第 1 章

Java 语言基础知识

Java 语言是一个功能强大的跨平台程序设计语言，是目前应用最为广泛的计算机语言之一。本书介绍 Java 语言与面向对象的程序设计方法以及 Java 语言应用的几个专题。作为全书的开篇，本章首先介绍面向对象程序设计的基本概念和 Java 语言的特点；然后简要介绍 Java 程序，并通过几个简单而典型的例子使读者对 Java 程序有个感性认识；接下来详细介绍 Java 的基本数据类型和表达式；最后讲解数组的概念、创建和引用。

在阅读本章时，读者对面向对象程序设计的一些概念、Java 的特点以及某些例题可能不完全理解，这没有关系。对这些内容在本章只是概要性地介绍，在后续章节将有深入、细致的讲解。

1.1 Java 语言与面向对象的程序设计

Java 语言是一个面向对象的程序设计语言，因此在系统地学习 Java 语言之前，首先需要对面向对象的程序设计思想有一个初步了解，在以后的学习过程中将不断加深理解并掌握面向对象的方法。除了面向对象的特点以外，Java 语言还在安全性、平台无关性、支持多线程、内存管理等许多方面具有卓越的优点。本节将概要性地介绍面向对象程序设计与 Java 语言的特点。

1.1.1 面向对象的程序设计思想

计算机程序设计的本质就是把现实生活中遇到的问题，抽象后利用计算机语言转化到机器能够理解的层次，并最终利用机器来寻求问题的解。此过程涉及下面两方面问题：

(1) 如何把问题抽象化？使问题能够很好地被抽象语言描述。

(2) 如何把已经抽象的问题映射到机器能够理解的语言。

第一个方面体现了程序设计思想，而第二个方面则体现了程序设计语言的应用。

面向对象的编程语言与以往编程语言的根本不同点在于抽象机制的不同。

程序设计语言从最开始的**机器语言**到**汇编语言**到各种**结构化高级语言**，到目前使用的支持面向对象技术的**面向对象语言**，关键之处就是抽象机制不断提高。机器语言和汇编语言几乎没有抽象，对于机器而言是最合适的描述，它可以直接操作机器的硬件，并且任何操作都是面向机器的，这就要求人们在使用机器语言或者汇编语言编写程序时，必须按照机器的方式去思考问题。因为没有抽象机制，所以程序员不得不陷入复杂的事物之中。C语言、FORTRAN语言等结构化高级语言的诞生，使程序员可以离开机器层次，在

更抽象的层次上表达意图。

开发一个软件是为了解决某些问题，这些问题所涉及的业务范围称为该软件的问题域。非面向对象的语言，比如机器语言、汇编语言、面向过程的高级语言，都是通过数据的定义和函数的调用来实现一定的功能、解决某些问题。这些语言和人类自然语言之间有很深的鸿沟。比如面向过程的程序设计，它所关注的只是处理过程，即执行预期计算所需要的算法。

面向对象的编程语言将客观事物看作具有状态和行为的对象，通过抽象找出同一类对象的共同状态（静态特征）和行为（动态特征），构成模型——类。世间万事万物皆对象，都可以抽象为包括状态和行为的类。而程序需要解决的问题便反映为各种不同属性的对象以及对象之间的关系和消息传递。面向对象的方法在程序设计领域是一种相对较新的方法，它更接近于人类处理现实世界问题的自然思维方法。假设你面对现实世界的一个对象，会不会把它的状态属性和行为方法分开来看待？当然不会。面向对象的程序设计也是一样：把一类对象的状态属性和处理方法封装在一起作为一个整体。

比如现实生活中的一类对象——汽车，在程序中的模型可以是：

```
class Car {
    int color_number;
    int door_number;
    int speed;
    ……
    void brake() { … };
    void speedUp() {…};
    void slowDown() { …  };
    ……
}
```

在程序中用 color_number、door_number、speed 等数据成员描述汽车的颜色、车门个数、速度等状态属性；用 brake()、speedUp()、slowDown()等方法描述汽车的刹车、加速、减速等处理行为。而数据成员和方法组合在一起构成类，用来描述汽车这类对象。

