

巍巍交大 百年书香
www.jiaodapress.com.cn
bookinfo@sjtu.edu.cn



策划编辑 李 勇
责任编辑 胡思佳
封面设计 刘文东

Java 程序设计

Java程序设计

主编 舒 坚 徐国庆



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS



扫描二维码
关注上海交通大学出版社
官方微信

ISBN 978-7-313-32452-8



9 787313 324528 >

定价:69.80元

免费提供

精品教学资料包

服务热线: 400-615-1233
www.huatengedu.com.cn



Java 程序设计

主编 舒 坚 徐国庆



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS



Java 程序设计

主 编 舒 坚 徐国庆
副主编 温洪波 熊建浩
郑 强 杨国强



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

内容提要

本书深入浅出地介绍了 Java 编程语言的基础知识。全书共分为 10 章,主要包括 Java 语言入门,Java 语法基础,数据类型,Java 面向对象,异常与容错,Java 函数指针与函数式编程,Java 文件与 I/O 机制,拓展:Java 界面编程,并发与多线程,拓展:虚拟机机制。

本书既可作为高等院校计算机类专业学生的教材,也可作为软件开发人员的参考资料,还可作为 Java 语言自学者的参考用书。

图书在版编目 (CIP) 数据

Java 程序设计 / 舒坚, 徐国庆主编. -- 上海 : 上海交通大学出版社, 2025. 4. -- ISBN 978-7-313-32452-8

I . TP312. 8

中国国家版本馆 CIP 数据核字第 2025AV4391 号

Java 程序设计

Java CHENGXU SHEJI

主 编:舒 坚 徐国庆

出版发行:上海交通大学出版社

邮政编码:200030

印 制:三河市龙大印装有限公司

开 本:850 mm×1 168 mm 1/16

字 数:619 千字

版 次:2025 年 4 月第 1 版

书 号:ISBN 978-7-313-32452-8

定 价:69.80 元

地 址:上海市番禺路 951 号

电 话:021-64071208

经 销:全国新华书店

印 张:22.5

印 次:2025 年 4 月第 1 次印刷

版权所有 侵权必究

告读者:如您发现本书有印装质量问题请与印刷厂质量科联系

联系电话:0316-3655788

前言

党的二十大报告指出：“教育、科技、人才是全面建设社会主义现代化国家的基础性、战略性支撑。”这明确了人才强国战略在新时代的科学内涵和使命，对加快世界重要人才中心和创新高地建设进行了科学擘画。

历时一年多，编者终于完成了本书的编写。编者最初之所以要编写本书，是因为深刻地体会到一些教材不注重理论与实践相结合，导致学生学习曲线长、学习效果不理想。于是，编者精心组织策划，致力于编写能够培养新时代大国工匠的教材。本书主要体现了以下五个“新”。

（1）技术新：本书紧跟时代的发展趋势，以辩证的思维阐述 Java 语言各个版本的新特性。

（2）教法新：全书对 Java 课程进行了架构设计，整理了教学顺序。本书使用了多种新的教学方法，如实践教学法、需求教学法、演绎教学法、渗透式教学法、拓展教学法和比较教学法等。这些教学方法相互结合，形成了联合拓展式的教学风格。通过示例具象地讲解理论，突出理论体系之间的关联，启发学生思考。理论与示例的结合使得本书内容深入浅出、通俗易懂。

（3）实践新：全书所有的示例都是编者辛勤努力的成果，设计精巧。例如，第 6 章中的示例展示了 Java 函数式编程的演进过程；在第 8 章中，在让读者学习 Java 语言的基础上，系统展示了从产品需求到设计再到开发的全过程，体现了编者的架构设计思想。

（4）高度新：本书从架构师和产品设计师的角度看待 Java 语言，体现有教无类的思想。编者深入查阅了大量的书籍，并通过 Spring Boot 源代码、MyBatis 源代码等技术资料对该思想进行印证。本书注重渗透式的启发和引导，力求不拘一格地培养学生的 Java 技术、产品 and 设计思维。

（5）团队新：本书旨在培养新时代的大国工匠，编写团队中既有行业领军人物，也有具有多年企业工作经验的架构师和分析师，还有任教多年的教师。这种优秀的组合使得本书更好地衔接了理论讲解与学生实践、知识传承与技术前沿、学校教学与市场需求。

编者建议，学习本书最好的办法是边操作边学习。编者认为，技术的出发点是实践，技术的归宿也是实践。首先，本书所有章节都有丰富的、经过精心构建的示例操作，这些示例

操作有助于读者直观地掌握相应的知识。其次，本书所有的操作都符合代码规范，思维缜密。通过本书的示例操作，读者可以培养良好的编码风格和软件操作习惯。再次，编者将对技术的理解 and 应用融入每一步操作之中。通过示例操作，读者可以更好地融入技术的应用场景，理解编者的思维方式，学到编者的编程思想。最后，本书从第 1 章开始就讲解软件开发中可能出现的问题，以及如何解决这些问题。通过示例操作，读者可以辩证地解决软件开发中出现的问题。

在软件开发过程中，通常会出现“二八定律”现象，即 20% 的时间用来编写代码，80% 的时间用来分析设计和解决问题。因此，排除故障和解决问题的技能与知识本身同等重要。

本书由舒坚（南昌航空大学）、徐国庆（江西软件职业技术大学）担任主编，温洪波（江西软件职业技术大学）、熊建浩（江西软件职业技术大学）、郑强（江西省科技干部培训中心）、杨国强（江西省计算技术研究所）担任副主编，王丙炎（江西软件职业技术大学）、杨廷通（江西软件职业技术大学）、白柳儿（江西软件职业技术大学）、蔡艳霞（江西软件职业技术大学）、李莹（江西软件职业技术大学）参与编写。

由于计算机科学技术发展迅速，加之编者自身水平有限，书中可能会存在不足之处，敬请广大读者提出宝贵意见或建议，以便于我们进一步完善本书。

编 者

目录

第 1 章 Java 语言入门

学习目标	1	1.4 Java 程序执行和编译	14
1.1 JVM/JRE/JDK	2	1.5 JAR 包及其执行	15
1.2 JDK 环境安装	2	1.5.1 JAR 包的调用	15
1.2.1 安装 JDK 安装包	2	1.5.2 可执行 JAR 包	19
1.2.2 配置环境变量	4	1.5.3 JAR 文件结构	21
1.2.3 下载安装 Eclipse	7	习题	22
1.3 第一个 Java 程序	10		

第 2 章 Java 语法基础

学习目标	24	2.2.8 运算符优先级	39
2.1 Java 程序结构	25	2.3 Java 流程控制	40
2.1.1 包的声明和使用	25	2.3.1 条件判断语句	40
2.1.2 类型的定义	26	2.3.2 while 循环语句	45
2.1.3 变量、参数、属性的使用	27	2.3.3 do-while 循环语句	46
2.2 表达式与运算符	30	2.3.4 for 循环语句	47
2.2.1 算术运算符	30	2.3.5 多重循环语句	51
2.2.2 关系运算符	32	2.3.6 continue 语句	52
2.2.3 位运算符	33	2.3.7 break 语句	53
2.2.4 逻辑运算符	35	2.3.8 拓展: break point	54
2.2.5 赋值运算符	36	2.4 拓展: 代码规范	55
2.2.6 条件运算符	38	习题	55
2.2.7 instanceof 运算符	38		

第 3 章 数据类型

学习目标	58	3.1.3 内存访问	61
3.1 计算机数据组织	59	3.2 基本数据类型	61
3.1.1 进制及转换	59	3.3 引用数据类型	62
3.1.2 符号编码	60	3.3.1 类	62

3.3.2 基础类 Object	63	3.5 包装类	76
3.3.3 数组	67	3.6 常见数据类型	78
3.3.4 枚举类型: Enum	70	3.6.1 String	78
3.3.5 注解	73	3.6.2 时间	80
3.4 数据转换	73	习题	83

第4章 Java 面向对象

学习目标	86	4.3 静态与非静态	103
4.1 面向过程与面向对象	87	4.4 类之间的关系	104
4.2 面向对象的特性	88	4.5 对象生命周期	106
4.2.1 抽象与接口	88	4.6 反射	108
4.2.2 封装: 包与可视域	90	4.7 注释和注解	110
4.2.3 继承与实现	92	4.7.1 注释	110
4.2.4 多态	95	4.7.2 注解	111
4.2.5 拓展: this 与 super 关键字	99	习题	114

第5章 异常与容错

学习目标	117	5.2.1 异常的捕获	129
5.1 编译器类型检查: 泛型编程	118	5.2.2 异常的种类	132
5.1.1 泛型的引入	118	5.3 Java 空指针处理	133
5.1.2 方法的泛型	120	5.4 对象析构	135
5.1.3 泛型的定义	121	5.5 拓展: 代理和消息模式	135
5.1.4 泛型的继承	121	5.5.1 设计模式与设计原则	135
5.1.5 泛型的使用	123	5.5.2 代理模式	136
5.1.6 泛型的限定	124	5.5.3 观察者模式	138
5.1.7 拓展: 通配符	125	5.5.4 构建者模式	140
5.2 Java 异常处理	129	习题	142

第6章 Java 函数指针与函数式编程

学习目标	145	6.5.1 JDK 8 中的 getter/setter	152
6.1 内部类	146	6.5.2 JDK 8 的 putter	157
6.1.1 方法内部类	146	6.5.3 andThen 方法	159
6.1.2 成员内部类	147	6.6 集合操作	160
6.1.3 静态内部类	147	6.6.1 集合数据类型	160
6.2 匿名类	148	6.6.2 对数组的封装	161
6.3 函数指针 (Lambda 表达式)	149	6.6.3 hash 去重	163
6.4 方法指针	149	6.6.4 排序去重	167
6.5 拓展: 函数式编程	152	6.7 拓展: Stream	171

6.7.1 Stream 的创建	172	6.7.4 终止操作	180
6.7.2 中间操作	173	6.7.5 集合类型转化	182
6.7.3 中间操作原理及 Stream 本质	177	习题	183

第 7 章 Java 文件与 I/O 机制

学习目标	185	7.3.1 TCP 编程	193
7.1 I/O 流	186	7.3.2 拓展：NIO 通信机制	194
7.2 文件读写	188	7.3.3 拓展：UDP 报文	202
7.2.1 Reader 和 Writer	189	7.3.4 拓展：NIO 机制的 UDP	204
7.2.2 RandomAccessFile	190	7.3.5 拓展：URL 中的协议	206
7.2.3 拓展：配置文件	191	习题	208
7.3 网络编程	193		

第 8 章 拓展：Java 界面编程

学习目标	209	8.3.2 流布局	221
8.1 Swing 的特点及其包结构	210	8.3.3 边界布局	223
8.1.1 Swing 的特点	210	8.3.4 网格布局	225
8.1.2 Swing 的包结构	210	8.3.5 网格组布局管理器	227
8.2 Swing 组件	211	8.4 面板	232
8.2.1 常用组件	211	8.4.1 JPanel 面板	232
8.2.2 JFrame 窗体	212	8.4.2 JScrollPane (滚动面板)	235
8.2.3 JDialog 对话框窗体	214	8.5 常用事件监听器	237
8.2.4 JLabel 标签	216	8.5.1 监听事件简介	237
8.2.5 图片标签的使用	217	8.5.2 动作事件监听器	237
8.3 布局管理	219	8.5.3 焦点事件监听器	241
8.3.1 绝对布局	219	习题	243

