# 解析

# Java虚拟机开发：

## 权衡优化、高效和安全的最优方案

介绍Java虚拟机开发的核心知识，内容深入、语言通俗易懂
全书精心筛选了虚拟机开发中最具代表性、最典型的知识点
采用了理论加实践的教学方法，兼顾理论、案例的完美展现



张善香 编著

清华大学出版社

# 解析 Java 虚拟机开发：权衡优化、高效和安全的最优方案

张善香　编　著



清华大学出版社
北京

## 内 容 简 介

本书细致分析了 Java 虚拟机开发的基本知识，为读者权衡出优化、高效和安全的最优方案。本书内容新颖、知识全面、讲解详细，全书共 17 章，第 1 章讲解一起走进 Java 世界的基本知识；第 2 章讲解 JDK 编译测试的基础知识；第 3 章讲解安全性考虑的核心知识；第 4 章讲解通过网络实现移动性的知识；第 5 章浅谈 Java 虚拟机内部机制的基础知识；第 6 章深入分析 class 文件的核心知识；第 7 章详细讲解栈和局部变量操作的知识；第 8 章深入详解内存异常和垃圾处理的基本知识；第 9 章讲解高效手段之性能监控工具和优化部署的核心知识；第 10 章讲解 JVM 参数分析和调优实战的知识；第 11 章讲解虚拟机类加载机制的基本知识；第 12 章讲解研究高效之魂；第 13 章讲解类加载器和执行子系统的基本知识；第 14 章讲解编译优化的基本知识；第 15 章讲解运行期优化的基本知识；第 16 章讲解内存模型和线程的基本知识；第 17 章讲解如何将安全和优化合二为一。全书内容循序渐进，并且逐一做到了深入剖析。

本书适合 Java 各个级别的程序员、研发人员及在职程序员，也可以作为相关培训学校和大专院校相关专业的教学用书。



图书在版编目(CIP)数据

解析 Java 虚拟机开发：权衡优化、高效和安全的最优方案/张善香编著. —北京：清华大学出版社，2013
ISBN 978-7-302-31494-3

Ⅰ. ①解… Ⅱ. ①张… Ⅲ. ①Java 语言—程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2013)第 024772 号

**责任编辑**：魏　莹
**装帧设计**：杨玉兰
**责任校对**：王　晖
**责任印制**：宋　林

**出版发行**：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn,010-62791865

**印 装 者**：清华大学印刷厂
**经　　销**：全国新华书店
**开　　本**：185mm×260mm　　**印　　张**：31　　**字　　数**：751 千字
**版　　次**：2013 年 6 月第 1 版　　**印　　次**：2013 年 6 月第1 次印刷
**印　　数**：1～4000
**定　　价**：59.00 元

产品编号：046016-01

# 前　言

自从 Java 语言自诞生以来，经过十多年的发展和应用，已经成为当今最流行的编程语言之一。根据某权威编程语言排行榜显示，它始终居于第一位。现在全球已有超过 15 亿部手机和手持设备应用 Java 技术。同时，Java 技术因其跨平台特性和良好的可移植性，成为广大软件开发技术人员的挚爱，是全球程序员的首选开发平台之一。

日益成熟的 Java 语言编程技术现在已无处不在，使用该编程技术可以进行桌面程序应用、Web 应用、分布式系统和嵌入式系统应用开发，并且在信息技术等各个领域得到广泛应用。

