OCA Java SE 7 Programmer I 认证学习指南

(Exam 1Z0-803)

[美] Edward Finegan 著
Robert Liguori
张方勇 译

清华大学出版社

北　京

Edward Finegan,Robert Liguori
OCA Java SE 7 Programmer I Study Guide (Exam 1Z0-803)
EISBN: 978-0-07-178942-4

图书在版编目(CIP)数据

OCA Java SE 7 Programmer I 认证学习指南(Exam 1Z0-803)/(美)法恩根(Finegan, E.)，(美)李戈里(Liguori, R.)著；张方勇 译. —北京：清华大学出版社，2014

书名原文：OCA Java SE 7 Programmer I Study Guide (Exam 1Z0-803)

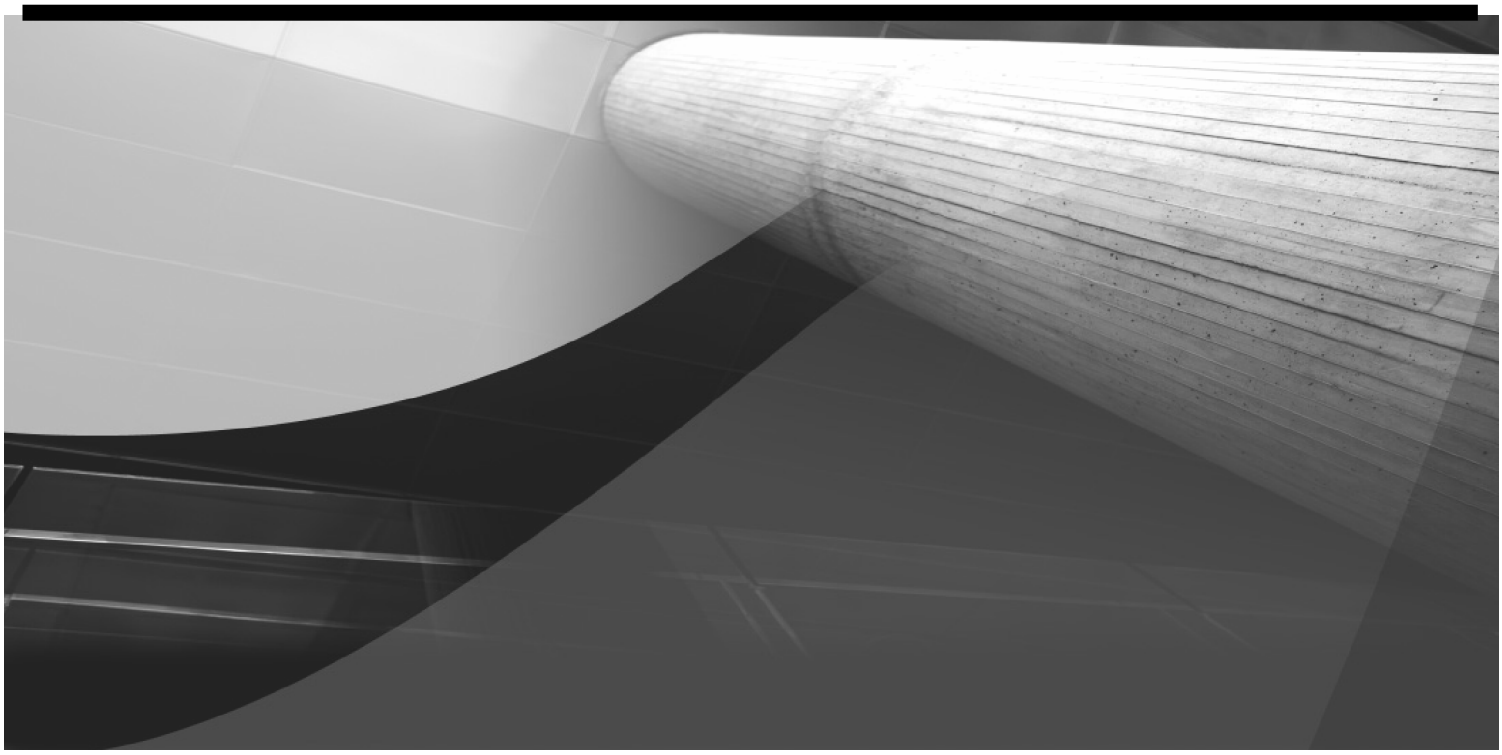
ISBN 978-7-302-35543-4

Ⅰ. ①O…　Ⅱ. ①法…　②李…　③张…　Ⅲ. ①JAVA 语言—程序设计—工程技术人员—资格考试—自学参考资料　Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2014)第 035556 号

责任编辑：王　军　刘伟琴
装帧设计：孔祥峰
责任校对：曹　阳
责任印制：
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　　**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：
装 订 者：
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：20.25　　**字　　数**：541 千字
版　　次：2014 年 3 月第 1 版　　**印　　次**：2014 年 3 月第 1 次印刷
印　　数：1～3000
定　　价：58.00 元

产品编号：



译 者 序

认证是敲门砖，是踏脚石，是进入行业大门的钥匙。它表示权威机构的认可。Oracle 认证工程师和 Java 程序员认证是 Java 权威机构对你掌握的 Java 基础知识的认可。

Java 语言风靡全球，让无数的人为之疯狂，前仆后继地想进入 Java 世界。但进入任何一行都有门槛，Java 也不例外。对于传统的程序员来说，Java 是面向对象的语言，与传统语言理念上大相径庭；而对于没有接触过编程语言的新人来说，Java 相关的技术太多，包括 Java SE、Java EE、EJB、JVM 等，众说纷纭。新手往往觉得千头万绪，不知该如何下手。几次碰壁之后，渐渐熄灭了胸中的一腔热血，Java 成了曾经的一个梦想，不由令人扼腕叹息。

OCA 认证能让人越过这道门槛，获得 Java 从业资格，让梦重新起航。

这本《OCA Java SE 7 Programmer I 认证学习指南(Exam 1Z0-803)》旨在帮助读者获得 OCA 认证，因此通俗易懂，消除了对 Java 的陌生感和神秘感，从而得以登门而入，甚至安身立命，实现人生理想。

本书作者具有丰富的 Java 实践经验，同样具有认证经验，本身也通过了认证考试，知道通过认证需要具备哪些知识，也了解考生的想法。因此本书可谓是：知其所需，投其所想。

本书从分包、编译和解释 Java 代码开始，循序渐进，介绍了 Java 的基本数据类型、语句、运算符、字符串编程。然后进一步引出面向对象的概念、类继承、接口实现、方法和变量的作用域，

层层递进，介绍了多态与类型转换。最后还介绍了异常处理，以及类之间的关系。此外，本书配备了大量的自测题，辅以详细的答案与解释，并穿插了许多“考试提示”、“实际经验”和“考试内幕”栏目，都是通过认证的强有力的手段与保障。读者在阅读本书时，有两点建议：对书中的练习要反复推敲，尝试不同的运行结果，分析代码的用意；对书中的实例要举一反三，透彻理解，分析更多的实用场景。

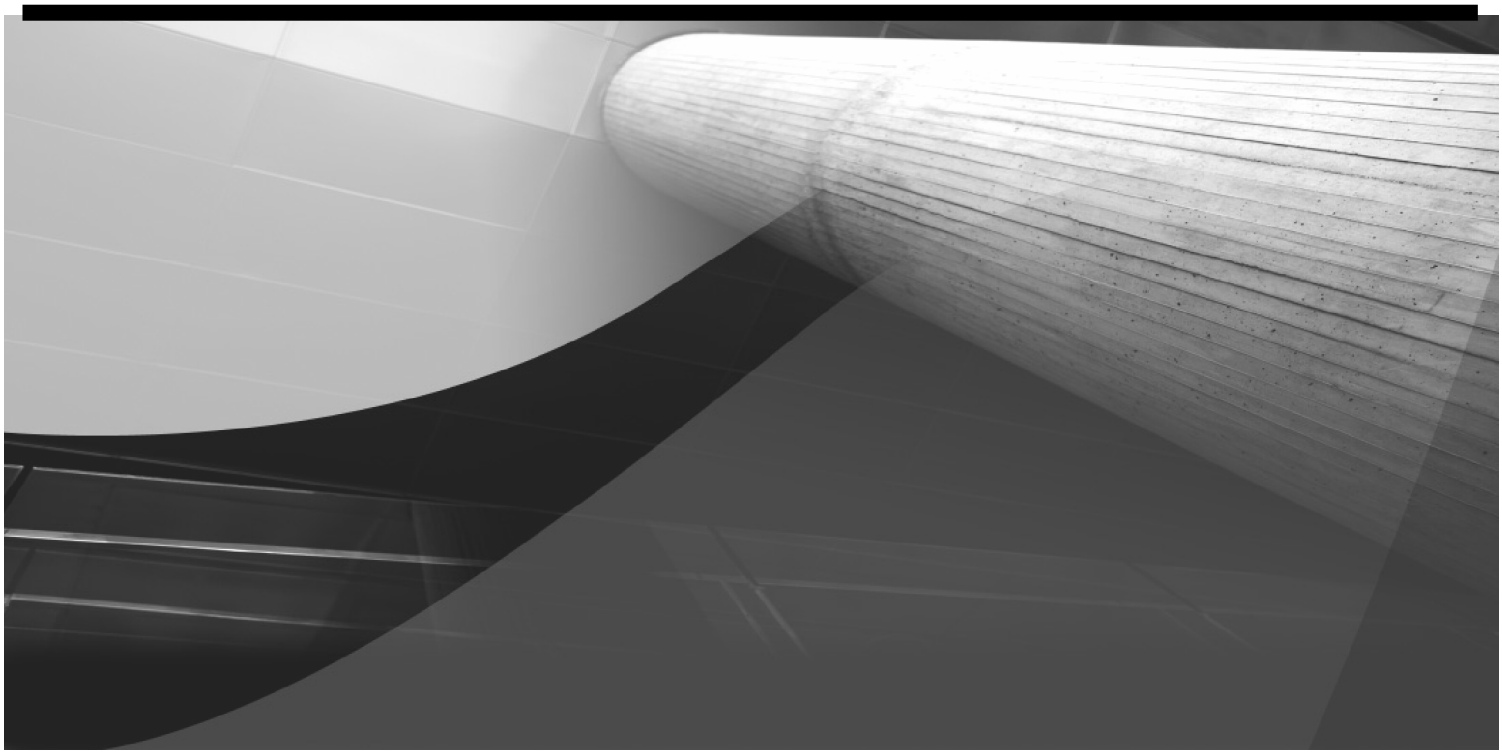
本书所有章节由张方勇翻译，参与本次翻译活动的还有李莹、马少伟、刘璇、李娟、桂华、陈永杰、刘新娟和李娟娟。另外我要特别感谢李莹，你提供了无数有价值的建议，帮助我审核校对。没有你们的帮助和鼓励，我不可能完成这本书的翻译工作。

鉴于译者水平有限，错误和失误在所难免，敬请广大读者批评指正。

最后，希望通过本书能让读者在短时间内对 Java 有一个全面的认识和了解，通过认证，能早日进入 Java 之门，享受 Java 之香。

张方勇

2013年11月于北京



作 者 简 介

Edward Finegan 是 Dryrain Technologies 公司的创始人。他的公司专门从事由企业 Java 后台支持的移动软件开发。他之前曾在赌场博彩业工作，在那里，他设计和实现博彩机的软件。他之前还具有空中交通管理系统和雷达协议的经验。

Finegan 拥有 Rowan 大学计算机科学的学士学位和 Virginia Commonwealth 大学计算机科学的硕士学位。他的论文 Intelligent Autonomous Data Categorization 研究了使用机器学习算法智能地和自主地对数据分类的可能性。

Finegan 是一个狂热的费城体育迷。他喜欢花时间陪家人，尤其是他的妻子 Shannon 和新生女儿 Adalyn。他的空闲时间都用在单打独斗的户外活动和家装项目以及最新的技术上。

可以通过 edward@ocajexam.com 与他联系。

Robert Liguori 是一位计算机科学家，自从 1996 年以来，他一直从事空中交通管理系统的开发、维护和测试工作。目前他为美国联邦航空管理局的空中交通管理系统提供 Web 服务和测试服务的支持。Liguori 是一位使用 STG 技术的软件/测试工程师。

Liguori 拥有新泽西州 Richard Stockton 学院的计算机科学和信息技术的学士学位。他获得了各种 Oracle 认证。

Liguori 喜欢摆弄有创造力的玩具，如 Capo Critters(坐在吉他顶部的独特小动物)。

在 2010 年，Liguori 联手 Ryan Cuprak 编写了书籍 *NetBeans IDE Programmer Certified Expert Exam Guide (Exam* 310-045)(McGraw-Hill，2010)。这本书的读者对象主要是准备 NetBeans IDE 考试的人。

在 2008 年，Liguori 同 Edward Finegan 密切合作，编写了这本 OCA 书的前身 *SCJA Sun Certified Java Associate Study Guide (CX-310-019)*(McGraw-Hill，2009)。这本书的读者对象主要是准备 SCJA 考试的人。

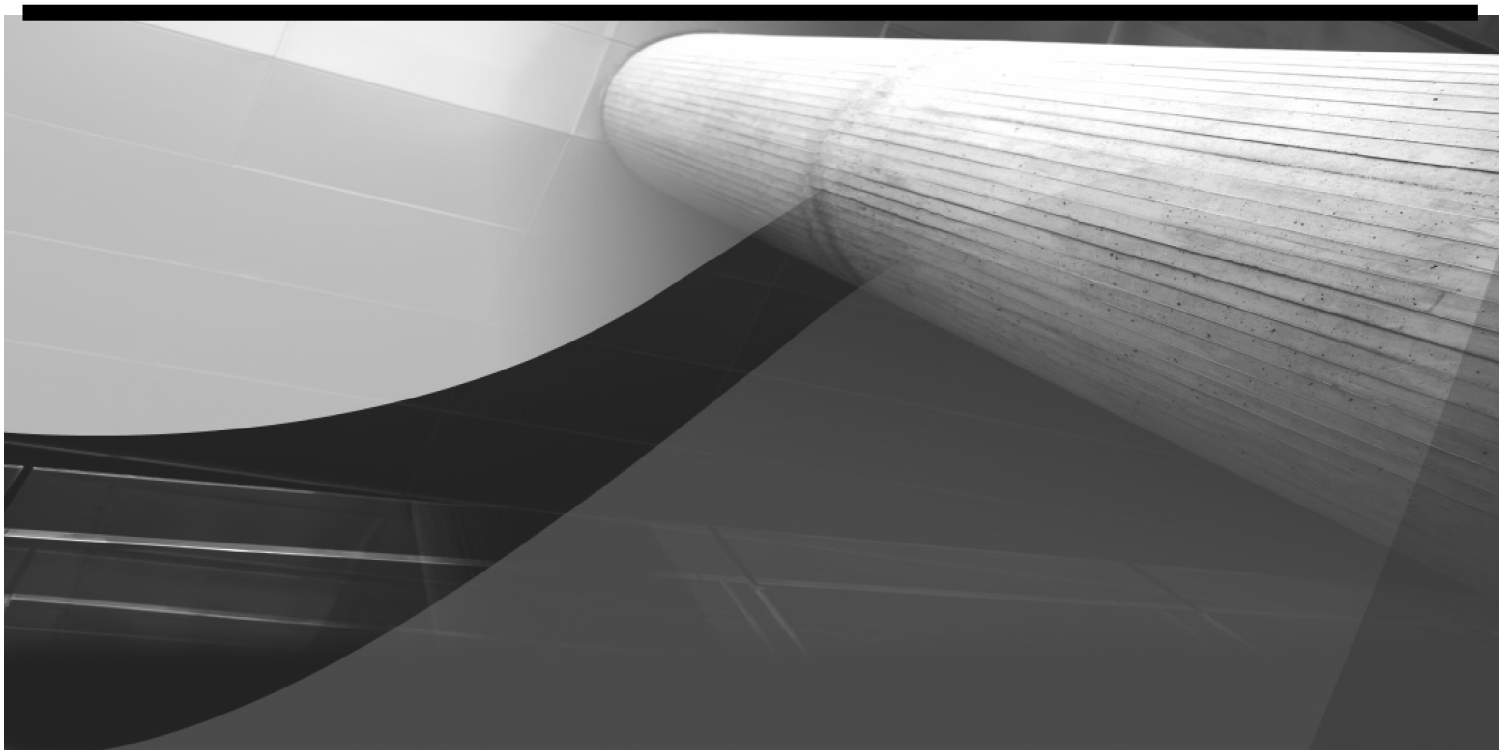
在 2007 年，Liguori 与他的妻子 Patricia Liguori，将空余时间花费在共同创作一本便携的 Java 参考指南上：*Java Pocket Guide*(O'Reilly Media Inc.，2008)。这本书聚集了同类书籍中的 Java 基础。

