八年打造，编程宝典　畅销丛书，全新改版

Java 编程技术大全（上）

◉ 魔乐科技（MLDN）软件实训中心　编著
◉ 张玉宏 主编　周喜平 副主编

44 小时 全程**同步**教学**视频** | **67** 小时 Java 项目**实战**教学**视频** | **17** 小时 Oracle 项目**实战**教学**视频** | **9** 个 超值**资源**大放送

☑ 全书配套**范例源码**与**实战练习答案** ☑ Java SE 类库**查询手册** ☑ Eclipse 常用**快捷键**说明文档 ☑ Eclipse 提示与技巧**电子书** ☑ Java 常见**面试题** ☑ Java 常见**错误**及**解决方案** ☑ Java 开发经验及技巧**大汇总** ☑ Java **职业规划** ☑ Java 程序员**面试技巧**

中国工信出版集团　人民邮电出版社 POSTS & TELECOM PRESS

人邮云课堂

Java 编程技术大全（上）

◉ 魔乐科技（MLDN）软件实训中心 编著
◉ 张玉宏 主编　周喜平 副主编

人民邮电出版社
北京

图书在版编目（CIP）数据

Java编程技术大全 / 魔乐科技（MLDN）软件实训中心编著 ; 张玉宏主编. -- 北京 : 人民邮电出版社, 2019.3
ISBN 978-7-115-50100-4

Ⅰ. ①J… Ⅱ. ①魔… ②张… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2019)第001454号

内 容 提 要

本书主要面向零基础读者，用实例引导读者学习，深入浅出地介绍 Java 的相关知识和实战技能。

本书第Ⅰ篇“基础知识”主要讲解 Java 开发环境搭建、Java 程序要素，并逐一介绍常量、变量、运算符、表达式、语句、流程控制、数组、枚举、类、对象以及方法等；第Ⅱ篇“核心技术”主要介绍类的封装、继承、多态，并逐一介绍抽象类、接口、Java 常用类库以及异常的捕获与处理等；第Ⅲ篇“高级应用”主要介绍多线程、文件 I/O 操作、GUI 编程、Swing GUI 编程、Java Web、常用设计框架以及 Android 编程基础等；第Ⅳ篇“项目实战”主要介绍智能电话回拨系统、理财管理系统、我的饭票网以及 Hadoop 下的数据处理等。

本书提供了与图书内容全程同步的教学视频。此外，还赠送大量相关学习资料，以便读者扩展学习。

本书适合任何想学习 Java 的初学者，无论初学者是否从事计算机相关行业，是否接触过 Java，均可通过对本书内容的学习快速掌握 Java 的开发方法和技巧。

◆ 编　　著　魔乐科技（MLDN）软件实训中心
主　　编　张玉宏
副 主 编　周喜平
责任编辑　张　翼
责任印制　马振武

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京市艺辉印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：52.75
字数：1325 千字　　2019 年 3 月第 1 版
印数：1－3 000 册　　2019 年 3 月北京第 1 次印刷

定价：119.00 元（上、下册）

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

前言
PREFACE

本书是专为 Java 初学者量身打造的一本学习用书，由专业计算机图书策划机构“龙马高新教育”精心策划编写而成。

本书主要面向 Java 初学者和爱好者，旨在帮助读者掌握 Java 基础知识、了解开发技巧并积累一定的项目实战经验。

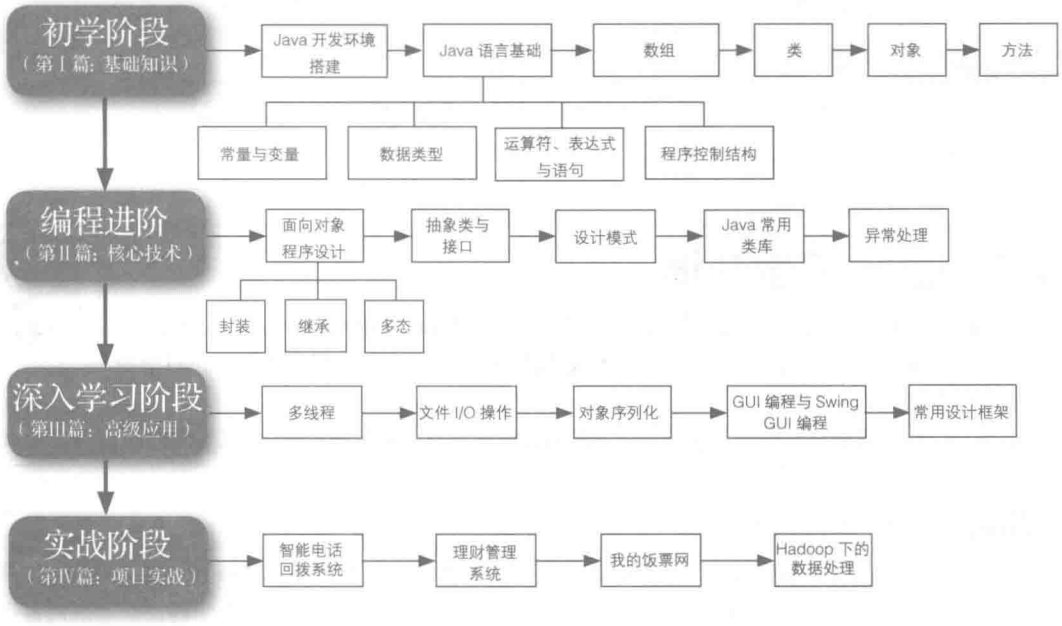
为什么要写这样一本书

荀子曰：“不闻不若闻之，闻之不若见之，见之不若知之，知之不若行之。”

