第1章 Java开发入门

- JDK的安装与使用
- 编写第一个Java程序
- 系统环境变量配置
- Java程序运行机制
- Eclipse安装与使用
- IntelliJ IDEA安装与使用

1.1.1 Java概述

计算机语言

计算机语言（Computer Language）是人与计算机之间通信的语言，它主要由一些指令组成，这些指令包括数字、符号和语法等内容，程序员可以通过这些指令指挥计算机进行工作。

计算机语言的种类非常多，总的来说可以分成机器语言、汇编语言、高级语言三大类。

1.1.1 Java概述

机器语言都是由二进制的0和1组成的编码，不便于记忆和识别。

汇编语言采用了英文缩写的标识符，容易识别和记忆；

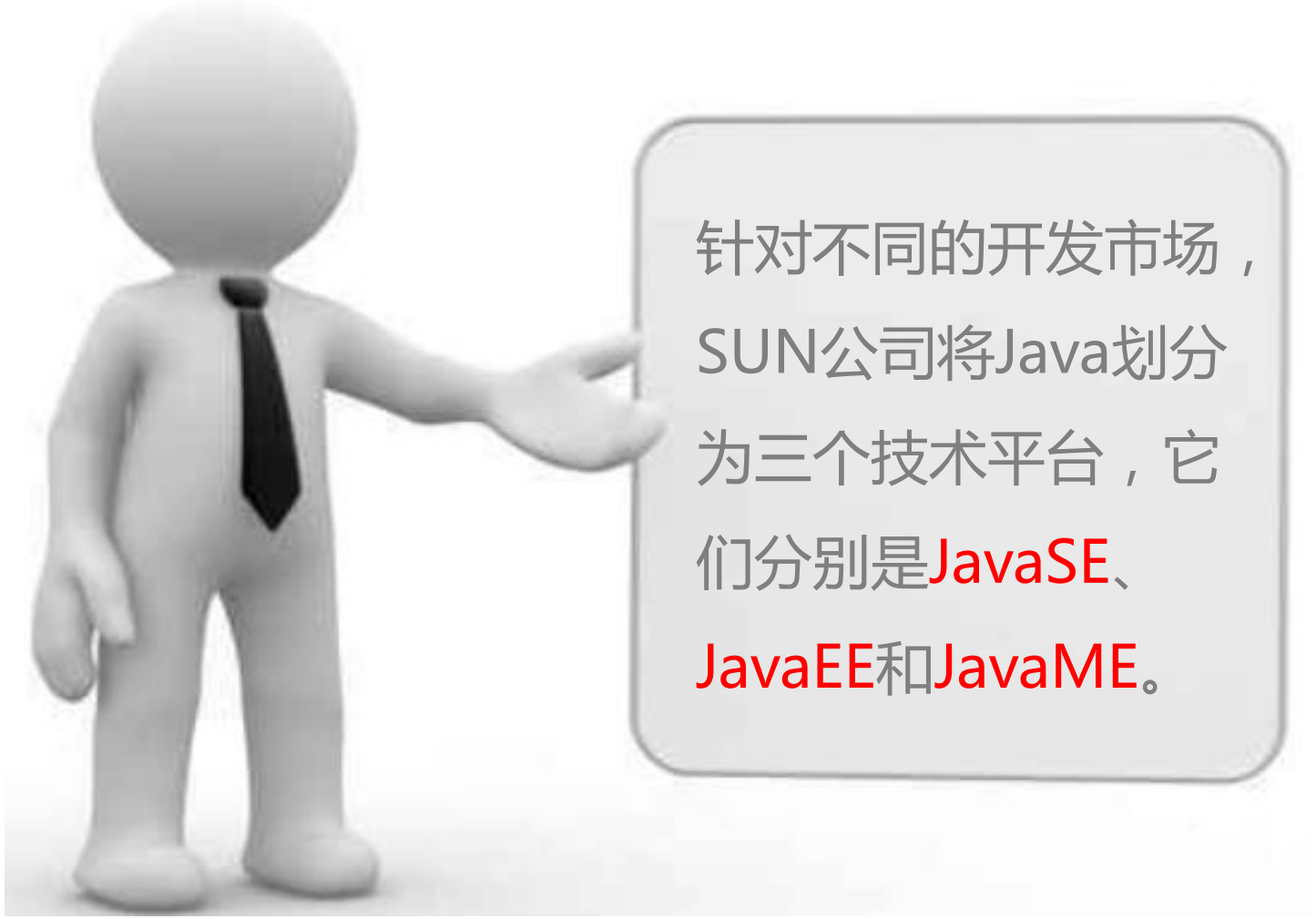
高级语言采用接近于人类的自然语言进行编程，进一步简化了程序编写的过程。

因此，目前编程语言大多是高级语言。

1.1.1 Java概述

Java是一种高级计算机语言，它是由SUN公司（已被Oracle公司收购）于1995年5月推出的一种可以编写跨平台应用软件、完全面向对象的程序设计语言。Java语言简单易用、安全可靠，自问世以来，与之相关的技术和应用发展得非常快。在计算机、移动电话、家用电器等领域中，Java技术无处不在。

1.1.1 Java概述


针对不同的开发市场，SUN公司将Java划分为三个技术平台，它们分别是JavaSE、JavaEE和JavaME。

1.1.1 Java概述

- Java SE（Java Platform Standard Edition）标准版，是为开发普通桌面和商务应用程序提供的解决方案。JavaSE是三个平台中最核心的部分，JavaEE和JavaME都是从JavaSE的基础上发展而来的，JavaSE平台中包括了Java最核心的类库，如集合、IO、数据库连接以及网络编程等。

1.1.1 Java概述

- Java EE(Java Platform Enterprise Edition) 企业版，是为开发企业级应用程序提供的解决方案。Java EE可以被看作一个技术平台，该平台用于开发、装配以及部署企业级应用程序，主要包括Servlet、JSP、JavaBean、JDBC、EJB、Web Service等技术。

1.1.1 Java概述

- Java ME(Java Platform Micro Edition) 小型版，是为开发电子消费产品和嵌入式设备提供的解决方案。JavaME主要用于小型数字电子设备上软件程序的开发。例如，为家用电器增加智能化控制和联网功能，为手机增加新的游戏和通讯录管理功能。此外，Java ME还提供了HTTP等高级Internet协议，使移动电话能以Client/Server方式直接访问Internet的全部信息，提供高效率的无线交流。

1.1.2 Java语言的特点

1．简单性

Java语言是一种相对简单的编程语言，它通过提供最基本的方法完成指定的任务。程序设计者只需理解一些基本的概念，就可以用它编写出适用于各种情况的应用程序。Java丢弃了C++中很难理解的运算符重载、多重继承等概念；特别是Java语言使用引用代替指针，并提供了自动的垃圾回收机制，使程序员不必担忧内存管理。

1.1.2 Java语言的特点

2．面向对象

Java语言提供了类、接口和继承等原语，只支持类之间的单继承，但支持接口之间的多继承，并支持类与接口之间的实现机制（关键字为implements）。Java语言全面支持动态绑定，而C++语言只对虚函数使用动态绑定。总之，Java语言是一个纯粹的面向对象程序设计语言。

1.1.2 Java语言的特点

3.安全性

Java安全可靠，例如，Java的存储分配模型可以防御恶意代码攻击。此外，Java没有指针，因此外界不能通过伪造指针指向存储器。更重要的是，Java编译器在编译程序时，不显示存储安排决策，程序员不能通过查看声明猜测出类的实际存储安排。Java程序中的存储是在运行时由Java解释程序决定。

1.1.2 Java语言的特点

4．跨平台性

Java通过JVM（虚拟机）以及字节码实现跨平台。Java程序由javac编译器编译成为字节码文件（.class）文件， JVM中的Java解释器会将.class文件翻译成所在平台上的机器码文件，执行对应的机器码文件就可以了。Java程序只要“一次编写，就可到处运行”。

1.1.2 Java语言的特点

5．支持多线程

Java语言支持多线程。所谓多线程可以简单理解为程序中多个任务可以并发执行，多线程可以在很大程度上提高程序的执行效率。

6．分布性

Java是分布式语言，既支持各种层次的网络连接，又可以通过Socket类支持可靠的流（stream）网络连接。用户可以产生分布式的客户机和服务器，在这个过程中，网络变成软件应用的分布式运载工具。

1.1.3 Java语言的发展史

Java语言是詹姆士·高斯林发明的，Java的名字来自于一种咖啡的品种名称，所以Java语言的Logo是一杯热气腾腾的咖啡。詹姆斯·高斯林等人于1990年初开发Java语言的雏形，最初被命名为Oak。随着1990年代互联网的发展，Sun公司看见Oak在互联网上应用的前景，于是改进了Oak，并于1995年5月以Java的名称正式发布。

1.1.3 Java语言的发展史

- 1995年5月23日，Java语言诞生。
- 1998年12月8日，JAVA2企业平台J2EE发布。
- 1999年6月，SUN公司发布Java的三个版本：标准版（J2SE）、企业版（J2EE）和微型版（J2ME）。
- 2001年9月24日，J2EE 1.3发布。
- 2002年2月26日，J2SE 1.4发布，自此Java的计算能力有了大幅提升。
- 2004年9月30日，J2SE 1.5的发布成为Java语言发展史上的又一里程碑。为了表示该版本的重要性，J2 SE 1.5更名为Java SE 5.0。

1.1.3 Java语言的发展史

- 2005年6月，JavaOne大会召开，SUN公司公开Java SE 6。此时，Java的各种版本进行了更名，取消了名称中的数字“2”，J2EE更名为Java EE，
- J2SE更名为Java SE，J2ME更名为Java ME。
- 2009年12月，SUN公司发布Java EE 6。
- 2011年7月28日，Oracle公司发布Java SE 7。

1.1.3 Java语言的发展史

- 2014年3月18日，Oracle公司发布Java SE 8(市场主流版本)。
- 2017年9月21日，Oracle公司发布Java SE 9。
- 2018年3月，Oracle公司发布Java SE10。
- 2018年9月，Oracle公司发布Java SE11。
- 2019年3月，Oracle公司发布Java SE12。

2019年9月，Oracle公司发布Java SE13。

1.2 JDK的使用

SUN公司提供了一套Java开发环境，简称JDK(Java Development Kit)。JDK包括Java编译器、Java运行工具、Java文档生成工具、Java打包工具等。

1.2.1 安装JDK

1.开始安装JDK

从Oracle官网下载安装文件“jdk-8u201-windows-x64.exe”，双击文件，进入JDK 8安装界面，如右图。

1.2.1 安装JDK

2．自定义安装功能和路径

在步骤1图中，单击【下一步】按钮进入JDK自定义安装界面，如右图。

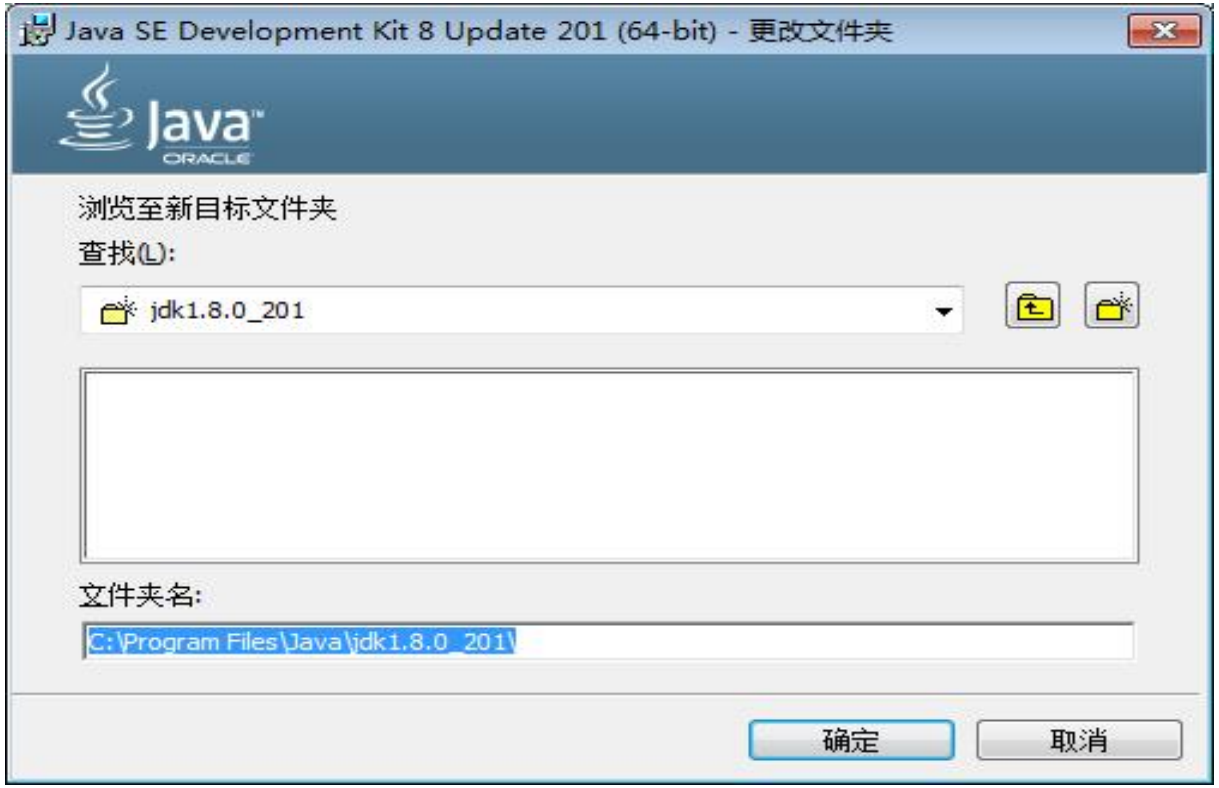
1.2.1 安装JDK

在本步骤图中，左侧有三个功能模块，每个模板具有特定功能如下：

- 开发工具：开发工具是JDK中的核心功能模块，包含一系列可执行程序，如javac.exe、java.exe等，还包含了一个专用的JRE环境。
- 源代码：是Java提供公共API类的源代码。
- 公共JRE：是Java程序的运行环境。由于开发工具中已经包含了一个JRE，因此没有必要再安装公共的JRE环境，此项可以不作选择。

