编译器 JApplet 类的位置在 javax.swing 包中，Graphics 类的位置在 java.awt 包中。当创建一个 Applet 小应用程序时，要加载 JApplet 类或 Applet 类。加载 Graphics 类为的是使程序能够画图（如线、矩形、椭圆和字符串）。

注意：java.applet 中有一个传统的 Applet 类，它没有包括在 Java 最新的 GUI 构件 javax.swing 包中。

Java API 中的所有包存放在 java 目录或 javax 目录中，这两个目录下还有许多子目录，包括 awt 目录和 swing 目录。注意：在磁盘上找不到这些目录，因为它们都存储在一个称为 JAR 的特殊的压缩文件中。在 J2SDK 安装结构中有一个名为 rt.jar 的文件，该文件包括了 Java API 里所有.class 文件。

与应用程序一样，每一个 Java Applet 至少由一个类定义组成。但是，用户几乎不必"从头开始"定义一个类。因为 Java 提供继承机制，使用户可以在已存在的类的基础上创建一个新类。程序中通过关键字 extends 实现。extends 前面为用户自定义的类，作为派生类或子类，extends 后边的类名为被继承的类，称为基类、父类或超类，如此程序中的系统类 JApplet。通过继承建立的新类具有其父类的属性（数据）和行为（方法），同时增加了新功能（尤其是在屏幕上显示 Welcome to Java Progrmming!的能力）。

实际上，一个 Applet 小应用程序需要定义 200 多个不同的方法，而在上边的程序中，我们只定义了一个 paint 方法。如果非得定义 200 多个方法，仅仅为了显示一句话，我们可能永远无法完成一个 Applet。使用 extends 继承 JApplet 类，这样 JApplet 的所有方法就已成为 WelcomeApplet 的一部分。使用继承机制，程序员不必知道所继承的基类的每一个细节，只需知道 JApplet 类具有创建一个 Applet 的能力即可。

注意：学习 Java 语言，一方面是学习用 Java 语言编写自己所需的类和方法，另一方面是学习如何利用 Java 类库中的类和方法。这样，有助于确保不会重复定义已提供的功能。

程序中重写了父类 JApplet 的 paint()方法，其中参数 g 为 Graphics 类的对象。在 paint()方法中，通过用 Graphics 对象 g 后的点操作符（.）和方法名 drawString 来调用 drawString()方法，在坐标（25，25）窗口处输出字符串，其中坐标是以像素点为单位，第一个坐标为 x 的坐标，它表示距离 Applet 框架左边界的像素个数；第二个坐标为 y 坐标，它表示距离 Applet 框架上边界的像素个数。Applet 程序没有 main( )方法是 Applet 与应用程序 Application 的区别之一。JApplet 类的方法 paint 在缺省情况下，不做任何事情。WelcomeApplet 类覆盖了或重新定义了这个行为，以使 appletviewer 或浏览器调用 paint 方法，在屏幕上显示一行字符串。

（3）用记事本编写与例 1.2 Java 源文件配合的 HTML 文件

```
<html>
<applet code="WelcomeApplet.class" width=400 height=50>
</applet>
</html>
```

HTML 标记是用尖括号括起来，成对出现。加斜杠表明标记结束。用<html>和</html>标记 HTML 文件的开始和结束，用<applet>和</applet>标记 Applet 的开始和结束。<applet>包含三个必需的参数：

- code：表示要打开的 Applet 字节码文件名。
- width：表示 Applet 所占用浏览器页面的宽度，以像素点为单位。
- height：表示 Applet 所占用浏览器页面的高度，以像素点为单位。

一般情况下，字节码文件和 HTML 文件处于同一目录下。否则字节码文件的路径要在 code 中给出。

（4）编译运行

① 编译 Java 源文件和例 1.1 一样，使用 javac 命令：

javac WelcomeApplet.java，如图 1-8 所示。

② 运行时使用如下命令格式：

appletviewer WelcomeApplet.html，如图 1-8 所示。

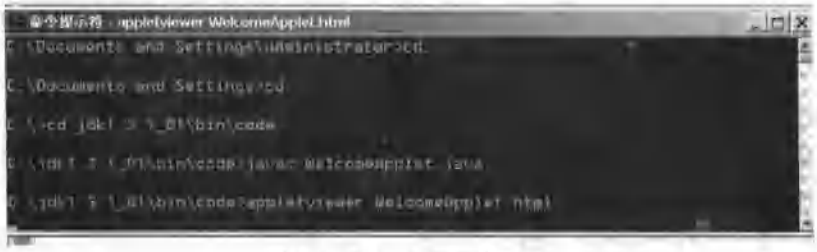

图 1-8　Java Applet 的操作步骤

运行结果如图 1-9 所示。

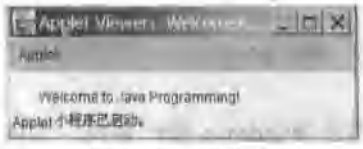

图 1-9　Applet 运行结果

注意：Java 的跨平台不是没有任何条件的。Application 的运行以各个平台上的虚拟机为前提条件，Applet 也是如此。当需要用 Java 的 Swing 编写 Applet 时，浏览器中需要安装支持它的插件。所以，为了在学习时调试 Applet 方便，本书示例均以 JDK 提供的 appletviewer 工具来显示嵌有 Applet 的 HTML 文件。

