

高等学校计算机应用规划教材

Java 基础教程

(第 4 版)

吴仁群 编著

清华大学出版社

北 京

内 容 简 介

本书是针对 Java 语言初学者编写的基础教程，包含 Java 程序设计的基础知识，以及大量实用性很强的编程实例。全书共分 11 章，包括 Java 语言概述、Java 语言基础、类与对象、继承与接口、数组与字符串、Java 的异常处理机制、Java 常见类库、输入输出及数据库操作、多线程、Applet 程序及应用和图形用户界面设计。

本书内容实用，结构清晰，实例丰富，可操作性强，可作为高等学校 Java 程序设计课程的教材，也可作为计算机相关专业的培训和自学教材。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。举报：010-62782989，beiqinquan@tup.tsinghua.edu.cn。

图书在版编目(CIP)数据

Java 基础教程 / 吴仁群编著. —4 版. —北京：清华大学出版社，2021.3

高等学校计算机应用规划教材

ISBN 978-7-302-55060-0

I. ①J… II. ①吴… III. ①JAVA 语言—程序设计—高等学校—教材 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2020)第 039173 号

责任编辑：刘金喜

封面设计：高娟妮

版式设计：孔祥峰

责任校对：牛艳敏

责任印制：丛怀宇

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

印 装 者：大厂回族自治县彩虹印刷有限公司

经 销：全国新华书店

开 本：185mm×260mm 印 张：19.25 字 数：556 千字

版 次：2009 年 4 月第 3 版 2021 年 3 月第 4 版 印 次：2021 年 3 月第 1 次印刷

定 价：59.00 元

产品编号：086655-01

前 言

《Java 基础教程(第 2 版)》已于 2013 年 10 月被评为北京市精品教材。为反映 Java 发展的新特点和新进展,以及为使教程的结构更清晰,内容描述更简洁明了,以便更适合于初学者学习,本书作者对部分章节进行了修改和完善,主要包括:①对概念进行统一,使得前后章节一致。如将操作元和操作数统一为操作元,将类变量和静态变量统一为静态变量,实例变量和非静态变量统一为非静态变量等。②对部分章节的观点表述进行进一步梳理,使其更简洁、更规范、更容易明白,且不容易出现二义性。③对许多章节的实例进行了更新或补充,以便更能反映所要表达的知识点或知识点的运用,如 `switch(expr)` 语句应用中,以前举例涉及的 `expr` 为整型或字符型,事实上 `expr` 也可以为字符串型,因此增加 `expr` 为字符串型的实例是必要的;又如增加接口继承的实例,以期对接口应用有更深刻的认识等。④对多线程一章进行大幅度修正,剔除一些淘汰技术,反映最新技术。⑤更正了部分编辑错误。

修改后的第 4 版仍然有 11 章:第 1 章讲述 Java 语言发展历程、Java 语言的特点及开发平台和开发过程;第 2 章介绍 Java 语言编程的基础语法知识;第 3 章和第 4 章讲述 Java 的面向对象技术,体现了 Java 作为一种纯粹的面向对象编程语言的编程特点;第 5 章介绍数组和字符串的特点及使用;第 6 章介绍 Java 语言的异常处理机制;第 7 章介绍 Java 类库结构及常用类库;第 8 章介绍 Java 语言中输入输出流和数据库操作方法;第 9 章介绍 Java 语言多线程的含义、特点及实现;第 10 章介绍 Applet 程序的概念及其应用;第 11 章介绍在 Java 语言中如何进行图形用户界面设计及处理功能的实现。

本书 PPT 课件、案例源文件和习题答案等教学资源可通过 <http://www.tupwk.com.cn/downpage> 下载。

本书由北京印刷学院吴仁群编著。在编写过程中,编者参考了本书“参考文献”所列举的图书,得到了清华大学出版社的大力支持,在此对“参考文献”中所列图书的作者及清华大学出版社表示深深的感谢。

