



JWorld@TW技术论坛版主
Java权威技术顾问与专业讲师

全新改版!

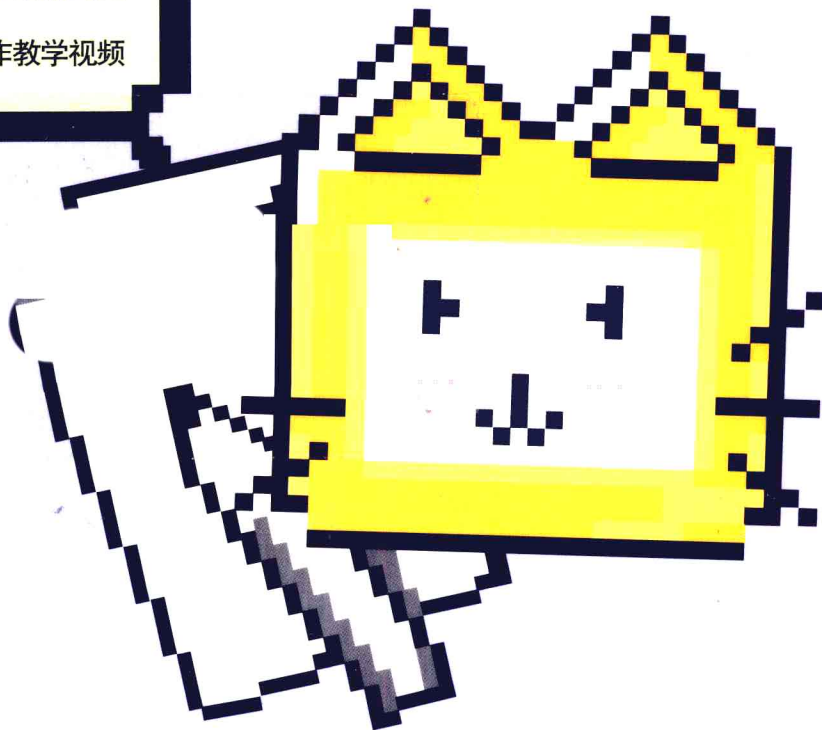
Java

JDK 7

学习笔记

林信良 编著

- 分享作者学习Java心得
- 涵盖OCP/JP(原SCJP)考试范围
- Java JDK 7新功能介绍
- JDK基础与IDE操作交相对应
- 提供Lab文档与操作教学视频





JWorld@TW技术论坛版主
Java权威技术顾问与专业讲师

全新改版!

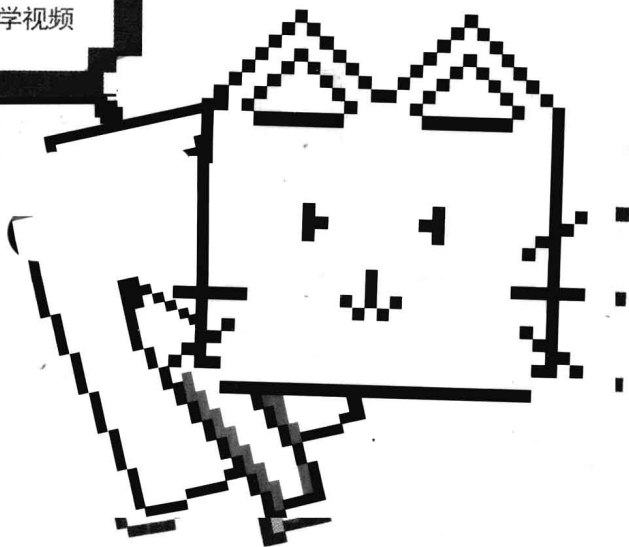
Java

JDK 7

学习笔记

林信良 编著

- 分享作者学习Java心得
- 涵盖OCP/JP(原SCJP)考试范围
- Java JDK 7新功能介绍
- JDK基础与IDE操作交相对应
- 提供Lab文档与操作教学视频



清华大学出版社

北 京

内 容 简 介

本书是作者多年来教学实践经验的总结,汇集了学生在学习 Java 或认证考试时遇到的概念、操作、应用等问题及解决方案。

本书针对 Java SE 7 新功能全面改版,无论是章节架构或范例程序代码,都做了重新编写与全面翻新。并详细介绍了 JVM、JRE、Java SE API、JDK 与 IDE 之间的对照关系。必要时从 Java SE API 的源代码分析,了解各种语法在 Java SE API 中如何应用。对于建议练习的范例提供 Lab 文档,以突出练习重点。此外,本书还将 IDE 操作纳为教学内容之一,让读者能与实践相结合,提供的教学视频让读者可以更清楚地掌握操作步骤。

本书适合 Java 的初中级读者以及广大 Java 应用开发人员。

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。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