第 9 章 并发与多线程

学习目标	246	9.2.5 线程安全类	270
9.1 Java 多线程	247	9.2.6 拓展：管道 (pipe)	271
9.1.1 线程池	250	9.2.7 拓展：死锁的产生	273
9.1.2 线程异步机制	253	9.2.8 拓展：死锁的检测	275
9.2 并发控制	254	9.2.9 拓展：消息队列	277
9.2.1 管程机制	256	9.3 资源阻塞与唤醒	289
9.2.2 原子锁	259	9.4 时钟 Timer	294
9.2.3 信号量	264	习题	299
9.2.4 读写锁	266		

第 10 章 拓展：虚拟机机制

学习目标	300	10.3.1 内存管理模型.....	315
10.1 JNI 机制	301	10.3.2 虚拟机栈.....	316
10.1.1 JNI 程序设计	302	10.3.3 内存配置.....	318
10.1.2 JNI 数据类型	311	10.3.4 内存溢出.....	322
10.2 类加载机制	313	习题	326
10.3 内存管理	315		

附录 TCP/UDP 测试工具

参考文献

第1章 Java 语言入门

Java 语言作为当今信息化社会中最流行且广泛应用的网络编程语言之一，在构建高效、可移植和安全的网络应用程序方面发挥了重要的作用。本章将围绕 Java 开发环境及其配置进行详细的介绍。



学习目标

知识目标：

- 掌握 Eclipse 的安装和配置方法。
- 掌握 JDK 的环境安装。
- 了解 JVM、JRE 和 JDK 之间的区别。

技能目标：

- 掌握 Java 环境的下载和安装方法。
- 掌握 Java 程序的编写方法。

素质目标：

- 通过 Java 语言的学习，培养和提高逻辑思维与分析能力，能够理解和分析复杂的程序逻辑，从而有效地解决问题。
- 形成自主学习的习惯，具备持续学习的意识，能够主动寻找和学习新的 Java 技术和知识，不断更新自己的知识体系。
- 培养团队协作精神，能够与他人有效地合作完成 Java 项目。同时，提高沟通能力，能够清晰地表达自己的观点和需求，与团队成员建立良好的合作关系。
- 树立职业道德观念，遵守编程规范和行业标准，保持诚信意识，不抄袭、不剽窃他人代码，尊重他人的知识产权。

Java 既是一种编程语言，也是一种运行环境。通常，Java 程序的源代码被编译成 .class 字节码后，需要放到 Java 虚拟机（JVM）环境中运行。为了满足开发者和用户的不同需求，Java 提供了两套方案：JDK 和 JRE。JVM、JRE、JDK 之间的关系如图 1.1-1 所示。

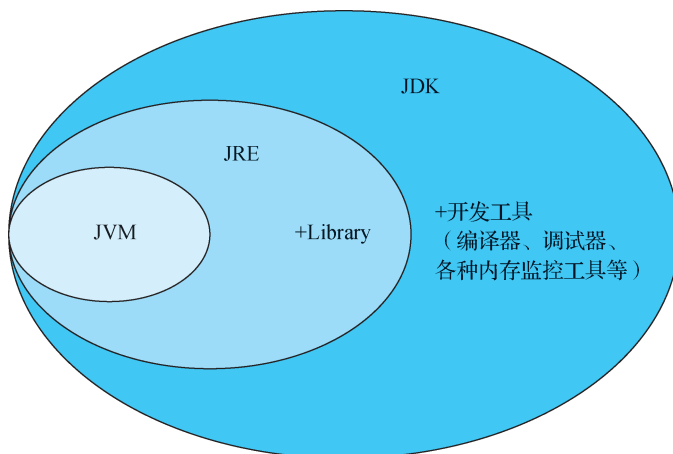


图 1.1-1 JVM、JRE、JDK 之间的关系

(1) JVM (Java virtual machine): Java 程序最终运行在虚拟机上, 通过 Java 命令和 Java 程序得以执行。由于 Java 程序在运行前会被编译成 .class 字节码, 而字节码并不是直接可被硬件执行的机器码, 因此需要 JVM 对类文件加载、解释和验证, 并以虚拟的方式逐条执行其中的指令。

(2) JRE (Java runtime environment): 是 Java 程序的运行时环境, 专为在设备终端上运行 Java 程序而设计, 因此只包含 JVM 和运行时所需要的类库。

(3) JDK (Java development kit): Java 程序开发工具包, 专为开发人员设计, 其中包含 JRE、编译器和调试器等开发工具。

在进行 Java 开发时，需要安装 JDK 而不是 JRE。如果使用 Eclipse 工具，那么引用的也是 JDK 而不是 JRE。因为只有 JDK 才具备编译功能，而 JRE 是不能用于编译的。下面将详细介绍 JDK 的安装、配置环境变量的方法，以及 Eclipse 的下载安装方法。

1.2.1 安裝 JDK 安裝包

第一步：下载JDK。进入JDK 官网 <https://jdk.java.net/java-se-ri/11>，其页面如图 1.2-1 所示。在该页面上，选择适合自己需求的 JDK 版本进行下载。

第二步：安装JDK。双击下载好的JDK安装包，系统会弹出安装窗口，单击“下一步”按钮，如图 1.2-2 所示。

单击“更改”按钮，选择安装目录，不建议安装在 C 盘中，然后单击“下一步”按钮，如图 1.2-3 所示。

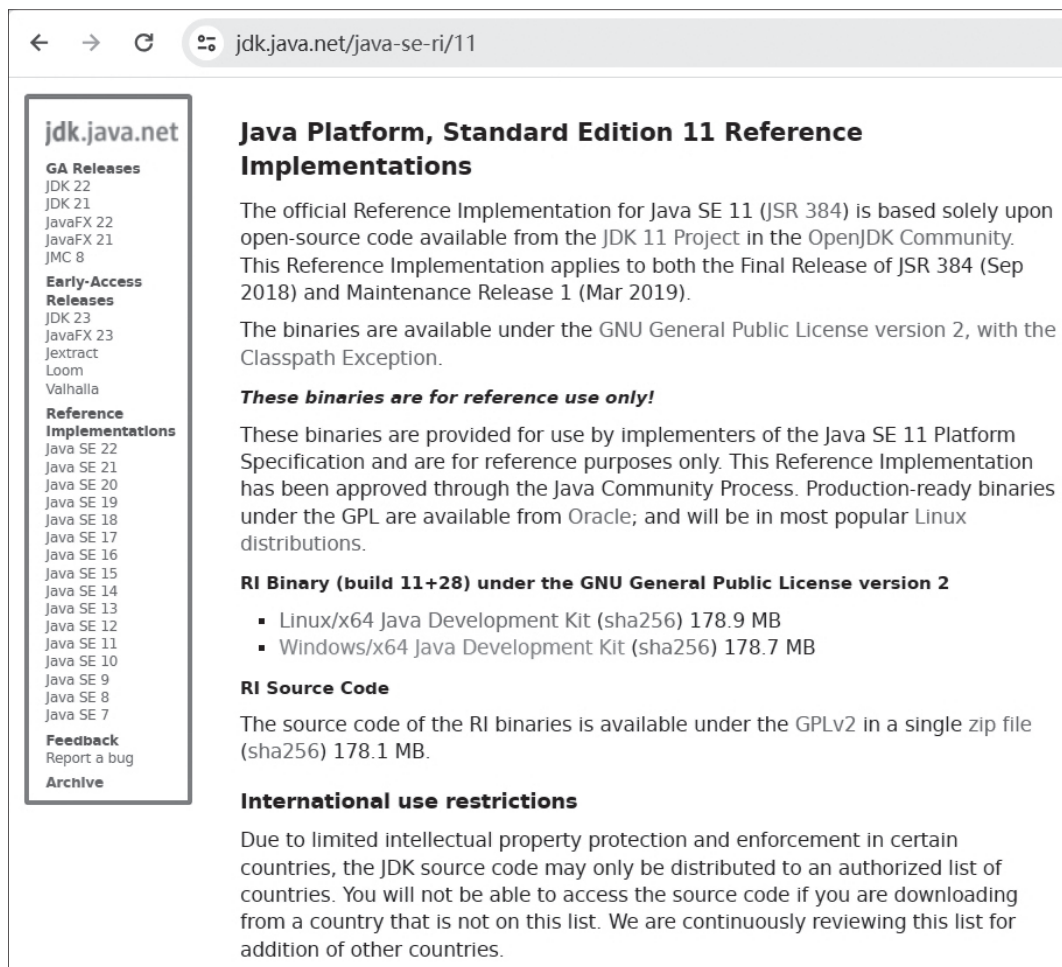


图 1.2-1 下载 JDK



图 1.2-2 安装 JDK

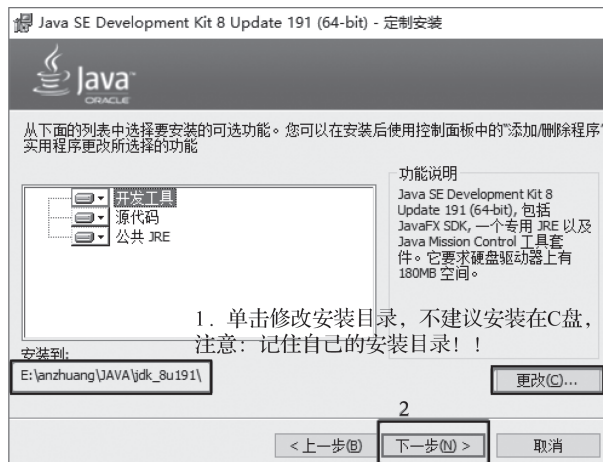


图 1.2-3 选择安装的目录

页面会显示安装进度, 如图 1.2-4 所示。
安装完成, 单击“关闭”按钮, 如图 1.2-5 所示。



图1.2-5 安装完成

以上步骤完成后，在 DOC 窗口运行 Java 命令，会出现图 1.2-6 所示的结果。

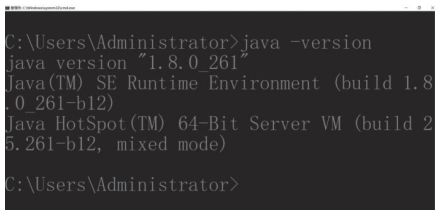


图 1.2-6 运行结果

有些初学者往往会认为，这样JDK 安装就完成了。但实际上还需要配置环境变量，否则无法进行程序开发。这是因为安装包安装完成后，往往只会将Java 命令的所在路径添加到环境变量中，但是编译所需要的命令并不在这个路径下。因此，必须重新设置环境变量，以确保编译命令能够被正确识别和执行，如图 1.2-7 所示。