Liguori 喜欢花时间陪家人，以及沿着美国东海岸的冲浪捕石斑鱼(鲈鱼)、tautog、石首鱼、青鱼、鲑鱼和比目鱼。

可以通过 robert@ocajexam.com 与他联系。

技术编辑简介

Ryan Cuprak 是 Enginuity PLM 的一位电子商务分析师和 Connecticut Java 用户组(从 2003 年开始运作)的主席。在 Enginuity PLM，他集中开发了数据集成来转换客户端数据，以及用户界面开发。加入 Enginuity 之前，他为初创分布式计算机公司 TurboWorx 和 Eastman Kodak 的 Molecular Imaging Systems(现在是 Carestream Health 的一部分)工作。在 TurboWorx，他是一名 Java 开发人员，还是一位支持售前和专业服务的技术销售工程师。Cuprak 拥有芝加哥 Loyola 大学计算机科学和生物学的学士学位。他是一位 Sun 认证的 NetBeans IDE 专家。可以通过 ryan@ocajexam.com 与他联系。



致　　谢

OCA 考试涵盖了 Java 基础到面向对象概念方面大量的详细信息。要完成像本书这样一项浩繁的项目，涵盖所有相关的目标，作者决定采取分而治之的办法，基于个人的专业知识来划分章节。Finegan 专注于涵盖面向对象基本概念的章节。Liguori 专注于 Java 核心基础的章节。

本书项目团队致谢

麦格劳-希尔(McGraw-Hill)和支持团队：Timothy Green、Stephanie Evans、Jody McKenzie、Jim Kussow、Melinda Lytle、Tania Andrabi、Vasundhara Sawhney、Lisa Theobald、Carol Shields 和 Rebecca Plunkett

Waterside 制作公司：Carole Jelen McClendon

技术编辑：Ryan Cuprak

非正式审校者：Shannon Reilly Finegan、Richard Tkatch、Wayne Smith (SCJA 版本)

个人致谢

感谢我所有的家人和朋友。像这样一个项目总是会耗费比预想更多的时间。他们的耐心和鼓励一直激励着我完成这次冒险。特别感谢我的妻子 Shannon。在这个项目的中途，她生下了我们的第

一个孩子 Adalyn，并且帮助我协调家庭和事业之间的时间。如果没有她的支持，本书绝不可能完成。

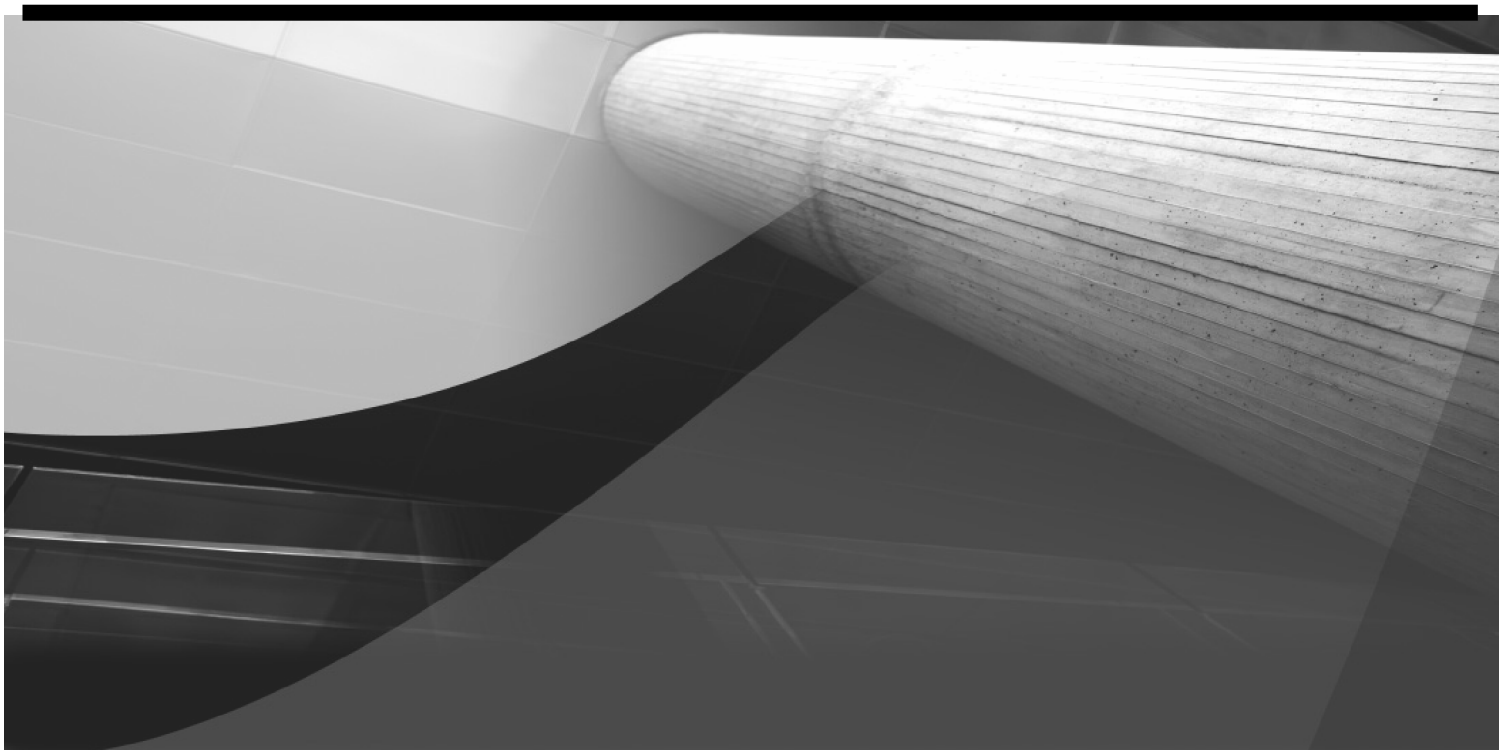
感谢我的合著者 Robert Liguori。同他合作这个项目是一次丰富的体验。Robert 是一位卓有才华的专业人士，衷心感谢他使我步入正轨。他对 Java 社区的热情和奉献真是了不起，而且在工作中也是显而易见的。

—Edward Finegan

感谢我亲爱的妻子兼朋友Patti。也感谢总是支持我的父母、家人和朋友。特别感谢我的女儿 Ashleigh Liguori 以及我的侄女 Ryley Wilson，因为她们最近总试图在饭桌上堆积如山的书籍、文章、论文、笔记本电脑等等下面找到我。感谢我的哥哥 Michael 绘制了下面这幅令人愉悦的图画，在本书的整个撰写期间，我已经将它设置为我的桌面背景。

最后，我要真诚地感谢我的合著者 Edward Finegan，他作为主要作者负责了本书的更新。因此，我由衷地感到，本书读者将大大受益于 Edward 和我使本书尽可能完整有用的努力投入。所以，亲爱的读者，祝你们考试顺利，感谢你们选择本书作为考试准备资料。

—Robert Liguori



序　言

本书的目的是帮助你准备OCA Java SE 7 Programmer I(1Z0-803)考试以获得 Oracle Certified Associate (Oracle 认证工程师)以及 Java SE 7 Programme (OCA Java SE 7 程序员)认证。这些准备需要你熟悉 Java 基础知识、概念、工具和考试中将出现的技术相关的必要知识。总之，这本书的范围是帮助你通过考试。因此，详细讲述了目标明确的领域。考试不需要用到的其他信息，可能没有包括在内，或者是以有限的篇幅呈现。由于这本书涵盖了 Java 基础和相关技术的大量信息，因此认证过程之后你也可以将它用作一般性的参考指南。

获得OCA Java SE 7 Programmer认证将巩固Java编程语言知识，为你打下相关技术进阶的基础，并标志着你已成为一名真正的Java专业人士。我们强烈向成熟的程序员和软件开发者推荐OCA Java SE 7 Programmer 认证。

关于 Java 的 Oracle 认证系列包括有关 Java SE 和 Java EE 两者的各种考试。本学习指南重点介绍了 Java SE 的 SE 相关考试的第一步。

通过了 Java SE 7 Programmer I(1Z0-803)考试可以获得 Oracle 认证工程师和 Java SE 7 程序员认证。一旦获得该认证，就可以参加 Java SE 7 Programmer II (1Z0-804)考试来获得 Oracle 认证专家和 Java SE 7 认证。如果你之前已经有了 Java 认证，可以选择升级到 Java SE 7 Programmer (1Z0-805)考试来直接获得 Oracle 认证专家以及 Java SE 7 认证。

本书预览

本书涵盖了 Java 的基础知识，包括开发工具、基本构造、运算符和字符串。另外本书还涵盖了面向对象的概念，包括类、类之间的关系以及面向对象的原则。一组附录涵盖了 Java 关键字、括号规范、Unicode 标准、伪代码算法、Java SE 包和统一建模语言(Unified Modeling Language，UML)。还提供了一个有用的术语表。祝你阅读愉快。

在线资源

本书中文支持网站 www.tupwk.com.cn/downpage 上提供了完整的 MasterExam 和一个企业架构项目文件。

MasterExam 软件易于安装到任何 Windows 2000/XP/Vista/7 计算机上，并且必须安装该软件才能访问 MasterExam 的功能。作为注册 MasterExam 的奖励，只需要单击主启动页面上的 Bonus MasterExam 链接并按照指示免费网上注册即可。

MasterExam 提供了真实考试的模拟。问题的数量、问题的类型以及允许的时间都试图还原真实的考试环境。你可以选择参加开卷考试(包括提示、参考和答案)、闭卷考试或者定时 MasterExam 模拟。当启动 MasterExam 时，在屏幕底部的右下角将会显示一个数字时钟。时钟将会持续倒计时到 0，除非你选择在时间到之前结束考试。

企业架构(EA)是一个 CASE 工具，支持 UML 建模和源代码逆向工程。本书使用 EA 来创建 UML 草图，如图 P-1 所示。因为了解 UML 是考试的要求，本书支持网站提供的文件包括了这些图表的项目文件，以帮助你学习。要打开项目文件，必须有一个 EA 版本。可以从 Spark Systems 网站 http://www.sparxsystems.com/products/ea/trial.html 上下载应用程序的 30 天试用版本。安装完成后，就可以查看和修改每一个 UML 图。你会发现被组织在 EA 项目中的图表，如同在每一章中出现的一样。关于 UML 的详细信息可以参考附录 G。

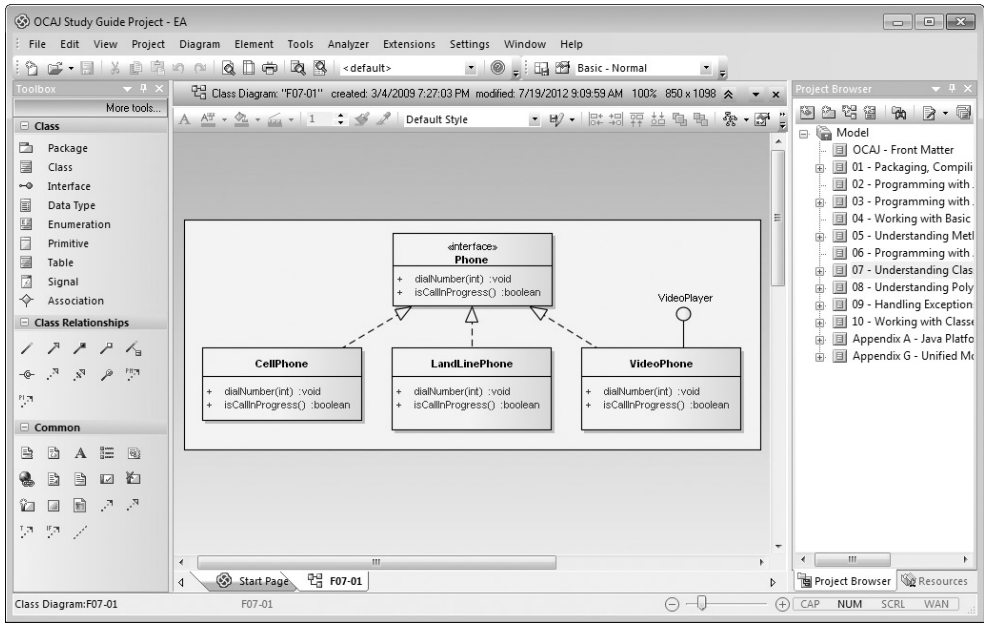


图 P-1　企业架构 CASE 工具

通过主页左下角的帮助按钮可以查看帮助文件，也可通过 MasterExam 使用独立的帮助功能。

MasterExam 安装在你的硬盘驱动器上。为了达到删除程序的最佳效果，使用 Start | All Programs | LearnKey | Uninstall 选项来删除 MasterExam。

LearnKey 提供了自学内容和多媒体传输解决方案，以提高个人技能和业务生产力。Learnkey 要求最大的库具有丰富的流媒体训练内容，使学员能够参与动态的富媒体课程，完成视频剪辑、音频、全动态图像和动画插图。可以在网络 www.LearnKey.com 上找到 LearnKey。对于 LearnKey 软件的技术问题(安装、操作、移除安装)，请访问 www.learnkey.com 或者发送电子邮件到 techsupport@learnkey.com。

关于本书内容相关的问题，请发送电子邮件到 international_cs@mcgraw-hill.com。

考试准备清单

在“前言”的末尾，你将会发现一个“考试准备清单”。通过这个表格你可以交叉引用官方的考试目标以及本书中涵盖的目标。该清单也可以让你在开始学习时衡量在每项目标上的专业水平。这应该能让你检查自己的进度，确保将时间花费在更难的或者不熟悉的章节上。对于厂商提出的每个目标，都可以在此清单中找到它位于本书的哪一章中。

编排方式

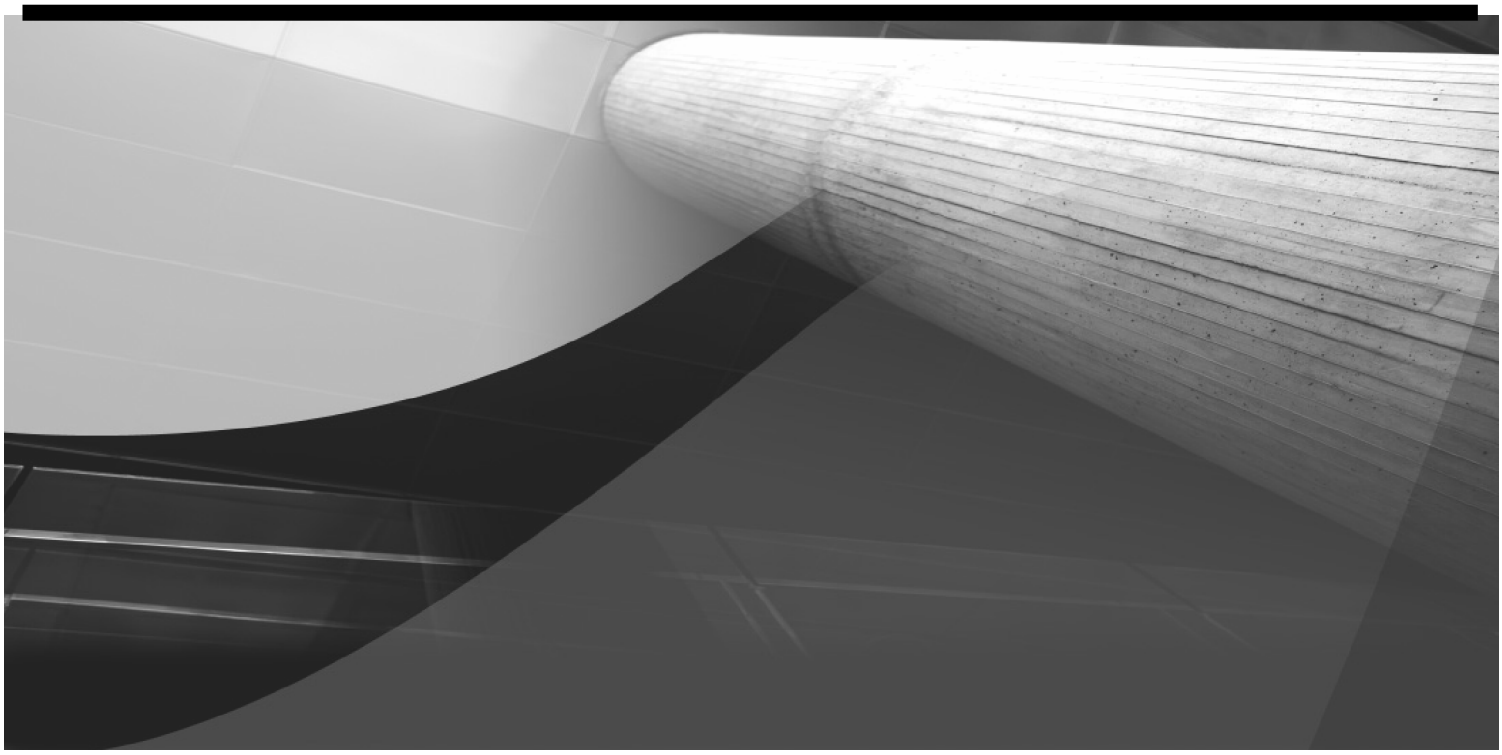
每章都有一组栏目，用来吸引你对重要项目引起注意，强调重点，并提供有用的考试提示。下面列出你将在每章中发现什么：