Java JDK 7 学习笔记/林信良 编著. —北京:清华大学出版社,2012.5

ISBN 978-7-302-28208-2

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

中国版本图书馆 CIP 数据核字(2012)第 035963 号

责任编辑:王 定 胡花蕾

封面设计:牛艳敏

版式设计:康 博

责任校对:成凤进

责任印制:王静怡

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 刷 者:清华大学印刷厂

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm 印 张:36.25 字 数:882 千字
(附光盘 1 张)

版 次:2012 年 5 月第 1 版

印 次:2012 年 5 月第 1 次印刷

印 数:1~6000

定 价:59.90 元

产品编号:044951-01

序

有时，教育培训的对象是刚步入社会的年轻人，在课程期间，我总会找机会提出这个问题：“你喜欢目前的工作吗？”

当然，这个问题很突然，刚进公司的年轻人也没办法回答这个问题。无论如何，请记得一段时间后再问一次自己：“喜欢目前的工作吗？”

如果不喜欢怎么办？是否明天就告诉老板不干了？这时，请再想一想：“自己真正喜欢的工作是什么？愿意每天付出多少时间……”

每天花点时间，一小时、三十分钟都可以，为自己将来想做的事情做准备。每天都没有时间？那么，至少也拨个十五分钟、十分钟，千万不要什么都不做，什么都没准备，只会每天抱怨现在的工作。虽然为了面包，仍得继续工作，然而积沙成塔、聚少成多，你为喜爱的工作付出的点滴时间，累积几年，将成为未来的本钱。

每天持续做些小事，累积起来也会成为巨大的力量！

那么，你喜欢程序设计的工作吗？

林信良

2011 年 11 月

目 录

Chapter 1 Java 平台概论	1
1.1 Java 不只是语言	2
1.1.1 前世今生	2
1.1.2 三大平台	5
1.1.3 JCP 与 JSR	6
1.1.4 建议的学习路径	7
1.2 JVM/JRE/JDK	11
1.2.1 什么是 JVM	11
1.2.2 区分 JRE 与 JDK	14
1.2.3 下载、安装 JDK	15
1.2.4 认识 JDK 安装内容	18
1.3 重点复习	19
1.4 课后练习	20
Chapter 2 从 JDK 到 IDE	21
2.1 从 Hello World 开始	22
2.1.1 撰写 Java 原始码	22
2.1.2 PATH 是什么	24
2.1.3 JVM(java)与 CLASSPATH	27
2.1.4 编译程序(javac)与 CLASSPATH	30
2.2 管理原始码与位码文档	31
2.2.1 编译程序(javac)与 SOURCEPATH	31
2.2.2 使用 package 管理类	33
2.2.3 使用 import 偷懒	36
2.3 使用 IDE	38
2.3.1 IDE 项目管理基础	38
2.3.2 使用了哪个 JRE	43
2.3.3 类文档版本	45
2.4 重点复习	48
2.5 课后练习	49
Chapter 3 基础语法	53
3.1 类型、变量与运算符	54
3.1.1 类型	54

3.1.2 变量	57
3.1.3 运算符	60
3.1.4 类型转换	66
3.2 流程控制	69
3.2.1 if...else 条件式	69
3.2.2 switch 条件式	72
3.2.3 for 循环	74
3.2.4 while 循环	75
3.2.5 break、continue	77
3.3 重点复习	78
3.4 课后练习	79
Chapter 4 认识对象	83
4.1 类与对象	84
4.1.1 定义类	84
4.1.2 使用标准类	87
4.1.3 对象指定与相等性	90
4.2 基本类型打包器	91
4.2.1 打包基本类型	91
4.2.2 自动装箱、拆箱	92
4.2.3 装箱的内幕	93
4.3 数组对象	96
4.3.1 数组基础	96
4.3.2 操作数组对象	99
4.3.3 数组复制	105
4.4 字符串对象	108
4.4.1 字符串基础	108
4.4.2 字符串特性	111
4.4.3 字符串编码	115
4.5 查询 Java API 文件	117
4.6 重点复习	119
4.7 课后练习	120
Chapter 5 对象封装	125
5.1 何谓封装	126
5.1.1 封装对象初始流程	126

5.1.2	封装对象操作流程	128
5.1.3	封装对象内部数据	131
5.2	类语法细节	134
5.2.1	public 权限修饰	134
5.2.2	关于构造函数	136
5.2.3	构造函数与方法重载	137
5.2.4	使用 this	139
5.2.5	static 类成员	142
5.2.6	不定长度自变量	148
5.2.7	内部类	150
5.2.8	传值调用	151
5.3	重点复习	154
5.4	课后练习	155
Chapter 6	继承与多态	161
6.1	何谓继承	162
6.1.1	继承共同行为	162
6.1.2	多态与 is-a	166
6.1.3	重新定义行为	170
6.1.4	抽象方法、抽象类	173
6.2	继承语法细节	174
6.2.1	protected 成员	174
6.2.2	重新定义的细节	176
6.2.3	再看构造函数	178
6.2.4	再看 final 关键字	180
6.2.5	java.lang.Object	181
6.2.6	关于垃圾收集	186
6.2.7	再看抽象类	189
6.3	重点复习	191
6.4	课后练习	192
Chapter 7	接口与多态	199
7.1	何谓接口	200
7.1.1	接口定义行为	200
7.1.2	行为的多态	204
7.1.3	解决需求变化	206
7.2	接口语法细节	213
7.2.1	接口的默认	213
7.2.2	匿名内部类	217
7.2.3	使用 enum 枚举常数	221