由于时间仓促,书中难免存在一些不足之处,敬请读者批评指正。

编 者
2020 年 10 月

目 录

第 1 章 Java 语言概述	1
1.1 Java语言的特点及相关概念.....	1
1.1.1 Java语言的发展历程.....	1
1.1.2 Java语言的特点.....	2
1.1.3 Java虚拟机(JVM).....	4
1.2 Java程序开发.....	5
1.2.1 运行平台.....	5
1.2.2 Java程序开发过程.....	9
1.3 综合应用.....	12
1.4 本章小结.....	13
1.5 思考和练习.....	14
第 2 章 Java 语言基础	15
2.1 Java程序概况.....	15
2.1.1 Java程序结构.....	15
2.1.2 Java注释.....	16
2.1.3 Java关键字.....	16
2.1.4 Java标识符.....	17
2.1.5 变量与常量.....	17
2.2 基本数据类型.....	18
2.2.1 基本数据类型概况.....	18
2.2.2 基本数据类型转换.....	22
2.3 运算符和表达式.....	24
2.3.1 算术运算符和算术表达式.....	24
2.3.2 关系运算符与关系表达式.....	26
2.3.3 逻辑运算符与逻辑表达式.....	27
2.3.4 赋值运算符与赋值表达式.....	27
2.3.5 位运算符.....	28
2.3.6 条件运算符.....	29
2.3.7 instanceof运算符.....	29
2.3.8 一般表达式.....	29
2.4 Java语句.....	31
2.4.1 Java语句概述.....	31
2.4.2 分支语句.....	31
2.4.3 循环语句.....	37
2.4.4 跳转语句.....	39
2.5 综合应用.....	42
2.6 本章小结.....	47
2.7 思考和练习.....	47
第 3 章 类与对象	49
3.1 面向对象基础.....	49
3.1.1 编程语言的4个发展阶段.....	49
3.1.2 面向过程的程序设计.....	50
3.1.3 面向对象的程序设计.....	50
3.1.4 两种程序设计语言的简单比较.....	52
3.2 类.....	53
3.2.1 类的定义.....	53
3.2.2 成员变量.....	54
3.2.3 成员方法.....	56
3.3 对象.....	58
3.3.1 对象的创建.....	58
3.3.2 对象的使用.....	59
3.3.3 对象的消亡.....	60
3.4 变量.....	61
3.4.1 类中变量的分类.....	61
3.4.2 变量的内存分配.....	62
3.4.3 实例变量和静态变量的简单比较.....	63
3.4.4 变量初始化与赋值.....	65
3.5 方法.....	68

3.5.1 方法概述	68	第5章 数组与字符串	121
3.5.2 方法分类	68	5.1 数组	121
3.5.3 方法调用中的数据传递	71	5.1.1 数组定义及说明	121
3.5.4 三个重要方法	74	5.1.2 数组应用举例	125
3.5.5 方法的递归调用	78	5.2 字符串	127
3.6 package和import语句	79	5.2.1 String类	128
3.6.1 package语句	79	5.2.2 StringBuffer类	130
3.6.2 import语句	81	5.2.3 应用举例	131
3.7 访问权限	82	5.3 综合应用	133
3.7.1 类的访问控制	82	5.4 本章小结	136
3.7.2 类成员的访问控制	84	5.5 思考和练习	137
3.8 综合应用	87	第6章 Java的异常处理机制	139
3.9 本章小结	91	6.1 异常的含义及分类	139
3.10 思考和练习	91	6.2 异常处理	140
第4章 继承与接口	94	6.2.1 异常处理的含义及必要性	140
4.1 继承	94	6.2.2 异常处理的基本结构	140
4.1.1 继承的含义	94	6.2.3 多个catch块	142
4.1.2 子类的继承性访问控制	95	6.2.4 finally语句	142
4.1.3 子类对象的构造过程	98	6.3 两种抛出异常的方式	144
4.1.4 子类的内存分布	98	6.3.1 throw——直接抛出异常	144
4.1.5 子类对象的成员初始化	100	6.3.2 throws——间接抛出异常(声明异常)	147
4.1.6 成员变量的隐藏	101	6.4 自定义异常	148
4.1.7 方法的重载与方法的覆盖	102	6.5 常见异常	149
4.1.8 this关键字	105	6.6 综合应用	150
4.1.9 super关键字	108	6.7 本章小结	152
4.1.10 对象的上下转型对象	109	6.8 思考和练习	152
4.2 接口	109	第7章 Java常见类库	153
4.2.1 abstract类	109	7.1 Java类库的结构	153
4.2.2 接口的定义	110	7.2 常用类	154
4.2.3 接口回调	113	7.2.1 System类	154
4.2.4 接口与抽象类的异同	114	7.2.2 Math类	158
4.3 特殊类	114	7.2.3 随机数类Random	160
4.3.1 final类	114	7.2.4 基本数据类型的包装类	161
4.3.2 内部类	115	7.2.5 Vector类	164
4.4 综合应用	116	7.2.6 Stack类	168
4.5 本章小结	119	7.2.7 Queue类	170
4.6 思考和练习	119	7.2.8 Arrays类	173