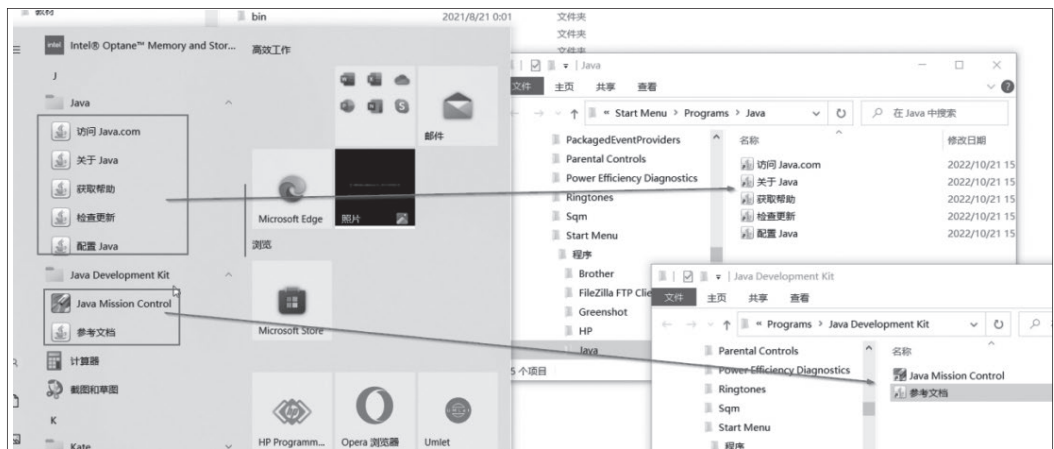


图 1.2-7 安装完成

打开控制面板，单击“系统和安全”→“系统”→“高级系统设置”，会弹出“系统属性”对话框，切换到“高级”选项卡，单击“环境变量”按钮，然后进行设置，如图 1.2-8 所示。



图 1.2-8 单击“环境变量”按钮

系统弹出“环境变量”对话框，单击最下面的“新建”按钮，新建一个名为“JAVA_HOME”（代表 JDK 安装路径）的变量，其值对应的是 JDK 的安装路径，如图 1.2-9 所示。

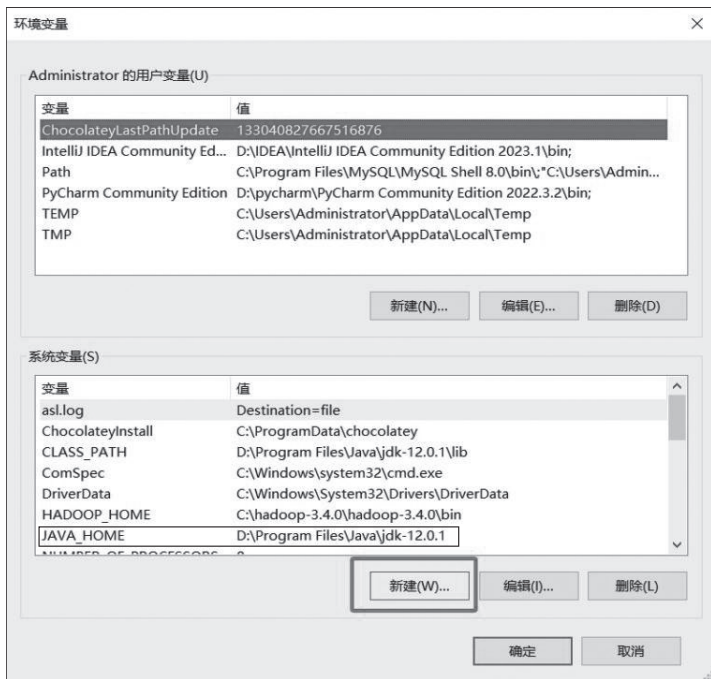


图 1.2-9 新建 JAVA_HOME

系统弹出“新建系统变量”对话框，设置变量名为JDK_HOME（代表JDK安装路径），变量值对应的是JDK的安装路径，如图1.2-10所示。

在系统变量中找一个名称为“Path”的变量（见图 1.2-11），如果没有就新建一个（见图 1.2-12）。在它的变量值里面追加代码“%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;”。

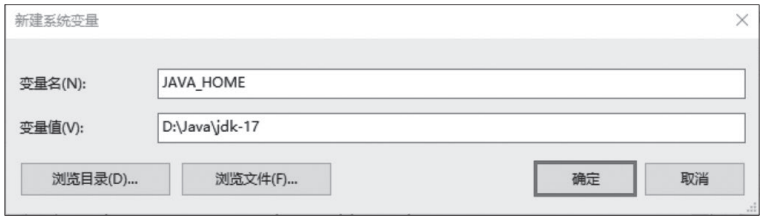


图 1.2-10 配置 JAVA_HOME

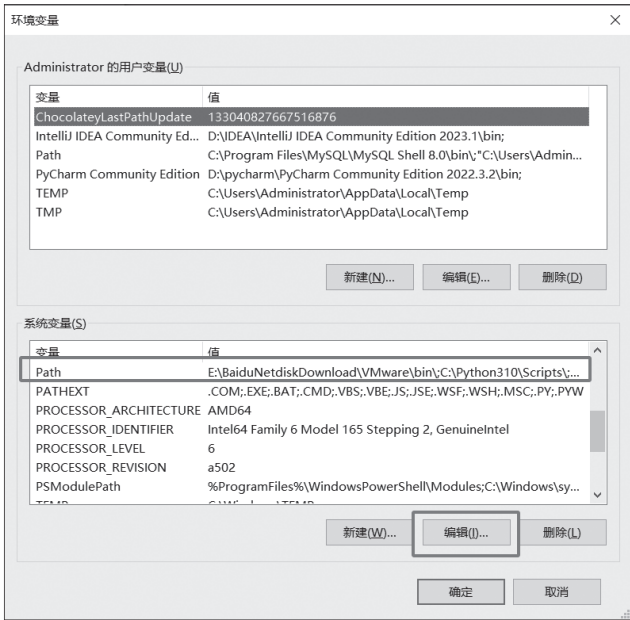


图 1.2-11 编辑 Path 变量

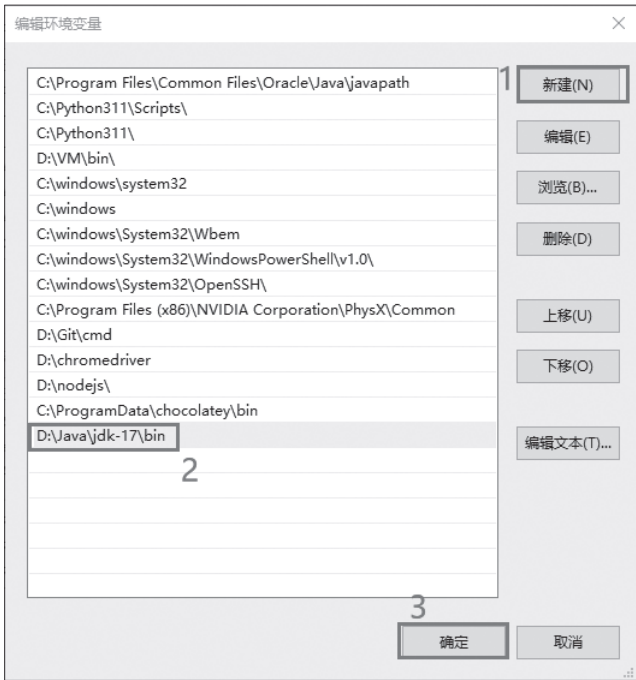


图 1.2-12 添加 Path 变量

然后根据提示单击“确定”按钮即可。至此，Java 环境就安装配置好了。若需要查看已安装的Java，则可打开命令行，使用 `javac -version` 命令查看版本。若出现图 1.2-13 所示的结果，则表示环境配置正确，否则表示配置出错。

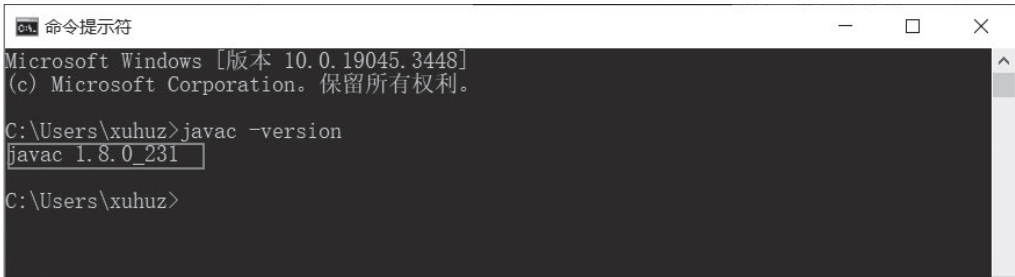


图 1.2-13 查看 Java 版本

1.2.3 下载安装 Eclipse

进入 Eclipse 官网（<https://www.eclipse.org/downloads/>）进行下载。登录官网之后，下载最新版本的安装包并进行安装。安装步骤如图 1.2-14 所示。