7.3	重点复习	224
7.4	课后练习	224
Chapter 8	异常处理	231
8.1	语法与继承架构	232
8.1.1	使用 try、catch	232
8.1.2	异常继承架构	235
8.1.3	要抓还是要抛	238
8.1.4	认识堆栈追踪	241
8.1.5	关于 assert	245
8.2	异常与资源管理	247
8.2.1	使用 finally	247
8.2.2	自动尝试关闭资源	249
8.2.3	java.lang.AutoCloseable 接口	251
8.3	重点复习	255
8.4	课后练习	256
Chapter 9	Collection 与 Map	261
9.1	使用 Collection 收集对象	262
9.1.1	认识 Collection 架构	262
9.1.2	具有索引的 List	263
9.1.3	内容不重复的 Set	266
9.1.4	支持队列操作的 Queue	270
9.1.5	访问对象的 Iterator	273
9.1.6	排序收集的对象	276
9.1.7	使用泛型	280
9.2	键值对应的 Map	284
9.2.1	常用 Map 操作类	284
9.2.2	访问 Map 键值	288
9.3	重点复习	291
9.4	课后练习	292
Chapter 10	输入输出	299
10.1	InputStream 与 OutputStream	300
10.1.1	串流设计的概念	300
10.1.2	串流继承架构	303
10.1.3	串流处理装饰器	306
10.2	字符处理类	311

10.2.1 Reader 与 Writer 继承 架构	311	12.4.2 操作路径	406
10.2.2 字符处理装饰器	313	12.4.3 属性读取与设定	409
10.3 重点复习	315	12.4.4 操作文档与目录	412
10.4 课后练习	316	12.4.5 读取、访问目录	414
Chapter 11 线程与并行 API	319	12.4.6 过滤、搜索文档	418
11.1 线程	320	12.5 重点复习	421
11.1.1 线程简介	320	12.6 课后练习	422
11.1.2 Thread 与 Runnable	323	Chapter 13 窗口程序设计	425
11.1.3 线程生命周期	324	13.1 Swing 入门	426
11.1.4 关于 ThreadGroup	331	13.1.1 简易需求分析	426
11.1.5 synchronized 与 volatile	334	13.1.2 Swing 组件简介	427
11.1.6 等待与通知	345	13.1.3 设计主窗口与菜单列	429
11.2 并行 API	349	13.1.4 关于版面管理	433
11.2.1 Lock、ReadWriteLock 与 Condition	349	13.1.5 事件处理	436
11.2.2 使用 Executor	357	13.2 文档打开、存储与编辑	442
11.2.3 并行 Collection 简介	370	13.2.1 操作打开文档	442
11.3 重点复习	373	13.2.2 制作存储、关闭文档	445
11.4 课后练习	375	13.2.3 文字区编辑、剪切、复制、 粘贴	448
Chapter 12 通用 API	377	13.3 重点复习	449
12.1 日志	378	13.4 课后练习	451
12.1.1 日志 API 简介	378	Chapter 14 整合数据库	453
12.1.2 指定日志层级	380	14.1 JDBC 入门	454
12.1.3 使用 Handler 与 Formatter	382	14.1.1 JDBC 简介	454
12.1.4 自定义 Handler、Formatter 与 Filter	383	14.1.2 连接数据库	458
12.1.5 使用 logging.properties	385	14.1.3 使用 Statement、 ResultSet	464
12.2 国际化基础、日期	387	14.1.4 使用 PreparedStatement、 CallableStatement	469
12.2.1 关于 i18n	387	14.2 JDBC 进阶	472
12.2.2 使用 Date 与 DateFormat	390	14.2.1 使用 DataSource 取得 联机	472
12.2.3 使用 Calendar	393	14.2.2 使用 ResultSet 卷动、 更新数据	476
12.3 规则表示式	395	14.2.3 批次更新	479
12.3.1 定义规则表示式	396	14.2.4 Blob 与 Clob	480
12.3.2 Pattern 与 Matcher	403	14.2.5 交易简介	481
12.4 NIO2 文件系统	405	14.2.6 metadata 简介	489
12.4.1 API 架构概述	405		

