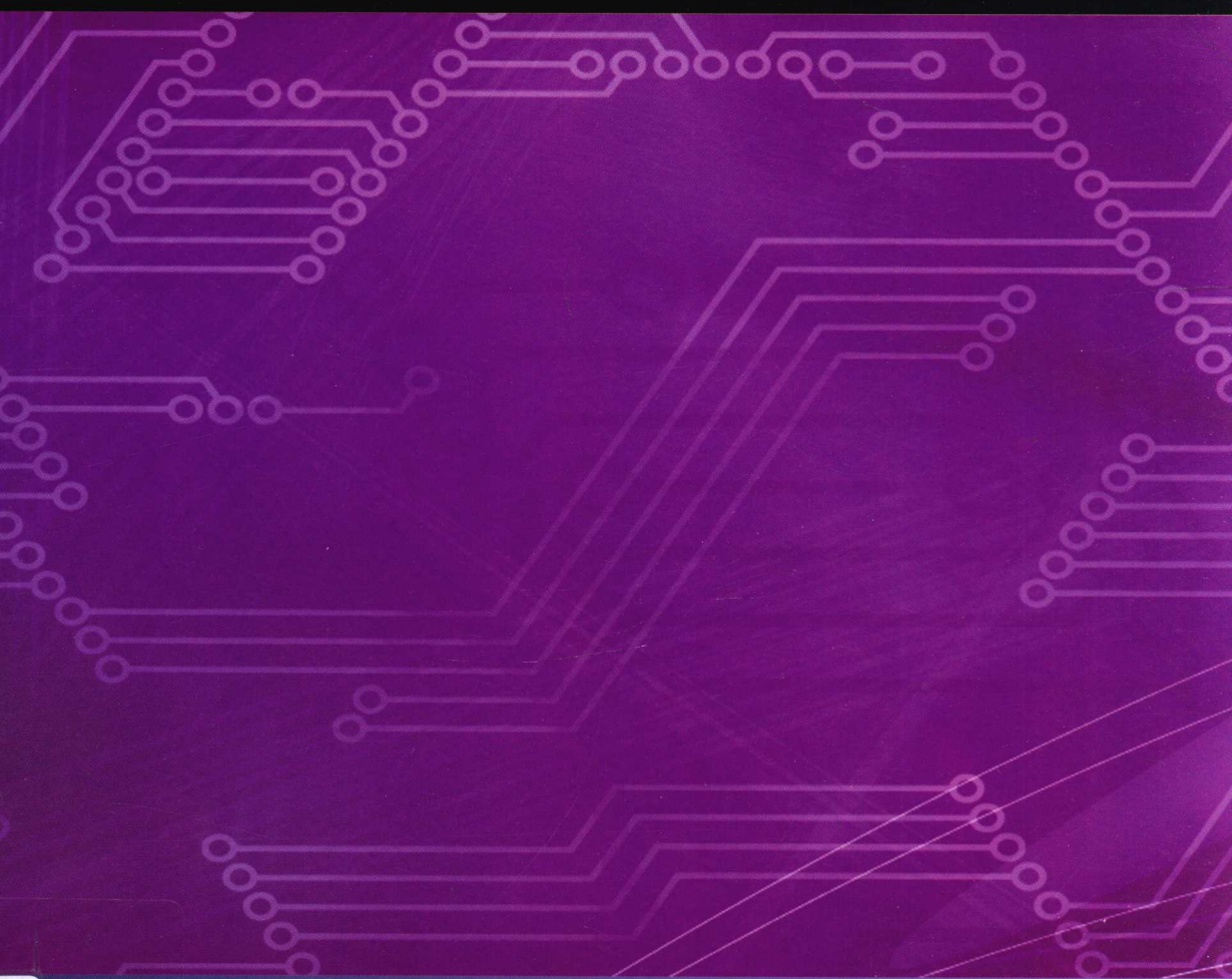
Java
Chengxu Sheji
Jichu Yu Yingyong

Java
程序设计基础与应用

李广建 编著



北京大学出版社
PEKING UNIVERSITY PRESS

Java 程序设计基础与应用

李广建 编著

北京大学出版社
PEKING UNIVERSITY PRESS

内 容 简 介

本书全面地介绍了Java语言的基础知识。全书分为九章，涵盖了Java开发环境、Java的基本数据类型、基本语法、类和接口及其特性、常用的类和接口、异常处理、多线程、网络通信、数据库编程、输入/输出操作、用户界面设计、网络爬虫、信息检索等内容。全书力图兼顾系统性、知识性、实用性，在介绍Java的基本语言现象的同时，也提供了大量的示例程序及相应的说明，以帮助读者提高编程实践能力。