1.7.4　良好的编程习惯

（1）所有的 Java 语句必须以“;”结束。

（2）Java 区分大小写，拼写时要注意关键字和标识符构成字母的大小写。

（3）花括号成对出现。在写左花括号时，立即再写一个右花括号。这样有助于防止漏写右花括号。类名称后面的花括号标识着类定义的开始和结束。

（4）习惯上，类名应以首字母大写开头，变量以小写字母开头，变量名有多个单词第一个单词后边的每个单词首字母应大写。当读一个 Java 程序时，寻找以大写字母开头的标识符，

这些通常代表 Java 类。

（5）程序段中适当增加空白行会增加程序的可读性。在定义方法内容的花括号中，将整个内容部分缩进一层，使程序结构清晰，程序易读。编译器会忽略这些空白行和空格字符。

（6）在程序中，一行最好只写一条语句。Java 允许一个长句分割写在几行中，但是不允许从标识符或字符串的中间分割。

（7）文件名与 public 类名在拼写及大小写上必须保持一致。

（8）如果一个.java 文件含有多于一个 public 类，则是一个错误。

（9）不以.java 为扩展名的文件名是一个错误。

（10）运行 appletviewer 时，文件扩展名不是.htm 或.html 是一个错误，这将导致无法使 appletviewer 装载 Applet。

1.8 Java 开发工具入门

1.8.1 JDK 的下载、安装

1. 下载 JDK

J2SDK 是 Java 2 Software Development Kit 的简称，其前身是 JDK（Java Development Kit），目前的最新版本是 JDK5.0。根据运行平台的不同，下载相应的版本并设置好 PATH 和 CLASSPATH。这个软件包提供了 Java 编辑器、Java 解释器，但没有提供 Java 编辑器，初学者推荐使用 Windows 的“记事本”。

J2SDK 或 JDK 是 Sun 公司免费提供的，可以在 Sun 公司网站上下载，下载网址——http://java.sun.com/product。也可以从国内的一些 ftp 站点中获得。从 ftp 站点上下载要比从国外站点上下载的速度快。还可在 www.google.com 中搜索后下载。

2. 安装 JDK

由于目前大多数用户使用的是 Windows 操作系统，下面以在 Windows 操作系统上安装 JDK1.3.1 为例，说明安装 JDK1.3.1 的过程。j2sdk-1_3_1_02-win.exe 是一个自解压文件，双击它就可以解压缩，同时进行安装工作。j2sdk-1_3_1_02-win.exe 中包含了 Java Runtime Environment，所以只要安装了 j2sdk-1_3_1_02-win.exe，浏览器就自然可以运行 Java Applet。

安装工作实际上分为两个步骤。安装程序首先会收集一些信息，用于安装的选择，然后才开始拷贝文件，设置 Windows 注册表等具体的安装工作。具体的操作步骤如下。

（1）双击 j2sdk-1_3_1_02-win.exe，文件会自动解压缩，如图 1-10 所示。

图 1-10 文件解压缩

（2）解压缩工作完成之后，出现 JDK1.3.1 的安装欢迎界面，如图 1-11 所示。

（3）单击“Next”按钮，出现 JDK1.3.1 的许可协议，如图 1-12 所示。

图 1-11　安装欢迎界面

图 1-12　安装协议

（4）单击“Yes”按钮，接受许可协议，安装程序会出现让用户选择安装目标路径的对话框。如图 1-13 所示。

在这个对话框中，系统让用户选择 JDK 程序的安装路径，系统默认的路径是 C:\jdk1.3.1_02，单击“Next”按钮，JDK 的所有程序就会被安装到 C:\jdk1.3.1_02 目录下。用户也可以在本对话框中单击“Browse”按钮，修改 JDK 程序的目标安装路径。

（5）下一个对话框让用户选择所使用的浏览器，如图 1-14 所示。

图 1-13　安装目标路径的选择

图 1-14　使用的浏览器的选择

Windows 用户常用的浏览器是 IE，所以选择“Microsoft Internet Explorer”复选框。单击“Next”按钮，进入下一个对话框，如图 1-15 所示。

图 1-15　组件选择

（6）在图 1-15 所示的对话框中选择要安装的 JDK 组件，这里选择全部的组件，单击“Next”按钮，系统就会开始正式的安装工作。

（7）JDK1.3.1 安装结束之后，会继续安装 Java 2 Runtime Environment，用于 Java 程序的执行。安装程序给出提示信息，如图 1-16 所示。