本书全面介绍了 Java 虚拟机技术的核心知识，全书内容深入并详解，作者用通俗的语言将大师级的知识展现在读者的眼前。

**本书的内容**

通过本书的内容，细致分析了 Java 虚拟机开发的基本知识，和读者们一起权衡出优化、高效和安全的最优方案。本书内容新颖、知识全面、讲解详细，全书分为 17 章，第 1 章讲解一起走进 Java 世界的基本知识；第 2 章讲解 JDK 编译测试的基础知识；第 3 章讲解安全性考虑的核心知识；第 4 章讲解通过网络实现移动性的知识；第 5 章浅谈 Java 虚拟机内部机制的基础知识；第 6 章深入分析 class 文件的核心知识；第 7 章详细讲解栈和局部变量操作的知识；第 8 章深入详解内存异常和垃圾处理的基本知识；第 9 章讲解高效手段之性能监控工具和优化部署的核心知识；第 10 章讲解 JVM 参数分析和调优实战的知识；第 11 章讲解虚拟机类加载机制的基本知识；第 12 章讲解研究高效之魂；第 13 章讲解类加载器和执行子系统的基本知识；第 14 章讲解编译优化的基本知识；第 15 章讲解运行期优化的基本知识；第 16 章讲解内存模型和线程的基本知识；第 17 章讲解如何将安全和优化合二为一。全书内容循序渐进，并且逐一做到了深入剖析。

**本书特色**

本书内容相当丰富，实例内容覆盖全面，满足 Java 程序员成长道路上的方方面面。我们的目标是通过一本图书，提供多本图书的价值，读者可以根据自己的需要有选择地阅读，以完善本人的知识和技能结构。在内容的编写上，本书具有以下特色。

(1) 专家写作，内容专业而深入

本书是国内一线著名的 Java 专家级作者的力作。为了本书确保广度和深度，本书并没有将大量篇幅用在规范和基本语法上，而是专注于各个基本知识的具体细节，尽量涉及了每个知识中最为重要的内容，并且讨论了相关的高级用法技术。

本书既是介绍性书籍，又是深入研究技术性书籍。本书实现了高级技术与介绍性知识并重的效果，为了达到这一目标，笔者做过大量研究。比如，参与论坛讨论，开发大量的实际项目，参加学术会议和研讨会，同时跟制定 Java 规范的专家组进行沟通，同全世界顶

级专家进行合作。

(2) 结构合理

从用户的实际需要出发，科学安排知识结构，内容由浅入深，叙述清楚，具有很强的知识性和实用性，反映了 Java 虚拟机的核心知识。同时全书精心筛选的最具代表性、读者最关心典型知识点，几乎包括虚拟机技术的各个方面。

(3) 易学易懂

本书条理清晰、语言简洁，可帮助读者快速掌握每个知识点；每个部分既相互连贯又自成体系，使读者既可以按照本书编排的章节顺序进行学习，也可以根据自己的需求对某一章节进行针对性的学习。

(4) 由浅入深

本书从 Java 语言的发展、开发环境及基本语法知识入手，逐步介绍了 JDK 编译测试、安全性、移动性、虚拟机的内部机制、class 文件、栈和局部变量操作、内存异常、垃圾处理、性能监控工具和优化部署、类的加载机制、类加载器和执行子系统、编译优化等知识。保证让读者在没有编程基础的情况下，也能够很快掌握 Java 虚拟机的各种技术。