实践对学习的重要性由此可见一斑。纵观当前编程图书市场，理论知识与实践经验的脱节，是一些 Java 图书中经常出现的情况。为了避免这种情况，本书立足于实战，从项目开发的实际需求入手，将理论知识与实际应用相结合。目的就是让初学者能够快速成长为初级程序员，并拥有一定的项目开发经验，从而在职场中拥有一个高起点。

Java 的学习路线

本书总结了作者多年的教学实践经验，为读者设计了合适的学习路线。



本书特色

- **零基础、入门级的讲解**

无论读者是否从事计算机相关行业，是否接触过 Java，是否使用 Java 开发过项目，都能从本书中获益。

- **超多、实用、专业的范例和项目**

本书结合实际工作中的范例，逐一讲解 Java 的各种知识和技术。最后，还通过实际开发项目总结本书所学内容，帮助读者在实战中掌握知识，轻松拥有项目经验。

- **随时检测自己的学习成果**

每章首页给出了“本章要点”，以便读者明确学习方向。每章最后的“实战练习”则根据本章的知识点精心设计而成，读者可以随时自我检测，巩固所学知识。

● 细致入微、贴心提示

本书在讲解过程中使用了“提示”“注意”“技巧”等小栏目，帮助读者在学习过程中更清楚地理解基本概念，掌握相关操作，并轻松获取实战技巧。

超值电子资源

● 全程同步教学视频

涵盖本书所有知识点，详细讲解每个范例和项目的开发过程及关键点，帮助读者更轻松地掌握书中的所有 Java 程序设计知识。

● 超多资源大放送

赠送大量电子资源，包括 Java 和 Oracle 项目实战教学视频、Java SE 类库查询手册、Eclipse 常用快捷键说明文档、Eclipse 提示与技巧电子书、Java 常见面试题、Java 常见错误及解决方案、Java 开发经验及技巧大汇总、Java 程序员职业规划、Java 程序员面试技巧。

读者可以申请加入编程语言交流学习群（QQ：829094243），可在群中获得本书的学习资料，并和其他读者进行交流，帮助你无障碍地快速阅读本书。

读者对象

- 没有任何 Java 基础的初学者。
- 有一定的 Java 基础，想精通 Java 的人员。
- 有一定的 Java 基础，缺乏 Java 实战经验的人员。
- 各类院校及培训学校的老师和学生。

二维码视频教程学习方法

为了方便读者学习，本书提供了大量视频教程的二维码。读者使用微信、QQ 的“扫一扫”功能扫描二维码，即可通过手机观看视频教程。

如下图所示，扫描标题旁边的二维码即可观看对应章节的视频教程。

1.1 Java 开发环境

学习 Java 的第一步，自然就是要搭建 Java 开发环境（Java Development Kit，JDK），在操作系统（如 Windows、Linux 等）下，JDK 是搭建 Java 最基本的开发环境之一，目前由 Oracle 公司维护开发并免费提供。

JDK 由一个处于操作系统层之上的开发环境和运行环境组成，如下图所示。JDK 除了包括编译（javac）、

创作团队

本书主编为张玉宏，副主编为周喜平、姜斌、曹鹤玲。其中第 0~9 章、第 12~16 章、第 18~21 章、第 30 章及附录由河南工业大学张玉宏编写，第 10~11 章、第 17 章、第 24 章及第 25 章由郑州大学西亚斯学院周喜平编写，第 27 章和第 29 章由郑州大学西亚斯学院姜斌编写，第 22~23 章、第 26 章及第 28 章由河南工业大学曹鹤玲编写。参与本书编写、资料整理、多媒体开发及程序调试的人员有孔万里、周奎奎、张田田、常俊杰、黄月、谢洋洋、刘江涛、张芳、江百胜、尚梦娟、张会锋、王金丽、贾祥锋、陈小杰、左琨、邓艳丽、崔姝怡、侯蕾、左花苹、刘锦源、普宁、王常吉、师鸣若、钟宏伟、陈川、刘子威、徐永俊、朱涛和翟桂花等。

在本书的编写过程中，我们竭尽所能地将更好的讲解呈现给读者，但也难免有疏漏和不妥之处，敬请广大读者不吝指正。若读者在阅读本书时遇到困难或疑问，或有任何建议，可发送邮件至 zhangtianyi@ptpress.com.cn。

编者

目录
CONTENTS

第 0 章 Java 学习指南

0.1 Java 为什么重要 002
0.2 Java 简史——给我们带来的一点思考 003
0.3 Java 应用领域和前景 007
0.4 Java 学习路线图 009

第 I 篇 基础知识

第 1 章 小荷才露尖尖角——Java 开发环境搭建

1.1 Java 开发环境 013
1.2 安装 Java 开发工具箱 013
1.2.1 下载 JDK 013
1.2.2 安装 JDK 016
1.3 Java 环境变量的配置 017
1.3.1 理解环境变量 017
1.3.2 JDK 中的 3 个环境变量 018
1.4 享受安装成果——开发第 1 个 Java 程序 022
1.5 Eclipse 的使用 023
1.5.1 Eclipse 概述 023
1.5.2 创建 Java 项目 025
1.5.3 创建 Java 类文件 026
1.5.4 在代码编辑器中编写 Java 程序代码 027
1.5.5 运行 Java 程序 029
1.6 探秘 Java 虚拟机 029
1.7 高手点拨 030
1.8 实战练习 032