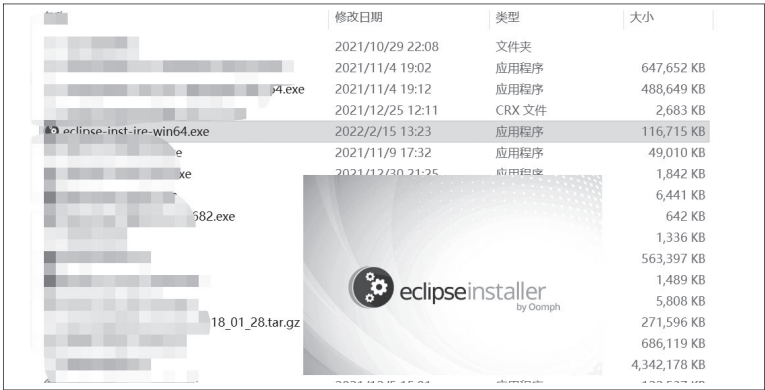


图 1.2-14 安装 Eclipse

选择 Eclipse 版本，一般选择第二个版本，如图 1.2-15 所示。

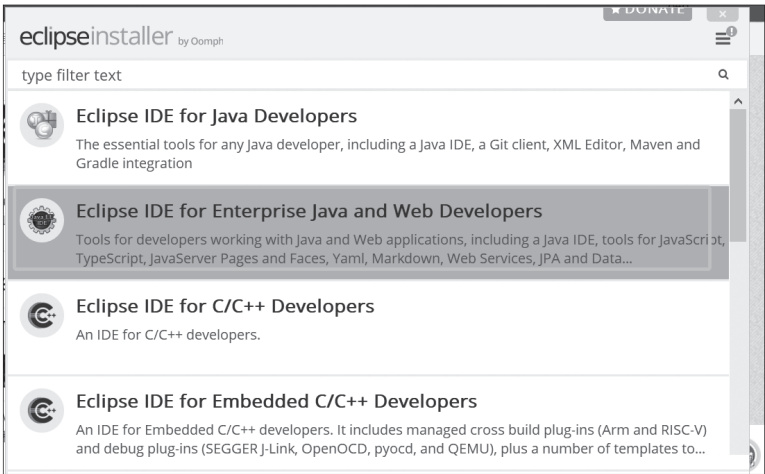


图 1.2-15 选择 Eclipse 版本

根据自己的计算机配置选择合适的安装路径，如图 1.2-16 所示。

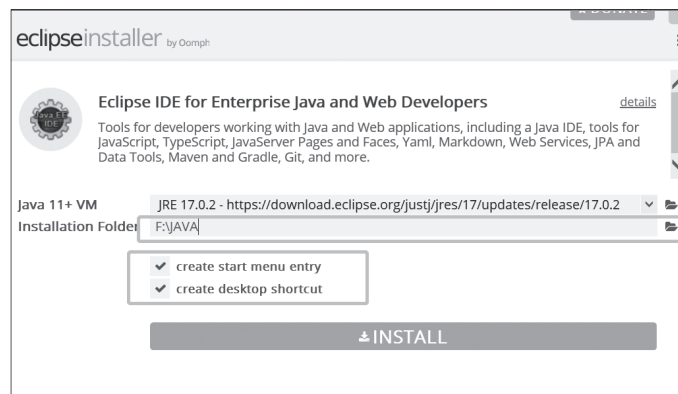


图 1.2-16 选择安装路径

选择安装 Eclipse 软件的版本，单击“Accept Now”按钮，如图 1.2-17 所示。

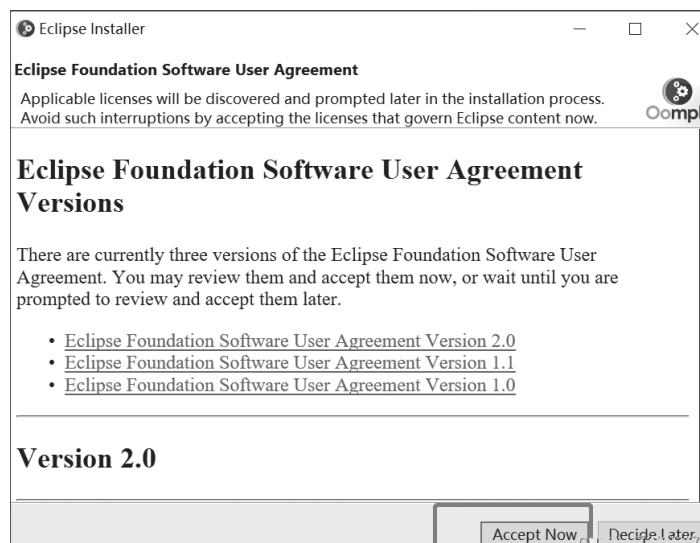


图 1.2-17 单击“Accept Now”按钮

接下来开始安装 Eclipse，如图 1.2-18 所示。

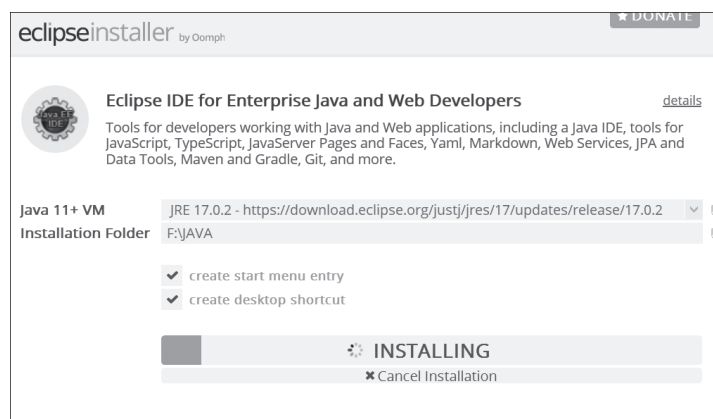


图 1.2-18 开始安装

安装完成后，系统会弹出图 1.2-19 所示的界面，单击“LAUNCH”按钮即可开始运行 Eclipse。

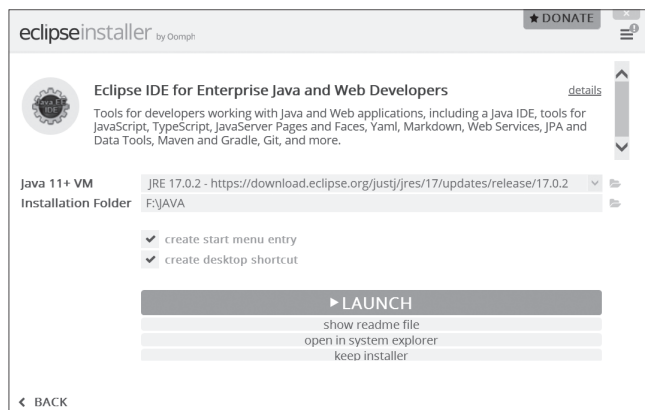


图 1.2-19 运行 Eclipse

系统弹出“Eclipse IDE Launcher”对话框，在其中选择 Workspace（工作区），就是开发的所有 Java 项目的地址。选中“Use this as the default and do not ask again”复选框后，下次不需要再选择工作区了，如图 1.2-20 所示。

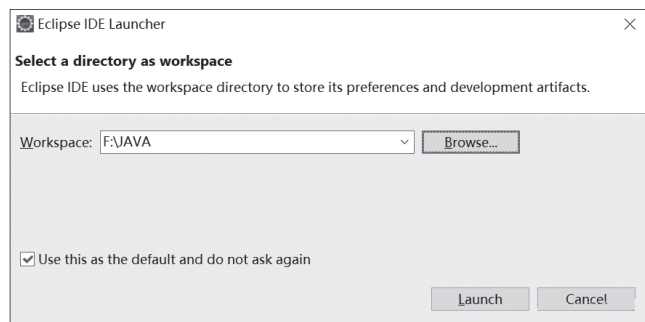


图 1.2-20 设置文件路径

Eclipse 软件首页如图 1.2-21 所示。

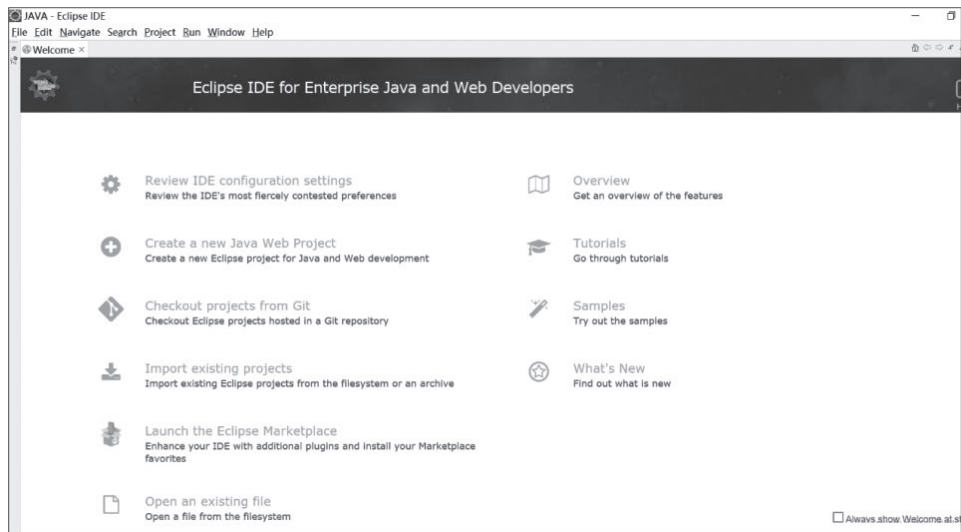


图 1.2-21 Eclipse 软件首页

1.3 第一个 Java 程序

使用 Eclipse 创建 Java 程序，首先打开 Eclipse 软件，单击“File”→“New”→“Project”，如图 1.3-1 所示，最后单击创建的 Java 项目。