(5) 实用性强

本书彻底摒弃枯燥的理论和简单的操作，注重实用性和可操作性，详细讲解了各个部分的源码知识，使用户掌握相关的操作技能的同时，还能学习到相应的基础知识。

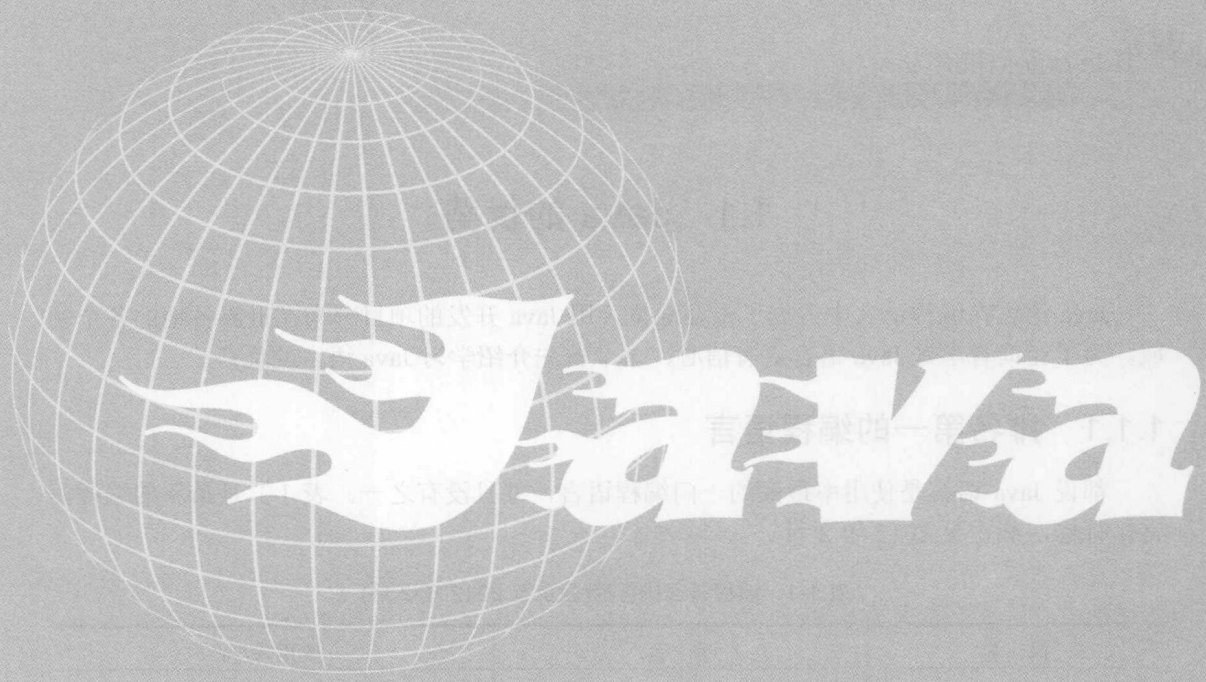
**读者对象**

初学编程的自学者
大中专院校的老师和学生
毕业设计的学生
程序测试及维护人员
在职程序员
编程爱好者
相关培训机构的老师和学员
初中级程序开发人员
参加实习的初级级程序员
资深程序员

本团队在编写编写过程中，得到了清华大学出版社工作人员的大力支持。但是本团队水平毕竟有限，如有纰漏和不尽如人意之处在所难免，诚请读者提出意见或建议，以便修订并使之更臻完善。另外，为了方便读者学习，我们特开通了技术支持 QQ 群，群号为 75593028，欢迎读者加入本群。

编　者

# 目　录




# 第1章

# 一起走进Java世界

几乎所有学习计算机的朋友都听说过 Java 语言，Java 语言“犀利无比”，已占据了当前软件应用的半壁江山。本章将首先带领读者领略 Java 这门强大的语言，仔细品味它的每一个强大之处，并且详细分析 Java 技术体系的构成，为 Java 程序员规划发展之路。

# 1.1 Java 的优势

Java 语言在编程语言中占据了重要地位，用 Java 开发的项目分布在世界各地的各个领域。为了让读者学好 Java 语言更有信心，本节首先介绍学习 Java 语言的优势。

## 1.1.1 排名第一的编程语言

都说 Java 语言是使用率最高的一门编程语言，并且没有之一。表 1-1 是最新的编程语言排行榜，截止至 2012 年 2 月。

表 1-1 编程语言排行榜(截止至 2012 年 2 月)

| 排 名 | 语 言 | 使用率/% |
|---|---|---|
| 1 | Java | 17.050 |
| 2 | C | 16.523 |
| 3 | C# | 8.653 |
| 4 | C++ | 7.853 |
| 5 | Object-C | 7.062 |
| 6 | PHP | 5.641 |

## 1.1.2 提供给我们美好的就业前景

由表 1-1 的统计数据可以看出，Java 语言是当今使用率最高的编程语言。使用率如此之高，也决定了 Java 程序员的就业机会要大于其他开发者。

Java 的功能比较强大，在服务器领域、移动设备、桌面应用和 Web 领域都占据了重要的地位。

- 服务器领域：Java 在服务器编程方面很强悍，拥有很多其他语言所没有的优势。
- 移动设备：Java 在手机领域的应用比较广泛，手机 Java 游戏随处可见，当前异常火爆的移动开发平台——Android，上面的应用项目基本上都是用 Java 开发的。
- 桌面应用：Java 和 C++、.NET 一样重要，影响着桌面程序的发展。
- Web 领域：Java Web 有着巨大的优势，无论是开发工具还是开发框架都是开源的，并且安全性更强。