第 2 章 初识庐山真面目——Java 程序要素概览

2.1 一个简单的例子 034
2.2 感性认识 Java 程序 035
2.2.1 Java 程序的框架 036
2.2.2 标识符 037
2.2.3 关键字 037
2.2.4 注释 038
2.2.5 变量 039
2.2.6 数据类型 040
2.2.7 运算符和表达式 040
2.2.8 类 041
2.2.9 输入与输出 041
2.3 程序的检测 044
2.3.1 语法错误 044
2.3.2 语义错误 045
2.4 提高程序的可读性 046
2.5 高手点拨 047
2.6 实战练习 047

第 3 章 九层之台，起于垒土——Java 编程基础

3.1 常量与变量 050
3.1.1 常量的声明与使用 050
3.1.2 变量的声明与使用 051
3.2 基本数据类型 055
3.2.1 数据类型的意义 055
3.2.2 整数类型 056
3.2.3 浮点类型 059
3.2.4 字符类型 060
3.2.5 布尔类型 062
3.3 数据类型的转换 063
3.3.1 自动类型转换 063

3.3.2 强制类型转换 064
3.4 高手点拨 065
3.5 实战练习 066

第 4 章 基础编程元素——运算符、表达式、语句与流程控制

4.1 运算符 070
4.1.1 赋值运算符 070
4.1.2 一元运算符 070
4.1.3 算术运算符 072
4.1.4 逻辑运算符 073
4.1.5 位运算符 076
4.1.6 三元运算符 077
4.1.7 关系运算符与 if 语句 078
4.1.8 递增与递减运算符 079
4.1.9 括号运算符 080
4.2 表达式 080
4.2.1 算术表达式与关系表达式 081
4.2.2 逻辑表达式与赋值表达式 082
4.2.3 表达式的类型转换 083
4.3 语句 084
4.3.1 语句中的空格 084
4.3.2 空语句 085
4.3.3 声明语句与赋值语句 086
4.4 程序的控制逻辑 086
4.4.1 顺序结构 087
4.4.2 分支结构 088
4.4.3 循环结构 088
4.5 选择结构 088
4.5.1 if 语句 089
4.5.2 if…else 语句 089
4.5.3 if…else if…else 语句 090
4.5.4 多重选择—— switch 语句 091
4.6 循环结构 093
4.6.1 while 循环 093
4.6.2 do…while 循环 095
4.6.3 for 循环 097
4.6.4 foreach 循环 098
4.7 循环的跳转 099
4.7.1 break 语句 099
4.7.2 continue 语句 101
4.7.3 return 语句 104
4.8 高手点拨 105
4.9 实战练习 106

第 5 章 常用的数据结构——数组与枚举

5.1 理解数组 108
5.2 一维数组 111
5.2.1 一维数组的声明与内存的分配 111
5.2.2 数组中元素的表示方法 112
5.2.3 数组元素的使用 113
5.3 二维数组 116
5.3.1 二维数组的声明与赋值 116
5.3.2 二维数组元素的引用及访问 117
5.4 枚举简介 118
5.5 Java 中的枚举 118
5.5.1 常见的枚举定义方法 118
5.5.2 在程序中使用枚举 119
5.5.3 在 switch 语句中使用枚举 120
5.6 高手点拨 121
5.7 实战练习 121

第 6 章 面向对象设计的核心——类和对象

6.1 理解面向对象程序设计 124
6.1.1 结构化程序设计简介 124
6.1.2 面向对象程序设计简介 124
6.1.3 面向对象程序设计的基本特征 125
6.1.4 面向对象编程和面向过程编程的比较 126
6.2 面向对象的基本概念 127
6.2.1 类 127
6.2.2 对象 128
6.2.3 类和对象的关系 128
6.3 类的声明与定义 129
6.3.1 类的声明 129
6.3.2 类的定义 130
6.4 类的属性 132
6.4.1 属性的定义 132
6.4.2 属性的使用 132
6.5 对象的声明与使用 135

6.5.1 对象的声明 135
6.5.2 对象的使用 136
6.5.3 匿名对象 138
6.5.4 对象的比较 139
6.5.5 对象数组的使用 141
6.6 this 关键字的使用 143
6.7 static 关键字的使用 145
6.8 final 关键字的使用 149
6.9 高手点拨 150
6.10 实战练习 152

第 7 章 重复调用的代码块——方法

7.1 方法的基本定义 154
7.2 方法的使用 156
7.3 方法中的形参与实参 157
7.4 方法的重载 158
7.5 构造方法 161
7.5.1 构造方法简介 161
7.5.2 构造方法的重载 163
7.5.3 构造方法的私有化 167
7.6 在方法内部调用方法 171
7.7 代码块 172
7.7.1 普通代码块 172
7.7.2 构造代码块 173
7.7.3 静态代码块 175
7.8 static 方法 177
7.8.1 自定义 static 方法 177
7.8.2 static 主方法 178
7.9 方法与数组 180
7.9.1 数组引用传递 180
7.9.2 让方法返回数组 183
7.10 包的概念及使用 185
7.10.1 包的基本概念 185
7.10.2 包的导入 186
7.10.3 JDK 中常见的包 187
7.11 高手点拨 188
7.12 实战练习 188

第 II 篇 核心技术

第 8 章 面向对象设计的精华——类的封装、继承与多态

8.1 面向对象的三大特点 191
8.1.1 封装的含义 191
8.1.2 继承的含义 191
8.1.3 多态的含义 192
8.2 封装的实现 194
8.2.1 Java 访问权限修饰符 194
8.2.2 封装问题引例 194
8.2.3 类的封装实例 195
8.3 继承的实现 202
8.3.1 继承的基本概念 202
8.3.2 继承问题的引入 202
8.3.3 继承实现代码复用 204
8.3.4 继承的限制 205
8.4 深度认识类的继承 208
8.4.1 子类对象的实例化过程 208
8.4.2 super 关键字的使用 210
8.4.3 限制子类的访问 213
8.5 覆写 216
8.5.1 属性的覆盖 216
8.5.2 方法的覆写 217
8.5.3 关于覆写的注解——@Override 221
8.6 多态的实现 223
8.6.1 多态的基本概念 223
8.6.2 方法多态性 225
8.6.3 对象多态性 225
8.6.4 隐藏 230
8.7 高手点拨 231
8.8 实战练习 234