1.2.1 安装JDK

开发人员可以根据自己的需求选择所要安装的模块，本教材选择“开发工具”模块。另外，在图中所示的界面右侧有一个【更改】按钮，单击该按钮进入更改JDK安装目录的界面，如右图。



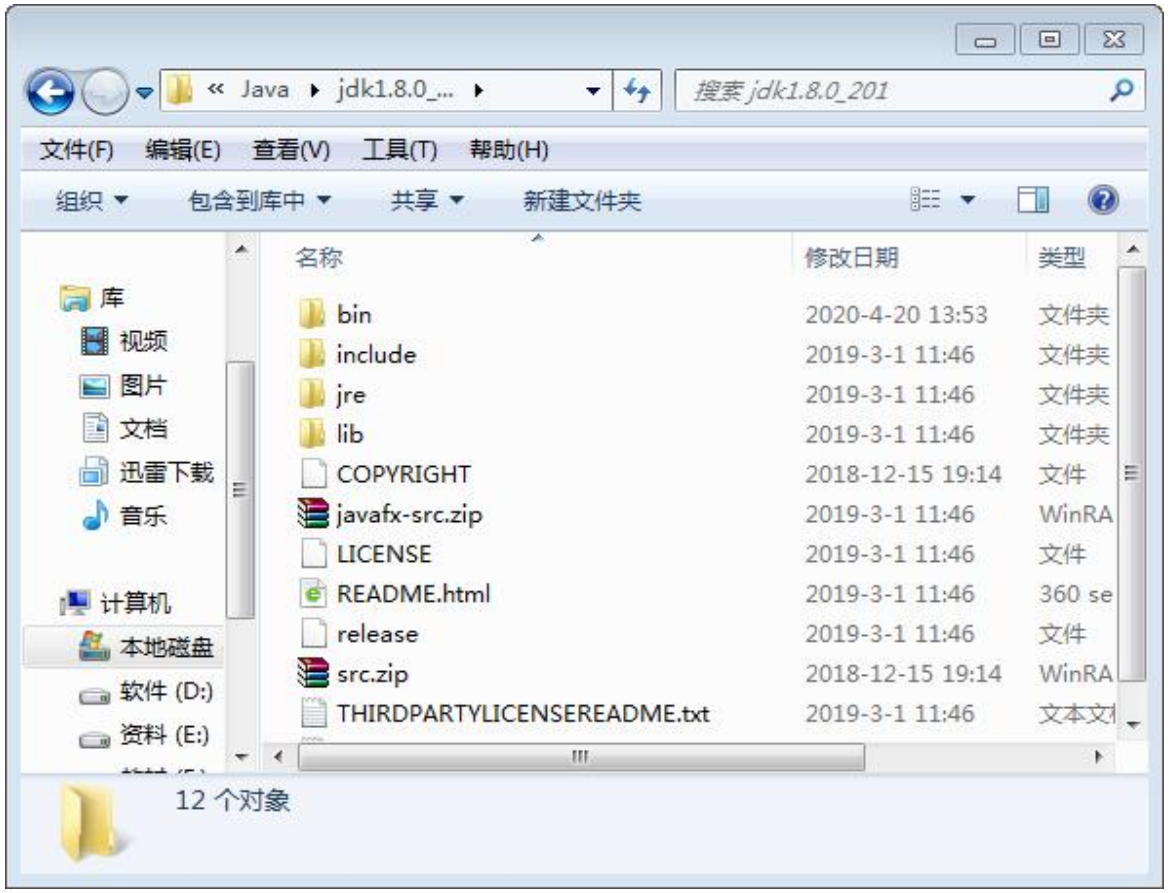
1.2.1 安装JDK

3．完成JDK安装

在步骤2中选择安装路径之后，单击【下一步】按钮开始安装JDK。安装完毕后会进入安装完成界面，如右图。

1.2.2 JDK目录介绍

JDK安装完毕后，会在磁盘上生成一个目录，该目录被称为JDK安装目录，如下图。



1.2.2 JDK目录介绍

（1）bin目录：该目录用于存放一些可执行程序，如javac.exe（Java编译器）、java.exe（Java运行工具）、jar.exe（打包工具）和javadoc.exe（文档生成工具）等。其中最重要的就是javac.exe和java.exe。

1.2.2 JDK目录介绍

- javac.exe是Java编译器，它可以将编写好的Java文件编译成Java字节码文件（可执行的Java程序）。Java源文件的扩展名为.java，如HelloWorld.java。编译后生成对应的Java字节码文件，字节码文件的扩展名为.class，如HelloWorld.class。
- java.exe是Java运行工具，它会启动一个Java虚拟机（JVM）进程，Java虚拟机相当于一个虚拟的操作系统，专门负责运行由Java编译器生成的字节码文件（.class文件）。

1.2.2 JDK目录介绍

（2）db目录：db目录是一个小型的数据库。从JDK 6开始，Java中引入了一个新的成员JavaDB，这是一个纯 Java 实现、开源的数据库管理系统。这个数据库不仅轻便，而且支持JDBC 4所有的规范，在学习JDBC时，不需要再额外地安装一个数据库软件，选择直接使用JavaDB即可。

1.2.2 JDK目录介绍

（3）jre目录：jre是Java Runtime Environment的缩写，意为Java程序运行时环境。该目录是Java运行时环境的根目录，它包含Java虚拟机、运行时的类包、Java应用启动器以及一个bin目录，但不包含开发环境中的开发工具。

（4）include目录：由于JDK是使用C和C++开发的，因此在启动时需要引入一些C语言的头文件，该目录就是用于存放这些头文件的。

1.2.2 JDK目录介绍

（5）lib目录：lib是library的缩写，意为Java类库或库文件，是开发工具使用的归档包文件。

（6）src.zip文件与javafx-src.zip文件：这两个文件中放置的是JDK核心类的源代码和JavaFX源代码，通过这两个文件可以查看Java基础类的源代码。

1.3 第一个Java程序

1．编写Java源文件

在JDK安装目录的bin目录下新建文本文档，重命名为HelloWorld.java。

用记事本打开HelloWorld.java文件，编写一段Java程序。

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

1.3 第一个Java程序

- class是一个关键字，用于定义一个类。在Java中，一个类就相当于一个程序，所有的代码都需要在类中书写。
- HelloWorld是类的名称，简称类名。class关键字与类名之间需要用空格、制表符、换行符等任意的空白字符进行分隔。类名之后要写一对大括号，它定义了当前这个类的作用域。

1.3 第一个Java程序

- 第2~4行代码定义了一个main()方法，该方法是Java程序的执行入口，程序将从main()方法开始执行类中的代码。
- 第3行代码在main()方法中编写了一条执行语句“System.out.println("hello world");”，它的作用是打印一段文本信息并输出到屏幕，执行完这条语句，命令行窗口会输出“hello world”。

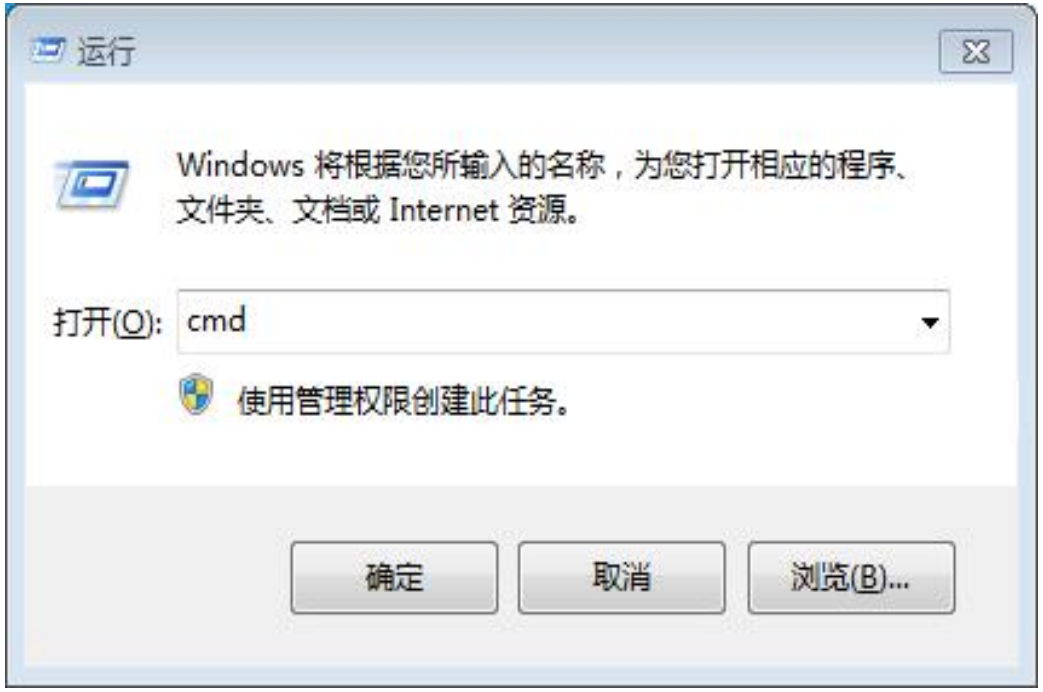
1.3 第一个Java程序

在编写程序时，程序中出现的空格、括号、分号等符号必须采用英文半角格式，否则程序会出错。

1.3 第一个Java程序

2．打开命令行窗口

单击【开始】→【所有程序】→【附件】→【运行】（或者使用快捷键Win+R），打开程序运行窗口，如右图。



1.3 第一个Java程序

在上图的运行窗口中输入“cmd”，单击【确定】按钮打开命令行窗口，如下图。

1.3 第一个Java程序

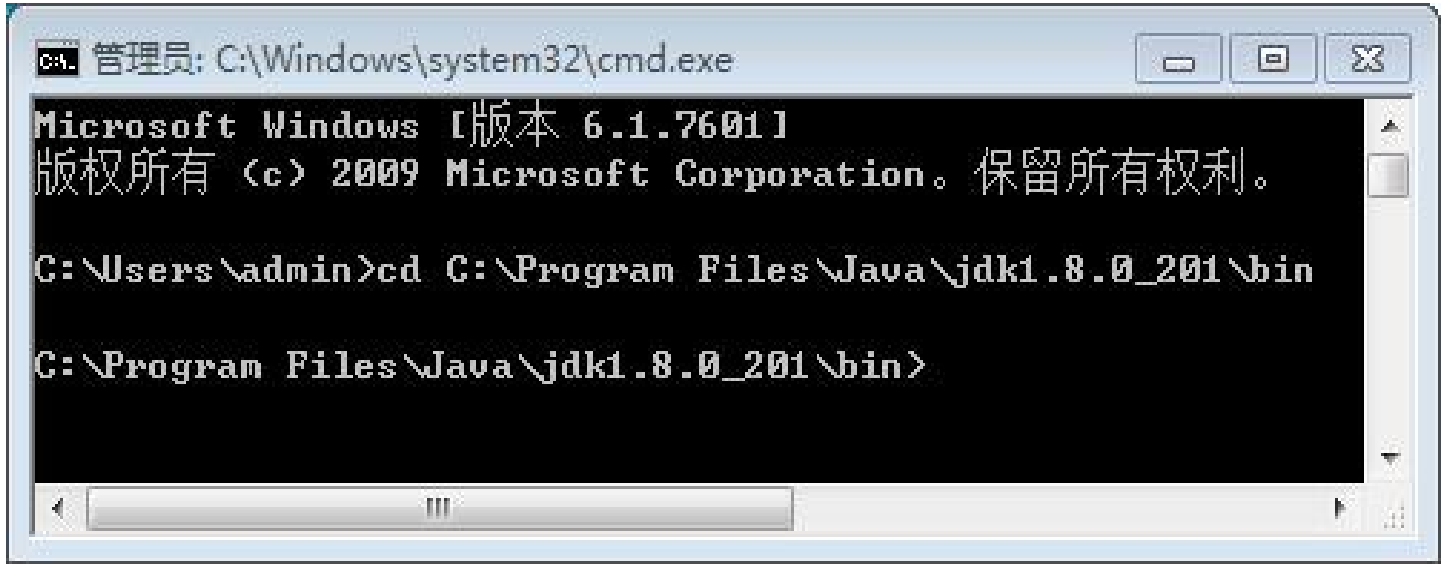
3．进入JDK安装目录的bin目录

编译和运行编写好的Java程序，首先需要进入Java程序所在的目录。例如，编译运行HelloWorld.java程序，需要进入JDK安装目录下的bin目录。在命令行窗口输入下面的命令：

```
cd C:\Program Files\Java\jdk1.8.0_201\bin
```