7.2.9 哈希表类Hashtable	176	9.4.1 共享变量和方法封装在一个类中	228
7.3 本章小结	179	9.4.2 通过系统方法实现线程通信	229
7.4 思考和练习	179	9.5 本章小结	233
第8章 输入输出及数据库操作	180	9.6 思考和练习	233
8.1 输入和输出	180	第10章 Applet 程序及应用	234
8.1.1 流的含义	180	10.1 Applet程序基础	234
8.1.2 流的层次结构	181	10.1.1 Applet程序概述	234
8.1.3 标准输入输出	182	10.1.2 Applet类	236
8.1.4 File类	183	10.1.3 Applet程序的生命周期	237
8.1.5 FileInputStream类和 FileOutputStream类	185	10.1.4 Applet的显示	238
8.1.6 DataInputStream类和 DataOutputStream类	187	10.1.5 Applet程序和Application程序 结合使用	239
8.1.7 随机访问文件	190	10.2 Applet程序典型应用	241
8.1.8 Reader类和Writer类	193	10.2.1 图形绘制	241
8.1.9 IOException类的4个子类	194	10.2.2 获取图像	245
8.1.10 综合应用	194	10.2.3 音频处理	246
8.2 数据库操作	201	10.2.4 动画处理	247
8.2.1 ODBC概述	201	10.3 综合应用	249
8.2.2 JDBC概述	202	10.4 本章小结	251
8.2.3 使用JDBC-ODBC技术访问 数据库	204	10.5 思考和练习	251
8.2.4 综合应用	206	第11章 图形用户界面设计	252
8.2.5 基本SQL语句	209	11.1 Java的AWT和Swing基础	252
8.3 建立数据源的操作	211	11.1.1 Java的AWT和Swing 概述	252
8.4 本章小结	213	11.1.2 Java的AWT组件和Swing组件	253
8.5 思考和练习	214	11.1.3 利用AWT组件和Swing组件进行 程序设计的基本步骤	255
第9章 多线程	215	11.2 常用容器	256
9.1 多线程的概念	215	11.2.1 框架	256
9.2 线程类	216	11.2.2 面板	258
9.2.1 多线程编程中常用的常量和方法	216	11.2.3 滚动窗口	259
9.2.2 线程的生命周期	217	11.2.4 菜单设计	261
9.2.3 创建多线程的方法	218	11.2.5 对话框	263
9.3 资源的协调与同步	223	11.3 布局管理器	266
9.3.1 线程调度模型	223	11.3.1 FlowLayout布局	266
9.3.2 资源冲突	224	11.3.2 BorderLayout布局	267
9.3.3 同步方法	225	11.3.3 GridLayout布局	269
9.4 线程间通信	228	11.3.4 CardLayout布局	269

11.3.5	null布局	271	11.5.5	复选框	285
11.4	事件处理	272	11.5.6	单选框	287
11.4.1	委托事件模型	272	11.5.7	选择框	290
11.4.2	键盘事件	275	11.5.8	列表	291
11.4.3	鼠标事件	276	11.6	综合应用	293
11.5	常用组件	278	11.7	本章小结	298
11.5.1	按钮	279	11.8	思考和练习	299
11.5.2	标签	281	参考文献		300
11.5.3	文本行	282			
11.5.4	文本域	284			

第 1 章

Java语言概述

Java 语言是目前使用最为广泛的编程语言之一，是一种简单、面向对象、分布式、解释、健壮、安全、与平台无关的并且性能优异的多线程动态语言。

本章的学习目标：

- 了解 Java 语言的发展历程。
- 理解 Java 语言的特点。
- 理解 Java 虚拟机 JVM。
- 掌握 Java 运行平台的安装与使用。
- 掌握 Java 程序开发的过程。
- 学会调试简单的 Java 程序。