面向对象的语言实现了封装，封装带来的好处是：隐藏类的数据，控制用户对类的修改和控制数据访问权限。

面向对象技术给软件发展带来如下益处。

- 可重用性：一个设计好的类可以在今后的程序开发中被部分或全部地重复利用。
- 可靠性：每一个类作为一个独立单元可以单独进行测试、维护，大量代码来源于成熟可靠的类库，因而开发新程序时的新增代码明显减少，这是程序可靠性提高的一个重要原因。

面向对象语言具有如下基本特征：

- 抽象和封装：抽象的结果形成类，类中的数据和方法是受保护的，可以根据需要设置不同的访问控制属性。这便保证了数据的安全性，隐藏了方法的实现细节，也方便了使用。

- 继承性：可以对已有类增加属性和功能，或进行部分修改来建立新的类，实现代码的重用。
- 多态性：在面向对象的程序中，同一个消息被不同的对象接收后可以导致不同的行为。

面向对象语言通过类的继承与多态可以很方便地实现代码重用，大大缩短软件开发周期，并使得软件风格统一。面向对象的编程语言使程序能够比较直接地反映问题域的本来面目，使软件开发人员能够利用人类认识事物所采用的一般思维方法进行软件开发。因而面向对象的语言缩小了编程语言与人类语言之间的鸿沟。

面向对象的程序设计语言经历了一个很长的发展阶段。例如 LISP 家族的面向对象语言、Simula67 语言、Smalltalk 语言以及 CLU、Ada、Modula-2 等语言，或多或少地都引入了面向对象的概念，其中 Smalltalk 语言是第一个真正的面向对象的程序语言。

目前，应用最广的面向对象程序语言是 Java 和C++语言。

1.1.2　Java 语言的特点

Java 语言的广泛应用和它具有的特点密不可分，这里简单介绍 Java 所具有的特点。

1. 面向对象

有别于传统语言，Java 是完全面向对象的语言。Java 语言提供了类的机制，在对象中封装了成员变量和方法，实现了数据的封装和信息隐藏；类提供了一类对象的模型，通过继承和多态，实现了代码的复用。

2. 安全性

安全性是网络环境下需要面对的最重要问题。Java 不支持指针，一切对内存的访问都必须经过对象的实例变量实现，防止了以不法手段访问对象的私有成员，同时避免了指针操作中容易产生的错误。Java 的内部安全措施保证 Java 程序在 Java 虚拟机规则下操作，防止未授权的程序访问含有专有信息的系统资源或危及客户机的完整性。

3. 操作平台无关性

Java 编译器生成与平台无关的字节码指令，只要安装了 Java 运行系统，其程序就可以在任意的处理器上运行。这些字节码对应于 Java 虚拟机中的表示，Java 解释器得到字节码后，对其进行解释，使之能够在不同的平台下运行。不同的操作系统有不同的虚拟机。与平台无关的特性使得 Java 程序可以方便地移植到不同的机器上。

4. 多线程

Java 是第一个在语言级提供内置多线程支持的高级语言，这大大简化了多线程程序的编写。而一些其他语言要通过调用操作系统的原语来支持多线程。

5. 动态内存分配

内存管理是 C 和C++程序中最容易产生错误的地方，如果内存分配与内存释放不符，就可能消耗系统资源直至耗尽，最后造成程序异常中止。Java 中所有的对象都是通过动态内存分配建立的，Java 对内存自动进行管理并进行垃圾回收，防止了因程序员失误而导致的内存分配错误，进而更好地利用了系统资源。

1.1.3 Java 类库

组成 Java 程序的最小单位是类，类封装了数据与处理数据的方法。当编写程序来描述和解决问题时，不是总需要从头编写代码，对于大多数常用的功能，有大量已经编译好的、经过测试的类可以直接使用，能够大大提高程序的开发效率。这些类的集合就是Java类库。Java 类库主要是随编译器一起提供，也有些类库是由独立软件开发商提供的，Java 类库也称为 Java API(application programming interface)。