执行上述命令进入bin目录。

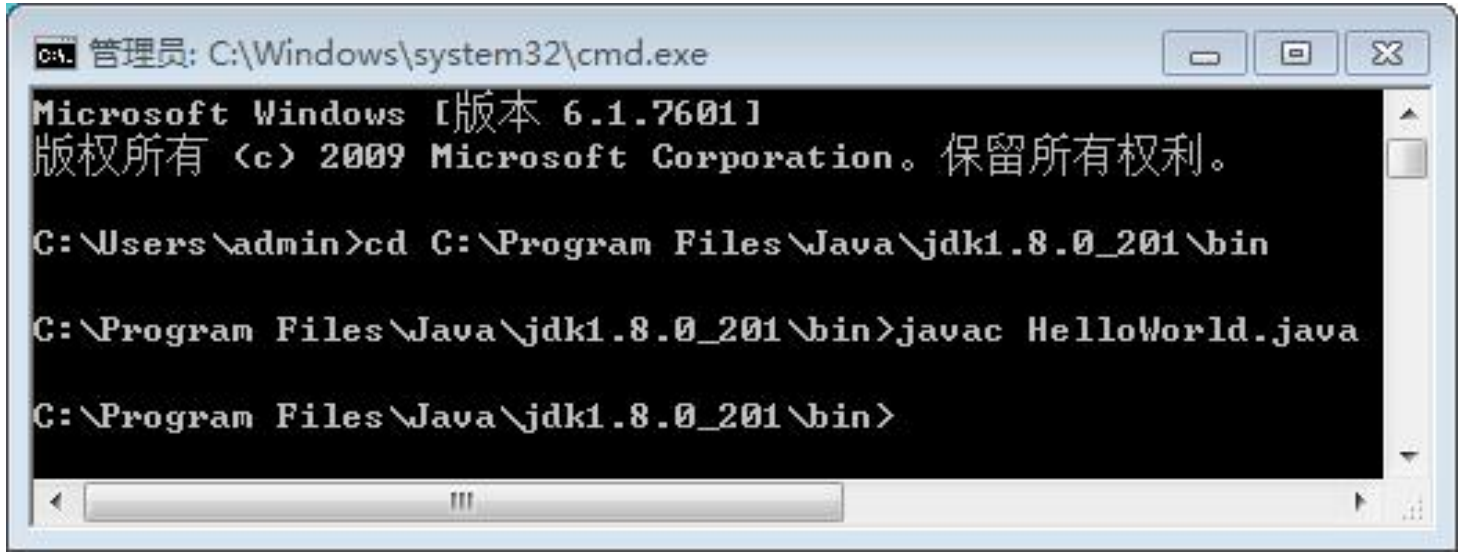
1.3 第一个Java程序



1.3 第一个Java程序

4．编译Java源文件

进入安装JDK的bin目录之后，输入“javac HelloWorld.java”命令，编译HelloWorld.java源文件，如下图。



1.3 第一个Java程序

5．运行Java程序

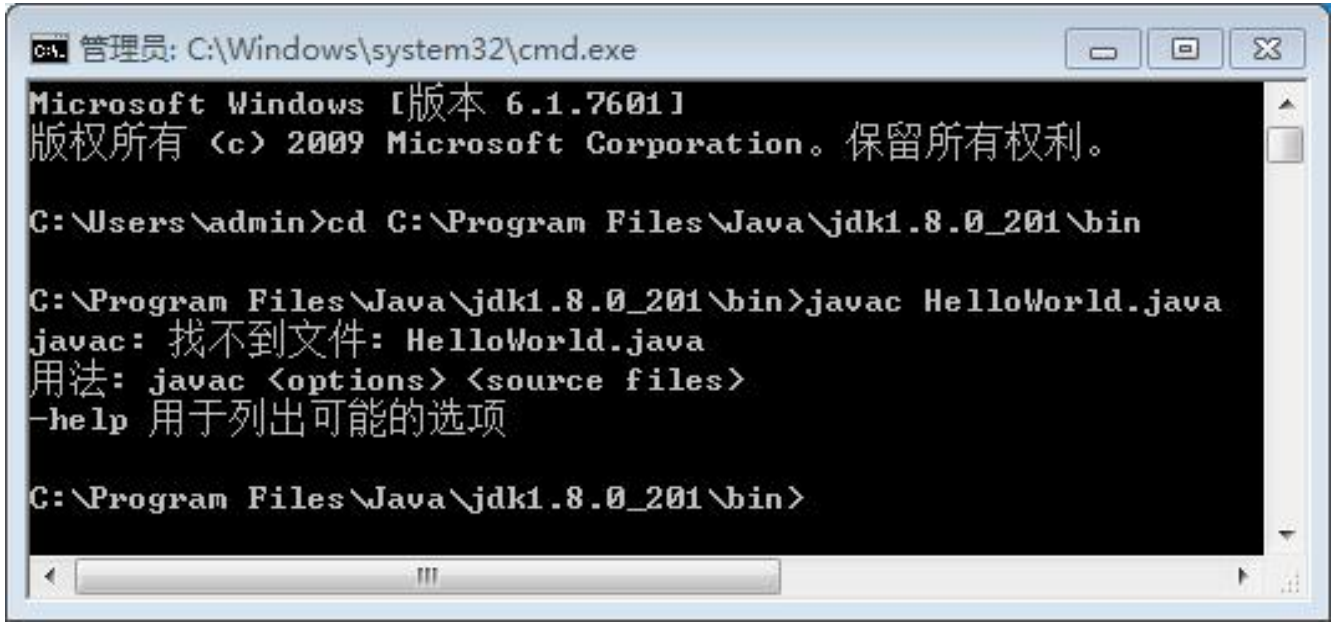
编译文件之后，javac命令执行完毕后，会在bin目录下生成HelloWorld.class字节码文件。在命令行窗口输入“java HelloWorld”命令，运行编译好的字节码文件，运行结果如下图。

1.3 第一个Java程序

在Java程序编写、编译以及运行的过程。第一，在使用javac命令进行编译时，需要输入完整的文件名。第二，在使用java命令运行程序时，需要的是类名，而非完整的文件名。

脚下留心：查看文件扩展名

在使用javac命令编译HelloWorld.java程序时，可能会出现“找不到文件”的错误，如下图。



脚下留心：查看文件扩展名

原因：

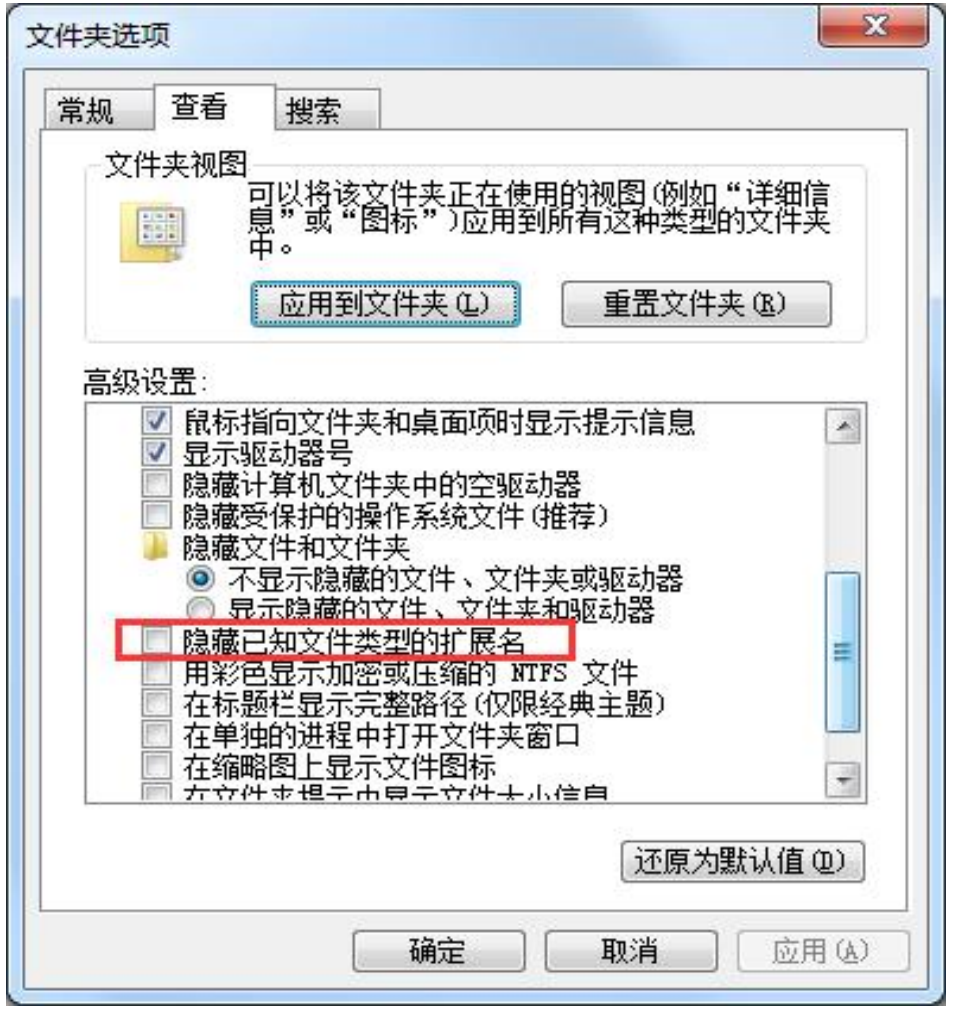
可能是文件的扩展名被隐藏了，虽然文本文件被重命名为“HelloWorld.java”，但实际上该文件的真实文件名为“HelloWorld.java.txt”，文件类型并没有得到修改。

解决方法：

让文件显示扩展名，文件显示出扩展名.txt后，将其重命名为HelloWorld.java即可。

脚下留心：查看文件扩展名

打开Windows的【文件夹选项】，在“高级设置”一栏中将“隐藏已知文件类型的扩展名”选项前面的“√”取消，单击【确定】按钮，如右图。



1.4.1 path环境变量

path环境变量用于保存一系列命令（可执行程序）路径，每个路径之间以分号分隔。当在命令行窗口运行一个可执行文件时，操作系统首先会在当前目录下查找是否存在该文件，如果未找到，操作系统会继续在path环境变量中定义的路径下寻找这个文件，如果仍未找到，系统会报错。

1.4.1 path环境变量

例如，在命令行窗口使用“javac”命令，系统提示错误，如下图。

1.4.1 path环境变量

错误原因：提示可以看出，系统没有找到javac命令。在命令行窗口输入“set path”命令可以查看当前系统的path环境变量，是否包含了javac命令所在目录。如果path环境变量没有包含javac命令所在目录，可以通过下面的命令将javac命令所在目录添加到path环境变量中。

```
set path=%path%;C:\Program Files\Java\jdk1.8.0_201\bin
```

1.4.1 path环境变量

在上述命令中，“%path%”表示引用原有的path环境变量；“C:\Program Files\Java\jdk1.7.0_60\bin” 表示javac命令所在的目录。整行命令的作用就是在原有的path环境变量值中添加javac命令所在的目录。

执行完成上述命令，再次使用set path命令查看当前系统的path环境变量，就会发现javac命令所在目录就包含在了path环境变量，此时再执行javac命令就不会再提示找不到。

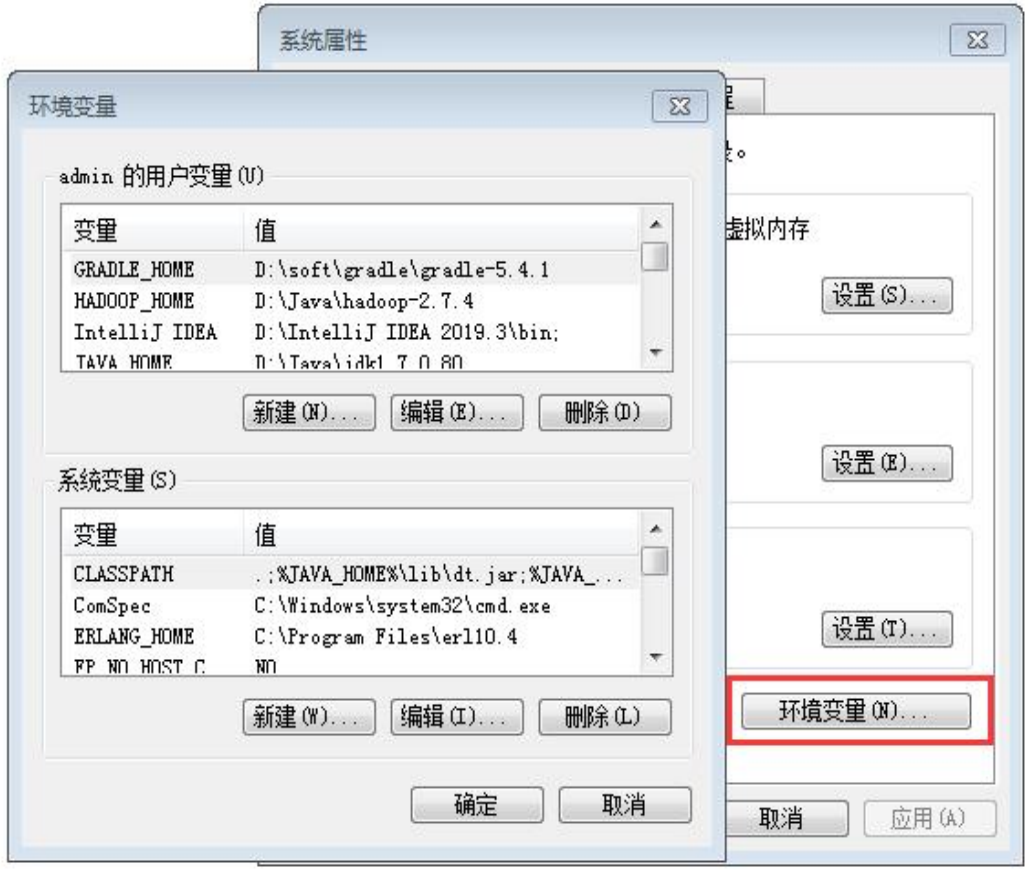
1.4.1 path环境变量

在命令行窗口中，对环境变量进行任何修改只对当前窗口有效，一旦关闭窗口，所有的设置都会失效。如果要让环境变量永久生效，就需要在系统中对环境变量进行配置，让Windows系统永久性地保存所配置的环境变量。