第 9 章 凝练才是美——抽象类、接口与内部类

9.1 抽象类 236
9.1.1 抽象类的定义 236

9.1.2 抽象类的使用 236
9.2 接口 240
9.2.1 接口的基本概念 240
9.2.2 使用接口的原则 241
9.2.3 接口的作用——Java 的回调机制 248
9.3 内部类 253
9.3.1 内部类的基本定义 253
9.3.2 在方法中定义内部类 255
9.4 匿名内部类 256
9.5 匿名对象 258
9.6 高手点拨 259
9.7 实战练习 262

第10章 更灵活的设计——泛型

10.1 泛型的概念 264
10.2 泛型类的定义 264
10.3 泛型方法的定义 265
10.4 泛型接口的定义 265
10.5 泛型的使用限制和通配符的使用 266
10.5.1 泛型的使用限制 266
10.5.2 通配符的使用 267
10.6 泛型的继承和实现 268
10.7 高手点拨 269
10.8 实战练习 270

第11章 更强大和方便的功能——注解

11.1 注解概述 272
11.2 常用内置注解 272
11.3 自定义注解 274
11.4 通过反射访问注解信息 277
11.5 高手点拨 280
11.6 实战练习 282

第12章 设计实践——常用的设计模式

12.1 设计模式概述 284
12.1.1 设计模式的背景 284
12.1.2 设计模式的分类 284
12.2 创建型模式 285
12.2.1 单例设计模式 285
12.2.2 多例设计模式 288
12.2.3 工厂模式 290
12.3 结构型模式 295
12.3.1 代理设计模式 296
12.3.2 桥接设计模式 299
12.4 行为型模式 307
12.4.1 行为型模式概述 307
12.4.2 责任链设计模式 307
12.5 高手点拨 310
12.6 实战练习 310

第13章 存储类的仓库——Java 常用类库

13.1 API 概念 312
13.2 基本数据类型的包装类 312
13.2.1 装箱与拆箱 313
13.2.2 基本数据类型与字符串的转换 315
13.3 String 类 317
13.3.1 字符串类的声明 317
13.3.2 String 类中常用的方法 319
13.4 System 类与 Runtime 类 321
13.4.1 System 类 321
13.4.2 Runtime 类 324
13.5 日期操作类 326
13.5.1 日期类 326
13.5.2 日期格式化类 328
13.6 正则表达式 329
13.6.1 正则的引出 329
13.6.2 正则标记 331
13.6.3 利用 String 进行正则操作 332
13.7 Math 与 Random 类 334
13.7.1 Math 类的使用 334
13.7.2 Random 类的使用 335
13.8 高手点拨 337
13.9 实战练习 338

第14章 防患于未然——异常的捕获与处理

14.1 异常的基本概念 340
14.1.1 为何需要异常处理 340
14.1.2 简单的异常范例 341
14.1.3 异常的处理 342
14.1.4 异常处理机制的小结 347
14.2 异常类的处理流程 348
14.3 throws 关键字 348
14.4 throw 关键字 350
14.5 异常处理的标准格式 350
14.6 RuntimeException 类 352
14.7 编写自己的异常类 353
14.8 高手点拨 354
14.9 实战练习 354

1. Java和Oracle项目实战教学视频
2. Java SE类库查询手册
3. Eclipse常用快捷键说明文档
4. Eclipse提示与技巧电子书
5. Java常见面试题
6. Java常见错误及解决方案
7. Java开发经验及技巧大汇总
8. Java程序员职业规划
9. Java程序员面试技巧

第 0 章

Java 学习指南

Java 是一门优秀的编程语言，它的优点是与平台无关，可以实现“一次编写，到处运行”。Java 是一门面向对象的语言，它简洁高效，具有高度的可移植性。本章介绍 Java 的来源、基本思想、技术体系、应用领域、前景以及学习 Java 的技术路线。

本章要点（已掌握的在方框中打钩）

- □ 了解 Java 的来源
- □ 了解 Java 的基本思想
- □ 了解 Java 的技术体系和应用前景

0.1 Java 为什么重要

目前，常用的编程语言就有数十种，令人应接不暇，到底哪一种语言更值得我们学习呢？要知道，学习任何一种语言，都需付出昂贵的时间成本（甚至金钱成本），如何选择一种真正需要的编程语言来学，就是一门学问了。

在现实生活中，有个很有意思的经验。当我们来到一个陌生的城市，自然想找一家比较有特色的饭馆，但面对矗立街头、琳琅满目的饭馆，该选择哪家最好呢？有人说，哪家人少去哪家，因为这样不用等！但有经验的“吃货”会告诉你，哪家人多，特别是等的人多，就去哪家。为什么呢？逻辑很简单，之所以人多，是因为好吃。之所以等的人多，是因为它值得人等。一句话，大样本得出的推荐建议，总还是比较信得过的。

对于初学者来说，编程语言的选择，犹如饭馆的挑选——追随多数人的选择，纵然可能没有满足你个性化的需求，但绝对不会让你错得离谱。目前，我们既然正处于大数据的时代，就要善于“让数据发声”。