14.2.7	RowSet 简介	492	16.1.3	使用 super 与?	530
14.3	重点复习	496	16.2	自定义枚举	533
14.4	课后练习	497	16.2.1	了解 java.lang.Enum 类	533
Chapter 15	反射与类加载器	499	16.2.2	进阶 enum 运用	536
15.1	运用反射	500	16.3	关于注释	542
15.1.1	Class 与.class 文档	500	16.3.1	常用标准注释	542
15.1.2	使用 Class.forName()	502	16.3.2	自定义注释类型	545
15.1.3	从 Class 获得信息	503	16.3.3	执行时期读取注释信息	549
15.1.4	从 Class 建立对象	506	16.4	重点复习	551
15.1.5	操作对象方法与成员	509	16.5	课后练习	551
15.1.6	动态代理	512	Appendix A	如何使用本书项目	553
15.2	了解类加载器	515	A.1	项目环境配置	554
15.2.1	类加载器层级架构	515	A.2	打开案例	554
15.2.2	建立 ClassLoader 实例	518	Appendix B	MySQL 入门	557
15.3	重点复习	520	B.1	安装、设定 MySQL	558
15.4	课后练习	521	B.2	MySQL 的数据类型	560
Chapter 16	自定义泛型、枚举 与注释	523	B.3	建立数据库、数据表	561
16.1	自定义泛型	524	B.4	进行 CRUD 操作	562
16.1.1	定义泛型方法	524			
16.1.2	使用 extends 与?	525			

Java SE 7 新功能索引

Unicode 6.0	54
新字面常量表示式	60
switchy 与字串	72
多重捕捉(Multi-catch)例外语法	237
更精确判断重抛(More-precise-rethrow)例外型態	241
自动尝试关闭资源语法(Try-with-resources)	249
Throwable 新增 addSuppressed()方法	250
AutoCloseable 介面	251
改良的泛型类型推断(Improved Type Inference for Generic)	284
Closable 介面继承 AutoCloseable 介面	301
Fork/Join 并行 API	366
NIO.2 档案系统 API	405
Connection、Statement、ResultSet 等介面继承 AutoCloseable 介面	462
RowSetFactory 介面与 RowSetProvider 类别	493
RowSet 介面继承 AutoCloseable 介面	495
@SagaVarargs 标注	543

Java 平台概论

Chapter

1

学习目标

- Java 版本迁移简介
- 认识 Java SE、Java EE、Java ME
- 了解 JVM、JRE 与 JDK
- 下载与安装 JDK

1.1 Java 不只是语言

从 1995 年至今, Java 已经超过 15 个年头, 经过这些年的演进, 正如本节标题所示, Java 已不仅是个程序语言, 也代表了解决问题的平台(Platform), 更代表了原厂、各个厂商、社群、开发者与用户沟通的成果。若仅以程序语言的角度来看待 Java, 正如冰山一角, 你仅看到 Java 身为程序语言的一部分, 而没看到 Java 身为程序语言之外, 更可贵也更为庞大的资源。

1.1.1 前世今生

一个语言的诞生有其目的, 因为这个目的而成就了语言的主要特性, 探索 Java 的历史演进, 对于掌握 Java 特性与各式可用资源, 着实有其帮助。

1. Java 诞生

Java 最早是 Sun 公司绿色项目 Green Project 中撰写 Star7 应用程序的程序语言, 当时名称不是 Java, 而是取名为 Oak。

绿色项目始于 1990 年 12 月, 由 Patrick Naughton、Mike Sheridan 与 James Gosling(James Gosling 被尊称为 Java 之父)主持, 目的是希望构筑出下一波计算机应用趋势并加以掌握, 他们认为下一波计算机应用趋势会集中在消费性数字产品(就像现在的 PDA、手机等消费性电子产品)的使用上。1992 年 9 月 3 日, Green Team 项目小组展示了 Star7 手持设备, 这个设备具备无线网络连接、5inLCD 彩色屏幕、PCMCIA 接口等功能, 而 Oak 在绿色项目中的目的, 是用来撰写 Star7 上应用程序的程序语言。

Oak 名称的由来, 是因为 James Gosling 的办公室窗外有一棵橡树(Oak), 就顺手取了这个名称。但后来发现 Oak 名称已经被注册了, 工程师们边喝咖啡边讨论着新名称, 最后灵机一动而改名为 Java。

Java 本身会见到许多为了节省资源而作的设计, 像是动态加载类别文档、字符串池(String pool)等特性, 这是因为 Java 一开始就是为了消费性数字产品而设计, 而这类小型装置通常有着有限内存与运算资源。

全球信息网(World Wide Web)兴起, Java Applet 成为网页互动技术的代表。

1993 年第一个全球信息网浏览器 Mosaic 诞生, James Gosling 认为因特网与 Java 的一些特性不谋而合, 利用 Java Applet 在浏览器上展现互动性媒体, 在当时而言, 对视觉感官是一种革命性的颠覆。Green Team 仿照 Mosaic 开发出以 Java 技术为基础的浏览器 WebRunner(原名为 BladeRunner), 后来改名为 HotJava, 虽然 HotJava 只是一个展示性产品, 但它使用 Java Applet 展现的多媒体效果立即吸引了许多人的注意, 如图 1.1 所示。