1.1 Java 语言的特点及相关概念

1.1.1 Java 语言的发展历程

Java 语言的前身是 Oak 语言。1991 年 4 月，Sun 公司(已被 Oracle 公司收购)以 James Gosling 为首的绿色计划项目组(green project)计划发展一种分布式系统结构，使其能够在各种消费性电子产品上运行。项目组成员在使用 C++编译器时发现其有很多不足之处，于是研发出 Oak 语言来替代它，但仅限于 Sun 公司内部使用。

1994 年下半年，由于 Internet 的迅速发展和 Web 的广泛应用，工业界迫切需要一种能够在异构网络环境下使用的语言，James Gosling 项目组在对 Oak 语言进行小规模改造的基础上于 1995 年 3 月推出了 Java 语言，并于 1996 年 1 月发布了包含开发支持库的 JDK 1.0 版本。该版本包括 Java 运行环境(JRE)和 Java 开发工具箱(Java Development Kit, JDK)，其中 JRE 包括核心 API、集成 API、用户界面 API、发布技术及 JVM(Java 虚拟机)5 个部分，而 JDK 包括编译 Java 程序的编译器(javac)。在 JDK 1.0 版本中，除了 AWT 外，其他的库并不完整。

1997 年 2 月，Sun 公司发布了 JDK 1.1 版本，为 JVM 增加了即时编译器(JIT)。与传统的编译器编译一条指令待其运行完后再释放掉不同的是，JIT 将常用的指令保存在内存中，这样在下次调用时就没有必要再编译。继 JDK 1.1 版本后，Sun 公司又推出了数个 JDK 1.x 版本。

虽然在 1998 年之前，Java 被众多的软件企业所采用，但由于当时硬件环境和 JVM 的技术尚不成熟，它的应用很有限，那时 Java 主要应用在前端的 Applet 及一些移动设备中，然而这并不等于 Java 的应用只限于这些领域。1998 年是 Java 迅猛发展的一年，在该年 Sun 发布了 JSP/Servlet、EJB

规范，以及将 Java 分成了 J2EE、J2SE 和 J2ME，标志着 Java 已经吹响了向企业、桌面和移动 3 个领域进军的号角。

1998 年 12 月，Sun 公司发布了 JDK 1.2 版本，该版本是 Java 语言发展过程中的一个关键阶段，从此 Sun 公司将 Java 更名为 Java 2。经过 10 年的发展，Java 语言已经发展到 1.6 版本。

JDK 1.2 版本可分为 J2EE、J2SE 和 J2ME 三大应用平台。JDK 1.2 版本的 API 分成了核心 API、可选 API 和特殊 API 三大类，其中核心 API 是由 Sun 公司制定的基本的 API，所有的 Java 平台都应该提供，可选 API 是 Sun 为 JDK 提供的扩充 API，特殊 API 是用于满足特殊要求的 API。同时，JDK 1.2 增加了 Swing 图形库，包含各式各样的组件。

从 JDK 1.2 版本开始，Sun 以平均两年一个版本的速度推出新的 JDK。2000 年 5 月，Sun 公司发布了 JDK 1.3 版本；2002 年 2 月，Sun 公司发布了 JDK 1.4 版本；2004 年 10 月，Sun 公司发布了 JDK 1.5 版本，同时，Sun 公司将 JDK 1.5 改名为 JDK 5.0。2006 年 4 月，发布了 JDK 6.0 测试版本，并于 2007 年初发布了 JDK 6.0 正式版本；2011 年 7 月发布了 JDK 7.0 版本。2014 年 3 月 18 日，Oracle 公司发表 Java SE 1.8。

在 Java 发展的十几年的时间里，经历了无数的风风雨雨，现在 Java 已经成为一种相当成熟的语言了。在这 10 年的发展中，Java 平台吸引了数百万的开发者，在网络计算遍及全球的今天，Java 已广泛应用于移动电话、桌面计算机、蓝光光碟播放器、机顶盒甚至车载，更是有 30 多亿台设备使用了 Java 技术。