正是因为 Java 语言可以在多个领域开发项目，所以市面上需要多个领域的 Java 程序员。据统计，当前全球有 30 亿 Java 器件正在运行着 Java 程序，500 多万 Java 开发者活跃在地球的每个角落，数以千万计的 Web 用户每次上网都亲历 Java 的威力。今天，Java 运行在 9.08 亿手机、10 亿智能卡和 10 亿 PC 上，并为 28 款可兼容的应用服务器提供了功能强大的平台。这么多应用，彻底改变了用户的生活。越来越多的企业，因为使用了 Java 而提高了生产效率。在中国，越来越多的用户，因为 Java 而降低了成本，享受了生活。

作为唯一在互联网上开发的语言，Java 平台以其移动性、安全性和开放性受到追捧。据

IDC 预计，自 2010 年起的其后 5 年内，采用 Java 的 IT 产品的价值将翻番，在 2015 年将达到 45.53 亿美元，年增长率为 14.9%。截止到 2010 年 5 月，Java 的注册开发商超过 1300 万人，对 JRE(Java 运行环境)的下载达 9200 万次。詹姆斯·戈士林博士预计在 3～5 年内 Java 技术开发商将发展到 2000 万，无线 Java 也在迅速攀升。

上述发展趋势在国内更是如此，当前我国对软件人才的需求已达 50 万，并且以每年 20%左右的速度增长。在未来 5 年内，合格软件人才的需求将远大于供给。在过去的 2011 年，我国软件人才的缺口已达 52.5 万人，其中尤以 Java 人才最为缺乏。

根据 IDC 的统计数字，在所有软件开发类人才的需求中，对 Java 工程师的需求达到全部需求量的 60%~70%。这也就不难解释在智联招聘和 51job 等主流招聘网站，随处可见 Java 工程师的招聘广告了。

## 1.2　学习 Java 需要了解的那些事

了解了学习 Java 语言的优势之后，接下来需要了解和这门神奇语言相关的几件大事，下面介绍这门神奇语言的发展历史和特点等，为读者学习本书后面的知识打下理论基础。

### 1.2.1　品 Java 语言的发展历史

Java 是由 Sun Microsystems 公司于 1995 年 5 月推出的 Java 程序设计语言(以下简称 Java 语言)和 Java 平台的总称。用 Java 实现的 HotJava 浏览器(支持 Java Applet)向我们展示了 Java 语言的魅力——跨平台、动态的 Web、Internet 计算。从那以后，Java 便被广大程序员和企业用户广泛接受，成为了当今最受欢迎的编程语言之一。

Java 平台由 Java 虚拟机(Java Virtual Machine)和 Java 应用编程接口(Application Programming Interface，API)构成。Java 应用编程接口为 Java 应用提供了一个独立于操作系统的标准接口，可分为基本部分和扩展部分。在硬件或操作系统平台上安装一个 Java 平台之后，Java 应用程序就可运行。现在 Java 平台已经嵌入了几乎所有的操作系统。这样 Java 程序可以只编译一次，就在各种系统中运行。Java 应用编程接口已经从 1.1x 版发展到 1.2 版。目前常用的 Java 平台基于 Java 1.6，最新版本为 Java 1.7。

当 1995 年 Sun 公司推出 Java 语言之后，全世界的目光都被这个神奇的语言所吸引。那么 Java 到底有何神奇之处呢？Java 语言其实最早诞生于 1991 年，起初被称为 OAK 语言，是 Sun 公司为一些消费性电子产品而设计的一个通用环境。Sun 公司的最初目的是为了开发一种独立于平台的软件技术，而且在网络出现之前 OAK 是默默无闻的，甚至差一点夭折。但是随着网络的出现彻底改变了 OAK 的命运。在 Java 出现以前，Internet 上的信息都是一些乏味死板的 HTML 文档。这对于那些迷恋于 Web 浏览的人们来说简直不可容忍。他们迫切希望能在 Web 中看到一些交互式的内容，开发人员也极希望能够在 Web 上创建一类无需考虑软硬件平台就可以执行的应用程序，当然这些程序还要有极大的安全保障。对于用户的这种要求，传统的编程语言显得无能为力。Sun 的工程师敏锐地察觉到了