1995 年 5 月 23 日(这一天公认为 Java 的生日), 正式将 Oak 改名为 Java, Java Development Kits(当时 JDK 全名)1.0a2 版本正式对外发表, 而在 1996 年 Netscape Navigator 2.0 也正式支持 Java, Microsoft Explorer 也开始支持 Java, 从此 Java 在因特网

的世界中逐渐风行起来。虽然 **Star7** 产品并不被当时消费性市场接受，绿色项目面临被裁撤的命运，然而全球信息网的兴起却给了 **Java** 新的生命与舞台。

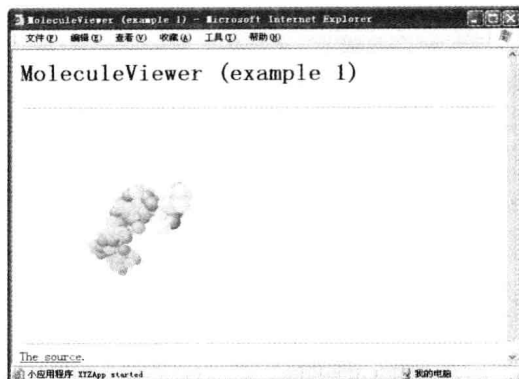


图 1.1 JDK 所附的 Java Applet 范例(JDK 文件夹\demo\applets\MoleculeViewer\ example1.html)

2. 版本演进

随着 **Java** 越来越受到瞩目，**Sun** 在 1998 年 12 月 4 日发布 **Java 2 Platform**，简称 **J2SE 1.2**，**Java** 开发者版本一开始是以 **Java Development Kit** 名称发表，简称 **JDK**，而 **J2SE** 则是平台名称，包含了 **JDK** 与 **Java** 程序语言。

Java 平台标准版约以两年为周期推出重大版本更新，1998 年 12 月 4 日发表 **J2SE 1.2**，2000 年 5 月 8 日发表 **J2SE 1.3**，2002 年 2 月 13 日发表 **J2SE 1.4**，**Java 2** 这个名称也从 **J2SE 1.2** 一直沿用至之后各个版本。

2004 年 9 月 29 日发表的 **Java** 平台标准版的版号不是 1.5，而直接跳到 5.0，称为 **J2SE 5.0**，这是为了彰显这个版本与之前版本有极大不同，如语法上的简化、增加泛型(**Generics**)、枚举(**Enum**)、注释(**Annotation**)等重大功能。

2006 年 12 月 11 日发表的 **Java** 平台标准版，除了版号之外，名称也有了变化，称为 **Java Platform, Standard Edition 6**，简称 **Java SE 6**，**JDK6** 全名则称为 **Java SE Development Kit 6**，也就是不再像以前 **Java 2** 带有 2 这个号码，版本号 6 或 1.6.0 都使用，6 是产品版本(**Product version**)，而 1.6.0 是开发者版本(**Developer version**)。

大部分的 **Java** 标准版平台都会取个代码名称(**Code name**)，例如 **J2SE 5.0** 的代码名称为 **Tiger**(老虎)，为了引人注目，在发表会上还真的抱了一只小白老虎出来作为噱头，而许多书的封面也相应地放上老虎的图片。有关 **JDK** 代码名称与发布日期，可以参考表 1.1 所示。

表 1.1 Java 版本、代码名称与发布日期

版 本	代 码 名 称	发 布 日 期
JDK 1.1.4	Sparkler(烟火)	1997/9/12
JDK 1.1.5	Pumpkin(南瓜)	1997/12/3
JDK 1.1.6	Abigail(圣经故事人物名称)	1998/4/24
JDK 1.1.7	Brutus(罗马政治家名称)	1998/9/28

(续表)

版 本	代 码 名 称	发 布 日 期
JDK 1.1.8	Chelsea(足球俱乐部名称)	1999/4/8
J2SE 1.2	Playground(游乐场)	1998/12/4
J2SE 1.2.1	无	1999/3/30
J2SE 1.2.2	Cricket(蟋蟀)	1999/7/8
J2SE 1.3	Kestrel(红隼)	2000/5/8
J2SE 1.3.1	Ladybird(瓢虫)	2001/5/17
J2SE 1.4.0	Merlin(魔法师名称)	2002/2/13
J2SE 1.4.1	Hopper(蚱蜢)	2002/9/16
J2SE 1.4.2	Mantis(螳螂)	2003/6/26
J2SE 5.0	Tiger(老虎)	2004/9/29
Java SE 6	Mustang(野马)	2006/12/11
Java SE 7	Dolphin(海豚)	2011/7/28