1.2 Java 程序概述

本节首先介绍 Java 的开发环境，然后举例介绍 Application、Applet、Servlet 和 JSP，使读者对 Java 程序有一个宏观上的了解。

1.2.1 Java 开发环境

Java 程序编译执行的过程如图 1-1 所示。

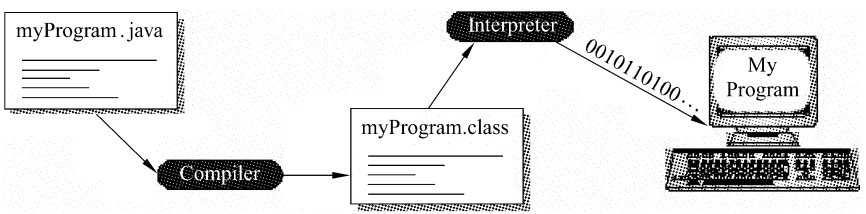

图 1-1 Java 程序编译过程

首先把源文件(.java 文件)编译成字节码文件，即类文件(.class)；然后由解释器负责解释执行类文件。

程序的运行需要一定的硬件和软件环境，这个环境被称为平台(platform)。Windows 2000、Linux、Solaris 和 MacOS 都是人们所熟知的一些平台。大部分平台可以看作是操作系统和硬件的集合，而 Java 平台不同，它是运行于其他平台上的软平台。Java 平台包括 Java 应用程序接口(API)和 Java 虚拟机(Java virtual machine，JVM)。Java 应用程序接口是指经过编译的，可在程序中直接使用的 Java 代码标准库。Java 虚拟机负责解释和执行 Java 程序。

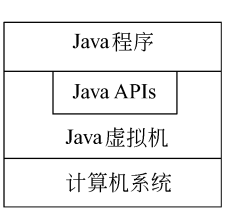

图 1-2 Java 平台

Java 平台如图 1-2 所示。Java 程序运行于 Java 平台之上，Java 虚拟机将程序和硬件隔离开来。

1. J2SDK

下面简单介绍 Java2 标准版软件开发工具包 J2SDK。J2SDK 主要包括开发工具、运行环境、附加库，Applets 和 Applications 的演示等内容。

J2SDK 中部分常用工具如下所示。

- javac. exe：Java 编译器，用于把 Java 源程序编译成字节码，即将. java 文件编译成. class 文件。
- java. exe：Java 解释器，用于解释执行 Java 字节码，即接受. class 文件，然后启动 Java 虚拟机解释并执行。
- appletviwer. exe：用于运行和调试已经编译成字节码的 Java Applet。
- javadoc. exe：Java 文档制作工具。
- jar. exe：压缩工具。
- javah. exe：C 头文件产生工具，用于编写本地方法。
- jdb. exe：Java 调试器，用来调试 Java 程序。

Java2 的运行环境包括 Java 虚拟机、组成 Java2 平台的 API 类和帮助文档。

首次应用 J2SDK 之前要先设置环境变量 path 和 classpath。在 Windows 2000 操作系统下，path 可以在“控制面板”→“系统”→“高级”→“环境变量”→“系统变量”下进行设置，如图 1-3 所示，添加“C:\j2sdk1. 4. 2\bin;”即可。

图 1-3　path 设置过程

在 Windows XP 操作系统下，可以在“开始”→“我的电脑”→“属性”→“高级”→“环境变量”下设置。其他操作系统下，请查阅相关文献。

classpath 也可以按上述方式设置，一般情况下 classpath 应该包括当前目录和 Java 类库所在的目录。

classpath 和 path 也可以在命令行方式下输入如下内容进行设置：

```
set  classpath=. ; C:\j2sdk1. 4. 2\lib\tools. jar;D:\java;
```