1.1.2 Java 语言的特点

作为一种面向对象且与平台无关的多线程动态语言，Java 具有以下特点。

1. 语法简单

Java 语言的简单性主要体现在以下 3 个方面。

- Java 的风格类似于 C++，C++ 程序员可以很快掌握 Java 编程技术。
- Java 摒弃了 C++ 中容易引发程序错误的地方，如指针和内存管理。
- Java 提供了丰富的类库。

2. 面向对象

面向对象编程是一种先进的编程思想，更加容易解决复杂的问题。面向对象可以说是 Java 最重要的特性。Java 语言的设计完全是面向对象的，它不支持类似 C 语言那样的面向过程的程序设计技术。Java 支持静态和动态风格的代码继承及重用。单从面向对象的特性来看，Java 类似于 SmallTalk，但其他特性，尤其是适用于分布式计算环境的特性远远超越了 SmallTalk。

3. 分布式

Java 从诞生起就与网络联系在一起，它强调网络特性，内置 TCP/IP、HTTP 和 FTP 协议类库，便于开发网上应用系统。因此，Java 应用程序可凭借 URL 打开并访问网络上的对象，其访问方式与访问本地文件系统完全相同。

4. 安全性

Java 的安全性可从两个方面得到保证：一方面，在 Java 语言中，如指针和释放内存等 C++ 中的功能被删除，避免了非法内存操作；另一方面，当 Java 用来创建浏览器时，语言功能和一些浏览器本身提供的功能结合起来，使它更安全。Java 语言在机器上执行前，要经过很多次的测试，其三级

安全检验机制可以有效防止非法代码入侵，阻止对内存的越权访问。

5. 健壮性

Java 致力于检查程序在编译和运行时的错误。除了运行时异常检查外，Java 还提供了广泛的编译时异常检查，以便尽早发现可能存在的错误。类型检查帮助用户检查出许多早期开发中出现的错误。Java 自己操纵内存减少了内存出错的可能性。Java 还实现了真数组，避免了覆盖数据的可能，这项功能大大缩短了开发 Java 应用程序的周期。Java 提供 Null 指针检测数组边界及检测异常出口字节代码校验。同时，在 Java 中对象的创建机制(只能用 new 操作符)和自动垃圾收集机制大大减少了因内存管理不当引发的错误。

6. 解释运行效率高

Java 解释器(运行系统)能直接运行目标代码指令。Java 程序经编译器编译，生成的字节码经过精心设计，并进行了优化，因此运行速度较快，克服了以往解释性语言运行效率低的缺点。Java 用直接解释器 1 秒钟内可调用 300 000 个过程。翻译目标代码的速度与 C/C++没什么区别。

7. 与平台无关

Java 编译器将 Java 程序编译成二进制代码，即字节码。字节码有统一的格式，不依赖于具体的硬件环境。

平台无关类型包括源代码级和目标代码级两种：C 和 C++属于源代码级与平台无关，意味着用它编写的应用程序不用修改只需重新编译就可以在不同平台上运行；Java 属于目标代码级与平台无关，主要靠 Java 虚拟机(JVM, Java Virtual Machine)来实现。

8. 多线程

Java 提供的多线程功能使得在一个程序里可同时执行多个小任务。线程有时也称作小进程，是从一个大进程中分出来的小的独立的进程。由于 Java 实现了多线程技术，所以比 C 和 C++更健壮。多线程带来的更大好处是更好的交互性能和实时控制性能。当然实时控制性能还取决于系统本身(UNIX、Windows、Macintosh 等)，在开发难易程度和性能上都比单线程要好。例如，上网时都会感觉为查看一幅图片而等待是一件很烦恼的事情，而在 Java 中，可用一个单线程来查看一幅图片，同时可以访问 HTML 中的其他信息而不必等待它。