提示>>> 撰写本书时，表 1.1 参考的数据源为：

<http://java.sun.com/j2se/codenames.html>

3. 江山易主

之前谈过，Java 约以两年为周期推出重大版本更新，正如表 1.1 所示，J2SE 1.2、J2SE 1.3、J2SE 1.4.0、J2SE 5.0、Java SE 6 推出的时间间隔，差不多都是两年。然而从 Java SE 6 之后，Java 开发人员足足等了四年多，才等到新版本的推出，不禁让人想问：Java 怎么了？

原因有许多，Java SE 7 对新版本的规划摇摆不定，涵盖许多不易实现的新特性，加上 Sun 一直营收低迷不振，影响了新版本的推动，新版本推出日期承诺不断推迟，从 2009 年推迟至 2010 年初，又突然宣布将加入原本不愿划入 Java SE 7 的 Closure 语法，并将推出日期推迟至 2010 年底，然后 2010 年年中传出 IBM 与 Sun 密谈并购失败，没隔几日，即爆出 Oracle 宣布并购 Sun，Java 也正式成为 Oracle 所属。

并购就会带来一连串的组织重整，导致 Java SE 7 推出日期再度推迟，为了对停滞不前的 Java 注入活水，决定先将现有已实现或较易实现的特性放入 Java SE 7 中，将未定方案或较难实现的特性放入 Java SE 8 中(像是 Jigsaw、Closure)，2010 年底 JCP(Java Community Process，稍后即会说明这个组织是什么)终于通过了 Java SE 7 与 Java SE 8 的规划地图(Roadmap)，并预计于 2011 年 7 月左右推出 Java SE 7，这次总算没有推迟，Java SE 7 正式于 2011 年 7 月 28 日发布。

提示>>> 正因为 Java SE 7 推出过程如此曲折，网络上很容易找到过时的 Java SE 7 数据。如果想快速了解 Java SE 7 有哪些特性，本书也标示出了哪些特性是 Java SE 7 所有，亦有个快速查询索引可作为参考。

1.1.2 三大平台

在 Java 发展的过程中，由于 Java 的应用领域越来越广，并逐渐扩及至各级应用软件的开
发，Sun 公司在 1999 年 6 月美国旧金山的 Java One 大会上，公布了新的 Java 体系架构，
该架构根据不同级别的应用开发区分了不同的应用版本：J2SE(Java 2 Platform, Standard
Edition)、J2EE(Java 2 Platform, Enterprise Edition)与 J2ME(Java 2 Platform, Micro
Edition)。

J2SE、J2EE 与 J2ME 是当时的名称，由于 Java SE 6 后 Java 不再带有 2 这个号码，
J2SE、J2EE 与 J2ME 分别被正名为 Java SE、Java EE 与 Java ME。

提示>>> 尽管 Sun 从 2006 年底，就将三大平台正名为 Java SE、Java ME 与 Java EE，但时至今
日，许多人的习惯显然还是没有改过来，J2SE、J2ME 与 J2EE 这些名词还是有很多人用。

1. Java SE

Java 是各应用平台的基础，想要学习其他的平台应用，必先了解 Java SE 以奠定基础，
Java SE 也正是本书主要的介绍对象。

图 1.2 所示是整个 Java SE 的组成概念图。

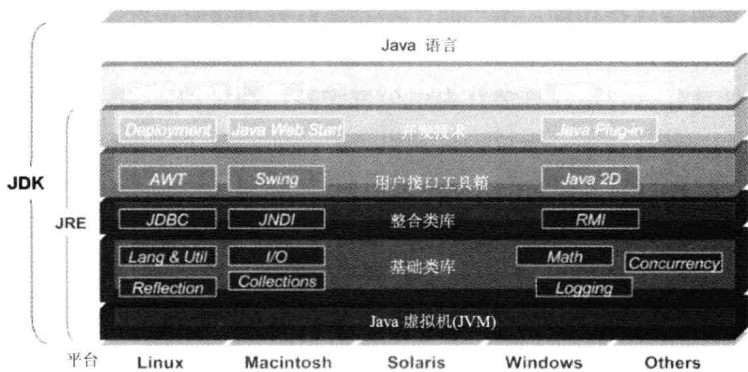


图 1.2 Java SE 的组成概念图

Java SE 可以分为四个主要的部分：JVM、JRE、JDK 与 Java 语言。

为了能够运行 Java 撰写好的程序，必须有 Java 虚拟机(Java Virtual Machine, JVM)。
JVM 包括在 Java 执行环境(Java SE Runtime Environment, JRE)中，所以为了要运行 Java
程序，必须安装 JRE。如果要开发 Java 程序，必须取得 JDK(Java SE Development Kits)，
JDK 包括 JRE 及开发过程中需要的一些工具程序，像是 javac、java、appletviewer 等工具
程序(关于 JRE 及 JDK 的安装与使用，会在第 2 章说明)。