其中“. ;”表示当前目录。

经过上述对环境变量的设置之后，就可以开始使用 J2SDK 的开发工具了。

2. 集成开发环境介绍

除了 J2SDK 以外，一些集成开发工具(integrated development environment，IDE)提

供了更为方便的交互式开发环境。下面介绍几种比较流行的集成开发工具。集成开发工具为 Java 提供了集成开发环境，为那些需要集成 Java 与 J2EE 的开发人员提供 Web application、Servlet、JSP、EJB、JDBC 等相关内容的强大支持。这里简要介绍 JBuilder、Sun ONE Studio 5、Eclipse、InteliJ IDEA。

1) JBuilder

可以说 JBuilder 是目前最好的 Java 集成开发工具之一，在协同管理、对 J2EE 和 XML 的支持等方面均走在其他产品的前面。

主要特性如下：

- 提供与 Tomcat 集成，使 Web 开发更容易；
- 提供了对企业应用的开发功能，可以集成多种应用服务器；
- 提供了更简单的程序发布功能，所有的应用都可以打包；
- 提供了团队开发能力，可以集成多种版本控制产品。

但是 JBuilder 也具有一些大家熟知的缺点，比如启动速度慢，占用内存多，中文系统中光标错位问题，显示中文问题等。

JBuilder 的版本升级很快，目前它的最新版本是 JBuilder X。

2) Sun ONE Studio 5

Sun ONE Studio 5 也是一个功能强大而直观的 Java 集成开发环境(IDE)，它基于开放源码的 NetBean 平台，完全使用 Java 语言编写，使用 Swing 支持图形用户界面。Sun ONE Studio 具有优良的性能以及可扩展的特性和功能，支持使用 Java Web Services Developers Pack (JWSDP)进行 J2SE、J2EE 和 Web 服务的开发。Sun ONE Studio 5 有中文版本。

Sun 网站对该工具的介绍如表 1-1 和表 1-2 所示。

表 1-1 Sun ONE Studio 概览

概览
开发、组装和部署 Java 2 Platform、Enterprise Edition (J2EE)组件和基于标准的 Web 服务应用程序，并将其部署到嵌入而紧密集成的 Sun ONE Application Server 7 和其他市场领先的平台上
支持 J2EE 1.3 规范的开发
创建独立的应用程序、Applet 和 JavaBean 组件，或者建立数据库相关的 Web 应用程序
支持 JavaBean 组件、JSP、Servlet 和 JDBC
提供集成的工具套件，包括用于测试的 JUnit、基于 XML 的脚本构建工具 Ant，以及 Pointbase，其中 Pointbase 是一种 100%纯基于 Java 的数据库服务器，用于数据访问和管理

表 1-2 Sun ONE Studio 特性与优点

特　　性	优　　点
Enterprise JavaBean (EJB) 2.0 Workshop	用于 Java 2 Platform、Enterprise Edition (J2EE) 1.3 企业应用程序的快速开发
在现有 EJB 组件的基础上开发和发布基于标准的 Web 服务	用服务相关的工具构建未来领先的 Web 服务应用程序

续表

特　　性	优　　点
用 JSP/Servlet 创建动态的 Web 内容	创建、测试和部署新的 Web 应用程序、Servlet 和来自 JSP 模板的 JSP
使用向导和模板快速开发组件	使用模板和向导大大缩短了开发时间。本产品包括超过 100 个由向导所支持的模板，从而加速了各种应用程序组件和功能的创建
用简化的 JDBC 开发来轻松地进行数据库访问	使用 Pointbase 简化数据访问和管理，这是一个 100 的 Java 数据库服务器，用于数据访问
创建基于 Swing 的图形用户界面(GUI)	使用 GUI Editor 来提高生产效率，还有专用的布局定置器 GridBagLayout，用以缩短开发时间

3) Eclipse

Eclipse 为 IBM 所开发，是替代 Visual Age for Java(以下简称 VAJ)的下一代 IDE 开发环境。Eclipse 是一个开放源代码的项目，任何人都可以免费下载 Eclipse 的源代码，(官方网站在 http://www.eclipse.org)，并且在此基础上开发自己的功能插件。近期还有包括 Oracle 在内的许多大公司也纷纷加入了该项目，并宣称 Eclipse 将来能成为可进行任何语言开发的 IDE 集大成者，使用者只须下载各种语言的插件即可。