本书可作为高等学校Java程序设计课程的教材，也可作为Java语言的培训教材或Java语言爱好者的自学用书。

图书在版编目(CIP)数据

Java程序设计基础与应用/李广建编著. —北京：北京大学出版社，2013.10
ISBN 978-7-301-23231-6

Ⅰ. ①J… Ⅱ. ①李… Ⅲ. ①JAVA语言—程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2013)第222598号

书　　名：Java程序设计基础与应用
著作责任者：李广建　编著
责 任 编 辑：王　华
标 准 书 号：ISBN 978-7-301-23231-6/TP·1308
出 版 发 行：北京大学出版社
地　　址：北京市海淀区成府路205号　100871
网　　址：http://www.pup.cn　　新浪官方微博：@北京大学出版社
电 子 信 箱：zpup@pup.pku.edu.cn
电　　话：邮购部62752015　发行部62750672　编辑部62765014　出版部62754962
印　刷　者：北京大学印刷厂
经　销　者：新华书店
787mm×1092mm　16开本　34.75印张　677千字
2013年10月第1版　2013年10月第1次印刷
定　　价：69.00元

前　言

本书试图较全面地介绍Java语言的基础知识，全书共分九章，介绍了Java语言的基础知识，包括Java语言概述、Java语言基础、Java面向对象程序设计、常用类与接口、Java编程技术、输入与输出、图形用户界面、Web搜索技术以及信息检索技术。

本书是根据我为信息管理与信息系统专业本科生开设的“面向对象程序设计Java”课程讲稿的一部分内容编写而成，承蒙北京大学主要专业课教材立项项目资助，使得本书得以正式出版。

徐树维、乔建忠、李芳、齐慧颖、王巍巍、李亚子、苏玉召、杨林等同志参加本书体例的讨论。在撰写本书的过程中，徐树维、乔建忠、李芳、齐慧颖、王巍巍同志参加了已有的课程讲稿的整理，并补充了一些新材料。

在开设“面向对象程序设计Java”课程的过程中，参阅和引用了大量的文献，不断对讲稿进行了补充，由于持续时间较长，加之篇幅有限，本书无法将参考和引用的文献一一列出，在此，对本书具名和未具名的参考文献的作者表示衷心的感谢。

北京大学出版社的王华同志耐心仔细地审读书稿，从书稿的内容到文字，与我多次交换意见，提出了非常中肯的修改意见。在此，对她深表谢意。

Java内容博大精深，由于水平所限，本书难免存在不足，甚至谬误，敬请专家、学者和读者批评指正。

李广建

2012年8月

目　录

第一章　面向对象的Java语言概述

Java是20世纪90年代出现的完全面向对象的程序设计语言，体现了计算机编程语言的新方法、新思想。本章首先介绍面向对象程序设计的基本概念、特点和基本思想；然后介绍面向对象的Java程序设计语言的发展概况、特点、运行机制和运行环境；最后简单介绍Java集成编程工具Eclipse。

1.1　面向对象技术程序设计

面向对象程序设计(Object-Oriented Programming，OOP)是计算机软件技术发展过程中的一个重大飞越，它能更好地适合软件开发在规模、复杂性、可靠性和质量、效率上的需求，因而被广泛应用，并逐渐成为当前的主流程序设计方法。

1.1.1　面向对象程序设计的基本思想

面向对象程序设计代表了一种全新的程序设计思路和表达、处理问题的方法。在解决问题的过程中，面向对象程序设计以问题中所涉及的各种对象为主要线索，关心的是对象以及对象之间的相互关系，以符合人们日常的思维习惯来求解问题，降低、分解了问题的难度和复杂性，提高了整个求解过程的可控性、可监测性和可维护性，从而能以较小的代价和较高的效率对问题进行求解。