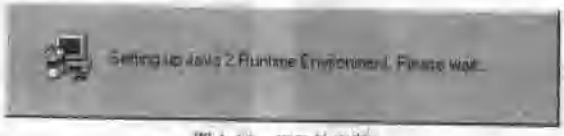

图 1-16　JRE 的安装

（8）Java 2 Runtime Environment 安装完成之后，出现如图 1-17 所示的对话框，单击“Finish”按钮，结束安装。

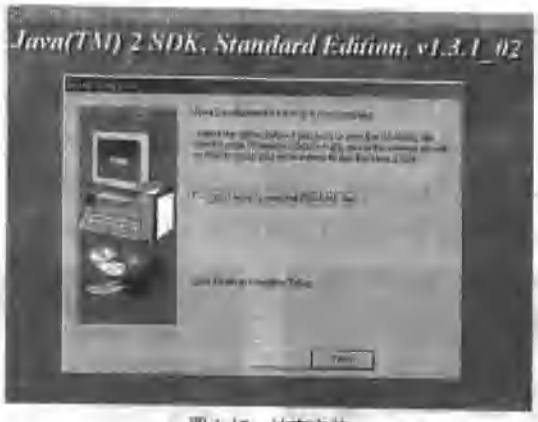

图 1-17　结束安装

1.8.2　环境变量介绍和配置

平台为 Windows 98 时，需要修改系统根目录下的 autoexec.bat 文件。使用记事本或任何文本编辑器打开 autoexec.bat 文件，在该文件的最后增加如下两行：

```
set path=c:\jdk1.3.1_02\bin; %path%
set  classpath=.;c:\jdk1.3.1_02\lib
```

其中，环境变量 path 和 classpath 分别指定了 JDK 命令搜索路径和 Java 类路径。在这里假设 JDK 安装在 C:\ jdk1.3.1_02 目录下，JDK 的所有命令都放在 c:\jdk1.3.1_02\bin 目录下。设置环境变量 path 的作用是使 DOS 操作系统可以找到 JDK 命令。%path%表示 path 环境变量的已有的当前的字符串取值。设置环境变量 classpath 的作用是告诉 Java 类装载器到哪里去寻找第三方提供的类和用户定义的类。在 classpath 环境变量中添加的（.）代表 Java 虚拟机运行时的当前工作目录。

path 环境变量的作用是设置供操作系统去寻找和执行的应用程序的路径，也就是说，如果操作系统在当前目录下没有找到想要的命令工具时，操作系统就会按照 path 环境变量指定的目录依次去查找，以最先找到的为准。path 环境变量可以存放多个路径，Windows 下路径和路径之间用分号（;）隔开。

平台为 Windows 2000 时，右键单击桌面上的“我的电脑”，打开菜单中的“属性”，在出现的属性面板中选择“高级”选项卡，如图 1-18 所示。单击“环境变量”按钮，打开环境

变量面板，在这里可以看到有上下两个窗口，上面的窗口为用户的环境变量，下面的窗口为系统变量，如图 1-19 所示。

图 1-18 属性面板

图 1-19 环境变量面板

可以在任意一个窗口进行设置，区别在于上面的窗口设置用于个人环境变量，只有以该用户身份登录时才有效，而下面窗口中的设置则对所有用户都有效。以设置系统变量为例，单击名为“path”的变量（如果没有要设置的环境变量选项，可在“用户变量”或“系统变量”中选择“新建”按钮来添加），选择“编辑”，打开如图 1-20 所示的“编辑系统变量”窗口，在“变量值”文本框中输入想设置的环境变量，应当在 path 原有值的末尾加上分号（;），然后加上 Java 编译器和解释器所在的路径（这里是 C:\jdk1.3.1_02\bin），最后单击“确定”按钮。

图 1-20 系统变量编辑窗口

path 环境变量设置好之后，在 DOS 窗口下用 cd 命令进入工作目录，程序就可以编译运行。如果有问题，可以进一步在 DOS 下，使用设置语句：set classpath=c:\myjava（myjava 为自己的源代码存放文件夹），再运行自己的源文件就可以了。

如果用户在安装 jdk1.3.1 时，选择了另外的 JDK 安装路径，则环境变量 path 和 classpath 要进行相应的调整。

环境变量设置完成后，在 DOS 窗口下，键入 javac 并按回车后，如果出现 javac 的用法参数提示信息，则安装正确；也可以在 DOS 窗口下使用 set 命令查看添加的系统路径，否则要检查环境变量设置是否正确。

1.8.3 JDK 开发工具简介

J2SDK 工具是以行命令方式应用的，即在 Windows 操作系统的 DOS 命令行提示符窗口中执行 J2SDK 命令。在 JDK 的 bin 目录下，存放着 Java 2 提供的一些可执行程序，为开发