这一点，从 1994 年起，他们开始将 OAK 技术应用于 Web 上，并且开发出了 HotJava 的第一个版本。并最终在 1995 年，将 Java 技术展现在了世人的面前。

Java 分为以下三个体系。

- JavaSE：是 Java 2 Platform Standard Edition 的缩写，即 Java 平台标准版。
- JavaEE：是 Java 2 Platform Enterprise Edition 的缩写，即 Java 平台企业版。
- JavaME：是 Java 2 Platform Micro Edition 的缩写，即 Java 平台微型版。

2009 年 4 月 20 日，Oracle(甲骨文)宣布成功收购 Sun 公司，从那以后，Java 成为了软件巨头甲骨文旗下的一款产品，并和 Oracle 数据库一起推动着世界科技继续向前发展。

### 1.2.2 Java 的特点

- 面向对象：Java 语言提供了类、接口和继承等特性。为了简单起见，Java 只支持类之间的单继承和接口之间的多继承，并且也支持类与接口之间的实现机制。总之，Java 语言是一门纯粹面向对象的程序设计语言。
- 简单：Java 语言的语法与 C 语言和 C++语言十分接近，这样大多数程序员可以很容易地学习和使用 Java。并且 Java 抛弃了 C++中的操作符重载、多继承、自动强制类型和指针等知识，这将更加利于我们学习并掌握它。另外，Java 还提供了自动废料收集机制，使得程序员不必再为内存管理而担忧。
- 分布式：Java 语言支持 Internet 应用开发，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)，通过这个接口提供了用于网络应用编程的类库，包括 URL、URLConnection、Socket、ServerSocket 等。Java 的 RMI(远程方法激活)机制也是开发分布式应用的重要手段。
- 健壮：Java 的强类型机制、异常处理、废料的自动收集等是 Java 程序健壮性的重要保证。Java 通过安全检查机制，使 Java 程序更具健壮性。
- 可移植：移植性是指能够在不同的开发平台和服务器平台上使用。因为 Java 的运行环境是用 ANSI C 实现的，所以 Java 系统本身具有很强的可移植性，可以在很多平台上运行。无论是微软的产品还是 IBM 的产品，都可以运行 Java 程序。
- 高性能：与解释型的高级脚本语言相比，Java 的确是高性能的。随着 JIT(Just-In-Time)编译器技术的发展，Java 的运行速度已经越来越接近于 C++。
- 多线程：当程序需要同时处理多项任务时就需要多线程开发，一个程序在同一时间只能做一件事情的功能过于简单，肯定无法满足现实的需求。在实际的应用中，多线程开发是必不可少的，多线程的目的是在同一时间可以做多件事情。并且可以开启多个线程同时做一件事情，这样可以提高效率。不管是 C 语言、C++还是其他的程序设计语言，线程都是一个十分重要的知识点，多线程是现代开发软件系统的发展方向，Java 作为当今的主流程序设计，它当然是支持多线程的，具有并发性，其执行的效率很高。
- 动态：Java 语言的设计目标之一是适应于动态变化的环境。Java 程序中的类需要动态地被载入到运行环境中，也可以通过网络来载入所需要的类。动态语言的好

处是有利于软件升级。另外，Java 中的类有一个运行时刻的表示，能进行运行时刻的类型检查。

## 1.3　剖析 Java 的运行机制

Java 语言是一门高级语言，它既有解释型语言的特性，也具有编译语言的特性。Java 需要先编译，然后再解释运行。本节将简要介绍 Java 语言的运行机制，更加深入地介绍这门神奇的语言。