简言之，面向对象程序设计的特点是使用对象模型对客观世界进行抽象，分析出事物的本质特征，从而对问题进行求解。面向对象程序设计的思想认为世界是由各种各样具有各自运动规律和内部状态的对象组成的，不同对象之间的相互通信和作用构成了现实世界。因此，人们应当按照现实世界本来的面貌理解世界，直接通过对象及其相互关系来反映世界，这样建立起来的系统才符合世界本来的面貌，才会对现实世界的变化有很好的适应性。所以，面向对象方法强调程序系统的结构应当与现实世界的结构相对应，应当围绕现实世界中的对象来构造程序系统。

所谓对象，是指现实世界的实体或概念在计算机程序中的抽象表示，具体地说，程序设计中的对象是指具有唯一对象名和一组固定对外接口的属性和操作的集合，它用来模拟组成或影响现实世界问题的一个或一组因素。其中，对象名是用于区别对象的标识；对象的对外接口是在约定好的运行框架和消息传递机制下与外界进行通信的通道；对象的属性表示它所处的状态；对象的操作(也称方法)是用来改变对象状态的特定功能。

具体地说，面向对象程序设计的思想主要体现在如下几个方面：

(1) 面向对象程序设计的核心和首要问题是标识对象，而不是标识程序中的功能(函数/过程)。从面向对象程序设计的角度来看，对象作为现实世界中事物的基本组成部分，是系统框架中最稳定的因素，对象描述清楚了，就能够很容易找出它们之间的关系，进而发现它

们之间的相互作用，从而解决问题。

(2) 正是由于把标识对象作为解决问题的出发点，面向对象程序设计在整体上说是一种自底向上的开发方法。面向对象的基本思想将程序看作是众多协同工作的对象所组成的集合，这些对象相互作用构成了系统的完整功能，因此，在设计开发程序时，面向对象的方法按照标识对象、定义对象属性和操作、明确对象之间事件驱动和消息传递关系，最后形成程序的整体结构这样一个顺序进行设计。这个过程，是一个典型的自底向上的过程。

(3) 同任何应用系统开发一样，面向对象的程序设计也要经历系统分析、系统设计和系统实施等主要阶段。但是，由于面向对象程序设计在概念模式与系统组成模式上的一致性，就使得面向对象程序设计过程中的各个阶段是一种自然平滑的过渡，各阶段的界限不是那么明显。系统分析阶段的结果能够直接映射成系统设计阶段的概念，系统设计阶段的结果也可以方便地翻译成实施阶段的程序组件，反之亦然。这样，系统设计和开发人员就能容易地跟踪整个系统开发过程，了解各个阶段所发生的变化，不断对各个阶段进行完善。

总之，面向对象程序设计方法更符合人们对客观世界的认识规律，开发的软件系统易于维护，易于理解、扩充和修改，并支持软件的复用。因而从 20 世纪 90 年代开始，面向对象程序设计的方法逐渐成为软件开发的主流方法。

1.1.2 面向对象程序设计的发展历史

面向对象程序设计方法作为一种程序设计规范，与程序设计语言的发展密切相关。事实上，最早的面向对象程序设计的一些概念正是由一些特定语言机制体现出来的。

20 世纪 50 年代后期，为了解决 Fortran 语言编写大型软件时出现的变量名在不同程序段中的冲突问题，Algol 语言设计者采用了“阻隔”(Barriers)的方式来区分不同程序段中的变量名，在程序设计语言 Algol60 中用 “Begin…End”为标识对程序进行分段，以便区分不同程序段中的同名变量，这也是首次在编程语言中出现保护(Protection)或封装(Encapsulation)的思想。

20 世纪 60 年代，挪威科学家 O. J. Dahl 和 K. Nygard 等人采用了 Algol 语言中的思想，设计出用于模拟离散事件的程序设计语言 Simula 67。与以往程序设计语言不同，Simula 67 从一个全新的角度描述并理解客观事实，首次在程序设计中将数据和与之对应的操作结合成为一个整体，提出“封装”的概念，它的类型结构和以后的抽象数据类型基本是一样的。尽管 Simula 67 还不是真正的面向对象程序设计语言，但它提出的思想标志着面向对象技术正式登上历史舞台。