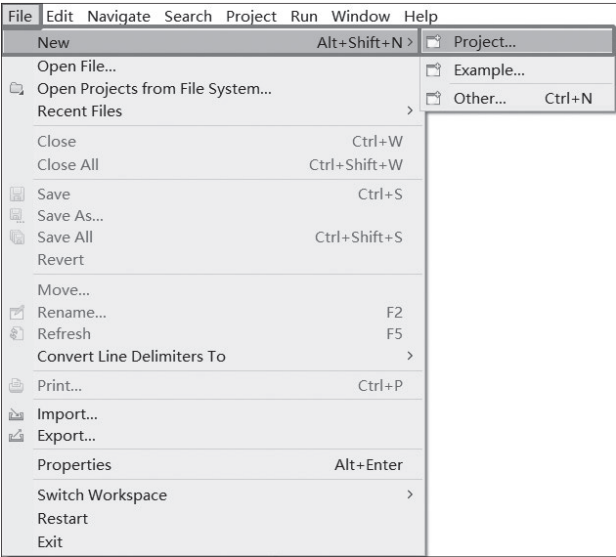


图 1.3-1 新建第一个 Java 项目

然后在弹出的“New Java Project”对话框中为新建的 Java 项目命名，如图 1.3-2 所示。

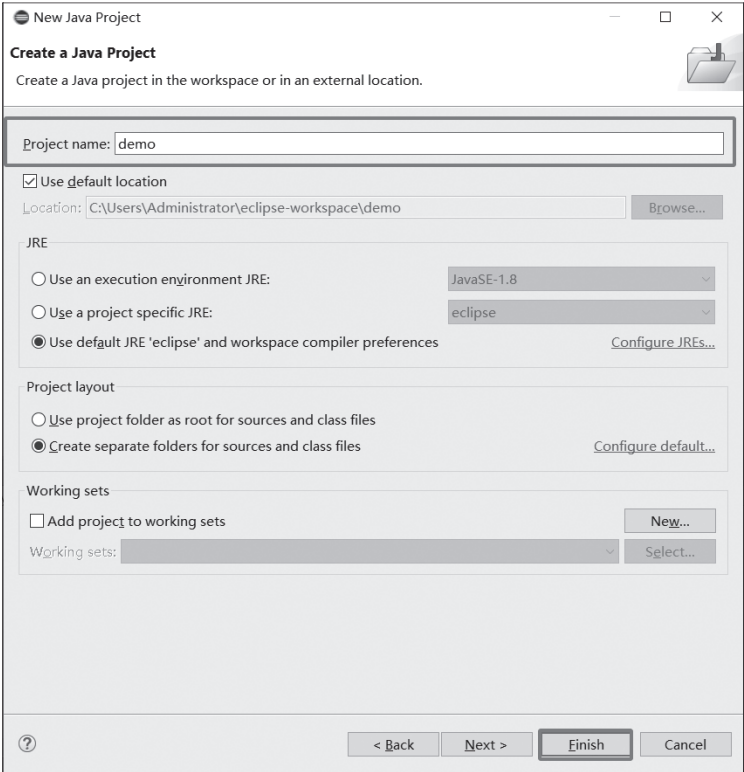


图 1.3-2 为 Java 项目命名

包 (Package)。

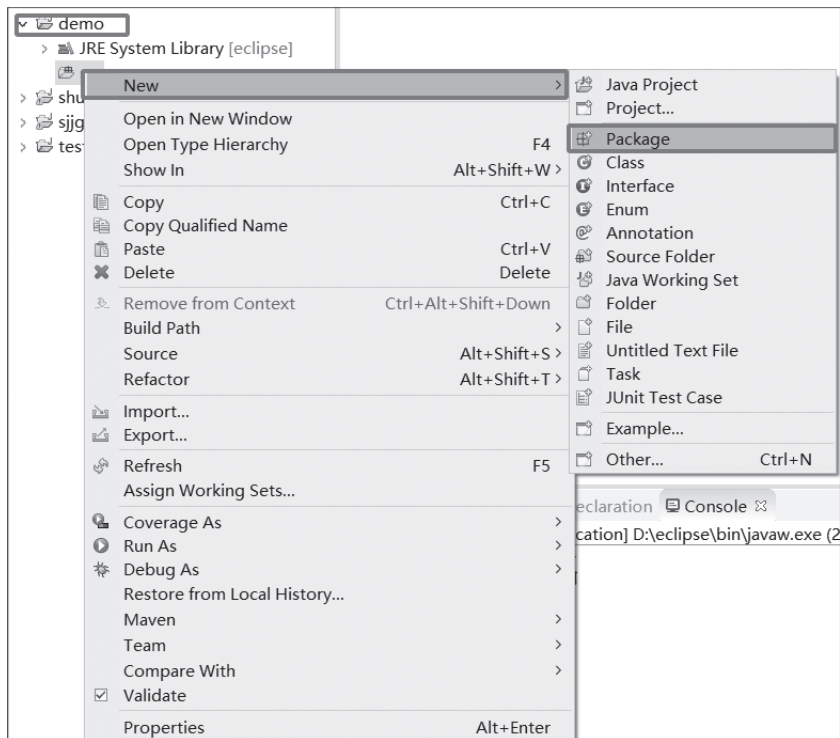


图 1.3-3 新建一个 Java 包

图 1.3-4 所示。

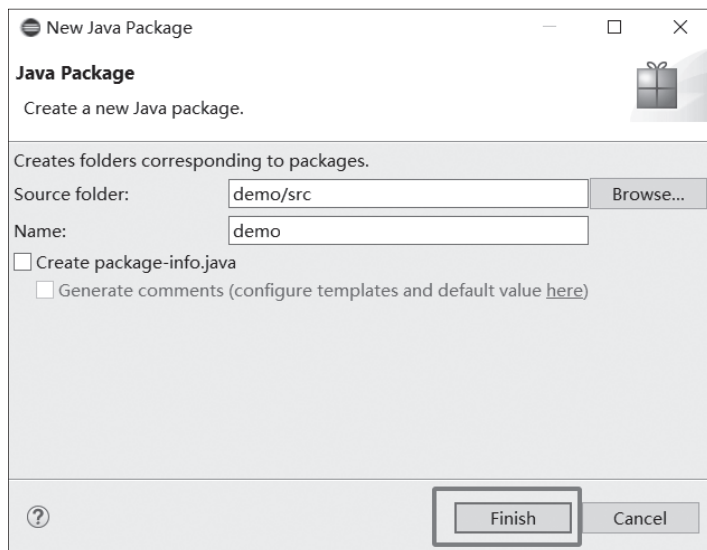


图 1.3-4 新建包名

一个 Java 类。

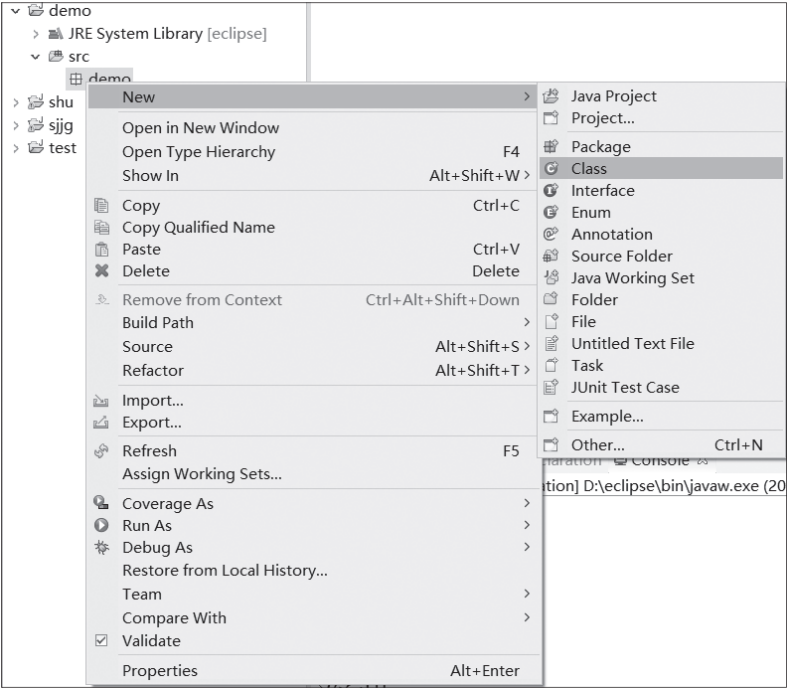


图 1.3-5 新建 Java 类

在弹出的“New Java Class”窗口中自定义一个类的名称，如图 1.3-6 所示。

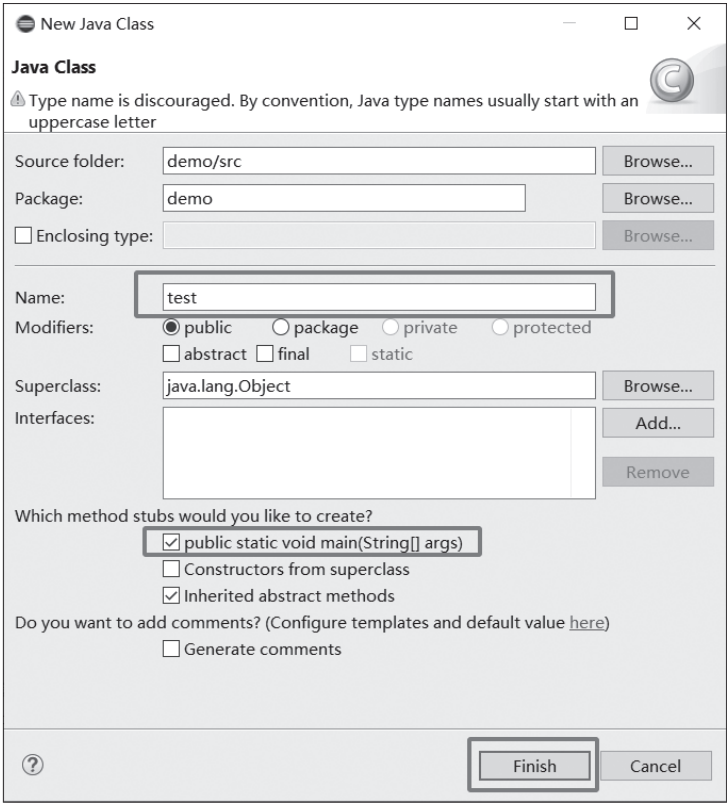


图 1.3-6 自定义类名

在弹出的代码框中输入一条输出语句“System.out.println(“HelloWorld”);” 如图 1.3-7 所示。

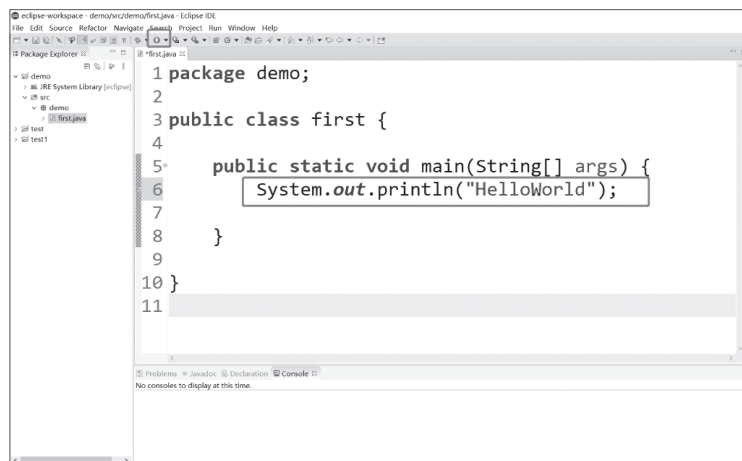


图 1.3-7 输入代码

在 `mian()` 方法所在的窗口右击，然后在弹出的快捷菜单中选择“RunAs”→“Java Application”选项，运行程序。如果代码未保存，就会弹出图 1.3-8 中的对话框。

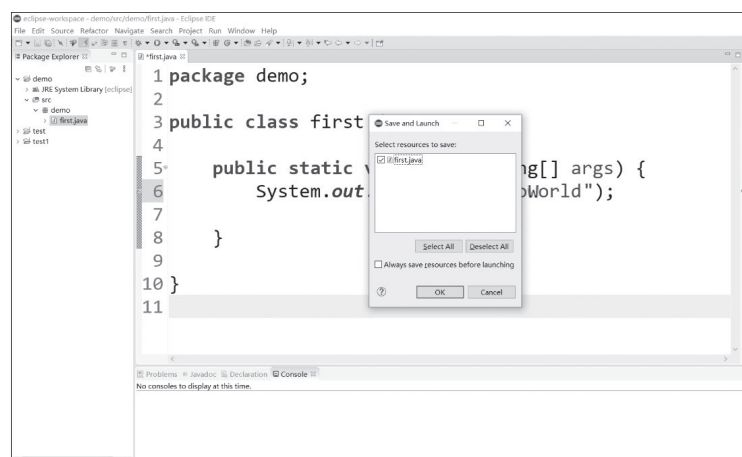


图 1.3-8 运行程序

单击“OK”按钮，程序运行结果就会在输出框中显示出来，如图 1.3-9 所示。

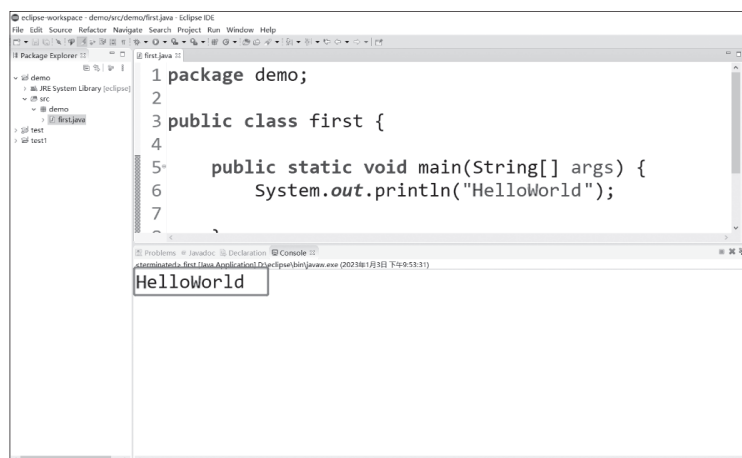


图 1.3-9 程序运行结果

前面已经新建了第一个 Java 类 HelloWorld，并在 Eclipse 中成功运行了它。

操作 接下来右击项目名称，在弹出的快捷菜单中选择“Properties”，系统会弹出“Properties Introduction”窗口。然后，在“Resource”选项卡中单击“Location”按钮，这样就可以定位到 HelloWorld 项目所在的目录，如图 1.4-1 所示。

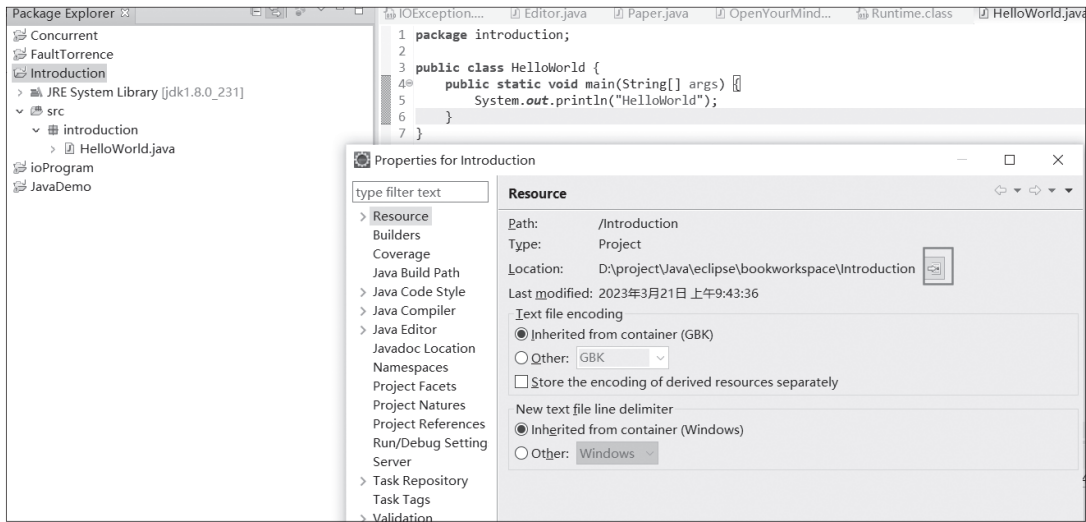


图 1.4-1 查看文件路径

在该目录下的 `/bin/introduction` 子目录中，可以发现 `HelloWorld.class` 文件。如果删除该文件，再次运行 `HelloWorld` 类时，将会出现以下错误。

