第一章

简介

1.1 历史点滴

Java 编程语言是一种通用的、并发的、面向对象的语言。它的语法类似于 C 和 C++，但它省略了许多使 C 和 C++复杂、混乱和不安全的特性。开发 Java 平台最初是为了解决为网络消费设备构建软件的问题。它被设计成支持多主机架构，并允许软件组件的安全交付。为了满足这些要求，编译后的代码必须能够在网络传输中存活下来，在任何客户端上运行，并向客户端保证它是安全运行的。

万维网的普及使这些属性变得更加有趣。网络浏览器使数百万人能够在网上冲浪，并以简单的方式访问富媒体内容。最后有了一种媒介，无论你使用的是哪台机器，也不管它是连接到快速网络还是低速调制解调器，你的所见所闻基本上都是一样的。

网络爱好者很快发现，网络的超文本标记语言文档格式支持的内容太有限了。表单等 HTML 扩展只突出了这些限制，同时明确指出，任何浏览器都不能包含用户想要的所有功能。可扩展性就是答案。

通过在 HTML 页面中嵌入程序，HotJava 浏览器首次展示了 Java 编程语言和平台的有趣属性。程序和显示它们的 HTML 页面一起被透明地下载到浏览器中。在被浏览器接受之前，程序要经过仔细检查，以确保它们是安全的。与 HTML 页面一样，编译后的程序是独立于网络和主机的。这些程序的行为方式是相同的，不管它们来自哪里，也不管它们被加载到哪种机器上并在哪种机器上运行。

结合 Java 平台的 Web 浏览器不再局限于一组预先确定的功能。访问包含动态内容的 Web 页面的访问者可以确保他们的机器不会被这些内容损坏。 程序员只需要编写一次程序，它就可以在任何提供 Java 运行时环境的机器上运行。

1.2 Java 虚拟机

Java 虚拟机是 Java 平台的基石。它是一种技术的组成部分，负责其硬件和操作系统的独立性，其编译代码的小尺寸，以及其保护用户免受恶意程序攻击的能力。

Java 虚拟机是一种抽象的计算机器。与真正的计算机一样，它有一个指令集，并在运行时

操作各种内存区域。使用虚拟机实现编程语言是相当常见的;最著名的虚拟机可能是 UCSD Pascal 的 P-Code 机器。

Java 虚拟机的第一个原型实现是在 Sun Microsystems 公司完成的，它模拟了由手持设备(类似于当代的个人数字助理(PDA))托管的软件中的 Java 虚拟机指令集。Oracle 当前的实现模拟了移动、桌面和服务器设备上的 Java 虚拟机，但是 Java 虚拟机没有假设任何特定的实现技术、主机硬件或主机操作系统。它不是固有的解释，但也可以通过将其指令集编译为硅 CPU 的指令集来实现。它也可以在微码中或直接在硅中实现。

Java 虚拟机对 Java 编程语言一无所知，只知道一种特定的二进制格式，即 class 文件格式。class 文件包含 Java 虚拟机指令（或字节码）和符号表以及其他辅助信息。

为了安全起见，Java 虚拟机对 class 文件中的代码施加了强大的语法和结构约束。但是，任何具有可用有效类文件表示的功能的语言都可以由 Java 虚拟机托管。被普遍可用的、与机器无关的平台所吸引，其他语言的实现者可以将 Java 虚拟机作为其语言的交付工具。

此处指定的 Java 虚拟机与 Java SE 19 平台兼容，并支持 Java 语言规范 Java SE 19 版中指定的 Java 编程语言。

1.3 规范的组织

第二章概述了 Java 虚拟机体系结构。

第三章介绍了用 Java 编程语言编写的代码编译到 Java 虚拟机的指令集中。

第四章指定 class 文件格式，即用于表示已编译类和接口的独立于硬件和操作系统的二进制格式。

第五章指定 Java 虚拟机的启动以及类和接口的加载、链接和初始化。

第六章指定 Java 虚拟机的指令集，按操作码助记符的字母顺序显示指令。

第七章给出了按操作码值索引的 Java 虚拟机操作码助记符的表。

在 Java 虚拟机规范第二版中，第 2 章概述了 Java 编程语言，该语言旨在支持 Java 虚拟机规范，但本身并不是该规范的一部分。在《Java 虚拟机规范，Java SE 19 版》中，读者可以参考《Java 语言规范，Java SE 19 版》，以了解有关 Java 编程语言的信息。这种形式的引用: (JLS §x.y) 指出了必要的地方。

在 Java 虚拟机规范的第二版中，第 8 章详细解释了 Java 虚拟机线程与共享主内存交互的底层操作。在 Java 虚拟机规范 Java SE 19 版本中，读者可以参考 Java 语言规范 Java SE 19 版本的第 17 章，了解有关线程和锁的信息。第 17 章反映了 JSR133 专家组产生的 Java 内存模型和线程规范。

1.4 符号

在本规范中，我们引用了来自 Java SE 平台 API 的类和接口。每当我们使用一个标识符 N 引用一个类或接口(除了在示例中声明的那些)时，预期的引用是指向 java.lang 包中名为 N 的类或接口。我们对来自 java.lang 以外的包的类或接口使用完全限定名。

每当我们引用包 java 或其任何子包中声明的类或接口时，预期引用的是引导类加载器加载的该类或接口（§5.3.1）。

每当我们引用名为 java 的包的子包时，预期的引用是由引导类加载器确定的子包。

本规范中字体的使用如下：

- 固定宽度字体用于 Java 虚拟机数据类型、异常、错误、class 文件结构、Prolog 代码和 Java 代码片段。
- 斜体用于 Java 虚拟机“汇编语言”，其操作码和操作数，以及 Java 虚拟机运行时数据区域中的项。它也用于引入新的术语，只是为了强调。

旨在澄清规范的非规范性信息以较小的缩进文本给出。

> 这是非规范性信息。它提供直觉、基本原理、建议、示例等。

1.5 反馈

欢迎读者向 jls-jvms-spec-comments@openjdk.java.net 报告 Java 虚拟机规范中的技术错误和歧义。

关于 javac (Java 编程语言的参考编译器)生成和操作 class 文件的问题可以发送到 compiler-dev@openjdk.java.net。