真正的面向对象程序设计语言是由美国 Alan Keyz 主持设计的 Smalltalk 语言。“Smalltalk”这个名字源自“Talk Small(少说话)”，意思是可以通过很少的工作量完成许多任务。Smalltalk 在设计中强调对象概念的统一，引入了对象、对象类、方法、实例等概念和术语，采用了动态联编和单继承机制。用 Smalltalk 编写的程序具有封装、继承、多态等特性，由此奠定了面向对象程序设计的基础。20 世纪 80 年代以后，美国 Xerox 公司推出了 Smalltalk-80，引起人们的广泛重视。

Smalltalk 语言出现，引发了学术界对面向对象程序设计的广泛重视，随之涌现出了很多面向对象的系统分析与设计方法，诞生了一系列面向对象的语言，如 C++、Eiffe、Ada 和

CLOS等。其中，C++不仅继承了C语言易于掌握、使用简单的特点，而且增加了众多支持面向对象程序设计的特性，促进了面向对象程序设计技术的发展。

20世纪90年代，美国Sun Microsystems公司提出的面向对象的程序设计语言Java，被认为是面向对象程序设计的一次革命，Java语言去除了C++中为了兼容C语言而保留的非面向对象的内容，使程序更加严谨、可靠、易懂。尤其是Java所特有的“一次编写、多次使用”的跨平台优点，使得它非常适合在Internet应用开发中使用，Java已经成为当前最为流行的面向对象的程序设计语言。

从面向对象程序设计语言的发展历程可以看出，面向对象程序设计语言是经过研究人员的不断改进与优化，才形成了今天的模样。正是由于这种语言更好地适应了软件开发过程中规模、复杂性、可靠性和质量、效率上的需求，并且在实践中得到了检验，逐渐成为当前主流的程序设计方法。

1.1.3　面向对象程序设计的特点

面向对象程序设计有许多特点，这里重点介绍其主要特点。

1. 抽象(Abstract)

抽象是日常生活中经常使用的一种方法，即去除掉被认识对象中与主旨无关的部分，或是暂不予考虑的这些部分，而仅仅抽取出与认识目的有关的实质性的内容加以考察。在计算机程序设计中所使用的抽象有两类：一类是过程抽象，另一类是数据抽象。

过程抽象将整个系统的功能划分为若干部分，强调功能完成的过程和步骤。面向过程的软件开发方法采用的就是这种抽象方法。使用过程抽象有利于控制、降低整个程序的复杂程度，但是这种方法本身自由度较大，难于规范化和标准化，操作起来有一定难度，质量上不易保证。

数据抽象是与过程抽象不同的抽象方法，它把系统中需要处理的数据和这些数据上的操作结合在一起，根据功能、性质、作用等因素抽象成不同的抽象数据类型。每个抽象数据类型既包含了数据，也包含了针对这些数据的授权操作，是相对于过程抽象而言更为严格、也更为合理的抽象方法。

面向对象程序设计的主要特点之一，就是采用了数据抽象的方法来构建程序的类、对象和方法。在面向对象程序设计中使用的数据抽象方法，一方面可以去除与核心问题无关的细节，使开发工作可以集中在关键、重要的部分；另一方面，在数据抽象过程中，对数据操作的分析、辨别和定义可以帮助开发人员对整个问题有更深入、准确的认识，最后抽象形成的抽象数据类型，则是进一步设计、编程的基础和依据。

2. 封装(Encapsulation)

封装是面向对象程序设计的重要特征之一，面向对象程序设计的封装特性与其抽象特性密切相关。封装是指利用抽象数据类型将数据和基于数据的操作结合并包封到一起，数据被保护在抽象数据类型的内部，系统的其他部分只有通过对象所提供的操作来间接访问对象内部的私有数据，与这个抽象数据类型进行交流和交互。

在面向对象程序设计中，抽象数据类型是用“类”这种面向对象工具可理解和可操作的结构来代表的。每个类里都封装了相关的数据和操作。在实际的开发过程中，类多用来构

建系统内部的模块。由于封装特性把类内的数据保护的很严密，模块与模块之间仅通过严格控制的接口进行交互，大大减少了它们之间的耦合和交叉，从而降低了开发过程的复杂性，提高了开发的效率和质量，减少了可能的错误，同时也保证了程序中数据的完整性和安全性。