错误: 找不到或无法加载主类introduction.HelloWorld

出现这种问题的原因在于，Java 是一种编译语言，其源代码（HelloWorld.java）必须先被编译成字节码文件（HelloWorld.class），然后这个字节码文件才能被 Java 虚拟机（JVM）执行。Eclipse 只是为开发者完成了编译和运行的步骤。因此，一旦缺少了 HelloWorld.class 文件，运行操作也就无法进行了。

Java 使用 `javac` 命令对源代码进行编译，而使用 `Java` 命令运行编译好的类。下面在 CMD 中进入的 `src` 目录，并编译该 Java 程序，如图 1.4-2 所示。



图 1.4-2 DOS 窗口编译

然后进入项目的 \bin 目录会发现编译好的目录和类文件 HelloWorld.class，如图 1.4-3 所示。

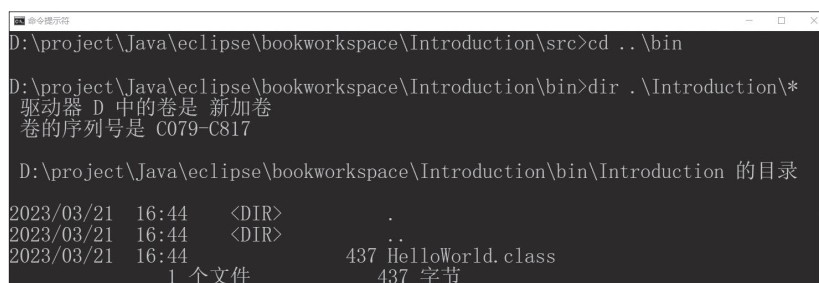


图 1.4-3 运行结果

这时，在该目录下执行 Java 命令，可以看到，Java 程序执行成功并在控制台上打印出 Hello World，如图 1.4-4 所示。

```
D:\project\Java\eclipse\bookworkspace\Introduction\bin>java introduction.HelloWorld
HelloWorld
D:\project\Java\eclipse\bookworkspace\Introduction\bin>
```

图 1.4-4 程序运行结果

值得注意的是：javac 编译源代码时，使用的命令行为“javac introduction\HelloWorld.java -d ..\bin”，此处参数直接是 HelloWorld.java 文件的相对路径；而在执行时，则使用命令行“java introduction.HelloWorld”，此处参数是类名。

1.5 JAR 包及其执行

Java 程序不仅可以在未打包的状态下执行，而且提供了一种将程序打包后执行的机制，具体如下。

(1) Java 程序中的部分代码可以从根目录或其他指定目录被打包成 .jar 文件（Java 归档文件），随后这些 .jar 文件可以被其他 Java 程序作为库或依赖进行调用。

(2) 还可以将整个 Java 主类及其依赖项打包到一个 .jar 文件中，然后通过 java -jar 命令直接执行该 .jar 文件。

1.5.1 JAR 包的调用

首先创建一个 HelloJar 项目，在该项目中创建一个 BeCalled 类，如图 1.5-1 所示。其代码如下。

```
package com.hello;

public class BeCalled {
    public void hello(){
        System.out.println("hello called");
    }
}
```

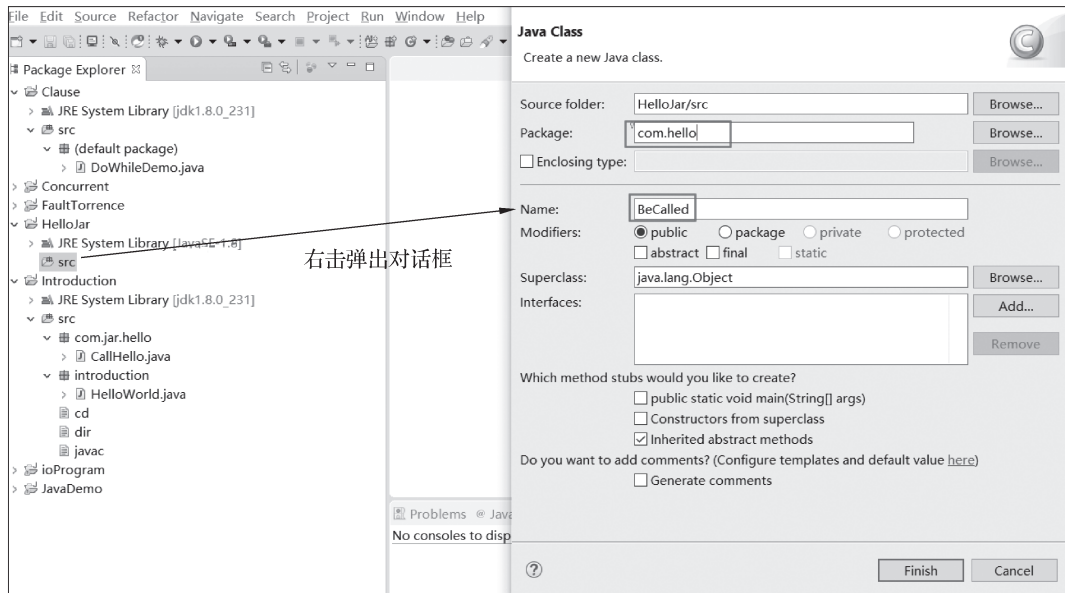


图 1.5-1 创建 BeCalled 类

按照以下步骤打包该项目。首先在“HelloJar”项目上右击，然后在弹出的快捷菜单中选择“Export”选项，如图 1.5-2 所示。

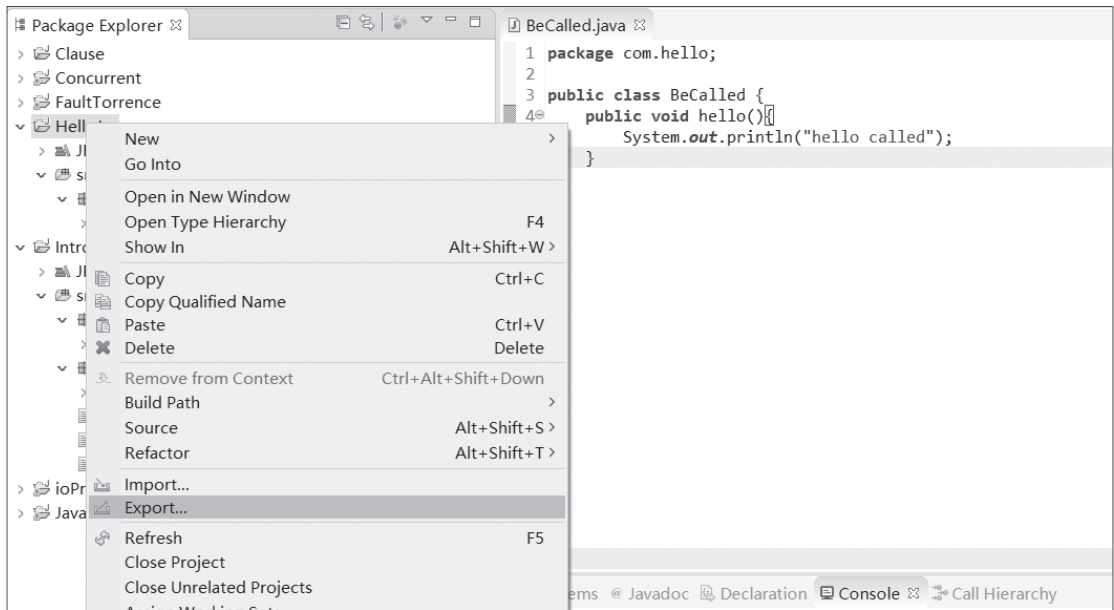


图 1.5-2 编写代码

系统弹出“Export”窗口，在其中选择“Java”→“JAR file”选项，然后单击“Next”按钮，如图 1.5-3 所示。

在弹出的窗口中选“Export generated class files and resources”复选框，然后将该项目生成的 .class 类文件打包到 .jar 文件，最后单击“Finish”按钮，如图 1.5-4 所示。

这样就打包了一个 .jar 文件。然后在 Introduction 项目中引入该 JAR 包文件。在 Introduction 项目上右击，然后在弹出的快捷菜单中选择“Properties”选项，会出现图 1.5-5 所示的界面。