Java 语言只是 Java SE 的一部分，除了语言之外，Java 最重要的就是提供庞大且强大
的标准 API，提供字符串处理、数据输入/输出、网络套件、用户窗口接口等功能，可以使用
这些 API 作为基础来进行程序开发，无须重复开发功能相同的组件。事实上，在熟悉 Java
语言之后，更多的时候，都是在学习如何使用 Java SE 提供的 API 来组成应用程序。

2. Java EE

Java EE 以 Java SE 为基础，定义了一系列的服务、API、协议等，适用于开发分布式、多层次(Multi-tiered)、以组件为基础、以 Web 为基础的应用程序，整个 Java EE 的体系是相当庞大的，比较为人熟悉的技术像是 JSP、Servlet、JavaMail、Enterprise JavaBeans(EJB) 等，其中每个服务或技术都可以使用专书进行说明，所以并非本书说明的范围。但可以肯定的是，必须在 Java SE 上奠定良好的基础，再来学习 Java EE 的开发。

3. Java ME

Java ME 是 Java 平台版本中最小的一个，目的是作为小型数字设备上开发及部署应用程序的平台，像是消费性电子产品或嵌入式系统等，最为人熟悉的设备如手机、PDA、股票机等。可以使用 Java ME 来开发出这些设备上的应用程序，如 Java 游戏、股票相关程序、记事程序、日历程序等。

1.1.3 JCP 与 JSR

Java 不仅是程序语言，还是标准规范。

先来看看没有标准会有什么问题？我们的身边有些东西没有标准，例如手机充电器，不同厂商的手机，充电器就不相同，家里面一堆充电器互不相容，换个手机，充电器就不能用的情况，相信你我都有过。

有标准的好处是什么？现在许多计算机外部设备，都采用 USB 作为传输接口，这让计算机中不用再接上一些转接器，跟过去计算机主机后面一堆不同规格的传输接口相比，实在方便了不少(现在有些手机的充电器，也改采用 USB 接口了，这真是件好事)。

回头来谈谈 Java 是标准规范这件事。你知道吗？编译/执行 Java 的 JDK/JRE，并不只有 Sun 才能实现，IBM 也可以撰写自己的 JDK/JRE，其他厂商或组织也可以撰写自己的 JDK/JRE，你写的 Java 程序，可以执行在这些不同厂商或组织写出来的 JRE 上。第 2 章将学到的第一个 Java 程序，其中会有这么一段程序代码：

```
System.out.println("Hello World");
```

这行程序目的是：请系统(System)的输出装置(out)显示一行(println)Hello World。谁决定使用 System、out、println 这些名称的？为什么不是 Platform、Output、ShowLine 这些名称？如果 Sun 使用 System、out、println 这些名称，而 IBM 使用了 Platform、Output、ShowLine 这些名称，用 Sun 的 JDK 写的程序，就不能执行在 IBM 的 JRE 上，那 Java 最基本的特性之一“跨平台”，就根本无法实现了。

Java 由 Sun 创造，为了让对 Java 感兴趣的厂商、组织、开发者与用户参与定义 Java 未来的功能与特性，Sun 公司于 1998 年组成了 JCP(Java Community Process)，这是一个开放性国际组织，目的是让 Java 演进由 Sun 非正式地主导，成为全世界数以百计代表成员公开监督的过程。

任何想要提议加入 Java 的功能或特性，必须以 JSR(Java Specification Requests)正式文件的方式提交，JSR 必须经过 JCP 执行委员会(Executive Committee)投票通过，方可成

为最终标准文件，有兴趣的厂商或组织可以根据 JSR 实现产品。

若 JSR 成为最终文件后，必须根据 JSR 成果做出免费且开发原始码的参考实现，称为 RI(Reference Implementation)，并提供 TCK(Technology Compatibility Kit)作为技术兼容测试工具箱，方便于其他想根据 JSR 实现产品的厂商或组织参考与测试兼容性。JCP、JSR、RI 与 TCK 的关系，如图 1.3 所示。

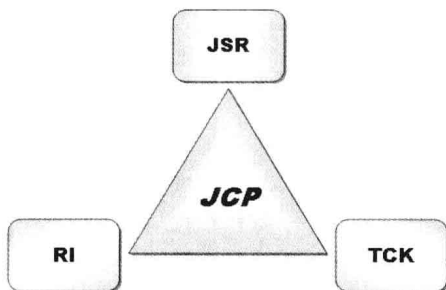


图 1.3 JCP、JSR、RI 与 TCK

提示>>> JCP 官方网站为 <http://jcp.org>。

现在无论 Java SE、Java EE 还是 Java ME，都是业界共同订制的标准，每个标准背后代表了业界所面临的一些问题，他们期待使用 Java 来解决问题，认为应该有某些组件、特性、应用程序编程接口等，来解决这些问题，因而制订 JSR 作为正式标准规范文件，不同的技术解决方案标准规范会给予一个编号。