面向对象程序设计的这种封装特性还有另一个重要意义，就是抽象数据类型，即类或模块的可重用性大为提高。封装使得抽象数据类型对内成为一个结构完整、可自我管理、自我平衡、高度集中的整体；对外则是一个功能明确、接口单一、在各种合适的环境下都能独立工作的有机单元。这样的有机单元特别有利于构建、开发大型、标准化的应用软件系统，可以大幅度地提高生产效率，缩短开发周期和降低各种费用。

3. 继承(Inheritance)

继承是面向对象程序设计中最具有特色，也与传统方法最不相同的一个特点。继承是存在于面向对象程序的两个类之间的一种关系，是组织、构造和重用类的一种方法。当一个类具有另一个类的所有数据和操作时，就称这两个类之间具有继承关系。被继承的类称为父类或超类，继承了父类或超类所有属性和方法的类称为子类。通过继承，可以将公用部分定义在父类中，不同的部分定义在子类中。这样公用部分可以从父类中继承下来，避免了公用代码的重复开发，实现了软件和程序的可重用性；同时，对父类中公用部分的修改也会自动传播到子类中，而无需对子类做任何修改，这样有利于代码的维护。

一个父类可以同时拥有多个子类，这时该父类实际是所有子类的公共属性的集合，而每个子类则是父类的特殊化，是在父类公共属性的基础上进行的功能和内涵的扩展及延伸。

在面向对象程序设计的继承特性中，有单重继承和多重继承之分。所谓单重继承，是指任何一个类都只有一个单一的父类；多重继承是指一个类可以有一个以上的父类，它的静态数据属性和操作从所有父类中继承。采用单重继承的程序结构比较简单，是单纯的树状结构，掌握、控制起来相对容易。支持多重继承的程序，其结构则是复杂的网状，设计、实现都比较复杂。在现实世界中，问题的内部结构多为复杂的网状，用多重继承的程序模拟起来比较自然，但会导致编程方面的复杂性。单重继承的程序结构简单，实现方便，但要解决网状的继承关系则需要其他的一些辅助措施。

在面向对象的程序设计中，采用继承的机制来组织、设计系统中的类，可以提高程序的抽象程度，使之更接近于人类的思维方式，同时也可以提高程序的开发效率、减少维护的工作量。

4. 多态(Polymorphism)

多态是面向对象程序设计的又一个特殊特性。所谓多态，是指一个程序中同名的不同方法共存的情况。在使用面向过程的语言编程时，主要工作是编写一个个的过程或函数，这些过程和函数各自对应一定的功能，它们之间是不能重名的，否则在用名字调用时，就会产生歧异和错误。而在面向对象的程序设计中，有时却需要利用这样的“重名”现象来提高程序的抽象度和简洁性。

面向对象程序设计中的多态有多种表现方式，可以通过子类对父类方法的覆盖实现多态，也可以利用重载在同一个类中定义多个同名的不同方法，等等。多态的特点大大提高了程序的抽象程度和简洁性，更为重要的是，它最大限度地降低了类和程序模块之间的耦合

性，使得它们不需要了解对方的具体细节，就可以很好地工作。这个优点，对应用系统的设计、开发和维护都有很大的好处。

1.2　Java 简介

Java 是美国 Sun Microsystems 公司研制的一种新型的程序设计语言。在高级语言已经非常丰富的背景下，Java 语言脱颖而出，独树一帜，在瑞士 TIOBE 公司每月发布的程序开发语言排行榜中，Java 连续多年名列榜首，说明了人们对 Java 的喜爱程度。

1.2.1　Java 产生的历史与现状

1994 年，美国 Sun MicroSystems 公司成立了 Green 项目开发小组，旨在研制一种能对家用电器进行控制和通信的分布式代码系统，当时这套系统被命名为 Oak 语言，这就是 Java 语言的前身。