May 2018	May 2017	Change	Programming Language	Ratings	Change
1	1		Java	16.380%	+1.74%
2	2		C	14.000%	+7.00%
3	3		C++	7.668%	+2.92%
4	4		Python	5.192%	+1.64%
5	5		C#	4.402%	+0.95%
6	6		Visual Basic .NET	4.124%	+0.73%
7	9	^	PHP	3.321%	+0.63%
8	7	v	JavaScript	2.923%	-0.15%
9	-	^^	SQL	1.987%	+1.99%
10	11	^	Ruby	1.182%	-1.25%

根据 TIOBE 统计的数据[①]，在 2018 年 5 月编程语言前 10 名排行榜中，Java 名列榜首。虽然在不同的年份，Java、C 和 C++ 的前 3 名地位可能有所互换，但多年来，Java 在整个编程领域前三甲的地位，基本上没有动摇。

从上表反映的情况可以看出，Java 作为一门编程语言，其关注度长期高居各种编程语言流行榜的榜首，这也间接说明了 Java 应用领域的广泛程度。事实上，Java 的开放性、安全性和庞大的社会生态链以及其跨平台性，使得 Java 技术成为很多平台事实上的开发标准。在很多应用开发中，Java 都是作为底层代码的操作功能的调用工具。

当下，不论是桌面办公还是网络数据库，不论是 PC 还是嵌入式移动平台，不论是 Java 小应用程序（Applet）还是架构庞大的 J2EE 企业级解决方案，处处都有 Java 的身影。

目前，随着云计算（Cloud Computing）、大数据（Big Data）时代的到来以及人们朝着移动领域的扩张，越来越多的企业考虑将其应用部署在 Java 平台上。无论是面向智能手机的 Android 开发，还是支持高并发的大型分布式系统开发，无论是面向大数据批量处理的 Hadoop 开发，还是解决公共云 / 私有云的部署，都和 Java 密不可分，Java 已然形成一个庞大的生态系统。

此外，Java 的开放性，也对打造其健壮的生态系统贡献非凡。基本上，无论我们有什么新的想法，都可以在 Java 的开源世界中找到对应的实现，而且其中很多解决方案还非常靠谱。例如服务器相关的 Tomcat、计算框架相关的 Hibernate、Spring 和 Struts，大数据处理相关的 ZooKeeper、Hadoop 和 Cassandra，等等。有了

① TIOBE 编程社区指数是编程语言流行趋势的一个指标，每月更新一次。排名数据的获取主要源自著名的搜索引擎或知名网站搜索（或点击）次数，这些网站大致为 Google、Blogger、Wikipedia、YouTube、Baidu、Yahoo!、Bing、Amazon。需要说明的是，这个排行榜利用了关键词搜索热度来代表流行程度：某个语言搜索的次数越多，说明这门语言越受人关注（也就更加流行）。TIOBE 编程社区指数，简而言之，只是一个关键字查询（点击）排行榜，在某种程度上，只反映一门编程语言的流行度，并不能据此说明某门编程语言的优劣高下。

基于 Java 开发的开源软件，开发者们就可以不用从零开始“重造轮子”，这样就大大减轻了开发组的负担，提高了解决问题的效率。

坦率来说，对于很多计算机相关领域的从业人员，找份好工作是学习某门编程语言本质的驱动力。而 Java 应用领域之广泛，也势必促使面向 Java 开发者的就业市场，呈现欣欣向荣之态势。根据国际数据公司（International Data Corporation，IDC）的统计数据，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%。这一高分数字，足以让 Java 语言初学者，跃跃欲试。

一言蔽之，学好用好 Java，可以解决诸多领域的问题，这就是 Java 如此重要的原因。

0.2 Java 简史——给我们带来的一点思考

著名人类学家费孝通先生曾指出，我们所谓的“当前”，其实包含着从“过去”历史中拔萃出来的投影和时间选择的积累。历史对于我们来说，并不是什么可有可无的点缀之饰物，而是实用的、不可或缺的前行之基础。

Java 从诞生（1995 年）发展到现在，已经度过了 20 多年。了解 Java 的一些发展历史，有助于我们更好地认识 Java，看清这纷杂的编程语言世界，进而用好 Java。

说到 Java 的发展历程，就不能不提到它的新老两个东家——Sun（太阳）公司和 Oracle（甲骨文）公司。先说 Sun 公司，事实上，Sun 的本意并非“太阳”，而是斯坦福大学校园网（Stanford University Network）的首字母缩写，跟“太阳”并没有关系。不过，由于这个缩写的蕴意不错，“太阳”就这样叫开了。

1982 年，Sun 公司从斯坦福大学产业园孵化而成，后来成为一家大名鼎鼎的高科技 IT 公司，其全称是太阳微系统公司（Sun Microsystems）。Sun 的主要产品是服务器和工作站，产品极具竞争力，自然市场表现斐然。在硬件方面，他们于 1985 年研制出了自己的 SPARC 精简指令（RISC）处理器，能将服务器的性能提高很多，在软件方面，他们引以为傲的操作系统 Solaris（UNIX 的一个变种）比当时的 Windows NT 能更好地利用计算机资源，特别是在用户数急剧上升，计算机系统变得非常庞大的情况下，Solaris 表现更佳。

20 世纪 90 年代，互联网兴起。Sun 公司就站在那个时代的潮流之上，所以它的服务器和工作站销量极佳，以至于这家公司在自己的广告中宣称：“我们就是 .com 前面的关键一点 (We are the dot in the .com)”。言外之意，没有我们这画龙点睛的一点（服务器 + 操作系统），互联网公司就难以开起来。其得意之情，溢于言表。