- **认证目标**——它位于每章的开头，是你需要知道的在考试中该章节所处理的内容。目标标题确定了该章的目标，所以阅读它时，应该总是明白目标。
- **考试提示**——吸引注意考试的有关信息，以及潜在的陷阱。这些有用的提示是由已经参加过考试并获得认证的作者所写。谁能更好地告诉你需要担心什么？他们知道你将要考的是什么！
- **分步练习**——穿插在整个章节中。这些练习通常设计为动手练习，让你感受一下真实世界的经验以便通过考试。它们有助于你掌握可能是考试重点领域的技能。不要只是阅读这些练习——它们应该是你乐于动手完成的实践。在实践中学习是一种用产品来提高你的竞争力的有效方式。
- **实际经验**——描述了在现实世界的设定中最经常出现的问题。它们对认证和产品相关的主题提供了有价值的观点。指出了常见的错误，并解决了从实际经验的讨论和经验中引出的问题。
- **考试内幕**——强调了学生参加现场考试时遇到的一些最常见的和最容易混淆的问题。预测了考试的重点，深入到考试内部，这将有助于确保你知道应该知道的内容，以通过考试。通过重点关注这些栏目，你可以明白如何应对那些难以理解的问题。
- **场景与解决方案**——以一种快速阅读的格式列举了潜在的问题和解决方案。
- **认证小结**——是该章的一个简洁复习以及考试相关要点的重述。
- **知识点回顾**——它位于每章的末尾，是该章要点的一个清单，可以用于最后复习。
- **自测题**——提供了类似于在认证考试中遇到的问题。这些问题的答案，以及答案的解释，可以在每章的末尾找到。阅读完每章后通过完成自测题，能够强化你从该章学到的内容，同时熟悉考试问题的结构。

学习指导

一旦阅读完本书，请花费一些时间来做一个彻底的复习。你可能需要多次翻阅本书，并利用书中提供的下列方法来进行复习：

- 重新阅读所有的“知识点回顾”，或者请别人考考自己。可以将这些知识点回顾制作成小卡片，临考试前快速过一遍。
- 重新阅读所有的“考试提示”和“考试内幕”。请记住，这些提示都是由那些参加过考试并且通过的作者编写的。他们知道什么是你期望的，以及什么是你应该注意的。
- 复习用于快速解决问题的所有“场景和解决方案”。
- 重做自测题。阅读该章后，马上做这些测试题是一个好方法，因为这些问题可以帮助强化刚学到的东西。然而，一个更好的主意是，稍后再回来并一次性地考虑本书中的所有问题。假设你正在参加现场考试。当你第一次通看问题时，应该在一张单独的纸上记下你的答案。通过这种方式，只要你需要，就可以多次回顾这些问题，直到你对这些材料感到不陌生。
- 完成练习。当你阅读完每一章时，你做这些练习了吗？如果没有，去做吧！这些练习涵盖了考试的内容，并且没有比通过练习来了解这些材料更好的方法。要确保理解每一个练习中的每一步为什么要这么执行？如果你对某个特定的主题不清楚，请重读相关章节。



前　言

本书旨在帮助你准备通过 Oracle 的 Java SE 7 Programmer I 考试以获得 OCA Java SE 7 Programmer 认证。本书信息的呈现方式包括文字内容、代码示例、练习以及其他方式。为了达到最佳效果，所有的代码示例都在 Macintosh OS X 和 Windows 计算机上验证过。关于所有考试目标的信息也都详细涵盖。本书涵盖的主要领域列举如下：

- Java SE 平台
- Java 开发和支持工具
- Java 基础，包括语句、变量、方法原语和运算符
- String 和 StringBuilder 类的方法和功能
- 基本的 Java 元素，包括基本数据类型、数组、枚举和对象
- 类和接口，包括类之间的关系
- 面向对象的原则
- 异常处理

本书后面包括多个附录，以辅助你学习。

关于 OCA Java SE 7 Programmer 认证考试的具体信息

OCA Java SE 7 Programmer 考试目标的具体信息详见 Oracle 的认证网站 http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=41&p_org_id=1001&lang=US&p_exam_id=1Z0_803。当你报名考试时，Pearson VUE 将提供关于考试注册流程的具体细节。但是，下面几小节详细讲述了报名和参加考试时需要知道的最重要的信息。关于考试目标最新的信息请访问 Oracle Certification 网站。

考试参考

这门考试的正式名称是 Java SE 7 Programmer I (1Z0-803)考试。如果你通过了考试，收到的证书是 Oracle 认证工程师以及 Java SE 7 程序员认证。

在网上用谷歌搜索，会看到人们和资源都称呼这个 1Z0-803 考试为 OCA 考试、OCAJ 考试、OCAJP 考试和 OCAJP7。这分别对应 OCA 认证、OCAJ 认证、OCAJP 认证和 OCAJP7 认证。

为简单起见，我们已经尽最大的努力将该考试称为 OCA 考试，将认证称为 OCA 认证。

Java SE 7 Programmer I (1Z0-803)考试动态

这门考试是面向希望获得基础 Java 认证的成熟 Java 程序员和开发者。无先决条件的考试由 90 个问题组成。必须有 75%的通过率——换而言之，90 个问题中至少有 68 个必须回答正确。设定的时间限制为 150 分钟(即 2 小时 30 分钟)。

目前美国参加该考试的价格是 300 美元。如果你受雇于一个技术组织，可以查看你的公司是否有教育援助政策。

Java 标准版 5 和 6，认证工程师(1Z0-850)考试动态

传统的 SCJA(CX-310-019)考试已经更新到 Java 标准版 5 和 6，通过注册工程师考试将获得 Oracle 认证工程师，以及 Java SE 5/SE 6 认证的资格。新的考试代码为 1Z0-850，并且针对以下认证考生：

- 入门级和初级程序员，希望开始或继续走使用 Java 技术的道路。
- 软件开发者和技术领导，希望巩固其 Java 相关的技能。
- 项目和程序经理，希望获得工作的真实情况以及他们团队所遇到的挑战，完善其规划和方向。
- 计算机科学和信息系统的学生，希望补充他们的学业。
- 寻找丰厚职位的 IT 求职者。
- 认证的求职者，希望完善自己的简历或履历。

如果你的目标是 Java SE 5/6 的考试版本而不是当前的 Java SE 7 版本，本书的前身 *SCJA Sun Certified Java Associate Study Guide* (*CX-310-019*)(McGraw-Hill，2009)是很好的资源，可以帮助你获得认证。

预约 OCA 考试

该考试必须在有监考的 Pearson VUE 考试地点进行。你可以通过三种方法中的一种来预约考试：

- 在线预约
- 通过电话预约
- 通过考试中心预约

关于以上这些选项所需要的所有信息，可以在 Pearson VUE 的网站(www.pearsonvue. com/oracle/)上找到。快速浏览一下，图 I-1 显示了在 Pearson VUE 网站上预约的过程。

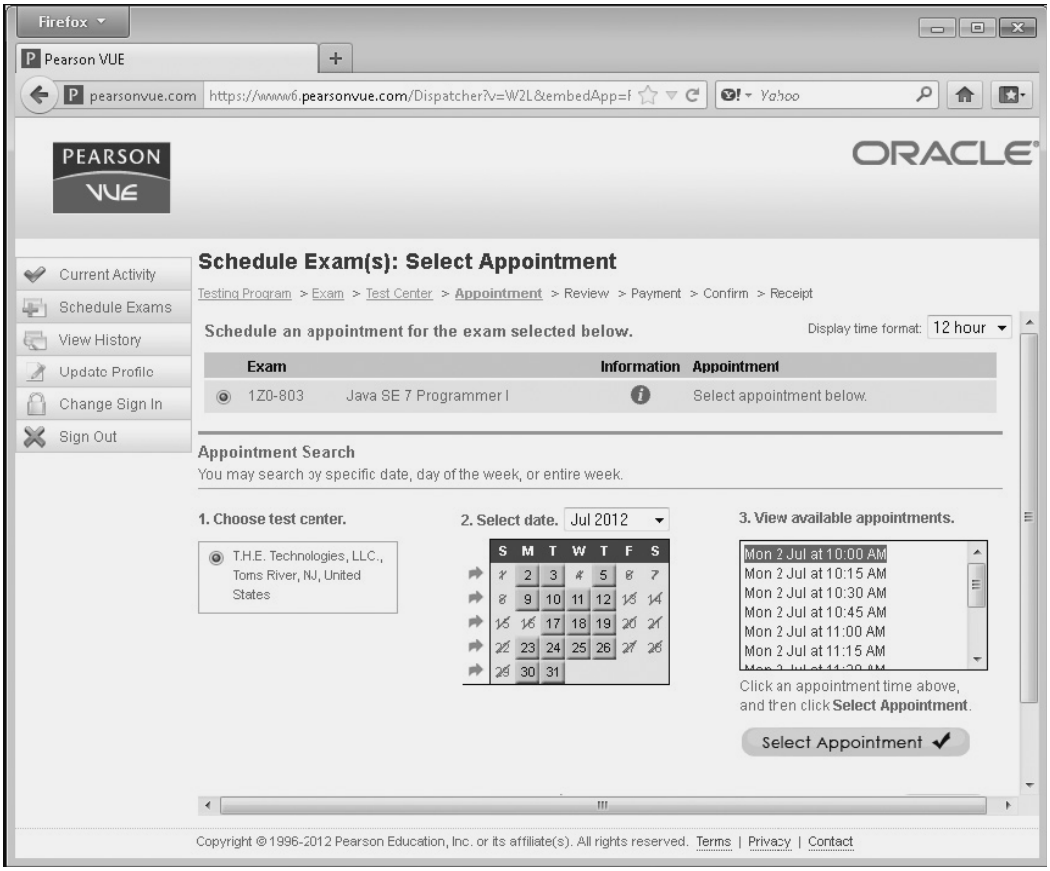


图 I-1　预约考试

Pearson VUE 的考试券

考试券(预付考试认证)可以用于购买OCAJP7考试。关于购买考试券的更多信息，请访问Pearson VUE 网站的试卷信息页面 www1.pearsonvue.com/vouchers/，或者访问 Pearson VUE 的考试券商店 www1.pearsonvue.com/contact/voucherstore/。

Pearson VUE Test Center Locator

如果你有兴趣寻找你附近的考试场地而不必注册，则可以使用www.vue.com/vtclocator/上的 Pearson VUE Test Center Locator。

准备 OCA 考试

在考试前好好休息并吃一顿健康的早餐，将有助于取得好的考试成绩，不过你应该已经知道这

些。考试的前晚不要临时抱佛脚。如果你发现需要补习，应该重新安排考试，因为你还没有准备好考试。

你需要带上一些东西参加考试，并且会留下一些东西。让我们来看一看该做什么和不该做什么。

待办事项

- 确保(至少)带两种形式的身份证明。有效的身份证件包括有效的护照、当前的驾驶执照、政府颁发的身份证、信用卡或者支票兑现卡。有两项必须包括，并至少有一项必须包括你的照片。当签名时，确保不要速写，因为它必须匹配你的身份证件。
- 提前到达。计划在考试预订开始时间前 15 分钟至半个小时内到达。你可能会发现，该中心需要为考试安排几个人。早点进入可能需要早点起床，或者至少确保不要迟到。
- 提前上厕所。这门考试需要接近两个小时才能完成。想要休息有些棘手，并且很有可能不允许。另外，你休息也不能暂停分配给你的时间。
- 打印出考试场地的路线。或者获得地址并输入到你的 GPS 导航仪或导航应用程序中(如果有的话)。

注意事项

- 不要把笔记本电脑、平板电脑、手机或寻呼机带入考场。此外，一些场所可能要求你不能携带手表或钱包进入考试区域。
- 不要携带书籍、笔记或书写用品。可能会给你一个可擦写的书写板使用。考试开始后才能使用书写板。
- 不要携带大件物品。可能没有用于存放书包和外套的大容量存储空间。但是，考试场所可能有小的安全储物柜供你使用。在存放手机或其他电子设备之前，先将其关闭。
- 不要带饮料或小吃进考场。但是在准备考试时，可以自由地享用茶点。
- 不要在考试中心学习。

参加 OCA 考试

在参加考试之前，监考人员可能需要采集你的照片。展示你最好的微笑。你可能需要签名并等待监考人员在 PC 上设置你的考试。在我的考试中，监考人员调整了房间里的网络摄像头，这样可以正面对着我。房间中可能还有其他人在参加其他的测试，所以如果你对鼠标点击和一般的噪音或者你身边的人敏感，请带上耳塞。监考人员应该会确保你获得第一个问题，然后他将离开房间。祝你好运！

考试结束后，可能会给你一项可选的基于计算机的调查。这项调查会询问你关于技术背景和其他相关信息。一个常见的误解是，这些问题的答案可能与你参加的考试问题有关；但是，本次调查与你收到的考试问题无关。调查可能需要几分钟才能完成。收集的信息对于那些开发和细化未来的考试很重要，所以要如实地回答问题。

完成考试后，你的结果将会显示在屏幕上，并且还会打印出来。找到监考人员(考试人员)，从打印机获取你的结果并签名。这里的要点是，你不应该在考试完成后就立即离开，而应该留下来并拿你的结果。

分享你的成功

我们希望知道你是如何通过考试的。你可以给我们发电子邮件到 results@ocajexam.com，或者将结果发布到 Java Ranch 的 Wall of Fame(http://faq.javaranch.com/java/Ocajp7WallOfFame)。

重新预约 OCA 考试

如果你需要重新预约(或者取消)考试，要在考试开始之前的 24 小时完成。使用 Pearson VUE 的 Test Taker Services 页面(www.vue.com/programs/)来帮助你重新预约或者直接联系 Pearson VUE。在考试前 24 小时重新预约则收取相同的考试费。如果你不露面参加考试(也就是说，如果你不出现)，将不给予退款。

其他的 OCA 资源

许多资源可以补充这本书，帮助你实现 OCA 认证的目标。这些资源包括 Java 软件实用工具、Java 社区论坛、语言规范和相关文档、OCA 相关的书籍、在线和可购买的模拟考试，以及软件工具(如 IDE 的 CASE 工具、UML 建模工具，等等)。虽然这些外围的工具和资源是非常有益和值得推荐的，但是对于通过 OCA 考试来说它们是可选的。本书试图涵盖所有必要的材料。

以下各小节详细讲述前面提到的资源。

Java 软件

- Java 开发工具包，www.oracle.com/technetwork/java/archive-139210.html
- Java 企业版(超出了考试范围，在此提供它只是为了让你了解)，www.oracle.com/ technetwork/java/ javaee/overview/index.html

Java 在线社区论坛

- Oracle 技术论坛，https://forums.oracle.com/forums
- Java Ranch 的 Big Moose Saloon Java 技术论坛，www.coderanch.com/forums
- Java Programming 论坛，http://javaprogrammingforums.com/
- </dream.in.code>，www.dreamincode.net/forums/forum/32-java/
- IBM-Java 技术论坛，www.ibm.com/developerworks/forums/dw_jforums.jspa
- Tek Tips Java 论坛，www.tek-tips.com/threadminder.cfm?pid=269
- Code Guru-Java Programming，http://forums.codeguru.com/forumdisplay.php?f=67
- Go4Expert，www.go4expert.com/forums/forumdisplay.php?f=21
- //javareference，www.javareference.com/
- Java 用户组，http://java.sun.com/community/usergroups/和 http://home.java.net/jugs/java-user-groups

Java 工具及技术规格和文档

- Java 教程，http://docs.oracle.com/javase/tutorial/
- Java SE 7 文档，http://docs.oracle.com/javase/7/docs/
- Java SE 7 API 规范，http://docs.oracle.com/javase/7/docs/api/