1994 年前后，正是 Internet，特别是 Web 的大发展时期，Sun MicroSystems 公司的研究人员发现 Oak 的许多特性更适合网络编程，于是在这方面进行一系列改进和完善，并获得了成功。1995 年初，Sun MicroSystems 公司要给这种语言申请注册商标，由于 Oak 已经被人注册，必须要为这种语言找到一个新的名字。在公司召开的命名征集会上，Mark Opperman 提出 Java 这个名字，据说，Mark Opperman 是因品尝咖啡时得到灵感的。Java 是印度尼西亚爪哇岛的英文名称，该岛因盛产高质量的咖啡而闻名，常被用来当做优质咖啡的代名词，Mark Opperman 的这个提议，得到了所有人的认可和律师的通过，Sun MicroSystems 公司用 Java 这个名字进行了注册，并以一杯热气腾腾的咖啡作为标志（logo），Java 语言由此诞生。

Java 从诞生到今天，不断进行改进和更新。其发展历程大致可以分成以下几个阶段。

1. 诞生期——Java1.0 和 Java1.1

Java1.0 版本的出现是为了帮助开发人员建立运行环境并提供开发工具。1996 年 1 月 23 日，Sun MicroSystems 公司发布了第一个 Java 开发工具 JDK1.0，JDK（Java Development Kit），JDK1.0 由运行环境（Java Runtime Enviroment，JRE）及开发工具（即 JDK）组成，其中运行环境又包括了 Java 虚拟机（Java Virtual Machine，JVM）、API（应用程序接口）和发布技术。随后的第二年（1997 年），Sun MicroSystems 公司对 1.0 版本进行了较大的改进，推出了 JDK1.1 版本，其中增加了 JIT（Just-In-Time）编译器。

2. 发展期——Java1.2 和 Java1.3

1998 年 12 月，Java1.0 诞生近三年后，Sun Microsystems 公司推出 Java1.2，并将其改名为 Java2，且把 Java1.2 以后的版本统称为 Java2，同时将 JDK1.2 也改名为 J2SDK（SDK 的全称为 Software Development Kit，意为软件开发工具），从此 Java 进入了快速发展的阶段。1999 年，Sun Microsystems 公司发布 Java 的三个版本：标准版（J2SE，Java2 Standard Edition）、企业版（J2EE，Java2 Enterprise Edition）和微型版（Java2 Micro Edition，J2ME），以适应不同的应用开发要求。2000 年，JDK1.3 版本发布，Java1.3 版本在 Java1.2 取得成功的基础上进行了一些改进，主要是对 API 做了改进和扩展。

3. 成熟期——Java1.4、Java1.5 和 Java1.6

自 Java2 平台开始，Java 的发展日趋成熟稳定，此后的 Java1.4、Java1.5 和 Java1.6 版本主要在分布式、稳定性、可伸缩性、安全性和管理方面进行了改进和提高。Java1.4 比 Java1.3 版本的运行效率提高了一倍，而从 Java1.5 版本开始，Java1.5 改名为 Java5.0，J2SE1.5 改名为 J2SE5.0，更好的反映出 J2SE 的成熟度和稳定性；Java1.6(J2SE6.0)则更强调管理的易用性，为外部管理软件提供更多的交互信息，并更好的集成了图形化用户界面。从 2004 年开始，为了更加突出 Java 本身，而不是 Java 的某个版本编号，Java 的三个版本陆续更名，去掉其中的编号 2，J2SE、J2EE、J2ME 更名为 Java SE、Java EE 和 Java ME。

经过不断完善和发展，Java 已经得到业界的广泛认可，主要体现在工业界认可、软件开发商青睐和编程人员欢迎等几个方面。

工业界认可。目前绝大部分计算机企业包括 IBM、Apple、DEC、Adobe、Silicon Graphics、HP、Oracle、Toshiba 以及 Microsoft 等公司都购买了 Java 的许可证，用 Java 开发相应的产品。这说明 Java 已得到了工业界的认可。

软件开发商青睐。除购买 Java 许可证，用 Java 开发新产品以外，众多的软件厂商还在自己已有的产品上增加 Java 接口，以使自己的产品支持 Java 的应用，例如 Oracle、Sybase、Versant 等数据库厂商开发了 CGI 接口，使得这些数据库支持 Java 开发。

编程人员欢迎。Java 的一个重要特点是其网络编程能力，因而成为网络时代编程人员最为欢迎的程序设计语言，各行业对掌握 Java 编程语言的人员需求量也非常大。使用 Java 是体现编程人员的能力的重要标志之一。