Sun 公司之所以敢于高抬自己，也不是吹嘘出来的，它实力的确非常雄厚，在当时足以傲视群雄。其重要的软实力，就是人才济济。在任何年代，人才都是稀缺的（不光是 21 世纪）。

Sun 公司创始人之一史考特·麦克尼里（Scott McNealy），可谓是一代“枭雄”，他非常重视研发。在他的主持下，Sun 公司先后开发了基于 SPARC 系列的处理器、工作站和 Solaris 操作系统，这些产品为 Sun 公司带来了丰厚的利润。

但如果我们把格局放大一些的话，从科技史的角度来看，可能 Sun 公司给人类带来的最有意义的产品，并不是前面提及软件和硬件，而是我们即将要介绍的重要内容——Java 编程语言。

现在，让我们简单地回顾一下 Java 诞生的背景。在 20 世纪 90 年代，世界上的计算机多处于两种状态：要么孤零零地“宅着”——不联网，要么小范围地“宅着”——企业内部局域网互联，那时可供公众分享的资源是非常有限的。

后来，互联网蓬勃发展，不同类型的计算机系统需要连接、信息需要共享的需求就产生了，亟需一种跨越不同硬件和不同操作系统的新平台——这就是那个时代的“痛点”。任何时候，能解决时代的痛点，就会出现划时代的产品。能解决时代的痛点，就抓住了时代的发展方向。

Sun 公司的创始人麦克尼里，对网络计算有着超前的洞察力。在他的带领下，Sun 公司的网络视野，并未仅仅定格于计算机之间的互联，它还看得更远——计算机与非计算机彼此也是隔断的，它们也需要彼此连接！

在 Sun 公司，麦克尼里一直在推行“网络即计算机（The Network is the computer）”的理念。这个关于无限连通世界观念的表述，推动着 Sun 公司参与时代的发展。事实上，这个理念和现在火热的云计算理念，也是一脉相承的。

2016 年 4 月 28 日，全球移动互联网大会（GMIC）在北京举行，当时的腾讯副总裁程武发表了《共享连接的力量》主题演讲。他提到，3 年前，腾讯就提出“连接一切”。无论连接人与人、服务、设备，互联网根本上是在满足人的延伸，让网络中的个体获得更多的资源和能力，去实现更大的价值。

这样的认知，其实是梅特卡夫定律（Metcalfe's Law）的体现，其内容是：网络的价值等于网络节点数的平方，网络的价值与联网的用户数的平方成正比。梅特卡夫认为，“连接”革命后，网络价值会飙升，网络中的个体有望实现更大的价值。

回顾起来，不论是现在流行的物联网（Internet of things，IoT）概念，还是腾讯的“连接一切”理念，其实和 Sun 公司 30 多年前的理念相差无几。因此可以说，Sun 公司在那个时代的视角，不可谓不“高瞻远瞩”。

Sun 公司认为，如果能把计算机和非计算机（主要指的是电子消费品，如家电等）系统这两者连接起来，将会带来一场计算机革命，这是一个巨大的机遇，而连接二者的媒介自然就是网络。

无限连通的世界，令人怦然心跳。但心动不如行动。Sun 公司行动的结果，就是 Java 语言的诞生。

后来被称为 Java 之父的詹姆斯·高斯林（James Gosling）说：“放眼当时的市场，两个领域的厂家各自为政，没有形成统一的网络。因此很多时候不得不重复大量的实验，但这些其实早在 30 年前的计算机科学中已得到解决。”

那么核心问题在于，当时的电子消费品制造者，压根并没有考虑使用网络，例如没有哪家生产商想生产一台会上网的冰箱。一流的企业，如苹果公司，是引导用户需求，而不是满足用户需求。因为有时候，用户压根也不能明确知道自己的需求。

为了解决计算机与计算机之间、计算机与非计算机之间的跨平台连接，麦克尼里决定组建一个名叫 Green 的、由詹姆斯·高斯林领衔的项目团队。其目的在于开发一种新的语言，并基于这种语言，研制专为下一代数字设备（如家电产品）和计算机使用的网络系统，这样就可以将通信和控制信息通过分布式网络，发给电冰箱、电视机、烤面包机等家用电器，对它们进行控制和信息交流。想一想，这不正是当下很热门的物联网思维吗？

最初 Green 项目的工程师们准备采用 C++ 实现这一网络系统。但 C++ 比较复杂，最后经过裁剪、优化和创新，1990 年，高斯林的研发小组基于 C++ 开发了一种与平台无关的新语言 Oak（即 Java 的前身）。Oak 的取名，缘于高斯林办公室外有一棵枝繁叶茂的橡树，这在硅谷是一种很常见的树。

Oak 主要用于为各种家用电器编写程序，Sun 公司曾以 Oak 语言投标一个交互式电视项目，但结果被 SGI（硅图公司，1982 年成立于美国）打败。由于当时智能化家电的市场需求比较低迷，Oak 的市场占有率并没有当初预期的高，于是“见风使舵”的 Sun 公司放弃了该项计划（事实上，“见风使舵”在市场决策中并不是一个贬义词，而是一种灵活的市场策略）。就这样，Oak 几近“出师未捷身先死”。其实也不能全怪 Sun 公司，想一想，即使在 30 多年后的今天，物联网、智能家居的概念虽然很火，但接地气、成气候的项目，至今也屈指可数。