1.4.1 path环境变量

1．查看Windows系统属性中的环境变量

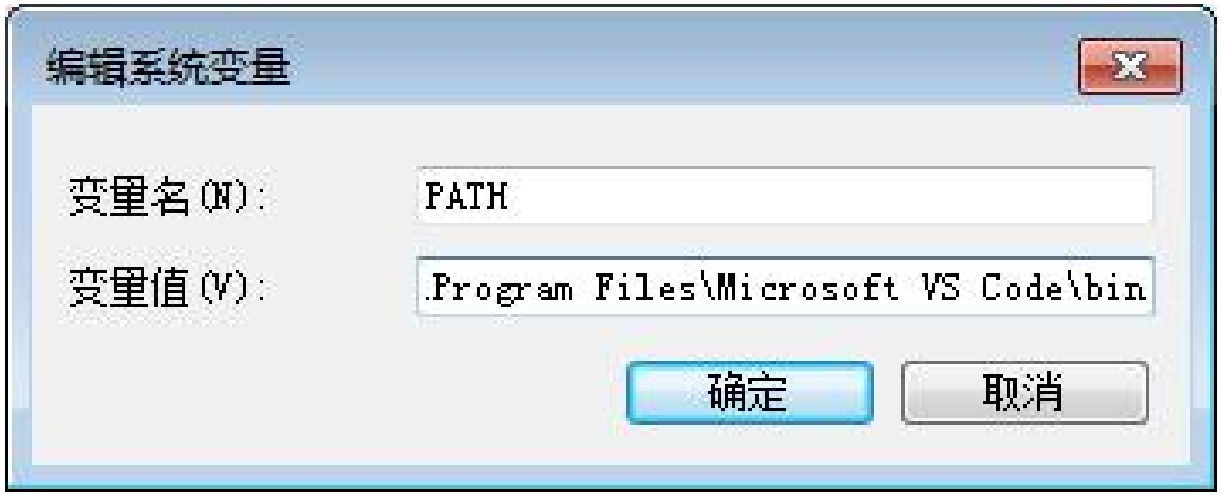
右键单击桌面上的【计算机】→【属性】，在弹出的【系统】窗口左边选择【高级系统设置】选项，弹出系统属性窗口，在系统属性窗口的【高级】选项卡下单击【环境变量】按钮，弹出【环境变量】窗口，如右图。



1.4.1 path环境变量

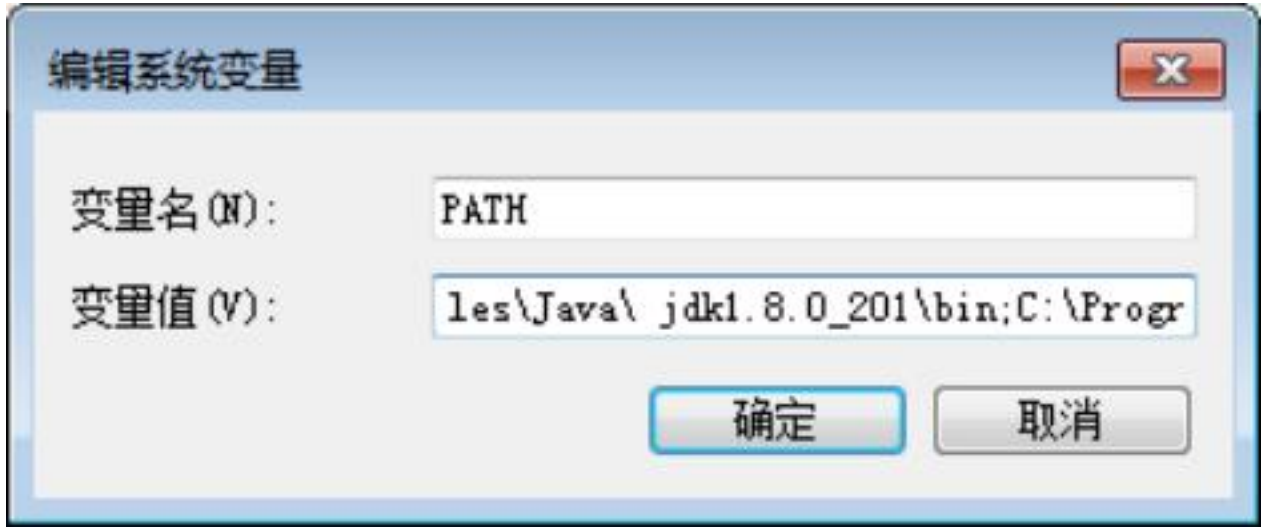
2．设置path系统环境变量

在步骤1图中，在【系统变量】区域选中名为“PATH”的系统变量，单击【编辑】按钮，打开【编辑系统变量】窗口，如下图。



1.4.1 path环境变量

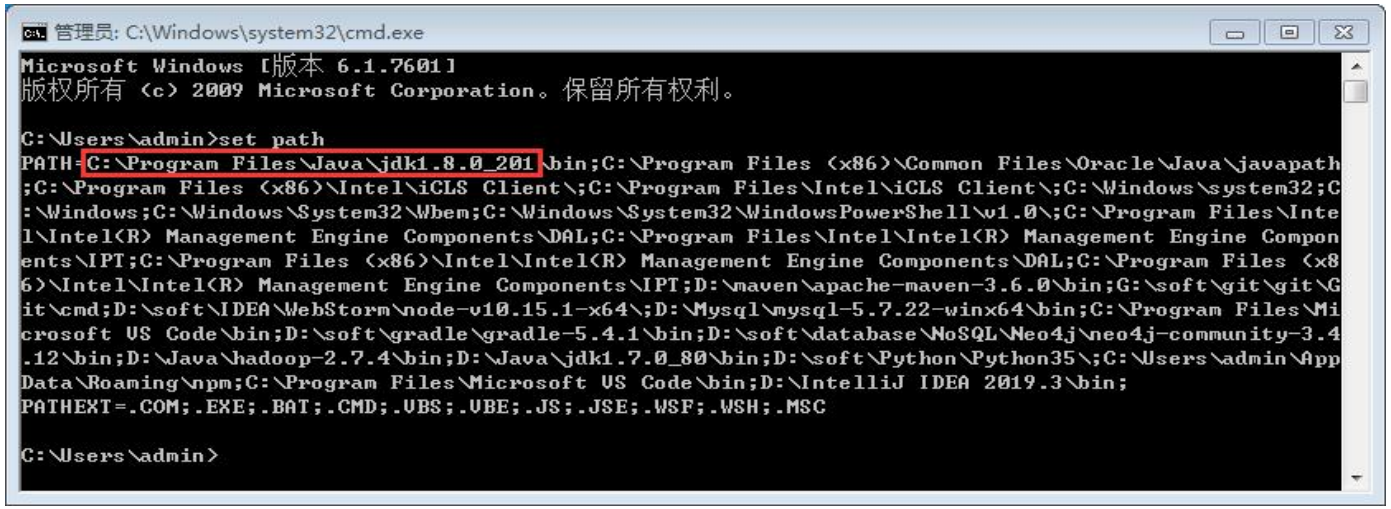
在上图的“变量值”文本区域内的开始处，添加“javac”命令所在的目录路径“C:\Program Files\Java\ jdk1.8.0_201\bin”，并在路径后面用英文半角分号(;)结束，将其与后面的路径隔开，如下图。



1.4.1 path环境变量

3．查看和验证设置的path系统环境变量

在上图中，添加完成后，依次单击所有打开窗口的【确定】按钮，完成path系统环境变量的设置。此时，打开命令行窗口，执行“set path”命令，查看设置后的path变量的变量值，如下图。



1.4.1 path环境变量

在上图中的环境变量的第一行，已经显示出了javac命令的路径信息。在命令行窗口中执行javac命令，如果能正常地显示帮助信息，说明系统path环境变量配置成功，这样系统就永久性地保存了path环境变量的设置。

1.4.2 classpath环境变量

classpath环境变量用于保存一系列类包的路径，它和path环境变量的查看与配置方式完全相同。当Java虚拟机需要运行一个类时，会在classpath环境变量定义的路径下寻找所需的class文件和类包。

1.4.2 classpath环境变量

打开命令行窗口，进入C盘根目录下，执行“java HelloWorld”命令，运行之前编译好的HelloWorld程序，结果会报错，如下图。

1.4.2 classpath环境变量

错误原因：

Java虚拟机在运行程序时无法找到HelloWorld.class文件，即在C盘根目录下没有HelloWorld.class文件。

解决方法：

对classpath环境变量进行设置，保存HelloWorld.class文件路径。在命令行窗口输入下面的命令：

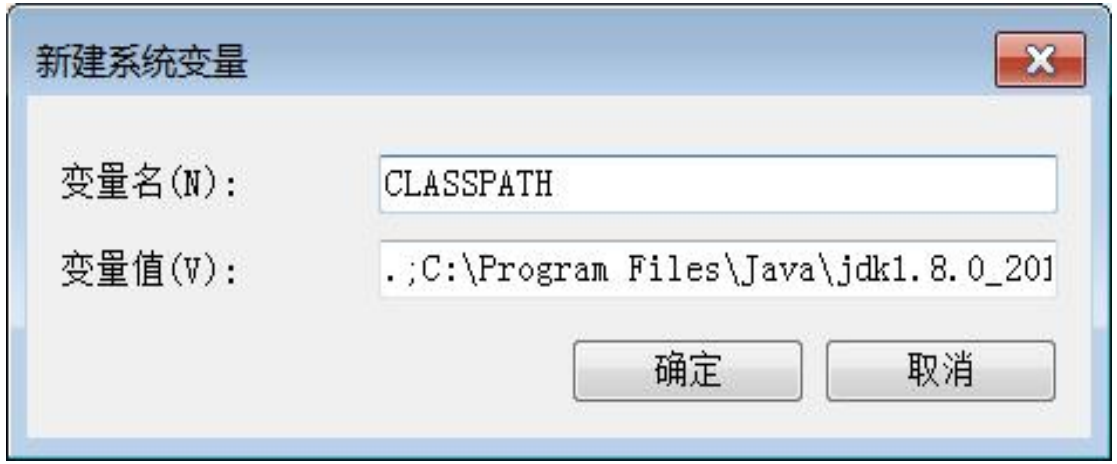
```
set classpath=C:\Program Files\Java\jdk1.8.0_201\bin
```

1.4.2 classpath环境变量

classpath除了可以指定类的路径外，还可以指定运行Java程序所需的标准类包的路径。JDK提供的标准类包有两个，分别是dt.jar和tools.jar，它们位于JDK安装目录的lib文件夹下。在配置环境变量时，通常会将这两个JAR包配置到classpath中，配置方式非常简单。

1.4.2 classpath环境变量

在【环境变量】窗口中的【系统变量】区域单击【新建】按钮，在弹出的【新建系统变量】窗口中，在变量名的文本区域输入“CLASSPATH”，变量值的文本区域输入dt.jar和tools.jar 两个类包的路径。



1.4.2 classpath环境变量

在设置CLASSPATH变量时，必须在配置路径前添加“.;”（当前目录），用于识别当前目录下的Java类。

1.5 Java运行机制

使用Java语言进行程序设计时，不仅要了解Java语言的特点，还需要了解Java程序的运行机制。Java程序运行时，必须经过编译和运行两个步骤。首先将后缀名为.java的源文件进行编译，生成后缀名为.class的字节码文件。然后Java虚拟机将字节码文件进行解释执行，并将结果显示出来。

1.5 Java运行机制

以HelloWorld.java为例，对Java程序的编译运行过程进行详细地分析：

（1）写HelloWorld.java文件。

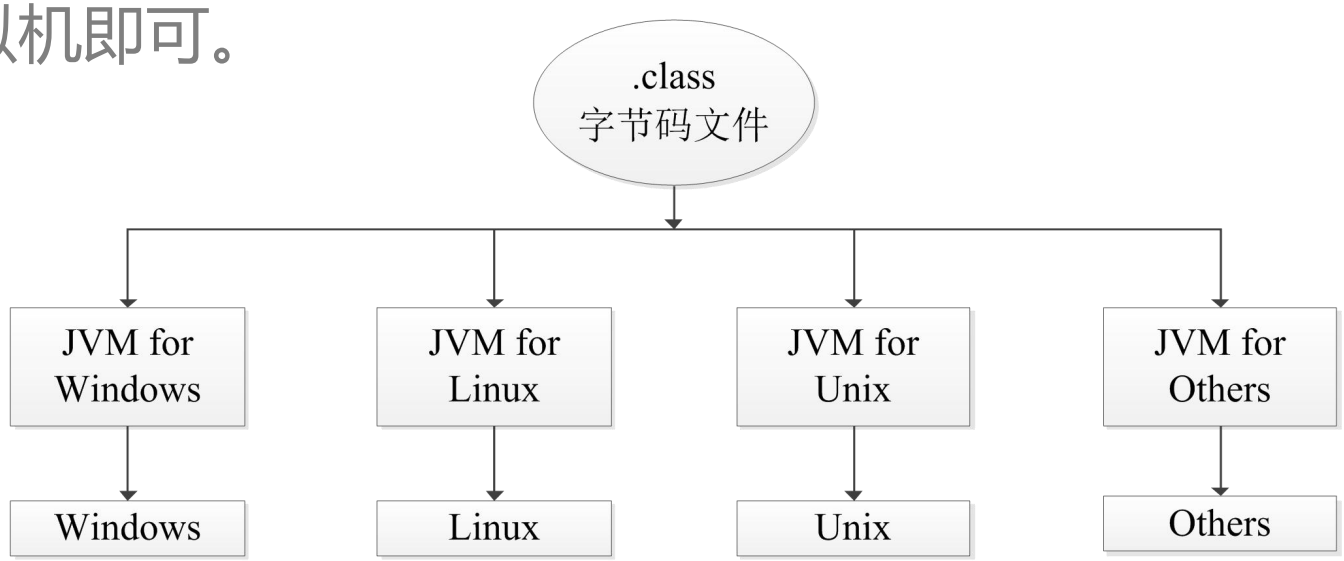
（2）使用“javac HelloWorld.java”命令开启Java编译器，编译HelloWorld.java文件。编译结束后，会自动生成一个名为HelloWorld.class的字节码文件。