### 1.3.1　高级语言的运行机制

高级语言按照执行方式可以分为解释型和编译型两种。

1) 解释型语言

计算机不能直接理解高级语言，只能直接理解机器语言，所以必须要把高级语言翻译成机器语言，计算机才能执行高级语言编写的程序。

翻译的方式有两种，一种是编译，一种是解释。

解释型语言的程序不需要编译，省了道工序，在运行程序的时候才翻译，比如解释型 Basic 语言，专门有一个解释器能够直接执行 Basic 程序，每个语句都是执行的时候才翻译。这样解释型语言每执行一次就要翻译一次，效率比较低，是一句一句地翻译。

2) 编译型语言

编译型语言写的程序执行之前，需要一个专门的编译过程，把程序编译成为机器语言的文件，比如 exe 文件，以后要运行的话就不用重新翻译了，直接使用编译的结果就行了(exe 文件)，因为翻译只做了一次，运行时不需要翻译，所以编译型语言的程序执行效率高。

编译型与解释型，两者各有利弊。前者由于程序执行速度快，同等条件下对系统要求较低，因此像开发操作系统、大型应用程序、数据库系统等时都采用它，像 C/C++、Pascal/Object Pascal(Delphi)等都是编译语言，而一些网页脚本、服务器脚本及辅助开发接口这样的对速度要求不高、对不同系统平台间的兼容性有一定要求的程序则通常使用解释型语言，如 Java、JavaScript、VBScript、Perl、Python、Ruby、MATLAB 等。

但随着硬件的升级和设计思想的变革，编译型和解释型语言越来越笼统，主要体现在一些新兴的高级语言上，而解释型语言的自身特点也使得编译器厂商愿意花费更多成本来优化解释器，解释型语言性能超过编译型语言也是必然的。

### 1.3.2　Java 的运行机制

Java 应用程序的开发周期包括编译、下载、解释和执行几个部分。Java 编译程序将 Java 源程序翻译为 JVM 可执行代码——字节码。这一编译过程同 C/C++的编译有些不同。当 C 编译器编译生成一个对象的代码时，该代码是为在某一特定硬件平台运行而产生

的。因此在编译过程中，编译程序通过查表将所有对符号的引用转换为特定的内存偏移量，目的是保证程序运行。Java 编译器不将对变量和方法的引用编译为数值引用，也不确定程序执行过程中的内存布局，而是将这些符号引用信息保留在字节码中，由解释器在运行过程中创立内存布局，然后再通过查表来确定一个方法所在的地址。这样就有效地保证了 Java 的可移植性和安全性。

运行 JVM 字节码的工作是由解释器(Java 命令)来完成的。整个解释执行过程分为以下三步进行。

- 代码的装入。
- 代码的校验。
- 代码的执行。

装入代码的工作由“类装载器”(Class Loader)完成。类装载器负责装入运行一个程序需要的所有代码，这也包括程序代码中的类所继承的类和被其调用的类。当类装载器装入一个类时，该类被放在自己的名字空间中。除了通过符号引用自己名字空间以外的类，类之间没有其他办法可以影响其他类。在本台计算机上的所有类都在同一地址空间内，而所有从外部引进的类，都有一个自己独立的名字空间。这使得本地类通过共享相同的名字空间获得较高的运行效率，同时又保证它们与从外部引进的类不会相互影响。当装入了运行程序需要的所有类后，解释器便可确定整个可执行程序的内存布局。解释器为符号引用同特定的地址空间建立对应关系及查询表。通过在这一阶段确定代码的内存布局，Java 很好地解决了由超类改变而使子类崩溃的问题，同时也防止了代码对地址的非法访问。

接下来，被装入的代码由字节码校验器进行检查。校验器可发现操作数栈溢出、非法数据类型转化等多种错误。通过校验后，代码便开始执行了。

有如下两种 Java 字节码的执行方式。

(1) 即时编译方式：解释器先将字节码编译成机器码，然后再执行该机器码。

(2) 解释执行方式：解释器通过每次解释并执行一小段代码来完成 Java 字节码程序的所有操作。

在 Java 中通常采用的是第二种方法，因为 JVM 规格描述具有足够的灵活性，这使得将字节码翻译为机器代码的工作具有较高的效率。对于那些对运行速度要求较高的应用程序，解释器可将 Java 字节码即时编译为机器码，从而很好地保证了 Java 代码的可移植性和高性能。