也就是说，未来只要有人需要，就会有建立在 Eclipse 之上的各种语言的开发插件出现。同时可以通过开发新的插件扩展现有插件的功能，例如在现有的 Java 开发环境中加入 Tomcat 服务器插件。可以无限扩展，而且有着统一的外观、操作和系统资源管理，这也正是 Eclipse 的潜力所在。但是现在 Eclipse 还没有支持对 EJBs 的开发。

主要特性如下：

- 很方便的对源文件进行导入和导出；
- 源代码的管理更加随心所欲；
- 支持团队开发；
- 支持插件开发功能。

4) IntelliJ IDEA

IntelliJ IDEA 是一个相对较新的 Java IDE，被称为是“最好的 Java IDE 开发平台”。该工具以其聪明的即时分析和方便的 Refactoring（重整）功能深为大家喜爱。它包括 J2EE 支持、Ant、JUnit、集成 CVS。还包含一个智能编辑器，代码辅助和增强的自动代码工具。高度优化的 IntelliJ IDEA 使普通任务变得相当容易，这是它广告语“Develop with pleasure”的原因之一。

1.2.2　Application 举例

本节和下一节简单介绍 Java 的两种程序类型：Application 程序和 Applet 程序，旨在使读者在学习 Java 语言之前对 Java 程序有一个初步认识。

Application 是一个运行在客户端 Java 虚拟机上的 Java 程序。它可在客户端机器中读写，可使用自己的主窗口、标题栏和菜单，程序可大可小。

Application 可以运行在最简单的环境中，能够以命令行参数的方式接收来自外部的数据。应用程序从命令行开始运行，其主类必须有一个主方法 main()，作为程序运行的入口。

例 1-1 Application 举例。

源程序如下：

```
// MyClass.java
public class MyClass {
  //用关键词 class 声明类 MyClass，public 表示这个类的访问权限是公有型
   private int val1,val2 ;
     //用关键字 int 声明两个整型变量，private 表示这两个变量的访问权限是私有型
    public void myFun(int x,int y)
    //定义了名为 myFun 的方法。x、y 为参数名，参数的类型为 int
    //myFun 访问权限为 public，返回值类型为 void，即没有返回值
    {
        val1=x;          // x 的值赋给私有变量 val1
        val2=y;          // y 的值赋给私有变量 val2
       System.out.println("The sum is: "+(val1+val2));       //输出求和运算结果
    }

  //每个 Application 必须有一个，而且也只能有一个 main()方法，作为程序的入口
    public static void main(String arg[])
    //String args[] 是传递给 main 方法的参数，类型为字符串
    {
        MyClass MyObj=new MyClass();      //创建 MyClass 类的一个实例 MyObj
        MyObj.myFun(1,2);      //通过 MyObj 调用方法 MyFun
    }
}
```

使用如下命令编译并运行程序：

```
javac    MyClass.java
java     MyClass
```

运行结果如下：

```
输出：The sum is: 3
```

读者对上述程序中的许多语句可能还不理解，不要紧，后面将对语法进行详细讲解。

1.2.3 Applet 举例

Applet 被称为小应用程序，运行于支持 Java 的 Web 浏览器中。浏览器的解释器把字节码转换成和机器匹配的指令，在网页中执行小程序。Applet 和 Application 的差别来自于运行环境的不同。Applet 需要来自 Web 浏览器的大量信息：它需要知道何时启

动、何时放在浏览器窗口中、何时何处激活或者关闭。小应用程序总是放在 Web 浏览器的图形用户界面中。

Applet 的优点在于 Web 浏览器软件包括很多小应用程序运行所需的功能；局限性是不能从客户端主机的文件系统中读/写，不能运行客户端主机的任何程序，仅能在服务器和客户端之间建立联系。