1.5 Java运行机制

（3）使用“java HelloWorld”命令启动Java虚拟机运行程序，Java虚拟机首先将编译好的字节码文件加载到内存，这个过程被称为类加载，由类加载器完成。然后Java虚拟机针对加载到内存中的Java类进行解释执行，输出运行结果。

1.5 Java运行机制

通过上面的分析不难发现，Java程序是由虚拟机负责解释执行的，并非操作系统。这样做的好处是可以实现Java程序的跨平台，也就是说，在不同的操作系统上，可以运行相同的Java程序，不同操作系统只需安装不同版本的Java虚拟机即可。



1.5 Java运行机制

Java程序通过Java虚拟机可以达到跨平台特性，但Java虚拟机并不是跨平台的。也就是说，不同操作系统上的Java虚拟机是不同的，即Windows平台上的Java虚拟机不能用在Linux平台上，反之亦然。

1.6.1 Eclipse概述

Eclipse是由蓝色巨人IBM花费巨资开发的一款功能完整且成熟的IDE集成开发环境，它是一个开源的、基于Java的可扩展开发平台，是目前最流行的Java语言开发工具。

Eclipse具有强大的代码编排功能，可以帮助程序开发人员完成语法修正、代码修正、补全文字、信息提示等编码工作，大大提高了程序开发的效率。

1.6.1 Eclipse概述

Eclipse的设计思想是“一切皆插件”。就其本身而言，Eclipse只是一个框架和一组服务，所有功能都是以插件组件的方式加入到Eclipse框架中实现的。Eclipse作为一款优秀的开发工具，自身附带了一个标准的插件集，其中包括了Java开发工具（JDK）。

1.6.2 Eclipse的下载与启动

1．下载Eclipse开发工具

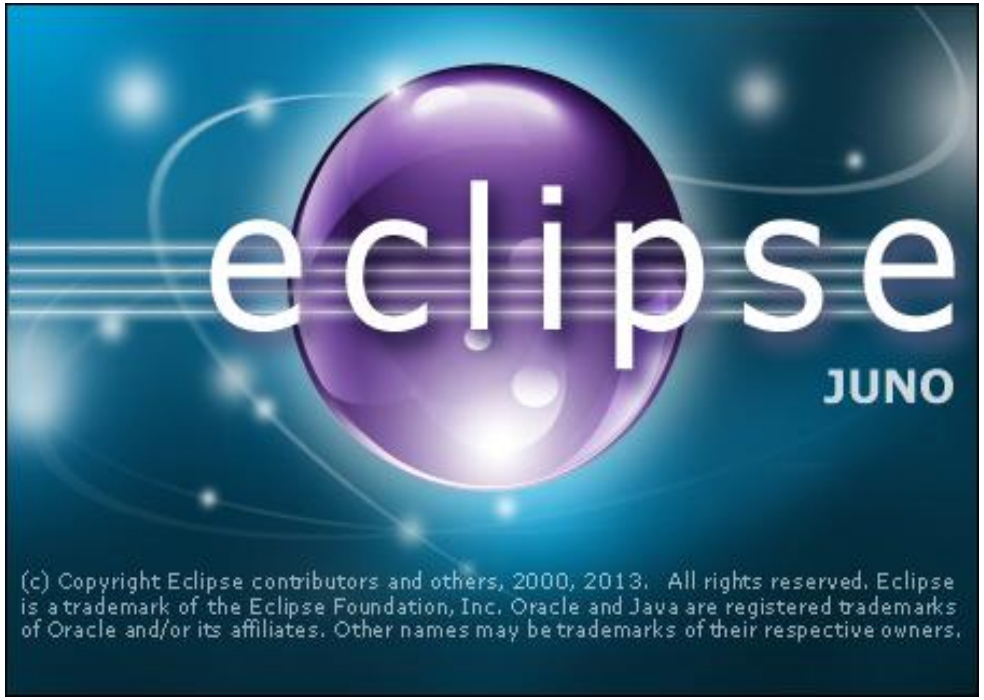
Eclipse是针对Java编程的集成开发环境（IDE），读者可以登录Eclipse官网免费下载，本教材使用的Eclipse版本是Juno Service Release 2。Eclipse安装时只需将下载好的ZIP包解压保存到指定目录下（例如D:\eclipse）就可以使用了。

1.6.2 Eclipse的下载与启动

2．启动Eclipse开发工具

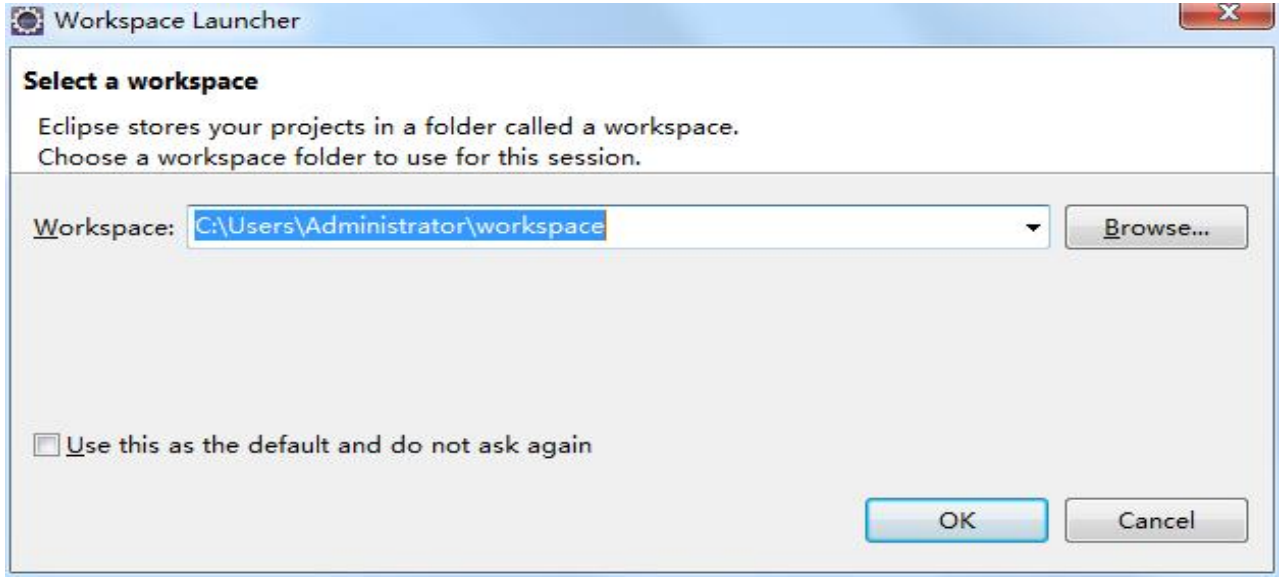
完成Eclipse的解压之后，接下来就可以启动Eclipse开发工具，具体步骤如下：

（1）在Eclipse解压文件中运行eclipse.exe文件，会出现启动界面如右图。



1.6.2 Eclipse的下载与启动

（2）Eclipse启动完成后会弹出一个对话框，提示选择工作空间（Workspace），工作空间用于保存Eclipse创建的项目和相关设置。可以使用Eclipse提供的默认路径为工作空间，也可以单击【Browse】按钮更改路径。如右图。

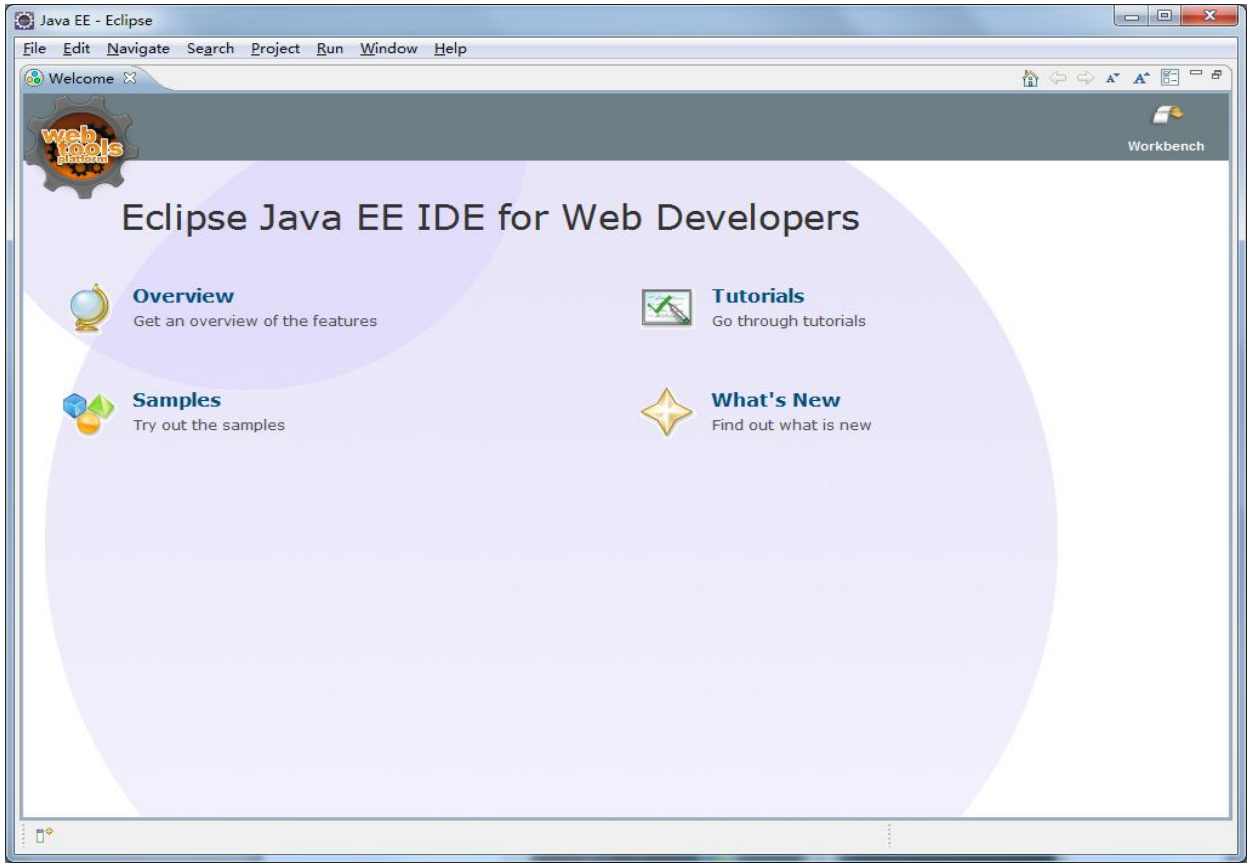


1.6.2 Eclipse的下载与启动

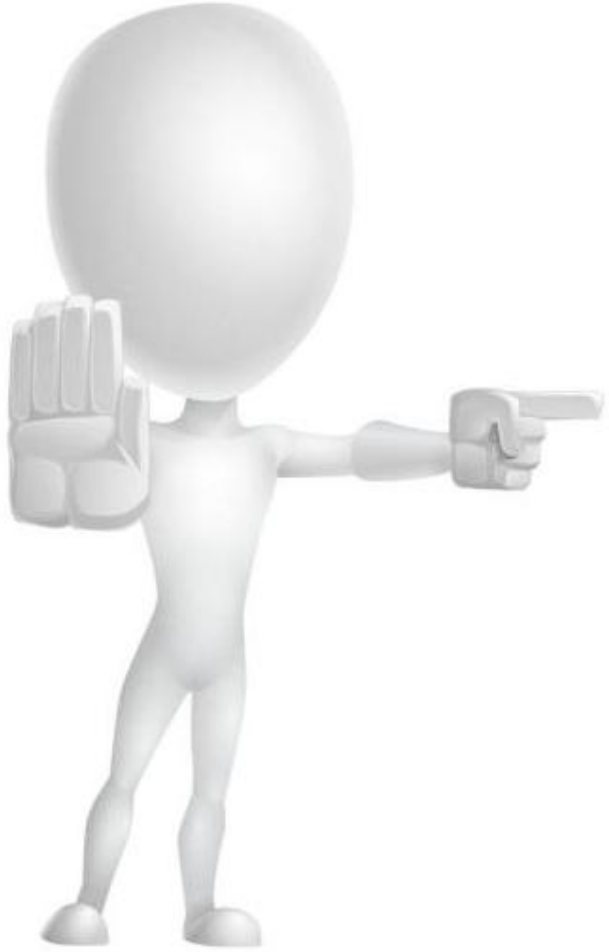
Eclipse每次启动都会出现选择工作空间的对话框，如果不想每次都选择工作空间，可以勾选上图中【Use this as the default and do not ask again】复选框，这就相当于为Eclipse工具选择了默认的工作空间，再次启动时就不会再出现提示对话框。