- jDocs Java 文档库，www.jdocs.com/
- UML 规范，www.omg.org/spec/UML/Current/
- Java 语言规范 Java SE 7 版，http://docs.oracle.com/javase/specs/jls/se7/jls7.pdf
- Java 语言规范第 3 版，http://java.sun.com/docs/books/jls/

涵盖 OCA 考试内容的其他书籍

你手头上的这本书足以涵盖通过考试需要知道的一切，补充阅读只是起帮助作用。考虑查看下列书籍来完善自己的技能：

- *Java Pocket Guide*，由 Robert 和 Patricia Liguori 合著(O'Reilly Media Inc.，2008)
- *NetBeans IDE Programmer Certified Expert Exam Guide (Exam-310-045)*，由 Robert Liguori 和 Ryan Cuprak 合著(McGraw-Hill，2010 年 7 月)
- *Java: The Complete Reference, Eighth Edition*，由 Herbert Schildt 所著(McGraw-Hill，2011 年 6 月)
- *UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Edition)*，由 Martin Fowler 所著(Addison-Wesley，2003)
- *SCJA Sun Certified Java Associate Study Guide for Test CX-310-019, 2nd Edition*，由 Cameron W. McKenzie 所著(PulpJava，2007)
- *SCJA Sun Certified Java Associate Mock Exam Questions*，由 Cameron W. McKenzie 所著(PulpJava，2007)

OCA 模拟考试

除了与本书相关的在线模拟考试，还存在其他各种免费的和商业的 OCA 和传统的 SCJA 模拟考试。这里列出了各种资源。

- Oracle 考试练习，http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=208
- uCertify PrepKit 和可下载的模拟考题，www.ucertify.com/exams/Oracle/CX310-019.html
- Enthuware 的模拟考试，http://enthuware.com/
- Whizlabs 的 SCJA Preparation Kit，www.whizlabs.com/scja/scja.html
- eJavaguru.com 的在线模拟考试，www.ejavaguru.com/scjafreemockexam.php
- SCJA.de 的电子书和在线模拟考题，http://scja.de/
- ExamsExpert，www.examsexpert.com/oracle-certifications.html
- Transcender(超越者)，www.selftestsoftware.com/certprep-materials/oracle.kap
- 自测软件，www.transcender.com/

集成开发环境

集成开发环境(Integrated Development Environment，IDE)是开发套件，允许开发者编辑、编译、调试、连接到版本控制系统、协作以及做更多的事情，这取决于具体工具。大多数现代的 IDE 对各种软件模块有增件(add-in)功能，以增强 IDE 的功能。没有理由不使用 IDE。因为其受欢迎程度、易用性和 Oracle 的支持，我们(作者)推荐你使用 NetBeans IDE 来准备考试。不过，下面所列举的任意 IDE 都够用：

- Netbeans IDE，www.netbeans.org
- Oracle JDeveloper IDE，www.oracle.com/technetwork/developer-tools/jdev/overview/
- IntelliJ IDEA，www.jetbrains.com
- Eclipse IDE，www.eclipse.org
- JCreator IDE，www.jcreator.com
- BlueJ，www.bluej.org

带 UML 建模功能的工具

一些工具和 IDE 具有 UML 功能。需要注意的是，OCA 考试的前身包括 UML 建模的问题。然而，在 OCA 中不包括关于 UML 的问题。如果想研究 UML 建模，可以查看以下几个工具，但再次声明，这不在考试范围内。但是，我们相信，软件程序员和开发者应该在其职业生涯的早期学习 UML。

- NetBeans IDE，http://netbeans.org/features/uml/
- JDeveloper IDE，www.oracle.com/technetwork/developer-tools/jdev/jdeveloper11g-datasheet-1-133040.pdf
- Enterprise Architect CASE 工具，www.sparxsystems.com/products/ea/
- UML 的可视化范式(提供流行的 IDE 插件)，www.visual-paradigm.com/product/vpuml/

其他资源

其他各种资源，例如下面这些资源，包括游戏和 Java 的新闻媒体，可以帮助你在考试中获得高分。

- Java Ranch Rules Round Up Game，www.javaranch.com/game/game2.jsp，如图 I-2 所示

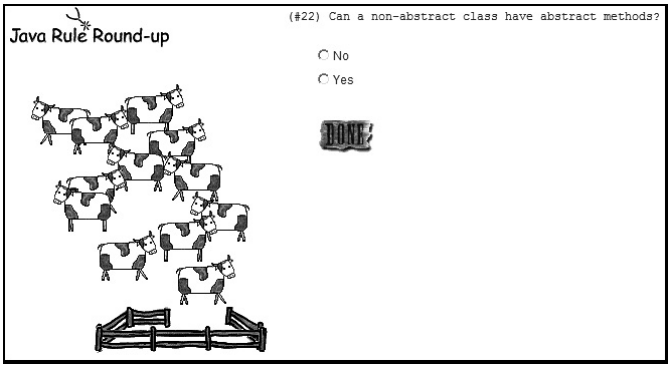


图 I-2　Java Ranch Rules Round Up 游戏

- DZone，http://java.dzone.com
- Server Side，http://theserverside.com
- Java Ranch 上的 OCA FAQ，www.coderanch.com/how-to/java/OcajpFaq
- Java Tutorial(Programmer Level I Exam)，http://docs.oracle.com/javase/tutorial/extra/certification/javase-7-programmer1.html
- Java 语言规范，http://docs.oracle.com/javase/specs/

Oracle 在 Java 技术方面的认证程序

本节将考试的目标映射到本书具体的涵盖内容。

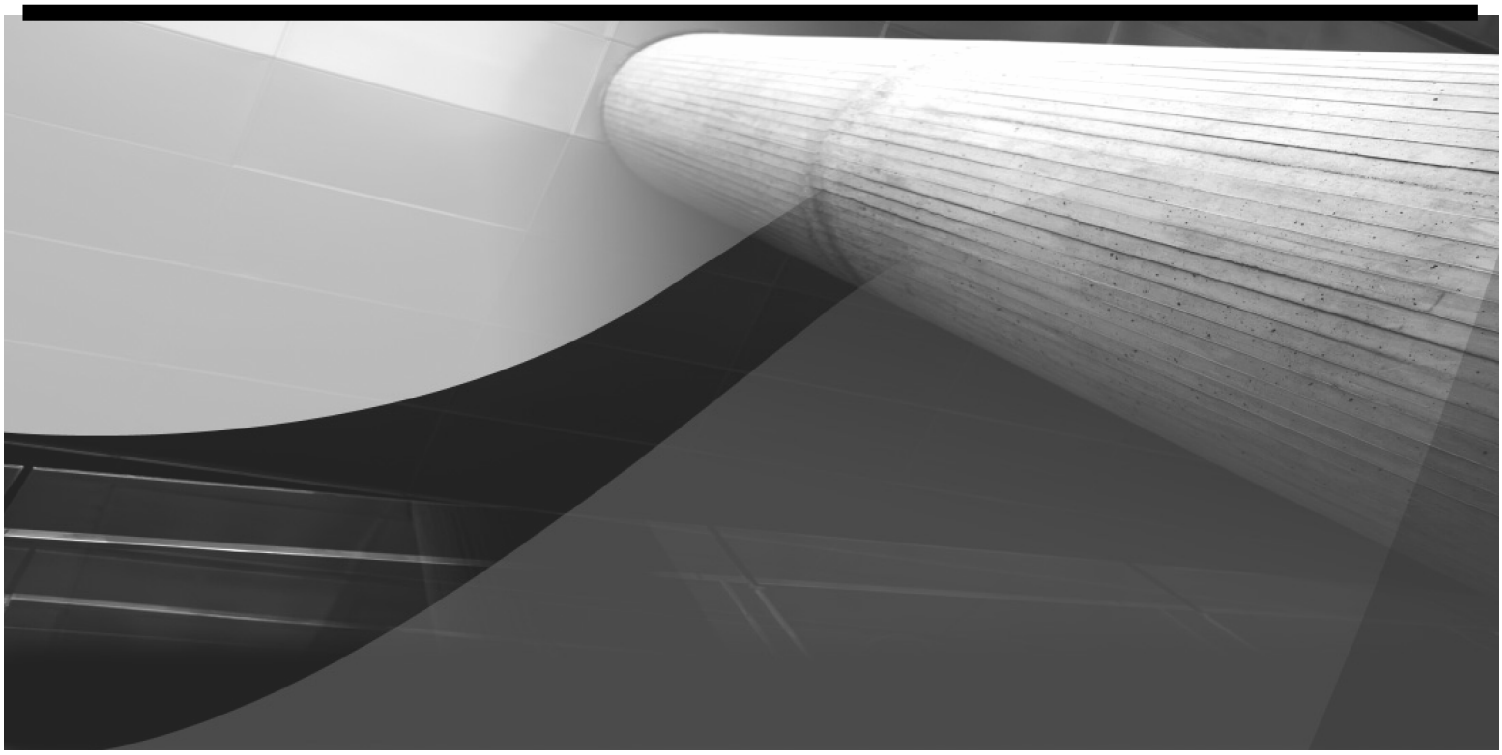
考试 1Z0-803

考试准备清单

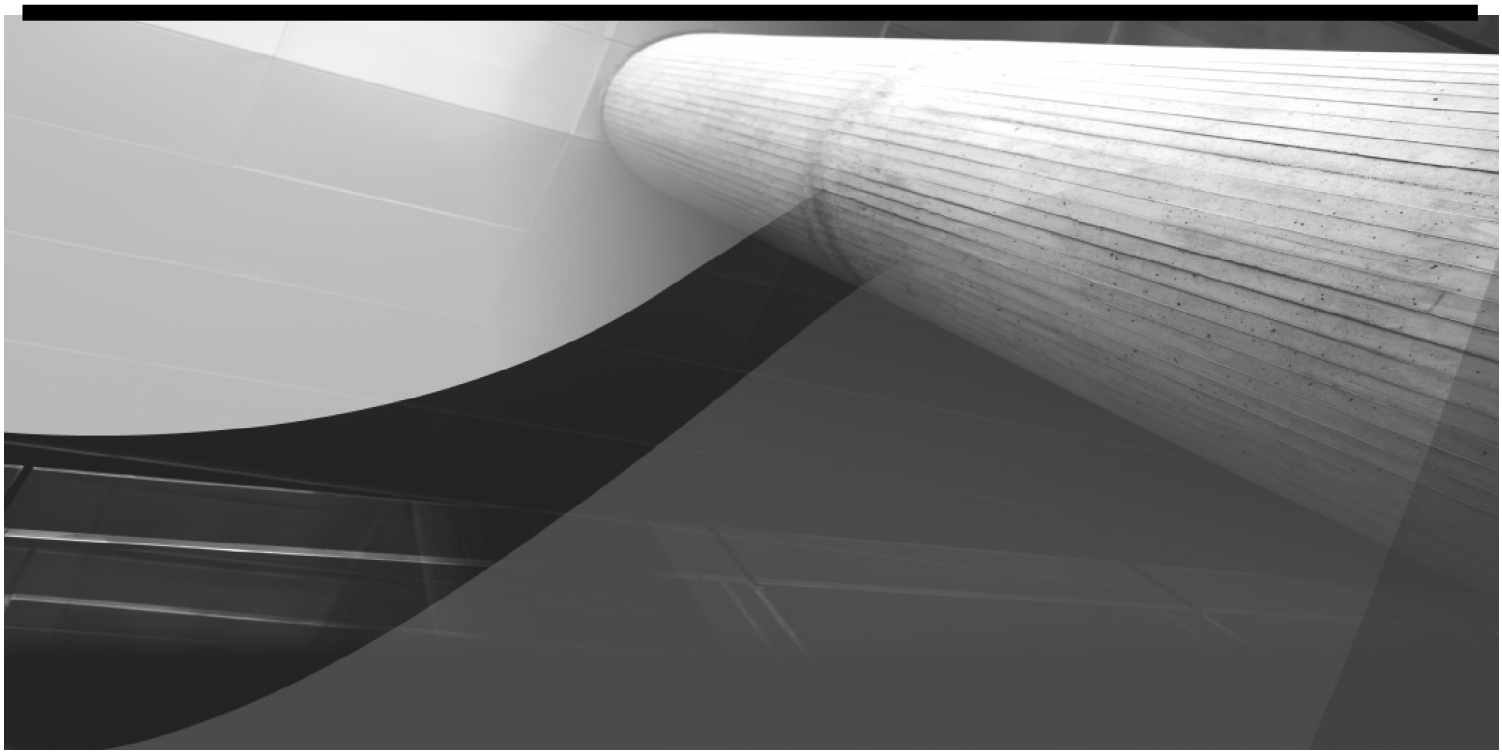
官 方 目 标	本书涵盖内容	章 号	初 学 者	中 级	专 家
1.4 导入其他 Java 包，使代码可以访问它们	理解包	1			
补充材料	理解包派生类	1			
1.2 定义 Java 类的结构	理解类结构	1			
1.3 用 main 方法创建可执行的 Java 应用程序	编译器和解释 Java 代码	1			
补充材料	理解赋值语句	2			
3.4 创建 if 和 if/else 结构	创建和使用条件语句	2			
3.5 使用 switch 语句	创建和使用条件语句	2			
5.2 创建和使用 for 循环，包括增强的 for 循环	创建和使用迭代语句	2			
5.1 创建和使用 while 循环	创建和使用迭代语句	2			
5.3 创建和使用 do/while 循环	创建和使用迭代语句	2			
5.4 对比循环结构	创建和使用迭代语句	2			
5.5 使用 break 和 continue	创建和使用控制转换语句	2			
3.1 使用 Java 运算符	理解基础运算符	3			
3.2 重写运算符优先级	理解运算符优先级	3			
2.7 创建和操作字符串	使用 String 对象及其方法	3			
2.6 使用 StringBuilder 类及其方法操作数据	使用 StringBuilder 对象及其方法	3			
3.3 使用==和 equals()测试字符串和其他对象之间的相等性	测试字符串和其他对象之间的相等性	3			
2.1 声明和初始化变量	理解基本数据类型、枚举和对象	4			
2.2 区别对象引用变量和基本数据类型变量	使用基本数据类型、枚举和对象	4			
1.1 定义变量的作用域	理解变量的作用域	5			
2.4 解释对象的生命周期	理解变量的作用域	5			
7.5 使用 super 和 this 访问对象和构造函数	使用 this 和 super 关键字	5			
6.1 创建带参数和返回值的方法	创建和使用方法	5			

(续表)

考试准备清单					
官 方 目 标	本书涵盖内容	章 号	初 学 者	中 级	专 家
6.5 创建和重载构造函数	创建和使用构造函数	5			
2.5 调用对象的方法	创建和使用方法	5			
6.2 对方法和字段应用 static 关键字	创建静态方法和实例变量	5			
6.3 创建重载方法	创建和使用方法	5			
6.4 区分默认的和用户定义的构造函数	创建和使用构造函数	5			
6.8 确定对象引用和基本数据类型值被传入改变其值的方法中时对它们的影响	通过引用和值传递对象	5			
4.1 声明、实例化、初始化和使用一维数组	使用 Java 数组	6			
4.2 声明、实例化、初始化和使用多维数组	使用 Java 数组	6			
4.3 声明和使用 ArrayList(新内容)	使用 ArrayList 对象及其方法	6			
7.1 实现继承	实现并使用继承和类的类型	7			
7.6 使用抽象类和接口	实现并使用继承和类的类型	7			
6.6 应用访问修饰符	理解封装原则	7			
2.3 读或写对象字段	类继承和封装的高级应用	7			
6.7 对类应用封装原则	理解封装原则	7			
7.2 开发演示使用多态的代码	理解多态	8			
7.3 区分引用类型和对象类型	理解多态	8			
7.4 确定何时类型转换是必要的	理解类型转换	8			
8.3 描述 Java 中异常的用途	理解异常的基本原理和类型	9			
8.1 区分检查异常、运行时异常和错误	理解异常的基本原理和类型	9			
8.2 创建 try-catch 块并确定异常如何改变正常的程序流程	改变程序流程	9			
8.4 调用抛出异常的方法	理解异常的本质	9			
8.5 识别常见异常类和类别	识别常见异常	9			
补充材料	理解类的组合与关联	10			
补充材料	类组合与关联的实践	10			