例 1-2　Applet 举例。

源程序如下：

```
import java.awt.Graphics;        //表明 applet 使用了 java.awt 包中的 Graphics 类
import java.applet.Applet;       //表明 applet 使用了 java.applet 包中的 Applet 类
public class MyApplet extends Applet {
// 声明了一个名为 MyApplet 公共类，它继承了 Applet 类
  public String s;               //声明一个字符串
  public void init(){            //Applet 的初始化方法
          s=new String("Hello World !");      //创建一个字符串
  }
  public void paint(Graphics g){
          g.drawString(s,60,40);
  }
}
```

下面讲解这个简单的 Applet 程序。

Graphics 类使得 Applet 可以绘制直线、矩形、椭圆形、字符串等。此处，绘制了“Hello World”字符串。

方法 init()实现了字符串的创建；paint()方法中，g 为 Graphics 类的对象。调用了 Graphics 的 drawString 方法绘制字符串。drawString 的第一个参数是要绘制的字符串 s，后面两个参数(60,40)说明字符串左下角所在的平面坐标。第二个参数(60)为横坐标，第三个参数(40)为纵坐标。Java 的坐标系原点在左上角，和数学中常常定义的坐标系不同，纵轴正方向指向下方。坐标单位是像素。此方法执行的结果就是从坐标(60,40)开始绘制出字符串 Hello World! 。

使用如下命令在 JDK 中将源程序编译为 class 文件：

```
javac MyApplet.java
```

Applet 中没有 main()方法作为 Java 解释器的入口，因此必须编写 HTML 文件，把 Applet 嵌入其中，然后用 appletviewer 来运行，或在支持 Java 的浏览器上运行。HTML 文件内容如下：

```
<html>
<head>
<title> My Applet </title>
</head>
<body>
```

```
<applet code= HelloWorldApp.class width=400 height=100>
</applet>
</body>
</html>
```

一般情况下，HTML 网页的第一个标记应该<html>，表示文件开始，与之对应的是</html>表示文件结束。同理<head>表示文件头开始，</head>表示文件头结束；<title>表示标题开始，</title>表示标题结束，此处设计标题为“My Applet”；<body>表示主体开始，</body>表示主体结束，其中<applet>标记用来启动 HelloWorldApp，code 指明字节码所在的文件，width 和 heigth 指明 applet 所占的大小。把这个文件存入到 Applet1.html。

用支持 Java 的浏览器，例如 IE 6.0，打开 Applet1.html，得到如图 1-4 所示的结果。

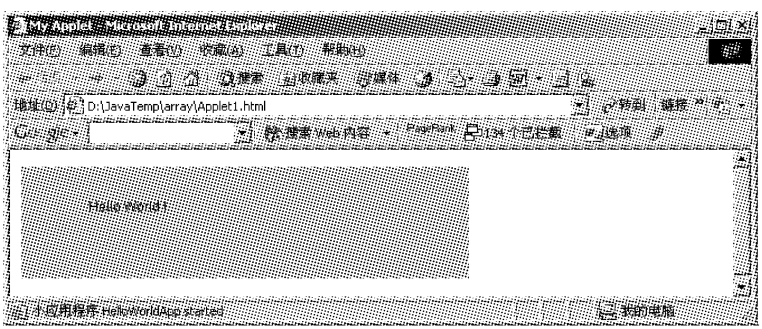

图 1-4 IE 浏览器运行结果

如果用 Java 自带的 appletviewer 浏览，需要输入如下命令，得到如图 1-5 所示的结果。

```
输入：appletviewer Applet1.html
```

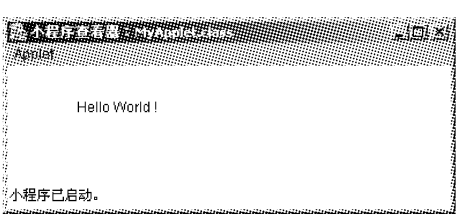

图 1-5 Appletviewer 运行结果

1.2.4 Servlet 举例

本节简单介绍 Java 的 Servlet 技术，在后面的章节中，还将对 Servlet 进一步深入讨论。

就像 Applet 扩展了浏览器的功能一样，Servlet 运行在服务器端，响应客户端请求，扩展了服务器的功能。Servlet 并没有跟客户端的特定协议绑定，但是通常使用的是超文