1.6.2 Eclipse的下载与启动

（3）在图1-27中，工作空间设置完成后，单击【OK】按钮，进入Eclipse欢迎界面，如右图。



1.6.2 Eclipse的下载与启动



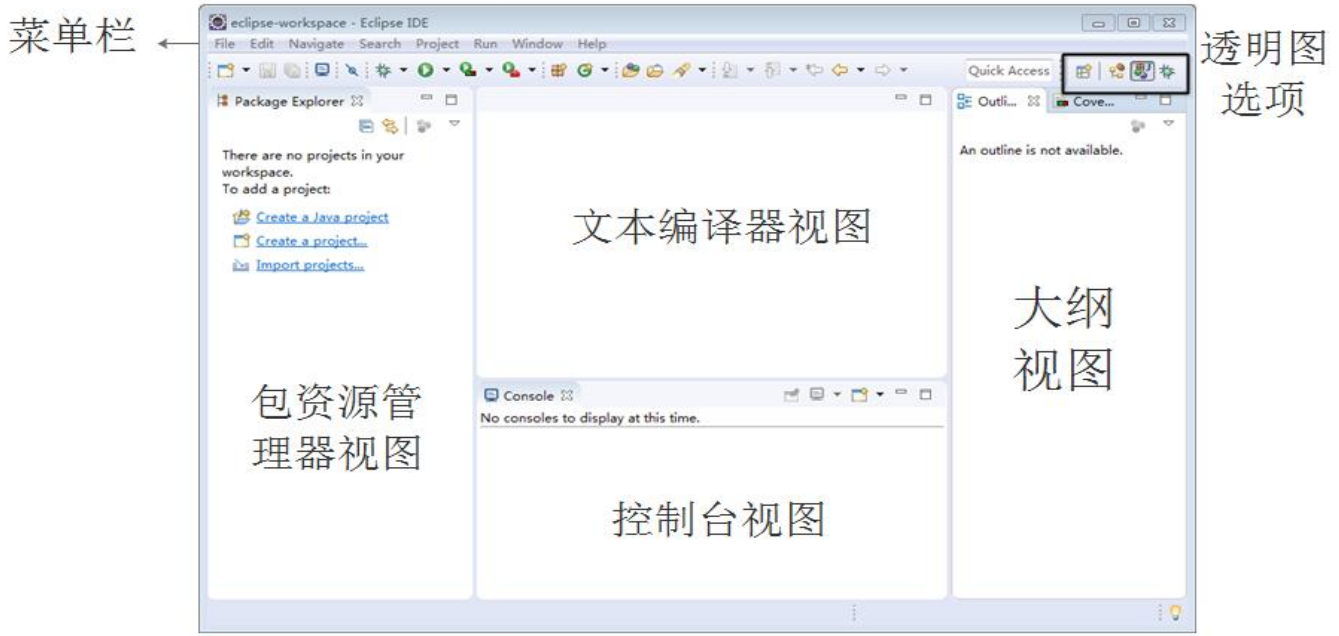
上图所示的欢迎界面中有四个功能图标，含义分别如下所示：

- “Overview” ：概述。
- “Tutorials” ：教程。
- “Samples” ：样本。
- “What’s new” ：新增内容。

1.6.2 Eclipse的下载与启动

3．Eclipse工作台

在上图中关闭欢迎界面窗口，进入Eclipse工作台界面，如下图。



1.6.2 Eclipse的下载与启动

Eclipse工作台主要由标题栏、菜单栏、工具栏、透视图四部分组成。一个工作台中最重要的部分就是透视图。

下面分别介绍Eclipse工作台几种主要视图的作用：

- 包资源管理器视图（Package Explorer）：用于显示项目文件的组成结构。
- 文本编辑器视图（Editor）：用来编写代码的区域。
- 控制台视图（Console）：用于显示程序运行时的输出信息、异常和错误。
- 大纲视图（Outline）：用于显示代码中类的结构。

1.6.2 Eclipse的下载与启动

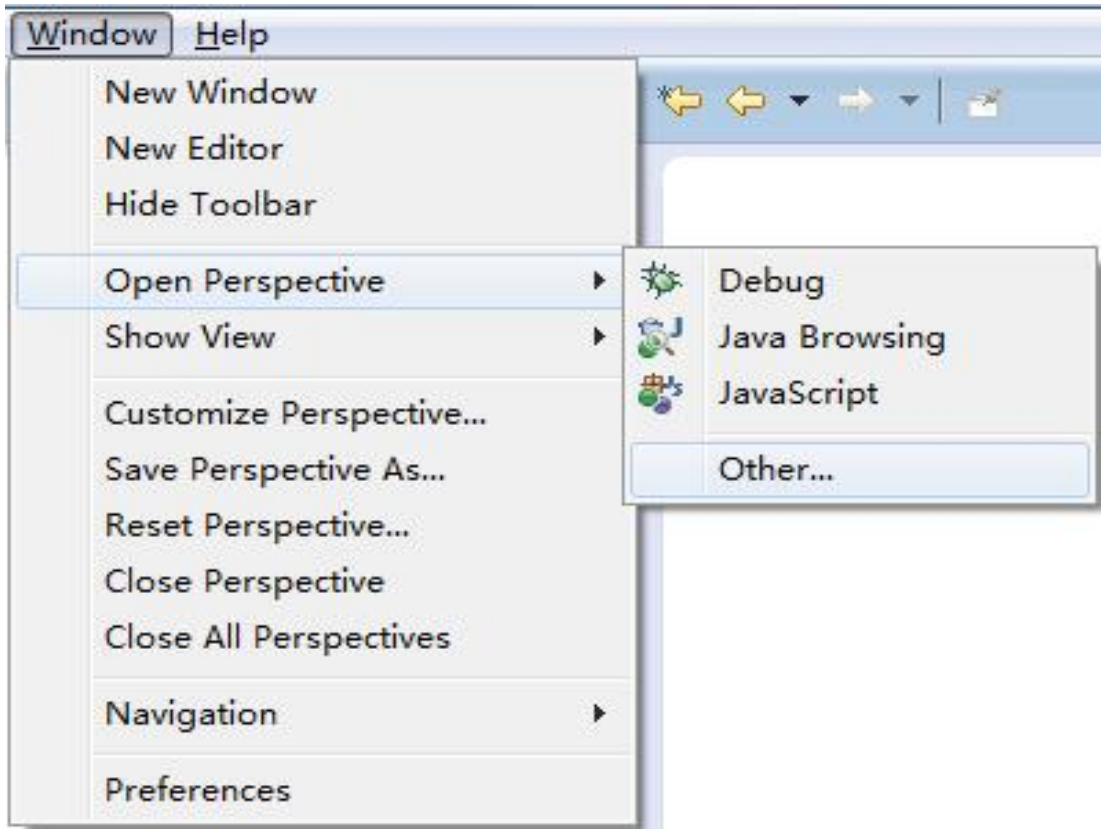
4．Eclipse透视图

透视图（Perspective）是比视图更大的一个概念，用于定义工作台窗口中视图的初始设置和布局，目的在于完成特定类型的任务或使用特定类型的资源。Eclipse提供了几种常用的透视图，如Java透视图、资源透视图、调试透视图、小组同步透视图等。