目　　录



第1章

分包、编译和解释 Java 代码

认证目标

- 理解包
- 理解包派生类
- 理解类结构
- 编译和解释 Java 代码

如果你拥有本书，或者正在阅读它的电子版，那么肯定对 Java 有认同感。你肯定也迫切地希望通过 Oracle Certified Associate (OCA) Java SE 7 Programmer 认证流程向大家证明你是一个真正的 Java 内行。因此，你应该是一个或期望成为一个 Java 程序员，并且从长远来看，是一个真正的 Java 开发者。你可能是或者计划成为管理 Java 程序员和/或开发者团队的项目经理。在这种情况下，你需要对 Java 语言及其技术有一个基本的理解。对于这两种情形，本书都适合你。

首先，你可能希望了解基本的 Java SE 平台关于类库和工具类提供的核心功能组件，以及这些组件是如何组织的。本章通过讨论 Java 包和类，以及它们的分包、结构、编译和解释过程，回答了这些问题。

当阅读完本章后，你将对 Java 类的分包、常用 Java SE 包高层次的细节以及 Java 编译和解释工具的基础有更深入的理解。

认证目标

1.1 理解包

考试目标 1.4　导入其他 Java 包，使代码可以访问它们

组织相关类和接口的一个常用方法是分包。大多数可重用的代码都是分包的。没有分包的类常见于书中和在线教程中，以及专注于狭窄领域的软件应用程序中。本节将展示如何以及何时对 Java 类进行分包，以及如何从 Java 包中导入外部类。本节将涵盖以下内容：

- 包设计
- 包和 import 语句

1.1.1 包设计

包被认为是类的容器，但实际上，它们定义了类在分层目录结构中的位置。Java 的编码规范鼓励使用分包来减少类冲突的可能性。对类分包同样也能促进代码的重用性、可维护性和面向对象的封装和模块化的原则。

当设计 Java 包时，例如，对类进行分组，应考虑表 1-1 中所示的关键领域。

表 1-1　应考虑的包特性

包　特　性	应用包特性的好处
类耦合	通过类耦合降低包的依赖性
系统耦合	通过系统耦合降低包的依赖性
包的大小	通常情况下，较大的包支持可重用性，而更小的包支持可维护性
可维护性	通常，当包具有集中的功能时，软件的变动可以被限制到一个单独的包中
命名	当命名包时需要考虑规范。对包结构使用反向域名。在包名中使用下划线分隔的小写字符单词

让我们来看一个现实世界中的示例。作为项目经理，假设你需要两套具有独特功能的类，它们将在同一个终端产品中使用。你给开发者 A 的任务是构建第一套类，而开发者 B 构建第二套类。你没

有定义这些类的名字，但是定义了包的用途以及它必须包括什么。开发者 A 将创建几个基于几何图形的类，包括一个点类，一个多边形类和一个平面类。出于模拟的目的，开发者 B 构建的类将被包含进来，包括如热气球、直升机和飞机的类。你让他们去创建各自的类(没有让他们将各自的类分包)。

到了交付时间，他们都会给你一个名为 Plane.java 的类，也就是说，一个几何平面类和一个用于飞机的类。现在有一个问题，因为所有的这些源文件(也是类文件)，不能共存于同一目录，因为它们具有相同的名称。解决方案就是分包。如果你曾经对开发者指定了包名，这种冲突绝对不会发生(如图 1-1 所示)。这给我们的教训就是：始终将代码分包，除非编码的项目在本质上是微不足道的。

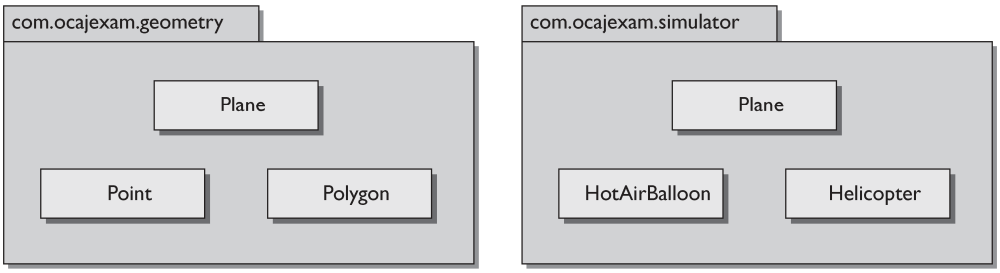


图 1-1　同名的类在不同的包中

1.1.2　包和 import 语句

你现在应该对何时以及为什么要对源码文件进行分包有一个总体的印象。接下来，需要确切地知道这究竟是如何完成的。在文件的开头使用 package 语句，将源文件放入一个包中。可以在每个源文件中使用 0 条或 1 条 package 语句。要从其他包中导入类到源文件中，可以使用 import 语句，也可以在每个类名前加上它的包名。容纳核心语言类的 java.lang 包是默认导入的。

下面的代码清单显示了 package 和 import 语句的用法。在本章详细讨论了 package 和 import 语句后，可以再返回这个代码清单。

```
package com.ocaj.exam.tutorial; // Package 语句
/*从 java.util 包中导入类 ArrayList */
import java.util.ArrayList;
/*从 java.io 包中导入所有的类*/
import java.io.*;
public class MainClass {
  public static void main(String[] args) {
    /* 从 java.io 包创建 console */
    Console console = System.console();
    String planet = console.readLine("\nEnter your favorite
    planet: ");
    /*创建 planet 的列表*/
    ArrayList planetList = new ArrayList();
    planetList.add(planet); // 增加用户输入到列表中
    planetList.add("Gliese 581 c"); // 增加一个字符串到列表中
    System.out.println("\nTwo cool planets: " + planetList);
  }
}
$ Enter your favorite planet: Jupiter
$ Two cool planets: [Jupiter, Gliese 581 c]
```

1. package 语句

package 语句包括 package 关键字，紧随其后是以点(.)分隔的包路径。表 1-2 显示了 package 语句的有效示例。

表 1-2　有效的 package 语句

package 语句	对应的目录结构
package java.net;	[目录路径]\java\net\
package com.ocajexam.utilities;	[目录路径]\com\ocajexam\utilities\
package package_name；	[目录路径]\包名\

package 语句包含下列特性：

- 它们是可选的。
- 每个源文件限制为一个。
- package 语句的标准编码规范是创建包的组织或团体的反向域名。例如，域名为 ocajexam.com 的拥有者可以对工具类包使用下面的包名：com.ocajexam.utilities。
- 包名等同于目录结构。包名 com.ocajexam.utils 就等同于目录 com/ocajexam/utils。如果一个类包含一条没有映射到对应目录结构的 package 语句，这个类将无法使用。
- 以 java.*和 javax.*开头的包名被保留。
- 包名应该小写。包名中各个独立单词应以下划线分隔开来。

Java SE 的 API 包含了几个包。Oracle 的在线 Javadoc 文档中详细描述了这些包：http://docs.oracle.com/javase/7/docs/api/。

在一个示例中，你将看到常见的包，Java 抽象窗口工具类的 API，Java Swing 的 API，Java 基础的输入/输出 API，Java 网络 API，Java 实用工具 API 以及 Java 语言的核心 API。你需要知道每个包/API 包括的基本功能。

2. import 语句

import 语句允许你在编译时将其他类的源代码包含到源文件中。import 语句包括 import 关键字，紧接着是以点(.)分隔的包路径，最后以类名或星号(*)结尾，如表 1-3 所示。这些 import 语句出现在可选的 package 语句之后，类定义之前。每条 import 语句只可以对应一个包。

表 1-3　有效的 import 语句

import 语句	定　义
import java.net.*;	导入 java.net 包中的所有类
import java.net.URL;	只导入 java.net 包中的 URL 类
import static java.awt.Color.*;	导入 java.awt 包(仅 J2SE 5.0 以上)中 Color 类的所有静态成员
import static java.awt.color.ColorSpace.CS_GRAY；	导入 java.awt 包(仅 J2SE 5.0 以上)中 ColorSpace 类的静态成员 CS_GRAY

场景与解决方案	
要绘制基本的图形和图像，应该使用哪个包？	使用 Java AWT API 包。import java.awt.*;
要创建 GUI 的轻量级组件，应该使用哪个包？	使用 Java Swing API 包。import javax.swing.*;
要使用数据流，应该使用哪个包？	使用 Java 基本的 I/O 包。import java.io.*;
要开发网络应用程序，应该使用哪个包？	使用 Java 网络 API 包。import java.net.*;
要使用集合框架、事件模型和日期/时间工具，应该使用哪个包？	使用 Java 实用工具 API 包。import java.util.*;
要使用核心的 Java 类和接口，应该使用哪个包？	使用 Java 语言的核心包，它是默认导入的。import java.lang.*;

实际经验：

出于维护的目的，最好是显式地导入类。这将让程序员能快速地确定在本类中使用了哪些外部类。举个示例，不是使用 import java.util.*，而是使用 import java.util.Vector。在这个现实世界的示例中，编码人员就能快速地看到(通过后者的做法)这个类只导入了一个类，并且它是一个集合类型。在这个示例中，因为它是一个传统的类型，所以能快速做出决定，使用一个新的集合类型来更新这个类。

C 和 C++程序员会看到 Java 的 import 语句和 C/C++的#include 语句之间一些外观和感觉上的相似之处，尽管在功能上没有直接的映射。

3. 静态 import 语句

静态导入时 Java SE 5.0 新功能(这里可以查看更多的功能：http://en.wikipedia.org/wiki/ Java_version_history)。简而言之，静态导入允许导入静态成员。以下示例语句在 Java SE 5.0 中是有效的，但是在 J2SE 1.4 中却是无效的：

```
/* 导入静态成员 ITALY */
import static java.util.Locale.ITALY;
...
System.out.println("Locale: " + ITALY); // 打印"Local: it_IT"
...

/* 导入类 Locale 中的所有静态成员*/
import static java.util.Locale.*;
...
System.out.println("Locale: " + ITALY); // 打印 "Local: it_IT"
System.out.println("Locale: " + GERMANY); // 打印 "Local: de_DE"
System.out.println("Locale: " + JAPANESE); // 打印 "Local: ja"
...
```

在此示例中如果没有静态导入，直接引用 ITALY、GERMANY 和 JAPANESE 将是无效的，并且会导致编译问题。

```
// 导入静态的 java.util.Locale.ITALY;
...
System.out.println("Locale: " + ITALY); // 将不能编译
```

练习 1-1

使用显式的 import 语句替换隐式的 import 语句

考虑下面的示例应用程序：

```
import java.io.*;
import java.text.*;
import java.util.*;
import java.util.logging.*;

public class TestClass {
  public static void main(String[] args) throws IOException {
    /* 确保已经创建了此目录*/
    new File("logs").mkdir();
    /*得到在文件名中使用的日期*/
    DateFormat df = new SimpleDateFormat("yyyyMMddhhmmss");
    Date now = new Date();
    String date = df.format(now);
    /*在 logs 目录中创建文件名*/
    String logFileName = "logs\\testlog-" + date + ".txt";
    /*创建 Logger */
    FileHandler myFileHandler = new FileHandler(logFileName);
    myFileHandler.setFormatter(new SimpleFormatter());
    Logger ocajLogger = Logger.getLogger("OCAJ Logger");
    ocajLogger.setLevel(Level.ALL);
    ocajLogger.addHandler(myFileHandler);
    /*记录日志信息*/
    ocajLogger.info("\nThis is a logged information message.");
    /* 关闭文件*/
    myFileHandler.close();
   }
 }
```

隐式导入允许导入所有必要的类包：

```
import java.io.*; // 隐式导入的示例
```

显式导入只允许导入包中指定的类或接口：

```
import java.io.File; // 显式导入的示例
```

此练习要求在示例应用程序中对所有必需的类使用显式的 import 语句来代替隐式的 import 语句。如果你对 Java 程序的编译和解释不熟悉，那么阅读完本章后再回到这个练习。否则，让我们开始吧。

(1) 将这个示例应用程序输入到一个新文件中，并命名为 TestClass.java。保存这个文件。

(2) 编译并运行应用程序以确保创建的文件内容没有错误：运行 javac TestClass.java 来编译，用 java TestClass 来运行。验证打印到屏幕上的日志信息。同样验证在日志子目录下创建的文件包含了同样的信息。

(3) 注释掉所有的 import 语句：

```
//import java.io.*;
//import java.text.*;
//import java.util.*;
//import java.util.logging.*;
```

(4) 编译应用程序： javac TestClass.java。你将会遇到几个缺少文件导入相关的编译错误。例如，图 1-2 演示了只有 java.io 包被注释后显示的错误。

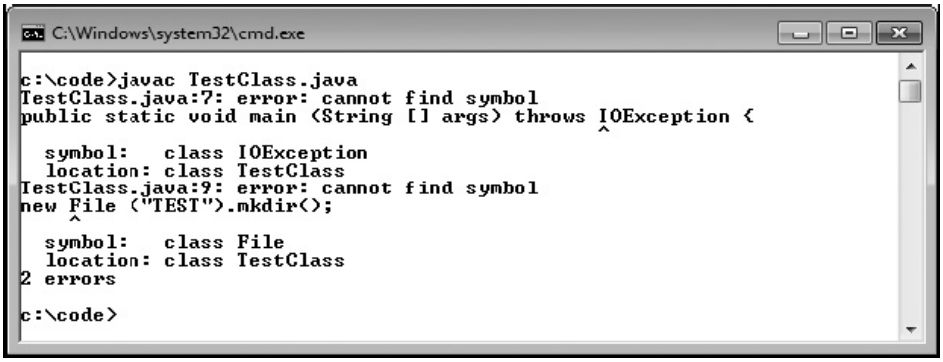


图 1-2　只有 java.io 包被注释后所显示的错误

(5) 对于每一个找不到的类，使用在线的 Java 规范 API 来确定它属于哪个包，然后用必要的显式 import 语句来更新源文件。一旦完成后，将用 9 条显式的 import 语句替换 4 条隐式的 import 语句。

(6) 再次运行应用程序，以确保这个应用程序使用显式的 import 语句与使用隐式的 import 语句时的行为一样。

认证目标

1.2　理解包派生类

Oracle 在 Java SE 7 的 API 中包括了 209 个包。每个包都有一个特定的关注点。幸运的是，对于 OCA 考试，你只需要熟悉其中的几个包。这些包可能包括 Java 的实用工具包、基本的输入/输出包、网络包、抽象窗口工具(Abstract Window Toolkit，AWT)包以及 Swing 包。

以下各小节讲解这些 API：

- Java 实用工具 API
- Java 基本输入/输出 API
- Java 网络 API
- Java 抽象窗口工具包 API
- Java Swing API

1.2.1　Java 实用工具 API

Java 实用工具 API 包含在 java.util 包中。这个 API 提供了各种实用工具类的功能。该 API 的主要类和接口可以分为几类。各个类别的类在考试中可能都会看到，包括 Java 的集合框架、日期和时间工具、国际化以及一些其他的实用工具类。

这些类别中，Java 集合框架所占的比重最大，因为它经常被用到，并且为构建有价值的 Java 应用程序提供了必要的基础数据结构。表 1-4 详细讲述了考试中可能涉及的集合类和接口的 API。

表 1-4　Java 集合框架的各种类

接　口	实　现	描　述
List	ArrayList、LinkedList、Vector	基于位置访问的数据结构
Map	HashMap、Hashtable、LinkedHashMap、TreeMap	映射键值对的数据结构
Set	HashSet、LinkedhashSet、TreeSet	基于元素唯一性的数据结构
Queue	PriorityQueue	队列通常按照先入先出(FIFO)的方式排序元素。优先级队列根据提供的比较器排列元素

集合 API 提供了 Comparator 接口，帮助集合不按照自然顺序进行排序。类似地，java.lang 包中的 Comparable 接口则用于按照对象的自然顺序进行排序。

java.util 包中还具有其他各种类和接口。日期和时间工具通过 Date、Calendar 和 TimeZone 类表示。地理区域使用 Locale 类表示。Currency 类表示符合 ISO 4217 标准的货币。Random 类提供了随机数生成器。而 StringTokenizer 将字符串分隔成组。还有其他几个类存在于 java.util 包中，但是这些类(集合接口和类)都是在工作中经常使用的。这些实用工具类如图 1-3 所示。