在 JSR 规范的标准之下，各厂商可以各自操作成品，所以同一份 JSR，可以有不同厂商的操作产品。以 Java SE 为例，对于身为开发人员，或使用 Java 开发产品的公司而言，只要使用兼容于标准的 JDK/JRE 开发产品，就可以执行、兼容在标准的 JRE 上，而不用担心跨平台的问题。

Java SE 7 的主要规范是在 JSR 336 文件之中，而 Java SE 平台中的特定技术，则再规范于特定的 JSR 文件之中，若对这些文件有兴趣，可以参考以下网址：

<http://jcp.org/en/jsr/detail?id=336>

提示>>> 想要查询 JSR 文件，只要在 <http://jcp.org/en/jsr/detail?id=>加上文件编号就可以了，例如上面查询 JSR 336 文件网址就是：

<http://jcp.org/en/jsr/detail?id=336>

JSR 对于 Java 初学者而言过于艰涩，但 JSR 文件规范了相关技术应用的功能，将来有能力时，可以试着自行阅读 JSR，这有助于了解相关技术规范的更多细节。

1.1.4 建议的学习路径

Java 不仅是程序语言，还是标准规范，每个标准代表着厂商面临的问题，代表着解决问题的方案，也因此，学习 Java，就等于在面临各式问题如何解决，然而，这么多的问题，

衍生出如此多的解决方案，也因此对于初学 Java 的人，如同面临满载产品的庞大货轮，不知从何开始，也不知将来何去何从。

提示 >>> 如果程序语言被比喻为一艘船，会是如何呢？这里有篇有趣的文章：

<http://compsci.ca/blog/if-a-programming-language-was-a-boat/>

在 Java 的官方网站提供有一份 Java 技术观念地图(Java Technology Concept Map)的文件，如图 1.4 所示。这是份 PDF 文件，可以在以下的网址下载：

<http://java.sun.com/new2java/javamap/intro.html>

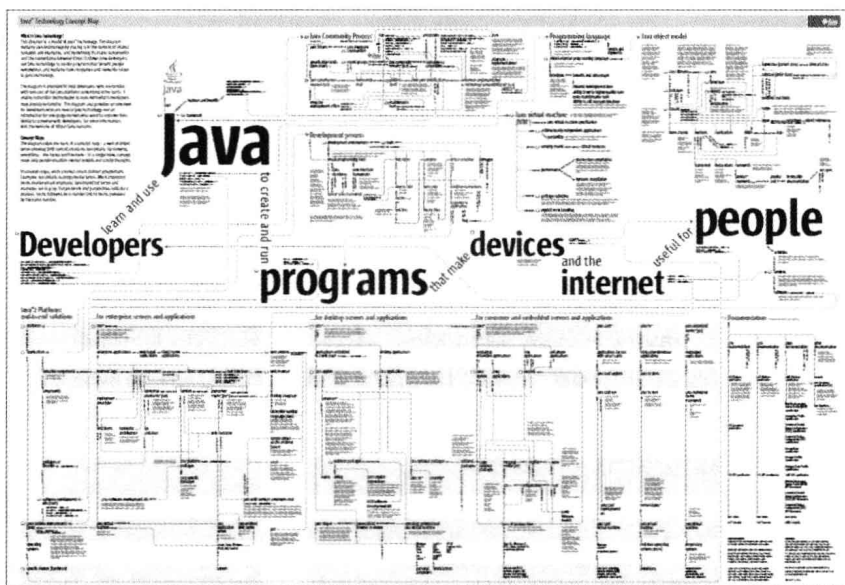


图 1.4 Java 技术观念地图

在这份文件中，密密麻麻地列出了大部分 Java 相关地图与简要说明，也代表了 Java 技术范畴的广泛，然而要从这么庞大的地图中找出一条适合初学 Java 的路线图绝非易事。以下是我基于经验与教学建议的学习路径。

1. 深入了解 JVM/JRE/JDK

许多书籍对于 JVM/JRE/JDK 的说明，通常以极短的篇幅介绍，就是在短短几页中，请使用者依书中步骤安装与设定 PATH、CLASSPATH 后，就开始介绍 Java 程序语言，而许多人到了业界后就开始使用 IDE(Integrated Development Environment)代劳所有 JDK 细节。这么做的结果就是，在 IDE 中遇到与 JDK 相关的问题，就完全不知道如何解决。

JVM/JRE/JDK 并不是用短短几页就可以说明，若没有“JVM 是 Java 程序唯一认识的操作系统，其可执行文件为.class 文档”的重要观念，就无法理解 PATH 与 CLASSPATH 并非同一层级的环境变量，JDK 中许多指令与选项，其实都可以对应至 IDE 中某个设定与操作，你对 JVM/JRE/JDK 有足够的认识，对 IDE 中相关选项就不会有疑问，也不会换个 IDE 就不