9. 动态性

Java 的动态特性是其面向对象设计时方法的发展，它允许程序动态装入运行过程中所需要的类，这是 C++语言进行面向对象程序设计时所无法实现的。在 C++程序设计过程中，每当在类中增加一个实例变量或一种成员函数后，引用该类的所有子类都必须重新编译，否则将导致程序崩溃。Java 编译器不是将对实例变量和成员函数的引用编译为数值引用，而是将符号引用信息在字节码中保存下来传递给解释器，再由解释器在完成动态连接后，将符号引用信息转换为数值偏移量。这样，一个在存储器中生成的对象不是在编译过程中确定，而是延迟到运行时由解释器确定的，因此对类中的变量和方法进行更新时就不至于影响现存的代码。解释执行字节码时，这种符号信息的查找和转换过程仅在一个新的名字出现时才进行一次，随后代码便可以全速执行。在运行时确定引用的好处是可以使用已被更新的类，而不必担心会影响原有的代码。如果程序连接了网络中另一系统中的某一类，则该类的所有者也可以自由对该类进行更新，而不会使任何引用该类的程序崩溃。Java 还简化了使用一个升级的或全新的协议的方法。当系统运行 Java 程序遇到了不知怎样处理的程序时，Java 能自动下载所需要的功能程序。

1.1.3 Java 虚拟机(JVM)

虚拟机是一种对计算机物理硬件计算环境的软件实现。虚拟机是一种抽象机器，内部包含一个解释器(interpreter)，可以将其他高级语言编译为虚拟机的解释器能执行的代码，我们称这种代码为中间语言(intermediate language)，实现高级语言程序的可移植性与平台无关性(system independence)。无论是运行在嵌入式设备还是多个处理器的服务器上，虚拟机都执行相同的指令，所使用的支持库也具有标准的 API 和完全相同或相似的行为。

Java 虚拟机是一种抽象机器，它附着在具体的操作系统上，本身具有一套虚拟机器指令，并有自己的栈、寄存器等运行 Java 程序不可少的机制。编译后的 Java 程序指令并不直接在硬件系统 CPU 上执行，而是在 JVM 上执行。在 JVM 上有一个 Java 解释器用来解释 Java 编译器编译后的程序，任何一台机器只要配备了解释器，就可以运行这个程序，而不管这种字节码是在何种平台上生成的。

JVM 是编译后的 Java 程序和硬件系统之间的接口，程序员可以把 JVM 看作一个虚拟处理器，它不仅解释执行编译后的 Java 指令，而且还进行安全检查，其是 Java 程序能在多平台间进行无缝移植的可靠保证，同时也是 Java 程序的安全检查引擎，如图 1-1 所示。

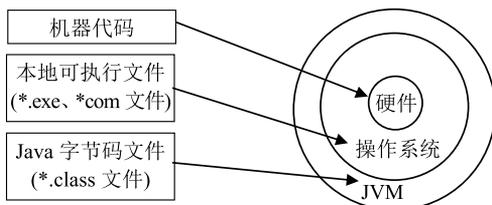


图 1-1 计算机硬件、操作系统、JVM 与各种可执行程序之间的关系

JVM 由多个组件构成，包括类装载器(class loader)、字节码解释器(bytecode interpreter)、安全管理器(Security Manager)、垃圾收集器(garbage collector)、线程管理(thread management)及图形(graphics)，如图 1-2 所示。

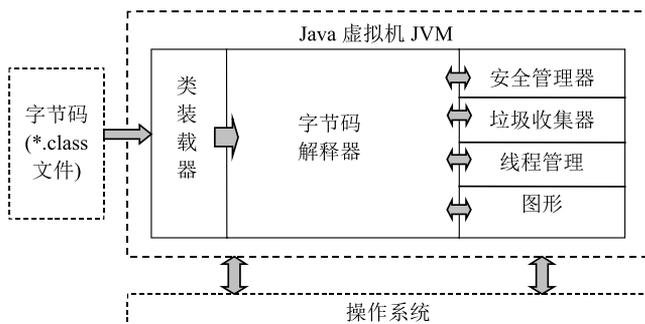


图 1-2 Java 虚拟机体系结构示意图

- **类装载器**：负责加载(load)类的字节码文件，并完成类的链接和初始化工作。类装载器首先将要加载的类名转换为类的字节码文件名，并在环境变量 CLASSPATH 指定的每个目录中搜索该文件，把字节码文件读入缓冲区。其次将类转换为 JVM 内部的数据结构，并使用校验器检查类的合法性。如果类是第一次被加载，则对类中的静态数据进行初始化。加载类中所引用的其他类，把类中的某些方法编译为本地代码。
- **字节码解释器**：它是整个 JVM 的核心组件，负责解释执行由类装载器加载的字节码文件中的字节码指令集合，并通过 Java 运行环境(JRE)由底层的操作系统实现操作。通过使用汇编