用户可以通过Elipse工具栏中的透视图按钮 在不同的透视图之间切换。

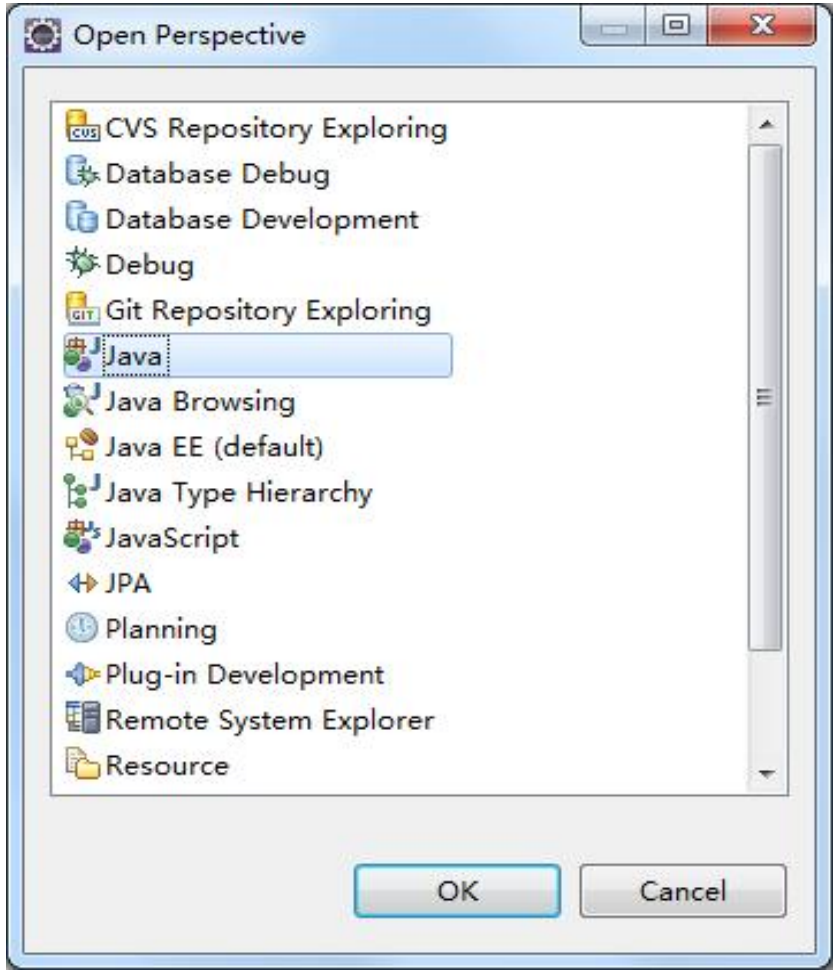
1.6.2 Eclipse的下载与启动

如果要选择进入某一个透视图，可以在菜单栏中选择【Window】→【Open Perspective】→【Other】打开其他透视图，如右图。



1.6.2 Eclipse的下载与启动

选择【Other】选项之后，弹出“Open Perspective”对话框，选择要打开的视图，如右图。

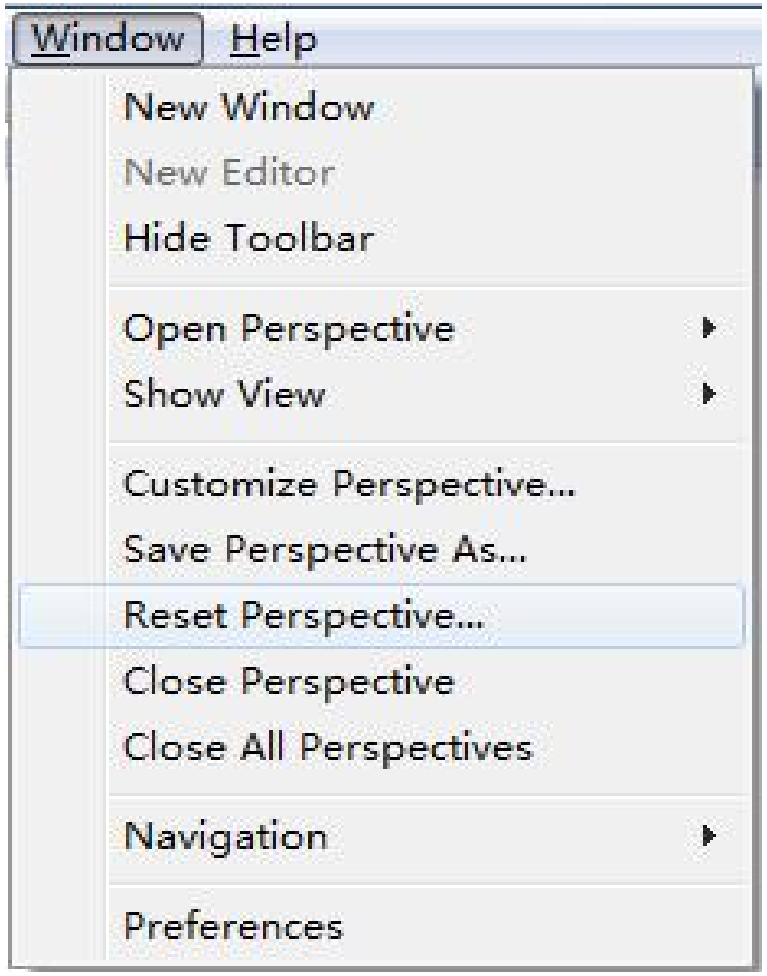


1.6.2 Eclipse的下载与启动

同一时刻只能有一个透视图是活动的，该活动的透视图可以控制哪些视图显示在工作台界面上，并控制这些视图的大小和位置，视图在透视图中的设置更改不会影响编辑器的设置。

1.6.2 Eclipse的下载与启动

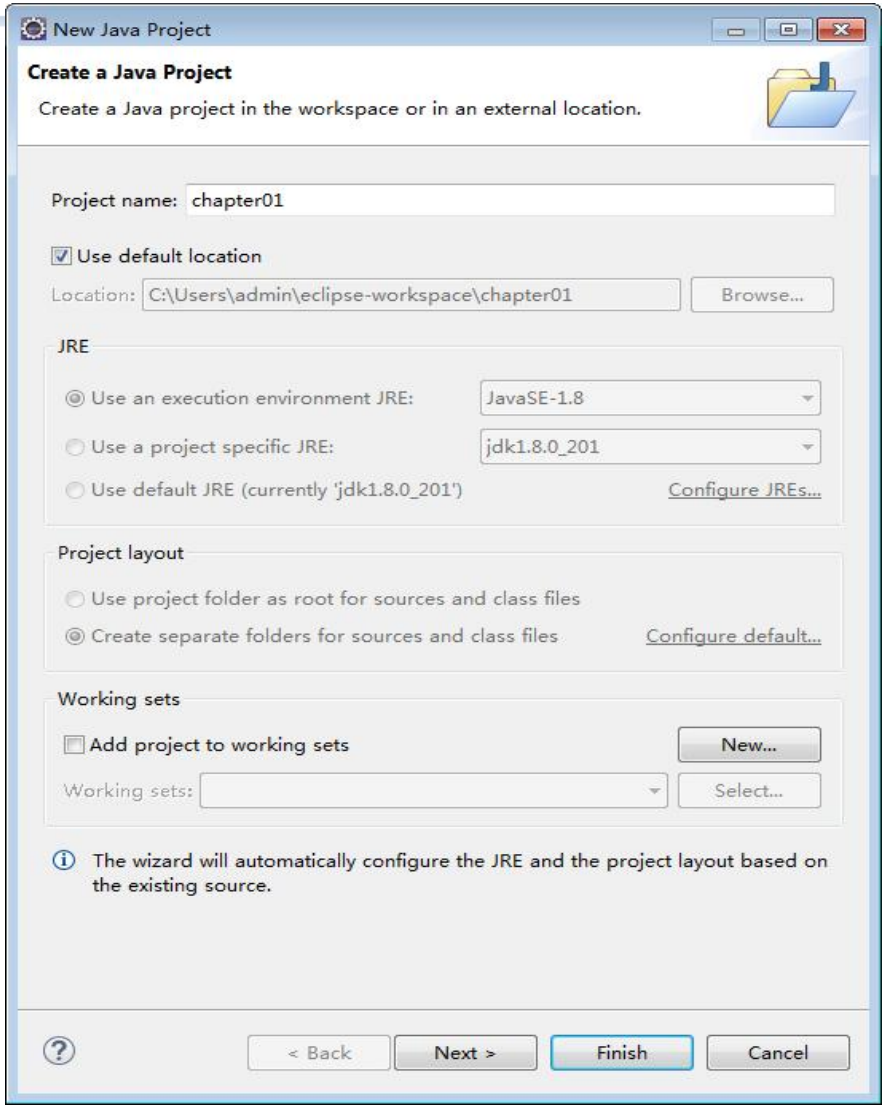
如果失误操作了透视图(Perspective)，例如，关闭了透视图中的包资源管理器视图，可以通过【Window】→【Show View】选择想要打开的视图，也可以重置透视图。在菜单栏选择【Window】→【Reset Perspective】可以重置透视图，将透视图回复至原始状态，如右图。



1.6.3 Eclipse的下载与启动

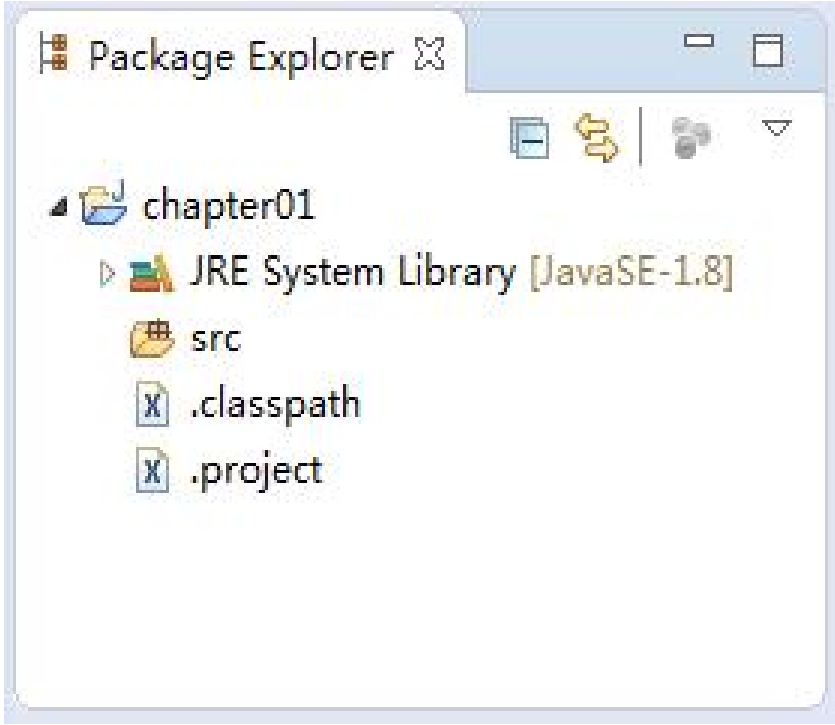
1．创建Java项目

在Eclipse窗口中选择菜单【File】→【New】→【Java Project】，或者在Package Explorer视图中单击鼠标右键，选择菜单【New】→【Java Project】，弹出一个“new Java Project”对话框，如右图。



1.6.3 Eclipse的下载与启动

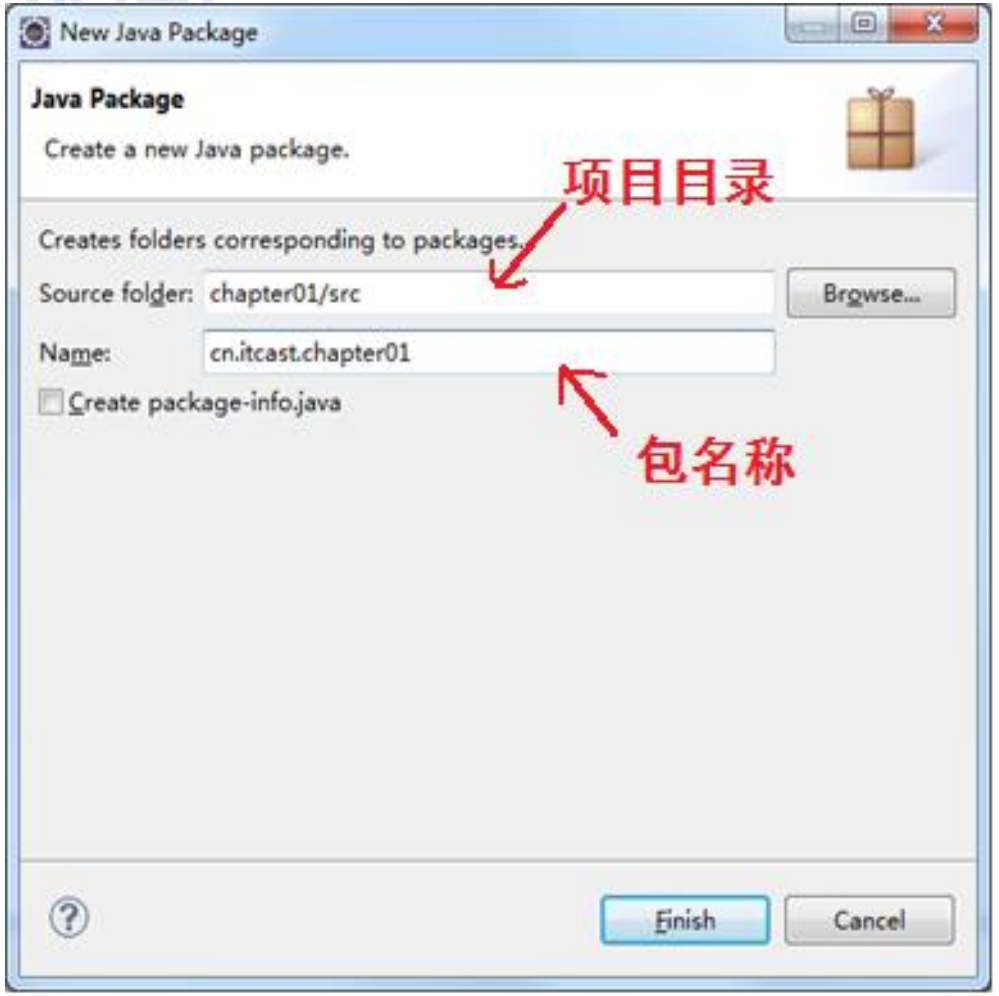
Project name表示项目的名称，在这里将项目命名为chapter01，其余选项保持默认，然后单击【Finish】按钮完成项目的创建。完成项目创建之后，在Package Explorer视图中便会出现一个名称为chapter01的Java项目，如右图。



1.6.3 Eclipse的下载与启动

2．在项目下创建包

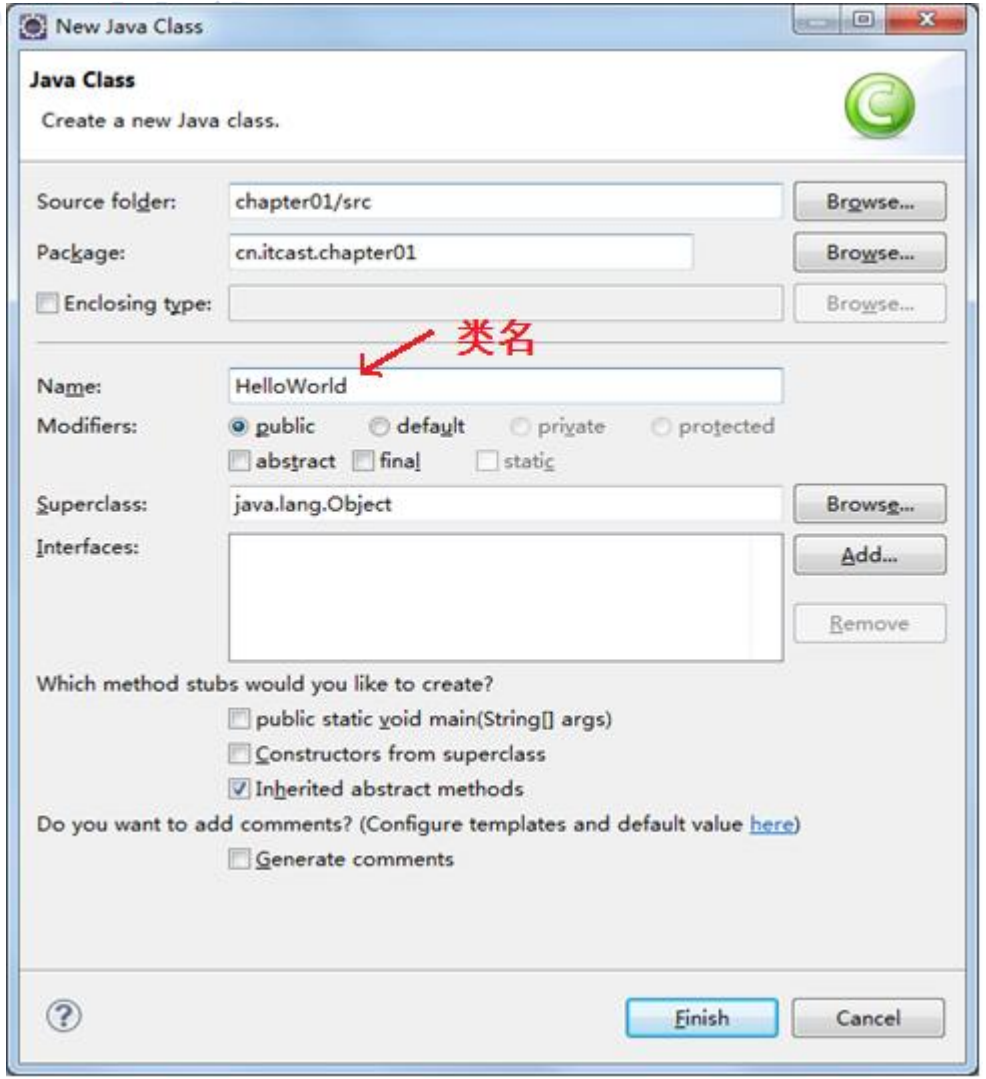
在上图中，鼠标右键单击chapter01项目下的src文件夹，选择【New】 → 【Package】，会弹出一个“New Java Package”对话框，如右图。