Java 程序的运行必须经过编写、编译、运行三个步骤。

- 编写是指在 Java 开发环境中进行程序代码的输入，最终形成后缀名为.java 的 Java 源文件。
- 编译是指使用 Java 编译器对源文件进行错误排查的过程，编译后将生成后缀名为.class 的字节码文件，这不像 C 语言那样最终生成可执行文件。
- 运行是指使用 Java 解释器将字节码文件翻译成机器代码，执行并显示结果。这一过程如图 1-1 所示。

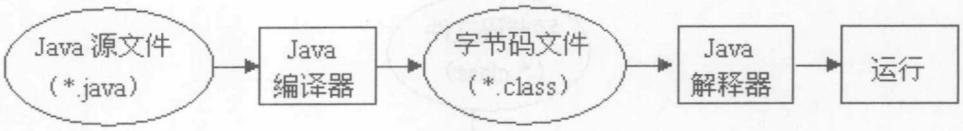



图 1-1　Java 程序的运行流程

在图 1-1 中，字节码文件是一种和任何具体机器环境及操作系统环境无关的中间代码，它是一种二进制文件，是 Java 源文件由 Java 编译器编译后生成的目标代码文件。编程人员和计算机都无法直接读懂字节码文件，它必须由专用的 Java 解释器来解释执行，因此 Java 是一种在编译基础上进行解释运行的语言。

Java 解释器负责将字节码文件翻译成具体硬件环境和操作系统平台下的机器代码，以便执行。因此 Java 程序不能直接运行在现有的操作系统平台上，它必须运行在被称为 Java 虚拟机的软件平台之上。

### 1.3.3　Java 虚拟机——JVM

JVM 是一种用于计算设备的规范，可用不同的方式(软件或硬件)加以实现。编译虚拟机的指令集与编译微处理器的指令集非常类似。Java 虚拟机包括一套字节码指令集、一组寄存器、一个栈、一个垃圾回收堆和一个存储方法域。

Java 虚拟机(JVM)是可运行 Java 代码的假想计算机。只要根据 JVM 规格描述将解释器移植到特定的计算机上，就能保证经过编译的任何 Java 代码能够在该系统上运行。

1) 为什么要使用 JVM

Java 语言的一个非常重要的特点就是与平台的无关性，而使用 Java 虚拟机是实现这一特点的关键。一般的高级语言如果要在不同的平台上运行，至少需要编译成不同的目标代码。而引入 Java 语言虚拟机后，Java 语言在不同平台上运行时不需要重新编译。Java 语言使用模式 Java 虚拟机屏蔽了与具体平台相关的信息，使 Java 语言编译程序只需生成在 Java 虚拟机上运行的目标代码(字节码)，就可以在多种平台上不加修改地运行。Java 虚拟机在执行字节码时,把字节码解释成具体平台上的机器指令执行。

2) JVM 的作用

Java 虚拟机(JVM)是运行 Java 程序的软件环境，Java 解释器就是 Java 虚拟机的一部分。在运行 Java 程序时，首先会启动 JVM，然后由它来负责解释执行 Java 的字节码，并且 Java 字节码只能运行于 JVM 之上。这样利用 JVM 就可以把 Java 字节码程序和具体的硬件平台以及操作系统环境分隔开来，只要在不同的计算机上安装了针对于特定具体平台的 JVM，Java 程序就可以运行，而不用考虑当前具体的硬件平台及操作系统环境，也不用考虑字节码文件是在何种平台上生成的。JVM 把这种不同软硬件平台的具体差别隐藏起来，从而实现了真正的二进制代码级的跨平台移植。JVM 是 Java 平台无关的基础，Java 的跨平台特性正是通过在 JVM 中运行 Java 程序实现的。Java 的这种运行机制可以通过图 1-2 来说明。