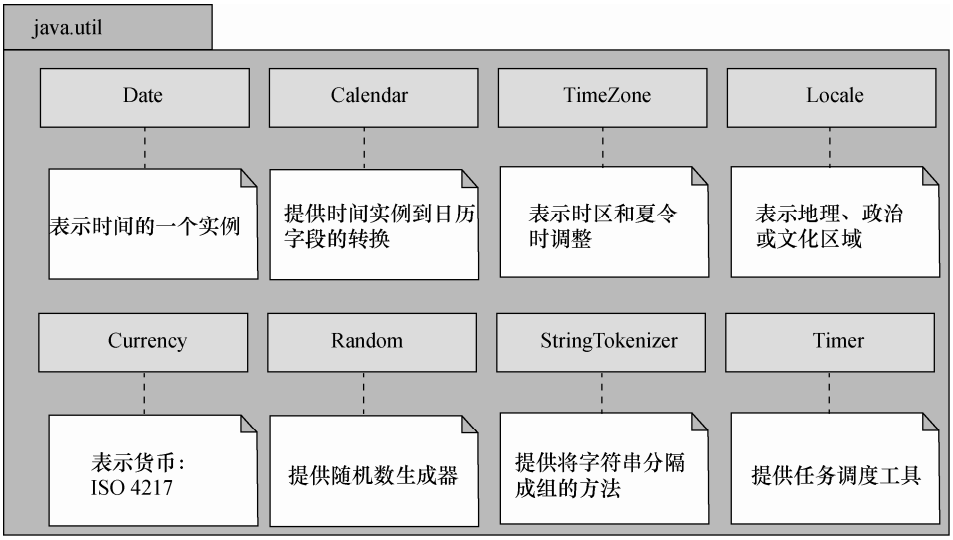


图 1-3　各个工具类

实际经验：

许多包中具有特有功能的相关类和接口，所以它们被包含在其子包中。例如，正则表达式被存储在 Java 实用工具包(java.util)的子包中。该子包名为 java.util.regex，并拥有 Matcher 和 Pattern 类。在必要的情况下，可以考虑为自己的项目创建子包。

1.2.2　Java 基本输入/输出 API

Java 基本输入/输出 API 包含在 java.io 包中。这个 API 为常见的系统输入和输出提供了对应的数据流、序列化和文件系统功能。数据流类包括了字节流 InputStream 和 OutputStream 类的子类。数据流类还包括字符流 Reader 和 Writer 类的子类。图 1-4 描绘了 Reader 和 Writer 抽象类的部分类层次结构。

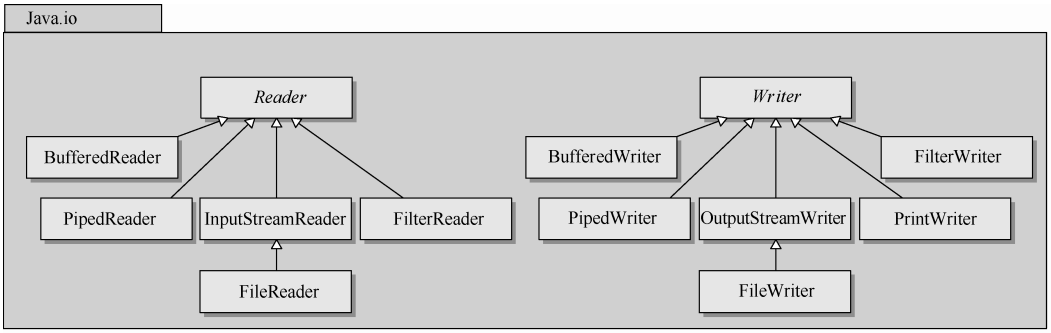


图 1-4　Reader 和 Writer 类的层次结构

其他重要的 java.io 类和接口包括 File、FileDescriptor、FilenameFilter 和 RandomAccessFile。File 类提供了文件和目录路径名的表示。FileDescriptor 类对打开的文件和套接字提供了一种句柄功能。FilenameFilter 接口，顾名思义，定义了过滤文件名的功能。RandomAccessFile 类允许读写指定位置的文件。

1.2.3　Java 网络 API

Java 网络 API 包含在 java.net 包中。这个 API 提供了支持创建网络应用程序的功能。此 API 的主要类和接口如图 1-5 所示。考试中可能几乎看不到这些类，但这幅图将帮助你概念化 java.net 包中的内容。提升性能的新 I/O(java.nio)包提供了不在考试范围中的无阻塞网络以及套接字工厂支持包(javax.net)。

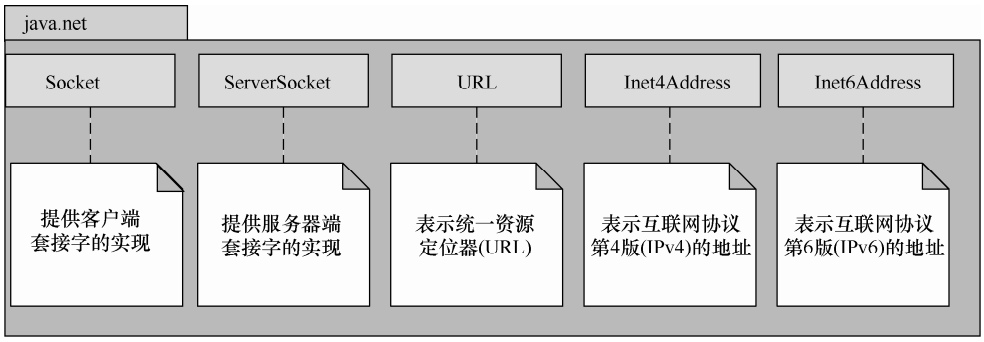


图 1-5　各种网络 API 的类

1.2.4　Java 抽象窗口工具包 API

Java 抽象窗口工具包 API 包含在 java.awt 包中。这个 API 提供了创建重量级组件的功能，用于创建用户界面和绘制相关的图形和图像。AWT API 是 Java 最初的 GUI API，并且已经被 Swing API 取代。现在推荐使用 Swing，某些 AWT API 仍然常用，例如，AWT 的焦点子系统在 J2SE 1.4 中重

写了。AWT 焦点子系统提供了组件之间的导向控制。图 1-6 描述了这些主要的 AWT 元素。

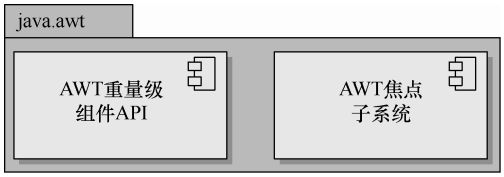


图 1-6　AWT 主要元素

1.2.5　Java Swing API

Java Swing API 包含在 javax.swing 包中。此 API 提供了创建轻量级(纯 Java)容器和组件的功能。Swing API 提供了一套更精密的 GUI 组件，取代了 AWT API。许多 Swing 类只是比遗留的对应等价的 AWT 组件增加了一个额外的“J”。例如，Swing 使用类 JButton 来表示按钮容器，而 AWT 使用类 Button。

场景与解决方案	
你需要创建基本的 Java Swing 组件，如按钮、面板和对话框。提供代码导入必需的类包	// Java Swing API 包 import javax.swing.*;
你需要支持文本相关方面的 Swing 组件。提供代码导入必需的类包	// Java Swing API 的 text 子包 import javax.swing.text.*;
你需要实现和配置基本的可插拔的外观支持。提供代码导入必需的类包	// Java Swing API 的 plaf 子包 import javax.swing.plaf.*;
你需要使用 Swing 的事件监听器和适配器。提供代码导入必需的类包	// Java Swing API 的 event 子包 import javax.swing.event.*;

Swing 还提供外观界面上的支持，允许改变 GUI 组件的统一风格。其他功能包括提示、辅助功能、事件模型和增强组件(例如表、树、文本组件、滑块和进度条)。Swing API 的一些主要的类如图 1-7 所示。

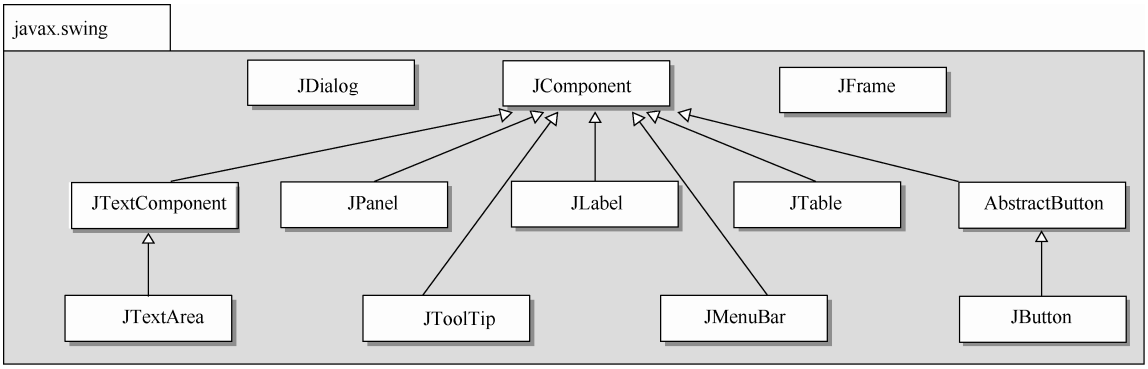


图 1-7　Swing API 的各个类

Swing API 很好地利用了子包，在 Java SE 7 中总共有 18 个子包。如前所述，当通用的类被分

隔到各自的包中时，代码的易用性和可维护性就得到了增强。

Swing采用了模型-视图-控制器架构(MVC)。模型表示每个组件的当前状态。视图是组件在屏幕上的表示。控制器的功能是维系 UI 组件到事件。虽然理解 Swing 的底层架构非常重要，但是对于考试却没有必要。关于 Swing API 的全面信息，请参看由 Herbert Schildt 所著的书籍 *Swing：A Beginner's Guide*(McGraw-Hill Professional，2007)。

实际经验：

熟悉以 java 和 javax 为前缀的包有好处。java 前缀通常用于核心包。javax 前缀通常用于以包含 java 标准扩展的包。特别注意在 AWT 和 Swing API 中使用的前缀：java.awt 和 javax.swing。还要注意的是，JavaFX 将取代 Swing 作为 Java SE 的 GUI 工具包。它的前缀是 javafx。

练习 1-2

理解 Java 实用工具 API 的扩展功能

在Java SE 6和7中，总共有10个包与Java 实用工具 API有直接关系，其基本包命名为java.util。J2SE 5.0 只有 9 个包；J2SE 1.4 只有 6 个包。此练习将展示 Java 实用工具 API 子包的详细信息，它们是在 Java SE 1.4 平台的后续版本中添加进来的。

(1) 进入在线的 J2SE 1.4.2 API 规范：http://docs.oracle.com/javase/1.4.2/docs/api/，如图 1-8 所示。

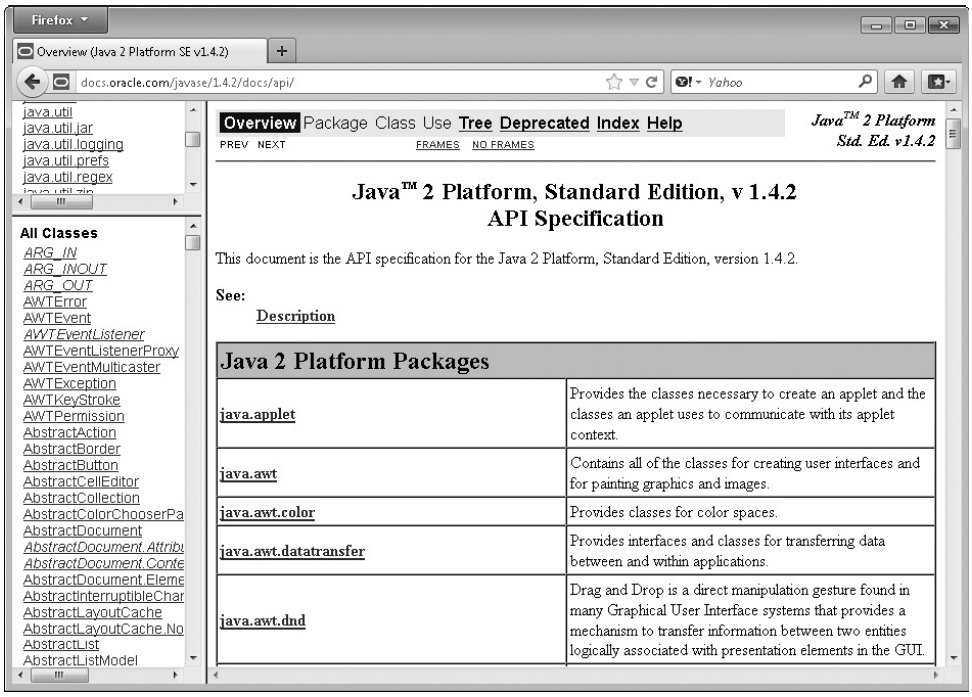


图 1-8　在线的 J2SE 1.4.2 API 规范

(2) 使用网页浏览器的滚动条，向下滚动到 Java 实用工具 API 包。

(3) 单击每个相关包的链接。探索每个包中类和接口的详细信息。

(4) 进入在线的 J2SE 5.0 API 规范：http://docs.oracle.com/javase/1.5.0/docs/api/，如图 1-9 所示。

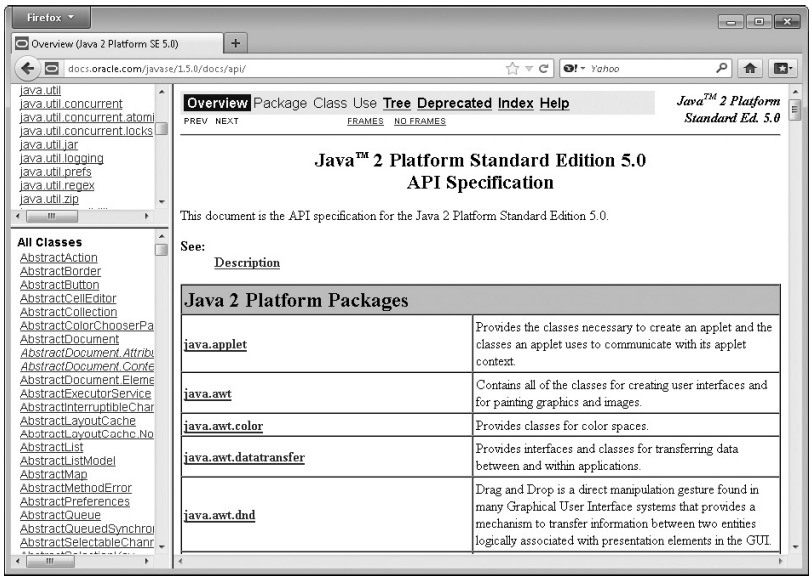


图 1-9　在线的 J2SE 5.0 API 规范

(5) 使用网页浏览器的滚动条，向下滚动到 Java 实用工具 API 包。

(6) 分辨是哪三个新的子包添加进了 Java 实用工具 API 中。单击这些新包的每一个链接。探索每个包中类和接口的详细信息。

(7) 进入在线的 Java SE 7 API 规范：http://docs.oracle.com/javase/7/docs/api/。这是考试应该参照的 API 规范，如图 1-10 所示。

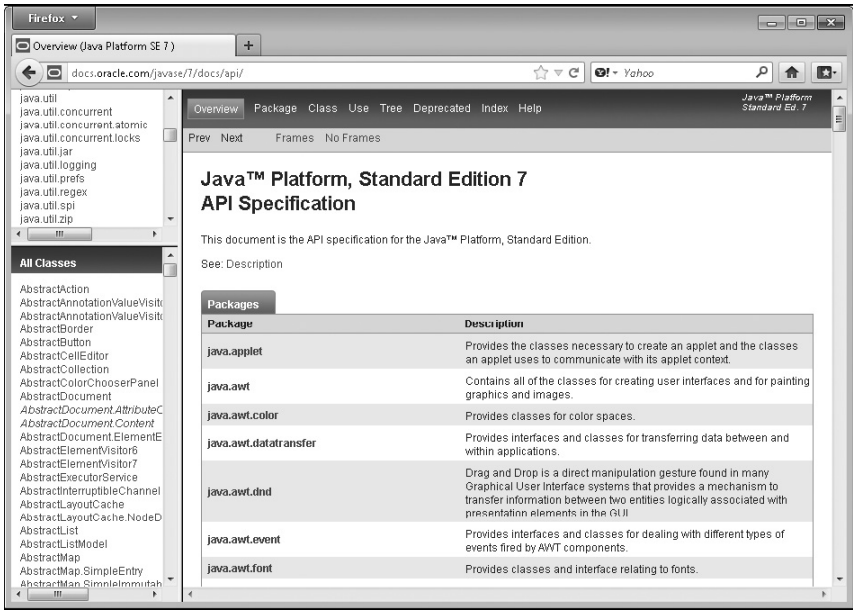


图 1-10　在线的 Java SE 7 API 规范

(8) 使用网页浏览器的滚动条，向下滚动到 Java 实用工具 API 包。

(9) 分辨自 J2SE 5.0 以来哪个新的子包添加进 Java 实用工具 API 中。单击这些新包的链接，探索包中类的详细信息。

认证目标

1.3　理解类结构

考试目标 1.2　定义 Java 类的结构

不管是为了应付考试，还是拥有一条光明的 Java 职业生涯，都必须理解 Java 类的结构。这将有助于对 Java 命名规范以及在 Java 源代码中看到的典型分隔符(例如，用于封闭实体的括号以及注释分隔符)的知识有个基本认识。下面几小节将涵盖这些内容：