恰逢这时，Mark Ardreesen（美国软件工程师，曾创办网景通讯公司）开发的 Mosaic 浏览器（互联网历史上第一个获普遍使用且能够显示图片的网页浏览器）和 Netscape 浏览器（网页浏览器，市占率曾位居主导地位）启发了 Oak 项目组成员，让他们预见 Oak 可能会在互联网应用上“大放异彩”，于是他们决定改造 Oak。

及时地调整战略，把握住了时代的需求，Oak 于是又迎来了自己的“柳暗花明又一村”。也就是说，计算机与非计算机之间的连接，由于太超前而失败了，但是计算机与计算机之间的连接需求（更加接近那个时代的地气）又救活了 Oak。

1995 年 5 月 23 日，Oak 改名为 Java。至此，Java 正式宣告诞生。Oak 之所以要改名，其实也是情非得已，因为 Oak 作为一个商标，已早被一家显卡制造商注册了。Oak 若想发展壮大，在法律层面上，改头换面，势在必行。

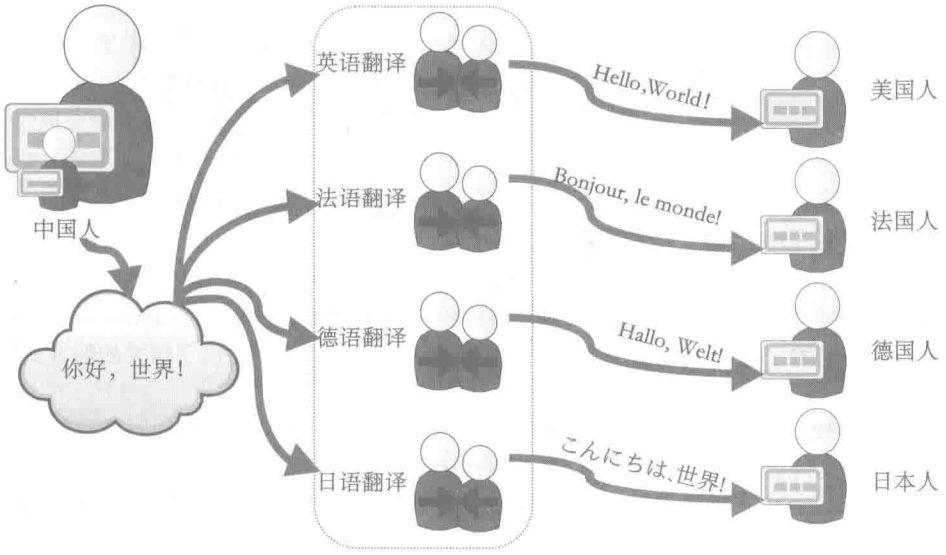
其实 Java 本身也韵意十足，它是印度尼西亚“爪哇”（注：Java 的音译）岛的英文名称，该岛因盛产咖啡而闻名。这也是 Java 官方商标为一杯浓郁咖啡的背后原因，而咖啡也是“爱”加班、“爱”熬夜的程序员们提神的最佳饮品之一。

当时，Java最让人着迷的特性之一，就是它的跨平台性。在过去，计算机程序在不同的操作系统平台（如UNIX、Linux和Windows等）上移植时，程序员通常不得不重新调试与编译这些程序，有时甚至需要重写。

Java的优点在于，在设计之初就秉承了“一次编写，到处运行”（“Write Once, Run Everywhere”，WORE；有时也写成“Write Once, Run Anywhere”，WORA）思想，这是Sun公司为宣传Java语言的跨平台特性而提出的口号。

传统的程序，通过编译，可以得到与各种计算机平台紧密耦合（Coupling）的二进制代码。这种二进制代码可以在一个平台运行良好，但是换一个平台就“水土不服”，难以运行。

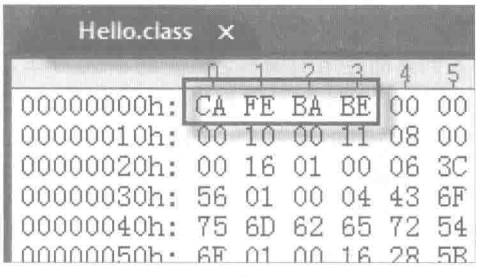
而Java的跨平台性，是指在一种平台下用Java语言编写的程序，在编译之后，不用经过任何更改，就能在其他平台上运行。比如，一个在Windows环境下开发出来的Java程序，在运行时，可以无缝地部署到Linux、UNIX或macOS环境之下。反之亦然，在Linux下开发的Java程序，同样可在Windows等其他平台上运行。Java是如何实现跨平台性的呢？我们可用下面的图来比拟说明。



比如说，中国人（一个平台）说了一句问候语：“你好，世界！”美国人、法国人、德国人及日本人（其他平台）都能理解中国人的“问候”。之所以能这样，这得益于英语、法语、德语及日语翻译们的翻译。

类似的，Java语言的聪明之处在于，它用一个名为Java虚拟机（Java Virtual Machine，JVM）的机制屏蔽了这些“翻译”的细节。各国人尽管尽情地表达（编写Java代码），通过编译，形成各个平台通用的字节码（Byte Code），然后JVM“看平台下菜”，在背后默默地干起了“翻译沟通”的活。正是因为有JVM的存在，Java程序员才可以做到“一次编写，到处运行”——这也是Java的精华所在。

在经过一段时间的Java学习后，读者就会知道由Java源代码编译出的二进制文件叫.class（类）文件，如果使用十六进制编辑器（如UltraEdit等）打开这个.class文件，你会发现这个文件最前面的32位将显示为“CA FE BA BE”，连接起来也就是词组“CAFE BABE”（咖啡宝贝），如下图所示。每个Class文件的前4个字节，都是这个标识，它们被称为“魔数”，主要用来确定该文件是否为一个能被虚拟机接受的Class文件。其另外一个作用就是，让诸如James Gosling等这类具有黑客精神的编程天才尽情表演，他们就是这样，在这些不经意的地方“雁过留声，人过留名”。