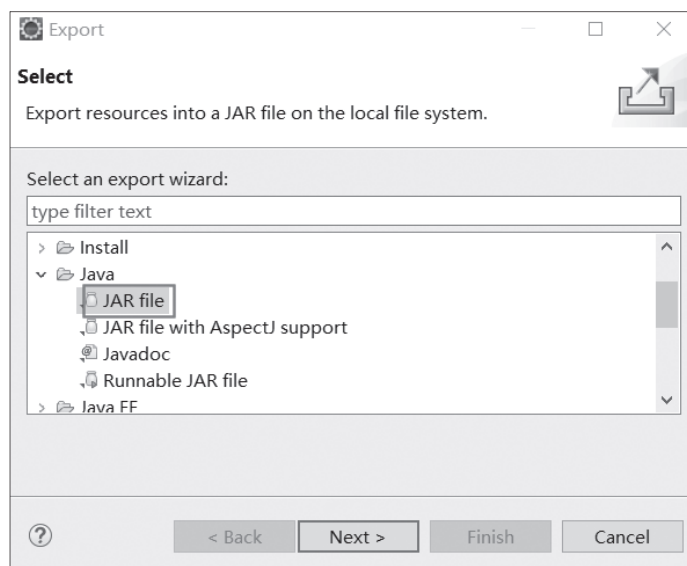


图 1.5-3 创建文件

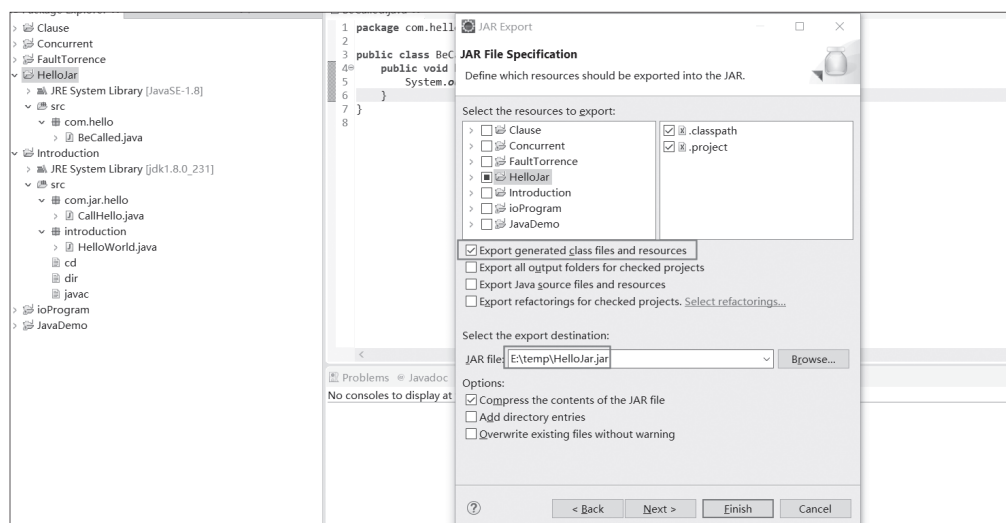


图 1.5-4 生成 .jar 文件

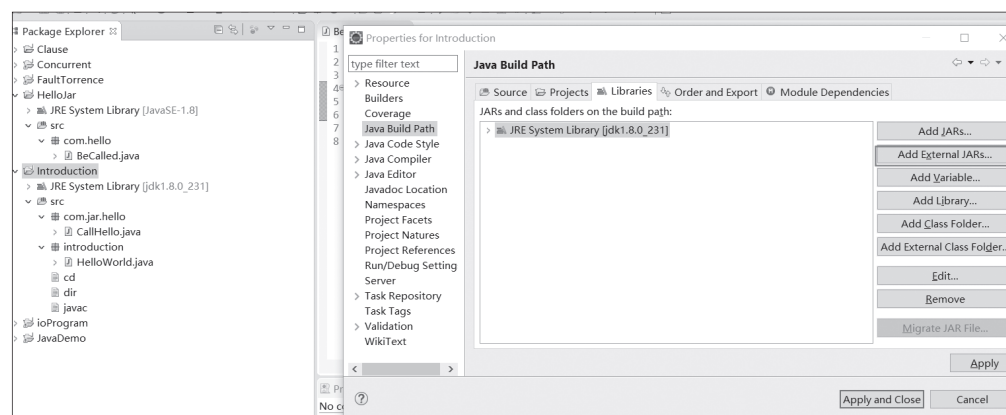


图 1.5-5 Introduction 项目

单击“Add External JARs”按钮，选择刚才打包的文件，然后在弹出的“JAR Selection”对话框中单击“打开”按钮，最后单击“Apply and Close”按钮，如图 1.5-6 所示。

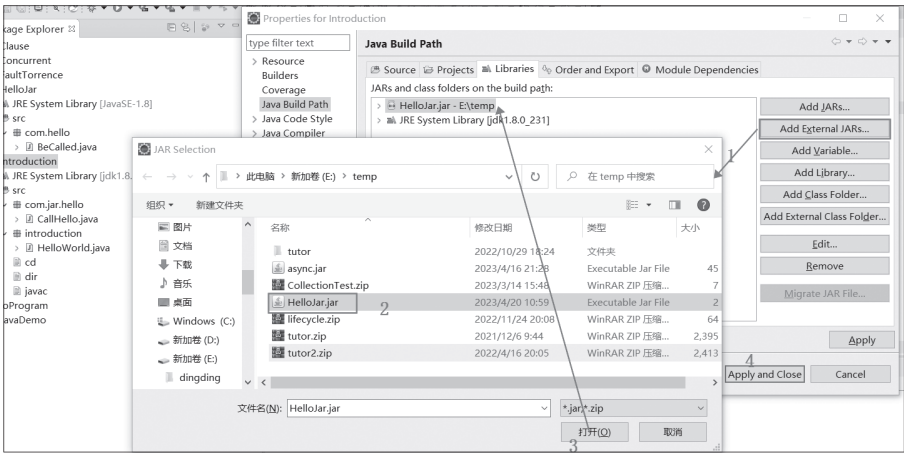


图 1.5-6 选择文件

创建一个 Java 程序时需要先创建类，其代码如下。

```

package com.jar.hello;
import com.hello.BeCalled;
public class CallHello {
    public static void main(String[] args) {
        BeCalled beCalled = new BeCalled();
        beCalled.hello();
    }
}
    
```

执行 CallHello 类，其运行效果如图 1.5-7 所示。

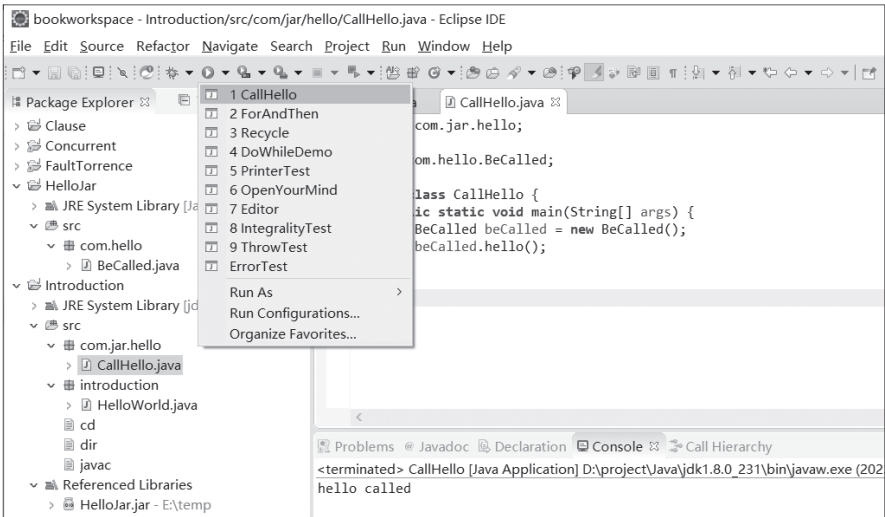


图 1.5-7 运行效果

1.5.2 可执行 JAR 包

下面打包一个可执行的 .jar 文件，如图 1.5-8 所示。

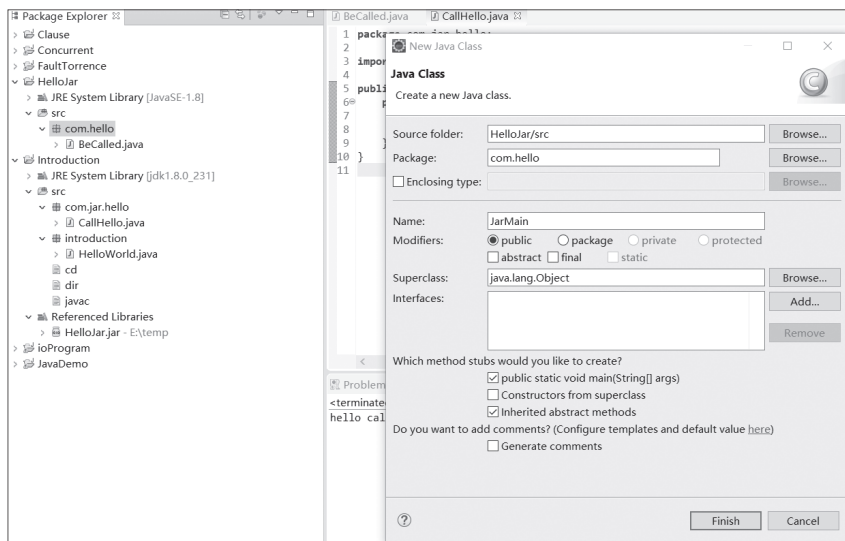


图 1.5-8 打包 .jar 文件

其代码如下。

```
public class JarMain {  
    public static void main(String[] args) {  
        System.out.println("JarMain say hello");  
    }  
}
```

现在将其打包。打包方式与前面几步相同。首先在项目名称上右击，然后在弹出的快捷菜单中选择“Export”选项，系统会弹出“Export”窗口，单击“Next”按钮，如图 1.5-9 所示。