- 命名规范
- 分隔符和其他 Java 源符号
- Java 类结构

1.3.1　命名规范

命名规范就是在创建标识符、方法、类名以及整个代码库时，字符用法/应用的规则。如果团队中的一些成员没有在代码中应用命名规范，为了代码部署后良好的可维护性，你应该鼓励他们这样做。

实际经验：

由 Roedy Green 所著的一篇流行文章“如何编写不可维护的代码”，值得阅读(http://thc.org/root/phun/unmaintain.html)。它以一种滑稽的方式揭示了维护公然或故意无视软件最佳开发实践的代码可能遇到的挑战。另一方面，由 Chad Fowler 所著的“充满激情的程序员，在软件开发中创造非凡的职业生涯”，鼓励软件开发者尽自己所能做得最好。

你可能会遇到使用自己的命名规范的人。尽管这比不应用命名规范强，但局外人试图去维护这个人的代码时，就需要学习最初的规范并应用它以保持一致性。幸运的是，Java 社区就 Java 中命名规范应该如何应用到许多不同的控件上达成了共识。表 1-5 简单描述了这些规范。当应用命名规范时，你应当努力使用有意义和没有歧义的名称。并记住命名规范的首要目标是让 Java 程序更可读，并因此可维护。Java 的命名规范在实践中使用驼峰命名法。驼峰命名法的做法是复合词中的第一个字符使用大写字母。

表 1-5　Java 命名规范

元　素	字　母	特　征	示　例
类名	大写开头，继续驼峰命名法	名词	SpaceShip
接口名	大写开头，继续驼峰命名法	当提供一种能力时，是以“able”或“ible”结尾的形容词，否则是名词	Dockable

(续表)

元　素	字　母	特　征	示　例
方法名	小写开头，继续驼峰命名法	动词，可以包括形容词或名词	corbit
实例和静态变量名	小写开头，继续骆峰命名法	名词	moon
参数和局部变量	小写开头，如果需要多个单词，继续驼峰命名法	单个单词、缩写或缩略语	lop(例如，线条的位置)
泛型参数	单个大写字母	推荐字母 T	T
常量	所有字母大写	以下划线分隔的多个单词	LEAGUE
枚举	大写开头，继续驼峰命名法；对象集合应全部大写	名词	enum Occupation {MANNED, SEMI_MANNED, UNMANNED}
包	所有字母小写	公共包应该是组织的域名反转	com.ocajexam.sim

1.3.2 分隔符和其他 Java 源符号

Java 编程语言使用了几个分隔符和符号，帮助在软件程序中创建源代码的结构。表 1-6 详细介绍这些分隔符号和符号。

表 1-6 符号和分隔符

符　号	描　述	目　的
( )	括号	方法参数的闭集，封闭的类型转换，在算术表达式中调整优先级
{ }	大括号	包围代码块和初始化数组
[ ]	方括号	声明数组类型和初始化数组
< >	尖括号	封闭泛型
;	分号	行尾的结束语句
,	逗号	变量声明中的分隔标识符，分隔值，在 for 循环中分隔表达式
.	点	分隔包名，选择对象的字段或方法，支持方法链
:	冒号	紧接循环标签
' '	单引号	定义字符
" "	双引号	定义字符串
//	正斜杠	表示单行注释
/* */	带星号的正斜杠	表示多行注释
/** */	带两个星号和一个星号的正斜杠	表示 Javadoc 注释

1.3.3 Java 类结构

每个 Java 程序至少有一个类。Java 类包含签名、可选的构造函数、可选的数据成员(字段)和可

选的方法，概貌如下：

```
[modifiers] class classIdentifier [extends superClassIdentifier]
[implements interfaceIdentifier1, interfaceIdentifier2, etc.] {

      [data members]

      [constructors]

      [methods]

}
```

每个类可以扩展一个并且只能一个超类。每个类可以实现一个或多个接口。接口之间用逗号分隔开。

下面的 SpaceShip 类展示了标有注释的典型元素。包含这个 SpaceShip 类的文件必须命名为 SpaceShip.java。

```
package com.ocajexam.craft_simulator;

public class SpaceShip extends Ship implements Dockable {

  // 数据成员
  public enum ShipType {
    FRIGATE, BATTLESHIP, MINELAYER, ESCORT, DEFENSE
  }
  ShipType shipType = ShipType.BATTLESHIP;

  // 构造函数
  public SpaceShip() {
    System.out.println("\nSpaceShip created with default shiptype.");
  }
  public SpaceShip(ShipType shipType) {
    System.out.println("\nSpaceShip created with specified shiptype.");
    this.shipType = shipType;
  }

  // 方法
  @Override
  public void dockShip () {
    // TODO
  }
  @Override
  public String toString() {
    String shipTypeRefined = this.shipType.name().toLowerCase();
    return "The pirate ship is a " + shipTypeRefined + " ship.";
  }
}
```

如下面的代码所示，SpaceShip 类可以被实例化：

```
package com.ocajexam.craft_simulator;
import com.ocajexam.craft_simulator.PirateShip.ShipType;
public class SpaceShipSimulator {
```

```
  public static void main(String[] args) {

    // 用默认的 ship 类型创建 SpaceShip 对象
    SpaceShip ship1 = new SpaceShip ();
    // 打印 "The pirate ship is a battleship."
    System.out.println(ship1);

    // 用指定的 ship 类型创建 SpaceShip 对象
    SpaceShip ship2 = new SpaceShip (ShipType.FRIGATE);
    // 打印 "The pirate ship is a frigate ship."
    System.out.println(ship2);
    }
}
```

实际经验：

重写注释(即@Override)表明方法声明打算重写超类中的方法声明。

第 4 章～第 7 章将更全面地讲解关于创建和应用类的细节。

认证目标

1.4 编译和解释 Java 代码

考试目标 1.3 使用 main 方法创建可执行的 Java 应用程序

Java 开发工具包(JDK)包括一些编译、调试和运行 Java 应用程序的工具。本节详细介绍其中的两个实用工具：Java 编译器和 Java 解释器。想了解 JDK 的更多信息和其他实用工具，请参阅第 10 章。

1.4.1 Java 编译器

我们需要一个示例应用程序来使用 Java 编译器和解释器。本节将采用简单的 GreetingsUniverse.java 源文件，如下所示：

```
public class GreetingsUniverse {
  public static void main(String[] args) {
    System.out.println("Greetings, Universe!");
  }
}
```

让我们来看看如何使用最基本的命令行选项来编译和解释这个简单的 Java 程序。

1. 编译源代码

Java 编译器是 JDK 中的几个工具之一。当有时间时，查看 JDK bin 文件夹中包含的其他工具，如图 1-11 所示。对于 OCA 的考试范围，只需要知道关于编译器和解释器的细节。

Java 编译器简单地将 Java 源文件转换成字节码。Java 编译器的用法如下：

```
javac [options] [source files]
```

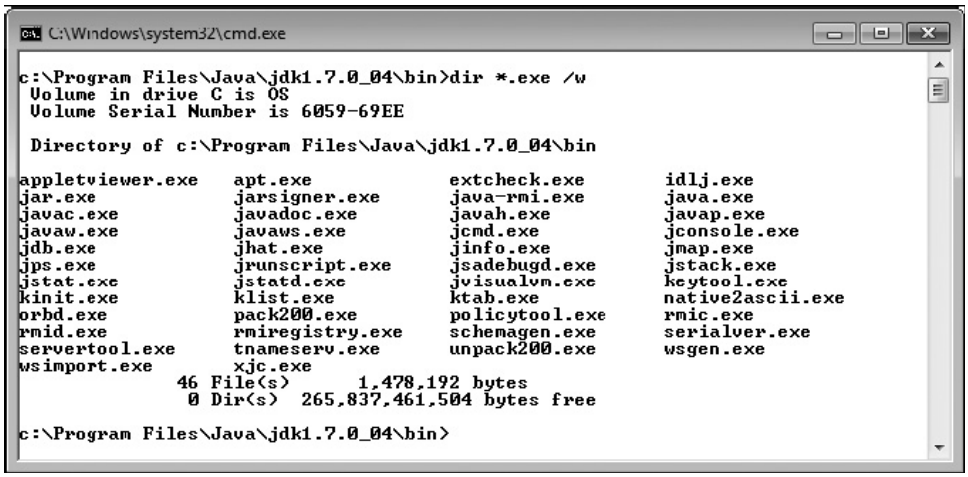


图 1-11　Java 开发工具包

编译 Java 类最直接的方法就是在命令行中，在 Java 源文件之前加上编译器工具：javac.exe FileName.java。.exe 是 Windows 机器上标准的可执行文件的扩展名，它是可选的。.exe 扩展名在类 UNIX 的系统中不显示。

```
javac GreetingsUniverse.java
```

这将导致所产生的字节码文件具有相同的前缀，如 GreetingsUniverse.class。此字节码文件将被放置到与源代码相同的文件夹，除非代码是分包的，并且/或者通过命令行选项告知它将被放置到其他地方。

实际经验：

你会发现，很多项目都使用 Apache Ant 和/或 Maven 来构建环境。了解命令行工具的基础知识，是编写/维护这些与构建产品相关的脚本的基础。

考试内幕：

命令行工具

大多数项目都使用集成开发环境(IDE)来编译和执行代码。使用 IDE 的明显好处是可以轻松地通过少数几个菜单选项，甚至只需要单击一个热键，就能构建和运行代码。它的缺点是，虽然可以通过配置对话框来创建设置，并在工作区窗口看到命令和后续参数，但是你没有获得通过手工反复创建命令和相关参数的完整结构的直接经验。考试的结构是验证你有调用编译器和解释器脚本的经验。不要轻易采用这个前提。只有在你已经掌握了何时以及如何使用工具、开关，以及相关参数之后再参加考试。在稍后的时间，可以考虑利用流行的 IDE，如 NetBeans、Eclipse、IntelliJ IDEA 和 JDeveloper 提供的“快捷”功能。

2. 使用-d 选项编译源代码

你可能希望显式地指定将编译后的字节码类文件放到哪里，可以使用-d 选项实现：

```
javac -d classes GreetingsUniverse.java
```

这个命令行结构将放置类文件到类目录，并且因为源代码是分包的(即源文件中包含 package 语句)，字节码将被放置到相应的子目录。

```
[当前的工作目录]\classes\com\ocajexam\tutorial\GreetingsUniverse.class
```

3. 使用-classpath 选项编译代码

如果你希望使用用户定义的类和包来编译应用程序，可能需要通过 classpath 来指定它们，告诉 JVM 到哪里去寻找。这个 classpath 的实现方式是用-cp 或-classpath 命令行来告诉编译器期望的类和包在哪里。在下面的编译器调用中，编译器在它编译时包括了位于 3rdPartyCode\classes 目录下的所有源文件，以及位于当前工作目录(.)下的所有类。-d 选项(再次)将放置编译后的字节码到类目录。

```
javac -d classes -cp 3rdPartyCode\classes\;. GreetingsUniverse.java
```

请注意，如果使用 CLASSPATH 环境变量设置了类路径，或者期望的文件在当前的工作目录下，则不需要包含 classpath 选项。

在 Windows 系统中，类路径中的目录是用反斜杠分隔，而路径则用分号分隔：

```
-classpath .;\dir_a\classes_a\;\dir_b\classes_b\
```

在基于 POSIX 的系统中，类路径中的目录用正斜杠分隔，而路径用冒号分隔：

```
-classpath .:/dir_a/classes_a/:/dir_b/classes_b/
```

点(.)仍然表示当前(或者目前)的工作目录。

考试提示：

了解开关。考试的设计者将试图通过混合匹配编译器和解释器开关的答案将你淘汰，你甚至可能看到一些在任何地方都不可能存在的选项。出于额外的准备，通过输入 java -help 或 javac -help 来查询命令开关的完整集合。开关通常也称为命令行参数、命令行开关、选项和标志。

1.4.2 Java 解释器

解释 Java 文件是创建 Java 应用程序的基础，如图 1-12 所示。让我们来看看如何调用解释器和它的命令行选项：

```
Java  [-options] class [args…]
```

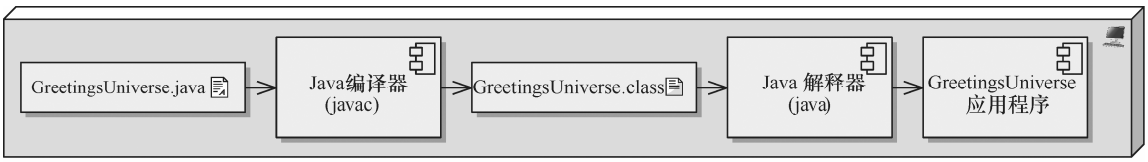


图 1-12　字节码转换

1. 解释字节码

使用 java[.exe]命令调用 Java 解释器。它用来解释字节码和执行程序。
可以很容易地在一个没有分包的类上调用解释器，如下所示：

```
java MainClass
```

在 Windows 系统中，可以选择使用 javaw 命令启动应用程序而不要命令窗口。对于基于 GUI 的应用程序，这是一个不错的功能，因为控制台窗口通常是没有必要的。

```
javaw.exe MainClass
```

同样，在基于 POSIX 的系统中，则可以使用&符号来运行应用程序为后台进程。

```
java MainClass &
```

2. 用-classpath 选项解释代码

当解释代码时，你可能需要定义其中某些类和包的位置。当包含-cp 或-classpath 选项到解释器时，可以在运行时找到类。如果你想要包含的类被分包了，则可以通过指定应用程序的全路径到类的基目录来启动应用程序，如下边的解释器调用：

```
java -cp classes com.ocajexam.tutorial.MainClass
```

-cp 和-classpath 选项的分隔语法相同。

3. 使用-D 选项解释字节码

-D 命令行选项允许设置新的属性值，用法如下：

```
java -D<name>=<value> class
```

下列包含 PropertiesManager 类的单文件应用程序打印出所有的系统属性：

```
import java.util.Properties;
public class PropertiesManager {
  public static void main(String[] args) {
    if(args.length == 0) {System.exit(0);}
    Properties props = System.getProperties();
      /* 新属性示例 */

      props.setProperty("new_property2", "new_value2");
      switch(args[0]) {
        case "-list_all":
        props.list(System.out); // 列举所有的属性
        break;
        case "-list_prop":
         /* 列举值 */
         System.out.println(props.getProperty(args[1]));
         break;
        default:
         System.out.println("Usage: java PropertiesManager[-list_all]");
         System.out.println("       java PropertiesManager[-list_prop [property]]");
```

```
            break;
        }
      }
    }
```

让我们运行这个应用程序，同时设定新的名为 new_property1 的系统属性，值为 new_value1：

```
java -Dnew_property1=new_value1 PropertiesManager -list_all
```

你将在标准输出中看到系统属性的列表，包括我们设定的新属性及其值：

```
...
new_property1=new_value1
java.specification.name=Java Platform API Specification
...
```

或者，可以通过实例化 Properties 类来设定值，然后使用 setProperty 方法来设置属性及其值。为了帮助你更好地概念化系统属性，表 1-7 详细介绍了标准系统属性的子集。

表 1-7　系统属性的子集

系 统 属 性	属 性 描 述
file.separator	平台相关的文件分隔符(POSIX 是/，Windows 是\)
java.class.path	类路径定义为系统环境变量
java.class.version	Java 类的版本号
java.home	Java 安装目录
java.vendor	Java 平台的供应商
java.vendor.url	供应商的统一资源定位器
java.version	Java 解释器/JVM 的版本
line.separator	平台相关的行分隔符(Mac OS 9 是\r，POSIX 是\n，Microsoft Windows 是\r\n)
os.arch	操作系统的架构
os.name	操作系统的名称
os.version	操作系统的版本号
path.separator	平台相关的路径分隔符(POSIX 是：，Windows 是；)
user.dir	用户的当前工作目录
user.home	用户的主目录
user.language	默认语言环境的语言代码
user.name	当前用户的用户名
user.timezone	系统默认时区

4. 使用-version 选项获取解释器的版本

Java 解释器使用-version 命令行选项来返回 JVM 的版本号并退出。不要想当然地使用简化的命令，因为考试的设计者可能会通过在命令之后包含附加的参数来试图让你上当。花点时间来熟悉包