上述事实说明，Java 是一种得到广泛应用并有很好发展前景的程序设计语言。

1.2.2 Java 语言的特点

Java 语言之所以能够受到如此众多的好评并拥有如此迅猛地发展速度，与其本身的特点是分不开的。Java 的主要特点如下：

1. 面向对象设计

面向对象设计是 Java 的标志特性。作为一种纯粹的面向对象程序设计语言，Java 不再支持面向过程的设计方法，而是从面向对象的角度思考和设计程序。Java 通过创建类和对象来描述和解决问题，支持封装、继承、重载、多态等面向对象特性，提高了程序的可重用性和可维护性。

2. 简单易用

Java 最初的产生源于对家用电器的控制，其设计以简单易用、规模小为原则。一方面，Java 的语法非常简单，它不再使用其他高级程序设计语言中诸如指针运算、结构、联合、多维数组、内存管理等复杂的语言现象，降低程序编写的难度；另一方面，Java 提供了极为丰富的类库，封装了各种常用的功能，程序设计人员无需对这些常用的功能再自行编写程序，只要直接调用即可，尽可能降低了程序设计人员的工作量。

3. 平台无关性

Java 的平台无关性主要体现在三个方面。首先，Java 运行环境是 Java 虚拟机，Java 虚拟机负责解释编译后的 Java 代码并将其转换成特定系统的机器码，再由机器加以执行。

Java虚拟机屏蔽了具体平台的差异性，用Java编写的应用程序无需重新编译就可以在不同平台的Java虚拟机上运行，实现了平台无关性。其次，Java的数据类型被设计成不依赖于具体机器，例如，整数总是32位，长整数总是64位。这样，Java基本数据类型及其运算在任何平台上都是一致的，不会因平台的变化而改变。第三，Java核心类库与平台无关，对类库的调用，不会影响Java的跨平台性。

4. 安全性和健壮性

Java去除了指针和内存管理等易出错的操作，在程序设计上增强了安全性。再有，Java作为网络开发语言，提供了多层保护机制增强安全性，例如不允许Applet运行和读写任何浏览器端机器上的程序等。此外，Java注重尽早发现错误，Java编译器可以检查出很多开发早期的错误，增强了程序设计的安全性和健壮性。

5. 性能优异

Java可以在运行时直接将目标代码翻译成机器指令，充分地利用硬件平台资源，从而可以得到较高的整体性能。另外，Sun Microsystems公司等与Java有关的厂商在不断完善Java的即时(Just-In-Time)编译器技术，旨在提高Java的运行速度，从现在的有基准测试来看，Java的运行速度超过了典型的脚本语言，越来越接近C和C++。

6. 分布式

分布式特性是指在由网络相连的不同平台上，可以在独立运行时间内运行不同程序。Java作为一种强大的网络开发语言，其能力主要体现在开发分布式网络应用。Java语言本身的特点很适合开发基于Internet的分布式应用程序，并且提供了完备的适应分布式应用的程序库。Java支持TCP/IP协议及其他协议，可以通过URL地址实现对网络上其他对象的访问，实现分布式应用。

7. 多线程

Java支持多线程技术，允许在程序中并发地执行多个指令流或程序片段，更好地利用系统资源，提高程序的运行效率。Java不仅支持多线程，而且对线程划分了优先级，更好地支持系统的交互和实时响应能力。此外，Java还具备线程同步功能，确保了计算结果的可预测性，有助于对程序做更好的控制。

1.2.3　Java程序的运行机制

Java程序的运行机制与C/C++等程序设计语言有较大的差别，这种差别也是保证Java具有更强动态性和平台无关性的基础。概括来说，Java的运行有三个步骤：编写、编译和运行。

编写是指利用编辑器生成Java程序代码，形成Java源文件。Java程序以.java为后缀。一个Java应用程序中可能会包括多个Java的类，这些类可以放在同一个Java源文件中，也可以为每一个类分别编写一个源文件。

编译是指Java编译器将编辑好的Java源程序转换成JVM可以识别的字节码的过程。字节码是一种独立于操作系统和机器平台的中间代码，用二进制形式表示，由JVM解释后才能在机器上执行。编译成功后，Java编译器生成后缀名为.class的字节码文件。如果一个Java源程序中包含了多个类，编译后会生成多个对应的.class文件。