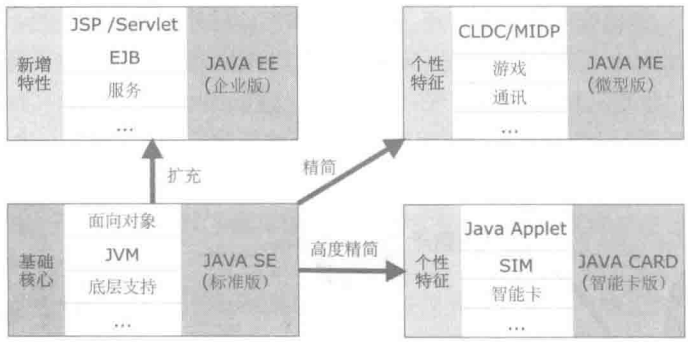
Java class文件的中“咖啡宝贝”

或许，正是麦克尼里看到了Java的这一优秀特性，在Java推出以后，Sun公司便赔钱做了大量的市场推广。仅3个月后，当时的互联网娇子之一——网景（Netscape）公司，便慧眼识珠，决定采用Java。

由于Java是新一代面向对象的程序设计语言，不受操作系

统限制，对网络支持很强，加之对终端用户是免费的，Java 一下子就火了。很快，很多大公司诸如 Oracle、Borland、SGI、IBM、AT&T 和英特尔等都纷纷加入了 Java 阵营。当 Java 逐渐成为 Sun 的标志时，Sun 公司索性就把它的股票代码“SUNW”直接改为了“JAVA”。Sun 公司对 Java 的重视程度，可见一斑。

1997 年，Sun 公司推出 64 位处理器，同年推出 Java 2，Java 渐渐风生水起，市场份额也越做越大。1998 年，Java 2 按适用的环境不同，被分化为 4 个派系（见下图）：Java 2 Micro Edition（J2ME）、Java 2 Standard Edition（J2SE）、Java 2 Enterprise Edition（J2EE）以及 Java Card。ME 的意思是小型设备和嵌入系统，这个小小的派系，其实是 Java 诞生的“初心”。



（1）Java SE（Standard Edition，标准版）：支持面向桌面级应用（如 Windows 下的应用程序）的 Java 平台，提供了完整的 Java 核心 API，这个版本 2005 年以前称为 J2SE。

（2）Java EE（Enterprise Edition，企业版）：以 Java SE 为基础向外延伸，增加了许多支持企业内部使用的扩充类，同时支持使用多层架构的企业应用（如 ERP——企业资源计划系统、CRM——客户关系管理系统的应用）的 Java 平台。除了提供 Java SE API 外，还对其做了大量的扩充并提供了相关的部署支持。这个版本 2005 年以前称为 J2EE。

（3）Java ME（Micro Edition，微型版）：Java ME 同样以 Java SE 为基础，但相对精简。它所支持的只有核心类的子集合，它支持 Java 程序运行在移动终端（手机、PDA——掌上电脑）上的平台，加入了针对移动终端的支持。这个版本 2005 年以前称为 J2ME。Java 的微型版主要是进行嵌入式开发，目前渐渐被 Android 开发所替代。

（4）Java Card（智能卡版）：由于服务对象定位更加明确化，Java Card 版本比 Java ME（微型版）更加精简。它支持一些 Java 小程序（Applets）运行在小内存设备（如容量小于 64KB 的智能卡）的平台上。

但是，Java 的技术平台不管如何划分，都是以 Java SE 为核心的，所以掌握 Java SE 十分重要，这也是本书的主要讲解范围。如果想进行 Java EE 的开发，Java SE 是其中必要的组成部分，这也是为什么在学习 Java EE 之前要求读者一定要有扎实的 Java SE 基础。

当时专为连接智能家电而开发的 Java，不曾想“有心栽花花不开，无心插柳柳成荫”，在家电市场毫无起色，却因其“一次编程，到处运行”的跨平台特性，赶上了互联网的高速发展时机，在企业级市场上大放异彩。

Java ME 一度在翻盖手机应用上得到极大推广，成为当时的标配。但后来，随着安卓（Android）的兴起，也慢慢中落。Java 之父高斯林后来也说，“ME 已经做得足够好了，在当时是最强大的智能电话开发平台之一。不过现在渐渐被遗忘，因为 Android 太耀眼了。”

有起有落，螺旋上升，是事物发展的常态。Java 的发展历程，也不例外。Oracle Java 平台开发副总裁、OpenJDK 管理委员会核心成员 Georges Saab 曾经这样说道：“在 20 世纪 90 年代，大多数开发者都把精力投入到桌面应用的编写之上。到了 2000 年，Pet.com（一家美国宠物网站）的成功吸引了大批的跟风者。业界又把焦点从桌面转移到了 HTML 应用。随着智能电话和平板电脑的到来，基于触摸屏的移动应用又站在了潮流前端。所以对于下一个流行趋势是很难把握的，这涉及天时、地利、人和。”

然而，Java 对于 Sun 来说，是“华而不实”的资产。因为，除了带来日渐高涨的声誉外，Java 并没有直接给 Sun 带来与其声誉对等的回报。用华尔街的话来说，Java 是赔钱赚吆喝。吆喝 Java 是赚到了，但赚钱盈利才是生存之道。

现在具备互联网思维的公司都知道，“免费”的目的是不免费，不免费的对象要发生转移，其实是要让