含附加参数的命令，以及放置-version 选项到命令的不同位置。不要对应用程序将会如何响应做任何假设。图 1-13 展示了在不同位置使用-version 选项时的不同结果。

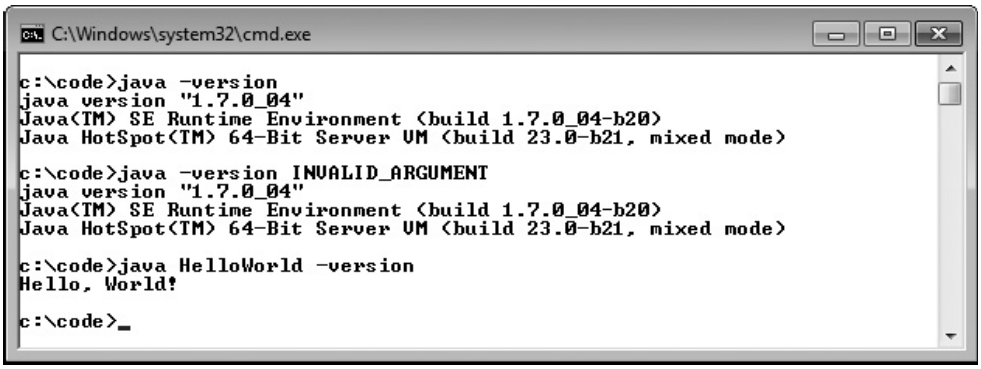


图 1-13　–version 命令行选项

实际经验：

检查可供使用的其他 JDK 工具，可以在 JDK 的 bin 目录中找到它们。JConsole 是一个特别有用的基于 GUI 的工具，可以用来检测和管理 Java 应用程序。在众多功能中，JConsole 允许查看内存和线程的使用情况。JConsole 是在 J2SE 5.0 中发布的。

练习 1-3

编译和解释分包的软件

当从 IDE 中编译和运行分包的软件时，执行过程只需要轻松地单击运行图标，因为 IDE 将会为你维护类路径，并且如果出现了任何问题，将会通知你。当试图使用命令行亲自编译和执行代码时，需要确切地知道文件的路径。考虑放置在 com.ocajexam.tutorial 包(即 com/ocajexam/tutorial 目录)中的示例应用程序。

```
package com.ocajexam.tutorial;
public class GreetingsUniverse {
  public static void main(String[] args) {
    System.out.println("Greetings, Universe!");
  }
}
```

这个练习将让你在不同包中创建新类来编译和运行应用程序。

(1) 编译程序：

```
javac -d . GreetingsUniverse.java
```

(2) 运行程序，以确保它没有错误：

```
java  -cp  .  com.ocajexam.tutorial.GreetingsUniverse
```

(3) 创建三个名为 Earth、Mars 和 Venus 的类，并将它们放置到 com.ocajexam.tutorial.planets 包

中。创建构造函数，将行星的名字打印到标准输出。这里给出 Earth 类的细节，作为你将要完成的一个示例：

```
package com.ocajexam.tutorial.planets;
public class Earth {
  public Earth {
    System.out.println("Hello from Earth!");
  }
}
```

(4) 通过将必要的代码添加到 GreetingsUniverse 类，从主程序中实例化每一个类。

```
Earth e = new Earth();
```

(5) 确保所有的源代码都在路径 src/com/ocajexam/tutorial/和 src/com/ocajexam/tutorial/planets/中。
(6) 确定需要编译完整程序的命令行参数。编译程序，并在必要时进行调试。
(7) 确定解释程序需要的命令行参数。运行程序。

标准输出的读取如下：

```
$ Greetings, Universe!
Hello from Earth!
Hello from Mars!
Hello from Venus!
```

1.5 认证小结

本章讨论了分包、结构、编译和解释 Java 代码。本章首先讨论了将类组织到包中的重要性，以及使用 package 和 import 语句来定义和包括不同的代码片段。在本章的中间，我们讨论了常用的 Java 包：java.awt、javax.swing、java.net、java.io 和 java.util 的几个主要功能。讨论了 Java 类的基本结构。然后本章以提供如何编译和解释 Java 源和类文件，以及如何应用它们的命令行选项的详细信息作为结束。现在，你应该可以独立地(在 IDE 之外)分包、构建和运行基本的 Java 程序了。

1.6 知识点回顾

1.6.1 理解包

- 包是类的容器。
- package 语句定义文件存储的目录路径。
- package 语句使用点(.)来分隔。
- 包名应该小写，并且单词之间使用下划线分隔。
- 以 java.*和 javax.*开头的包名被保留。
- 每一个源文件中没有或只有一条 package 语句。
- import 语句用来从外部类中包含源代码。
- import 语句出现在可选的 package 语句之后，类声明之前。

- import 语句可以定义导入指定的类名。
- import 语句可以使用星号来包含给定包中的所有类。

1.6.2　理解包派生类

- Java 的抽象窗口工具包(Abstract Window Toolkit)的 API 包含在 java.awt 包和子包中。
- java.awt 包包含了 GUI 创建和绘制图形图像的功能。
- Java Swing API 包含在 javax.swing 包和子包中。
- javax.swing 包包含的类和接口支持轻量级 GUI 组件功能。
- Java 基本输入/输出相关的类包含在 java.io 包中。
- java.io 包包括的类和接口支持文件系统、数据流和序列化的输入/输出功能。
- Java 网络类包含在 java.net 包中。
- java.net 包包含的类和接口支持基本的网络功能，并被 javax.net 包扩展。
- 基础的 Java 实用工具包含在 java.util 包中。
- java.util 包和子包包含的类和接口支持 Java 集合框架、遗留的集合类、事件模型、日期和时间工具以及国际化功能。

1.6.3　理解类结构

- 命名规范用来使 Java 程序更具有可读性和可维护性。
- 命名规范被应用到一些 Java 元素上，包括类名、接口名、方法名、实例和静态变量名、参数和局部变量名、泛型参数名、常量名、枚举名和包名。
- 类中出现的元素的首选顺序是数据成员，接着是构造函数，紧接着是方法。需要注意的是，包括的每一类元素都是可选的。

1.6.4　编译和解释 Java 代码

- Java 编译器使用 javac[.exe]命令调用。
- .exe 扩展名在 Microsoft Windows 机器上是可选的，在类 UNIX 系统上是不显示的。
- 编译器的-d 命令行选项定义了编译后的类文件应该被放置的位置。
- 如果类使用 package 语句声明，编译器的-d 命令行选项将包括包的位置。
- 编译器的-classpath 命令行选项定义了查找类的目录路径。
- Java 解释器通过 java[.exe]命令来调用。
- 解释器的-classpath 开关定义了运行时使用的目录路径。
- 解释器的-D 命令行选项允许设置系统属性值。
- 解释器的-D 命令行选项的语法是-Dproperty=value。
- 解释器的-version 命令行选项用来返回 JVM 的版本并退出。
- -h 命令行选项既可以应用到编译器，也可以在解释器上打印出工具的用法信息。

1.7 自测题

1.7.1 理解包

1. Which two `import` statements will allow for the import of the `HashMap` class?
   A. `import java.util.HashMap;`
   B. `import java.util.*;`
   C. `import java.util.HashMap.*;`
   D. `import java.util.hashMap;`
2. Which statement would designate that your file belongs in the package `com.ocajexam.utilities`?
   A. `pack com.ocajexam.utilities;`
   B. `Package com.ocajexam.utilities.*`
   C. `package com.ocajexam.utilities.*;`
   D. `package com.ocajexam.utilities;`
3. Which of the following is the only Java package that is imported by default?
   A. `java.awt`
   B. `java.lang`
   C. `java.util`
   D. `java.io`
4. What Java-related features are new to J2SE 5.0?
   A. Static imports
   B. `package` and `import` statements
   C. Autoboxing and unboxing
   D. The enhanced `for` loop

1.7.2 理解包派生类

5. The `JCheckBox` and `JComboBox` classes belong to which package?
   A. `java.awt`
   B. `javax.awt`
   C. `java.swing`
   D. `javax.swing`
6. Which package contains the Java Collections Framework?
   A. `java.io`
   B. `java.net`
   C. `java.util`
   D. `java.utils`
7. The Java Basic I/O API contains what types of classes and interfaces?
   A. Internationalization
   B. RMI, JDBC, and JNDI
   C. Data streams, serialization, and file system

D. Collection API and data streams

8. Which API provides a lightweight solution for GUI components?

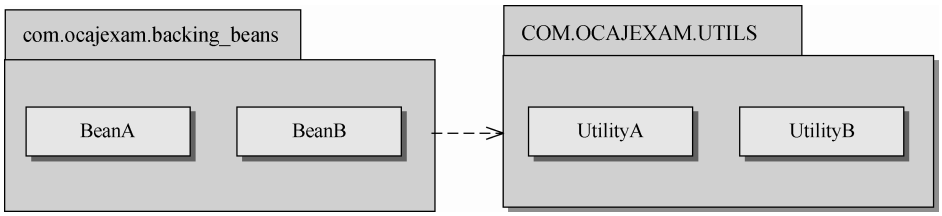
A. AWT

B. Abstract Window Toolkit

C. Swing

D. AWT and Swing

9. Consider the following illustration. What problem exists with the packaging? You may wish to reference Appendix G on the Unified Modeling Language (UML) for assistance.



A. You can have only one class per package.

B. Packages cannot have associations between them.

C. Package `com.ocajexam.backing_beans` fails to meet the appropriate packaging naming conventions.

D. Package `COM.OCAJEXAM.UTILS` fails to meet the appropriate packaging naming conventions.

1.7.3　理解类结构

10. When apply naming conventions, which Java elements should start with a capital letter and continue on using the camel case convention?

A. Class names

B. Interface names

C. Constant names

D. Package names

E. All of the above

11. When instantiating an object with generics, should angle brackets, box brackets, parentheses, or double-quotes be used to enclose the generic type? Select the appropriate answer.

A. `List<Integer> a = new ArrayList<Integer>();`

B. `List[Integer] a = new ArrayList[Integer]();`

C. `List{Integer} a = new ArrayList{Integer}();`

D. `List"Integer" a = new ArrayList"Integer"();`

12. When organizing the elements in a class, which order is preferred?

A. Data members, methods, constructors

B. Data members, constructors, methods

C. Constructors, methods, data members

D. Constructors, data members, methods

E. Methods, constructors, data members

1.7.4 编译和解释 Java 代码

13. Which usage represents a valid way of compiling a Java class?
    A. `java MainClass.class`
    B. `javac MainClass`
    C. `javac MainClass.source`
    D. `javac MainClass.java`
14. Which two command-line invocations of the Java interpreter return the version of the interpreter?
    A. `java -version`
    B. `java --version`
    C. `java -version ProgramName`
    D. `java ProgramName -version`
15. Which two command-line usages appropriately identify the classpath?
    A. `javac –cp /project/classes/ MainClass.java`
    B. `javac –sp /project/classes/ MainClass.java`
    C. `javac -classpath /project/classes/ MainClass.java`
    D. `javac -classpaths /project/classes/ MainClass.java`
16. Which command-line usages appropriately set a system property value?
    A. `java -Dcom.ocajexam.propertyValue=003 MainClass`
    B. `java -d com.ocajexam.propertyValue=003 MainClass`
    C. `java -prop com.ocajexam.propertyValue=003 MainClass`
    D. `java –D:com.ocajexam.propertyValue=003 MainClass`

1.8 自测题答案

1.8.1 理解包

1. 答案为 A 和 B。HashMap 类可以直接通过 import java.util.HashMap 或者使用通配符通过 import java.util.*;而导入。C 和 D 不正确。C 不正确，因为这个答案是一条静态的 import 语句，它导入 HashMap 类的静态成员，而不是类本身。D 不正确，因为类名是大小写敏感的，所以类名 hashMap 不等于 HashMap。

2. 答案为 D。使用关键字 package 是正确的，紧跟着是以点分隔的包名，后紧跟一个分号。A、B 和 C 不正确。A 不正确，因为单词 pack 不是有效的关键字。B 不正确，因为 package 语句必须以分号结尾，并且在 package 语句中不能使用星号。C 不正确，因为不能在 package 语句中使用星号。

3. 答案为 B。java.lang 包是所有类都默认导入的唯一包。A、C 和 D 不正确。java.awt、java.util 和 java.io 包中的类都不是默认导入的。

4. 答案是 A、C 和 D。静态导入、自动装箱/拆箱和增强的 for 循环都是 J2SE 5.0 的新功能。B 不正确，因为基本的 package 和 import 语句对于 J2SE 5.0 不是新的。

1.8.2　理解包派生类

5. 答案为 D。属于 Swing API 的组件通常以大写 J 为前缀。因此，JCheckBox 和 JComboBox 应该是 Java Swing API。的一部分，而不是 Java AWT API。Java Swing API 的基本包是 javax.swing。A、B 和 C 不正确。A 不正确，原因是包 java.awt 不包括 JCheckBox 和 JComboBox 类，因为它们属于 Java Swing API。请注意，包 java.awt 包括 CheckBox 类，而不包括 JCheckBox 类。B 和 C 不正确，因为包名 javax.awt 和 java.swing 不存在。

6. 答案为 C。Java 集合框架是 java.util 包中 Java 实用工具 API 的一部分。A、B 和 D 不正确。A 不正确，因为 Java 基本的 I/O API 的基本包名为 java.io，并且不包含 Java 集合框架。B 不正确，因为 Java 网络 API 的基本包名为 java.net，同样不包括集合框架。D 不正确，因为没有名为 java.utils 的包。

7. 答案为 C。Java 基本的 I/O API 包含了针对数据流、序列化和文件系统的类和接口。A、B 和 D 不正确，因为国际化(i18n)、RMI、JDBC、JNDI 和集合框架不包括在基本 I/O 的 API 中。

8. 答案为 C。Swing API 为 GUI 组件提供了轻量级的解决方案，这意味着 Swing API 的类是用纯 Java 代码构建的。A、B 和 D 不正确。AWT 和抽象窗口工具是同一个，并且为 GUI 组件提供了重量级的解决方案。

9. 答案为 D。COM.OCAJEXAM.UTILS 不满足正确的包命名规范。包名应该是小写的。包名的单词之间还应该有下划线。但是，ocajexam 中的单词是连接在 URL 中的。因此，这里不包含下划线是可以接受的。包名应该是 com.orajexam.utils。A、B 和 C 不正确。A 不正确，因为限制一个包内只能有一个类是可笑的，没有这样的限制。B 不正确，因为包可以并且经常与其他的包相关联。C 不正确，因为 com.ocajexam.backing_beans 符合正确的包命名规范。

1.8.3　理解类结构

10. 答案为 A 和 B。类名和接口名应该以大写字母开头，并且继续使用驼峰命名规范。C 和 D 不正确。C 不正确，因为常量名应该全部是以下划线分隔的大写字母。D 不正确，因为包名不包括大写字母，也不必遵守驼峰命名规范。

11. 答案为 A。泛型使用尖括号。B、C 和 D 不正确。泛型不能放入方括号、大括号和双引号中。

12. 答案为 B。在类中出现的元素的首选顺序是：首先出现数据成员，其次是构造函数，最后是方法。A、C、D 和 E 不正确。当以这些方式对元素进行排序时，不会造成任何功能或编译错误，这不是首选的。

1.8.4　编译和解释 Java 代码

13. 答案为 D。编译器通过 javac 命令调用。当编译 Java 类时，必须包括以.java 为扩展名的主类的文件名。A、B 和 C 不正确。A 不正确，因为 MainClass.class 是已经编译过的字节码。B 不正确，因为 MainClass 缺少.java 扩展名。C 不正确，因为 MainClass.source 对任何类型的 Java 文件都不是有效的名字。

14. 答案为 A 和 C。-version 标志应该用作第一个参数。应用程序将版本信息以适当的字符串返回到标准输出，然后立刻退出。第二个参数被忽略。B 和 D 不正确。B 不正确，因为版本标志不允许使用双破折号。你可能在工具中看见过双破折号的标志，尤其是那些遵循 GNU 许可证的标志。但是，双破折号不能应用于 Java 解释器的版本标志中。D 不正确，因为版本标志必须作为第一个参数，否则它的功能将被忽略。

15. 答案为 A 和 C。用于指定类路径的可选标志是-cp 或-classpath。B 和 D 不正确，因为可选标志-sp 和-classpaths 是无效的。

16. 答案为 A。属性设置是用于解释器的，而不是编译器。属性名必须被夹在-D 标志和等号之间。所期望的值应紧跟在等号之后。B、C 和 D 不正确，因为-d、-prop 和-D:不是指定系统属性的有效方式。