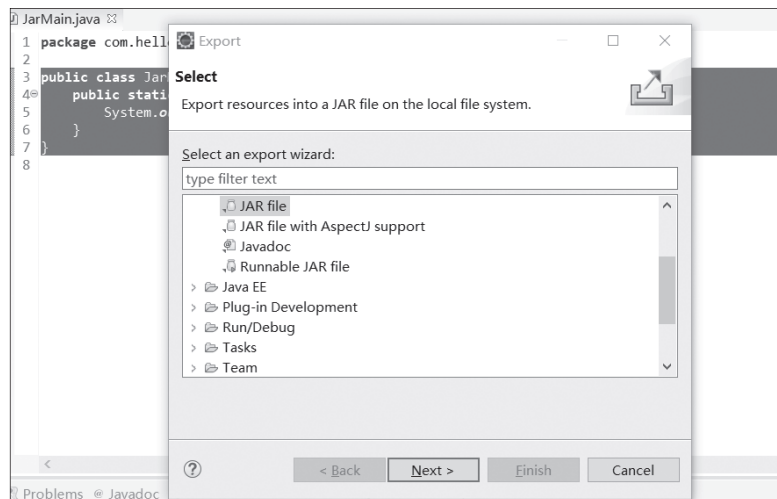


图 1.5-9 “Export” 窗口

在弹出的“JAR Export”窗口中指定JAR文件，并选中图 1.5-10 所示的复选框，再单击“Next”按钮。

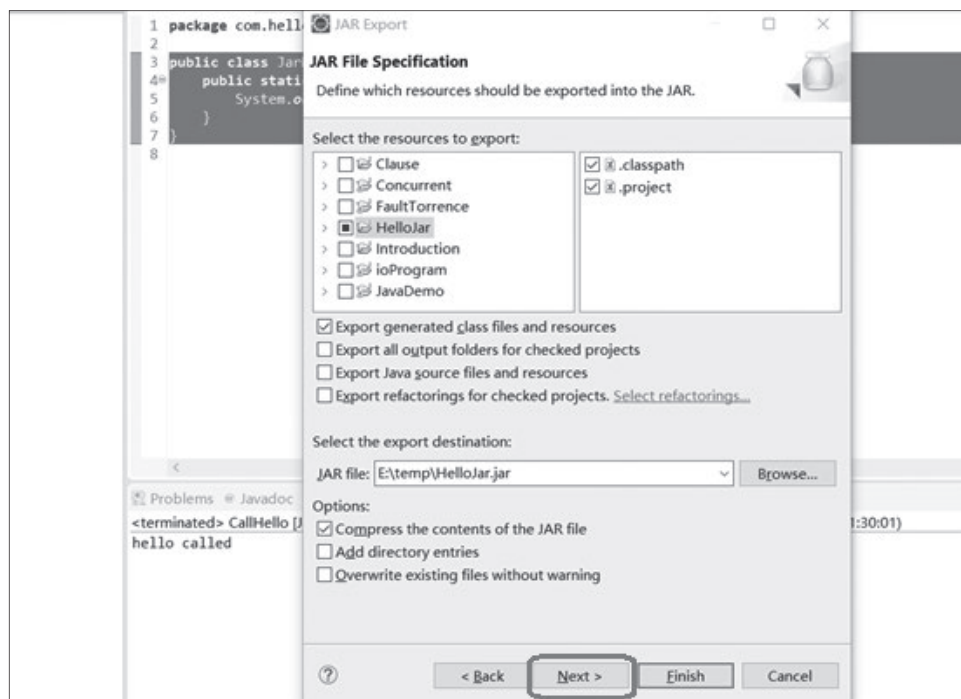


图 1.5-10 选择指定 JAR 文件

在弹出的窗口中单击“Next”按钮，如图 1.5-11 所示。

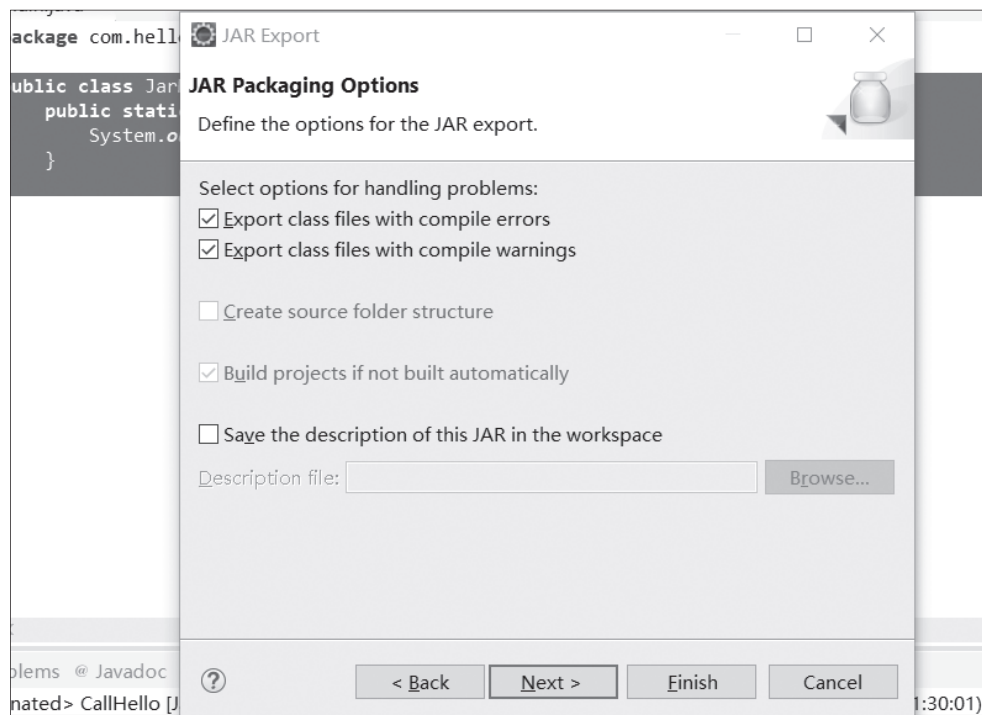


图 1.5-11 执 行

最后，在“Main class”文本框中输入“com.hello.JarMain”，再单击“Finish”按钮，如图 1.5-12 所示。

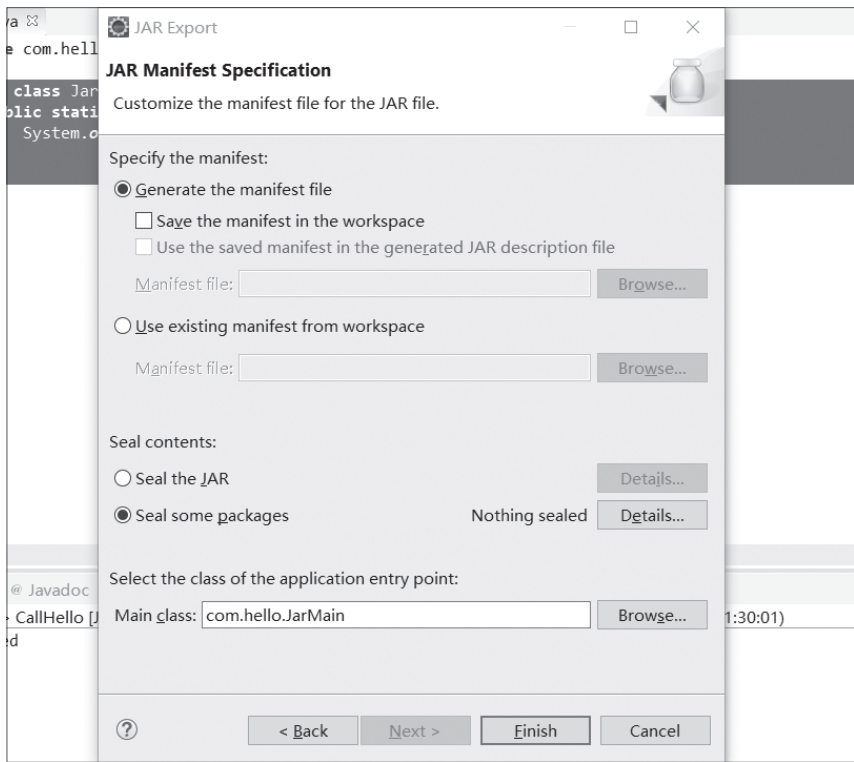


图 1.5-12 输入“com.hello.JarMain”

打开“命令提示符”（CMD）窗口，在 HelloJar.jar 所在的目录下执行 Java 命令，运行结果如图 1.5-13 所示。

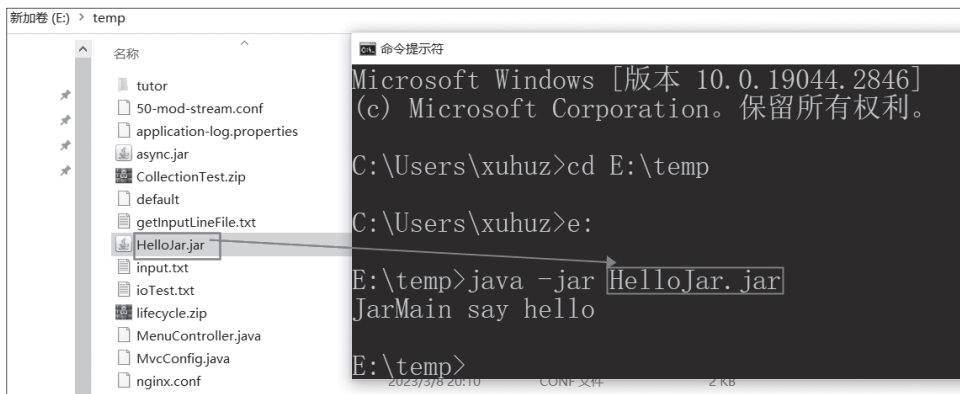


图 1.5-13 运行结果

1.5.3 JAR 文件结构

现在通过 HelloJar.jar 文件来分析 JAR 包文件的结构。首先，JAR 包文件本质上也是个 zip 文件，因此，可以通过一个具有 zip 解压功能的程序（如 WinZip 或 WinRAR）来解压或打开该文件，其结构如图 1.5-14 所示。

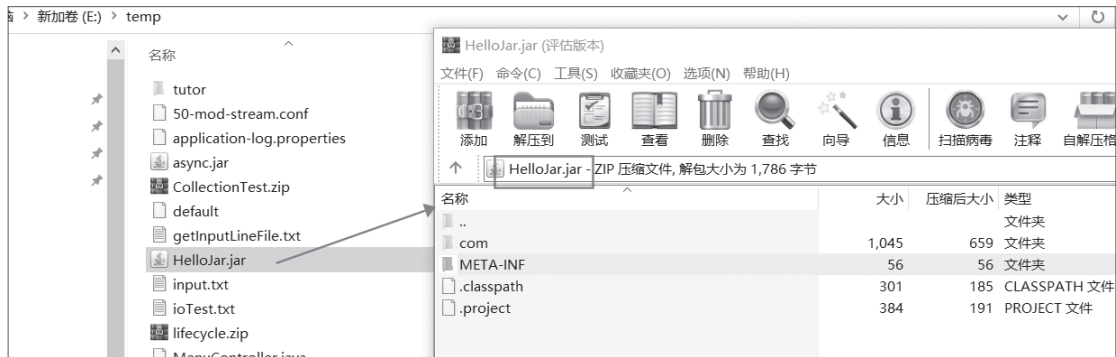


图 1.5-14 解压文件

分析这个结构可以发现，打包为 JAR 包实际上就是将相关的文件以 zip 格式压缩并打包成 .jar 文件。对于一般性的 JAR 包，只需要把所执行的 class 和资源文件从根目录开始逐级压缩进去即可，而对于可执行的 JAR 包，则需要包含一个名为 \META-INF\MANIFEST.MF 的文件，该文件内指定了需要执行的主类，如图 1.5-15 所示。

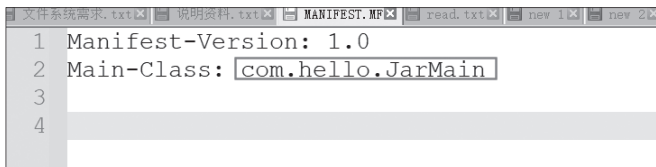


图 1.5-15 需要执行的主类名称

链接 JAR 资源文件的读取方式请参考 7.3.5 小节 URL 协议中的讲述。

习题

一、选择题

- 关于 JVM 的三个组成部分，下列（ ）是 Java 程序的运行环境。

A. JDK

B. JRE

C. Eclipse

D. JVM
- 在 JDK 环境安装的过程中，（ ）步骤是设置操作系统中 Java 程序的执行路径。

A. 安装 JDK 安装包

B. 配置环境变量

C. 下载安装 Eclipse

D. 第一个 Java 程序
- 执行 Java 程序的关键组件是（ ）。

A. JDK

B. JRE

C. JVM

D. Eclipse
- JAR 包是一种 Java 的（ ）文件。

A. 文本

B. 可执行

C. 压缩

D. 图像
- 可执行 JAR 包是指具有（ ）特性的 JAR 文件。

A. 包含 Java 源代码

B. 包含编译后的字节码

C. 包含文档和注释

D. 包含操作系统特定的二进制文件
- 关于 JAR 文件结构，（ ）部分用于指定包含主类的信息。

A. MANIFEST.MF

B. src/

- C. lib/ D. META-INF/
7. JVM 是 () 的缩写。
- A. Java Virtual Machine B. Java Runtime Environment
C. Java Development Kit D. Java Environment Variable
8. 在配置环境变量时, JAVA_HOME 通常指向 () 目录。
- A. JRE B. Eclipse 安装 C. JDK D. JVM
9. () 步骤用于下载和安装 IDE 集成开发环境。
- A. 安装 JDK 安装包 B. 配置环境变量
C. 下载安装 Eclipse D. 第一个 Java 程序
10. 关于 Java 程序的编译, 下列 () 阶段将 Java 源代码转换为字节码。
- A. 执行阶段 B. 编码阶段 C. 编译阶段 D. 链接阶段

二、填空题

1. Java 开发环境的三个组成部分分别是 _____、_____ 和 _____。
2. 在 JDK 环境安装中, 配置环境变量的目的是让操作系统能够找到 _____。
3. JAR 是 Java Archive 的缩写, 它是一种 _____ 文件格式。
4. 可执行 JAR 包的特性是它包含了 _____, 使得它可以直接运行。
5. JAR 文件结构中, 用于指定包含主类信息的文件是 _____。

三、简答题

1. 解释 JDK、JRE、JVM 之间的关系。
2. 简要描述 JAR 包的作用, 以及在 Java 开发中经常使用它的原因。
3. 解释 Java 程序的执行和编译过程。
4. 在安装 Java 开发环境时, 为什么需要配置环境变量?

四、操作题

使用 `jd-gui` 工具将 JAR 包反编译。