1.6.3 Eclipse的下载与启动

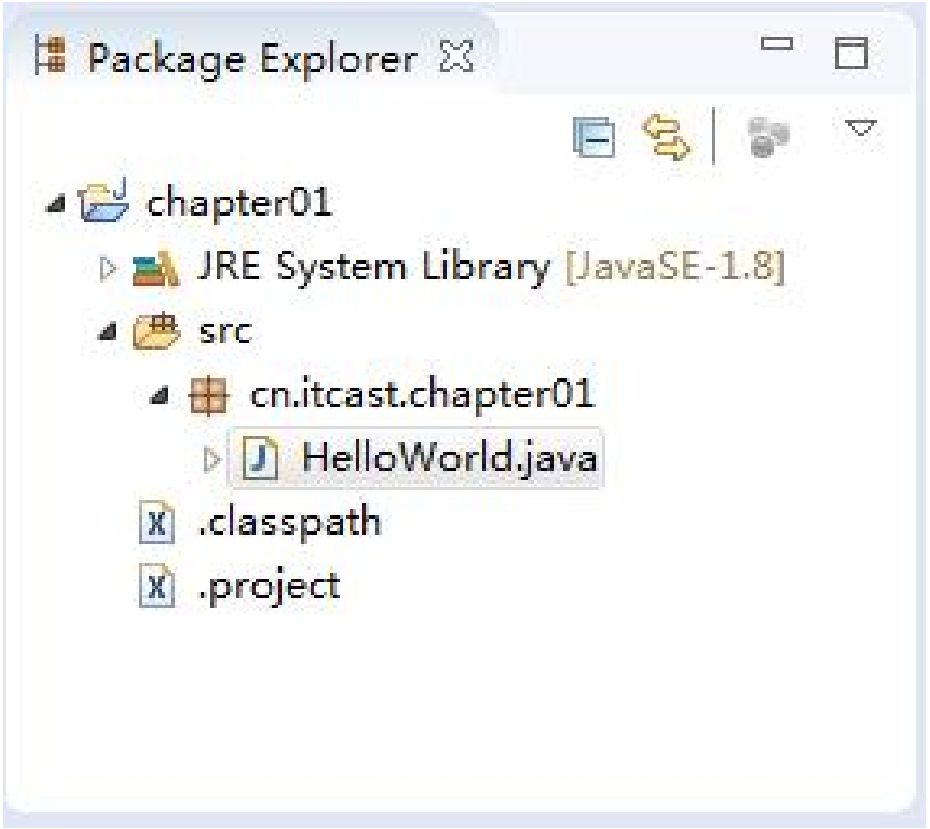
3．创建Java类

鼠标右键单击包名（cn.itcast.chapter01），选择【New】→【Class】，会弹出一个“New Java Class”对话框，如右图。



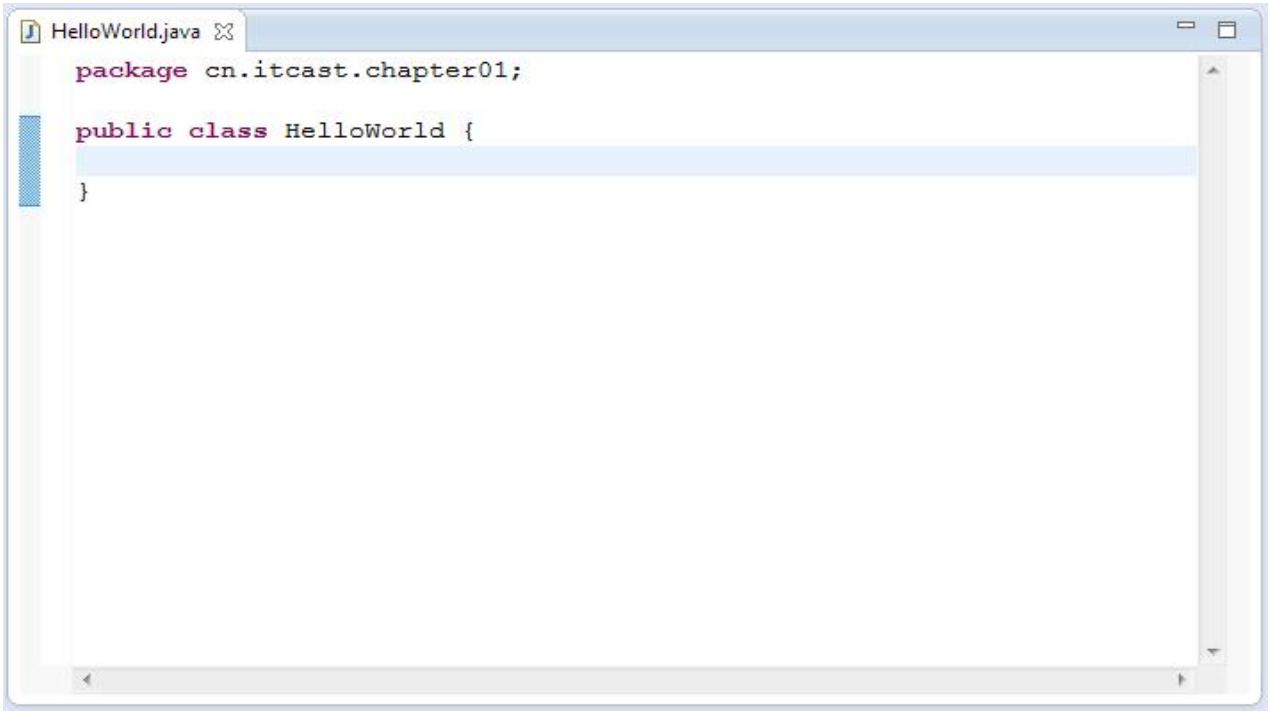
1.6.3 Eclipse的下载与启动

单击【Finish】按钮，完成HelloWorld类的创建。类名创建完成之后，在“cn.itcast.chapter01”包下就会出现一个HelloWorld.java文件，如右图。



1.6.3 Eclipse的下载与启动

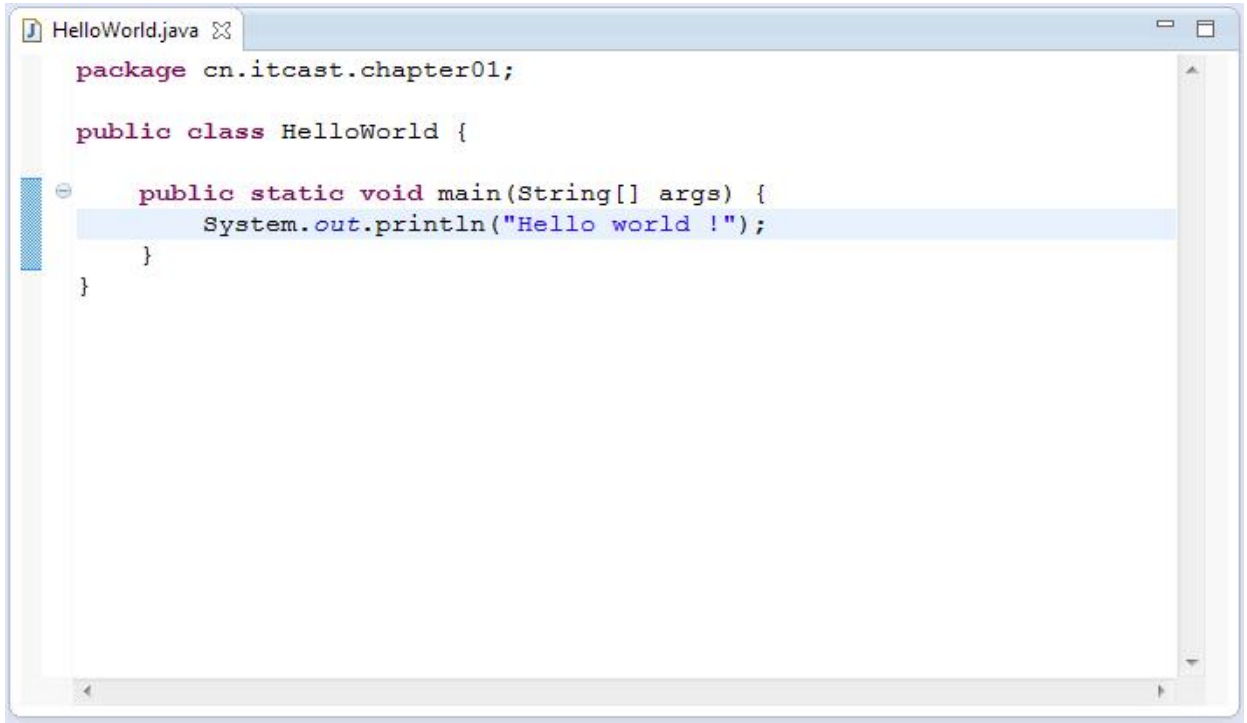
创建好的HelloWorld.java文件会在编辑区域自动打开。



1.6.3 Eclipse的下载与启动

4．编写程序代码

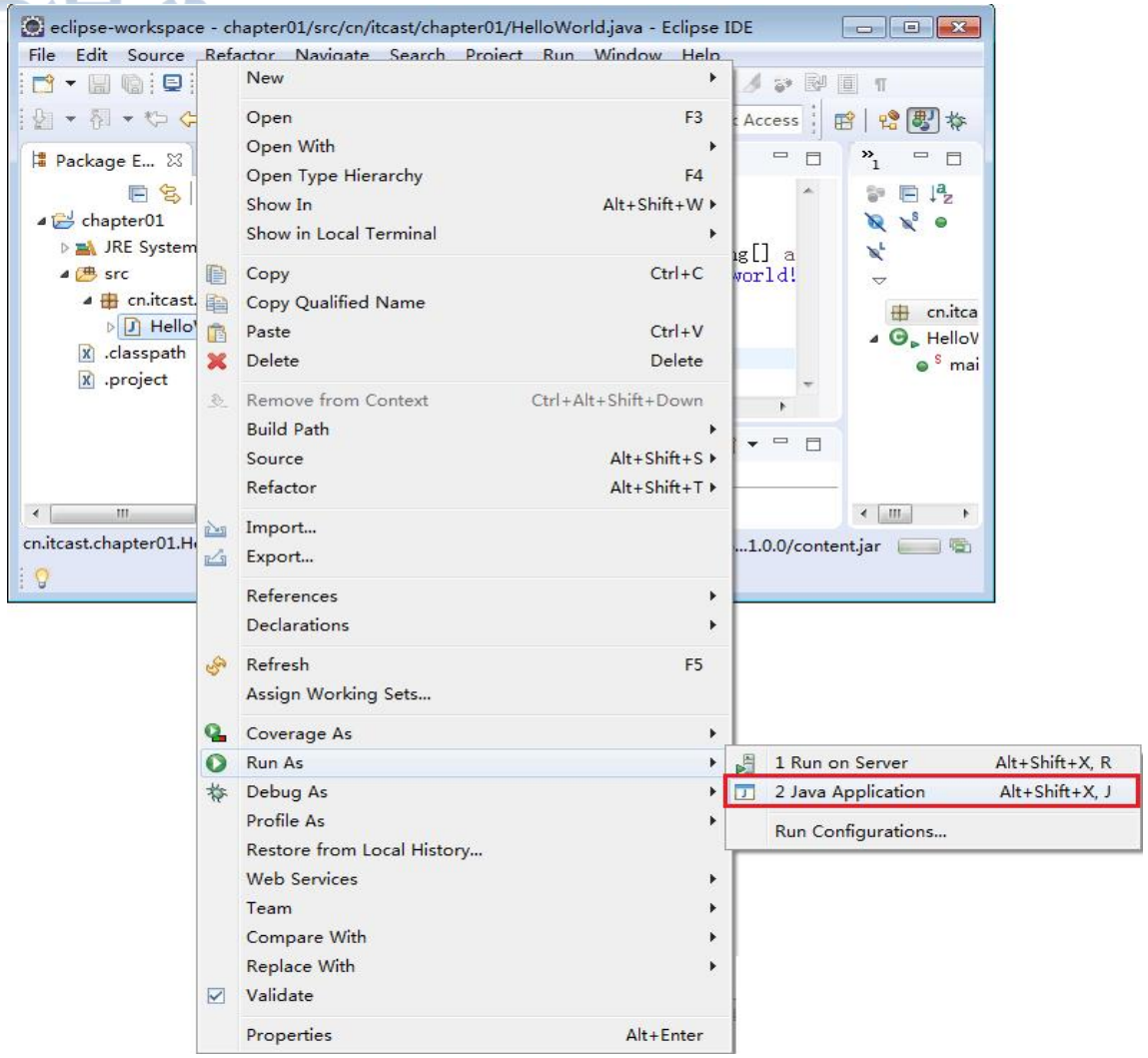
在上图中的文本编辑区域完成代码的编写，如下图。



1.6.3 Eclipse的下载与启动

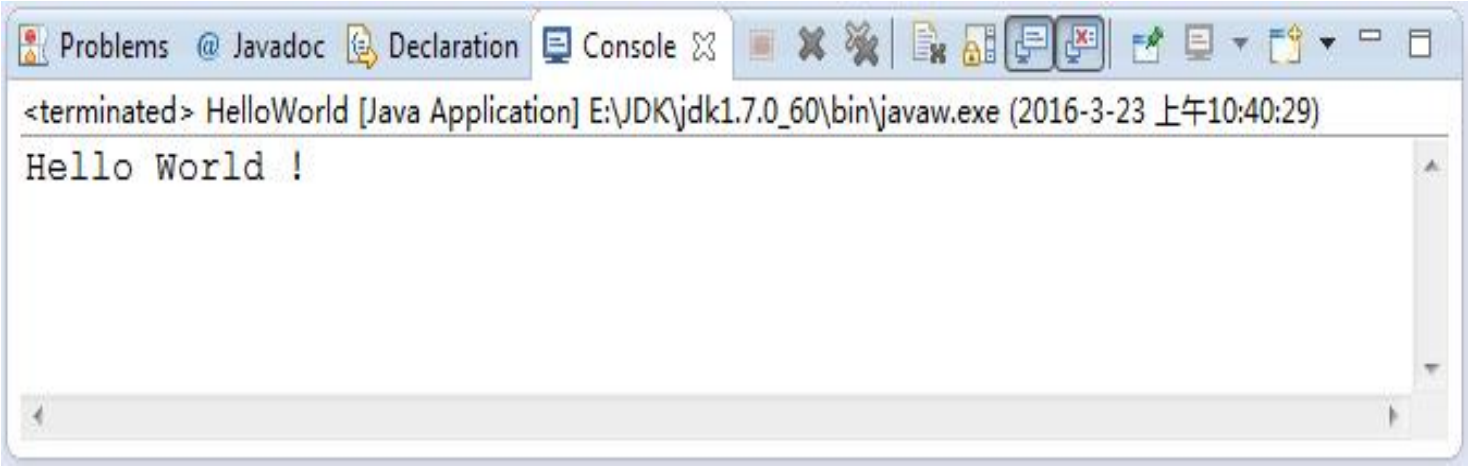
5．运行程序

在上图中，程序编写完成之后，鼠标右键单击Package Explorer视图中的HelloWorld.java文件，在弹出框中选择【Run As】→【Java Application】运行程序，如右图。



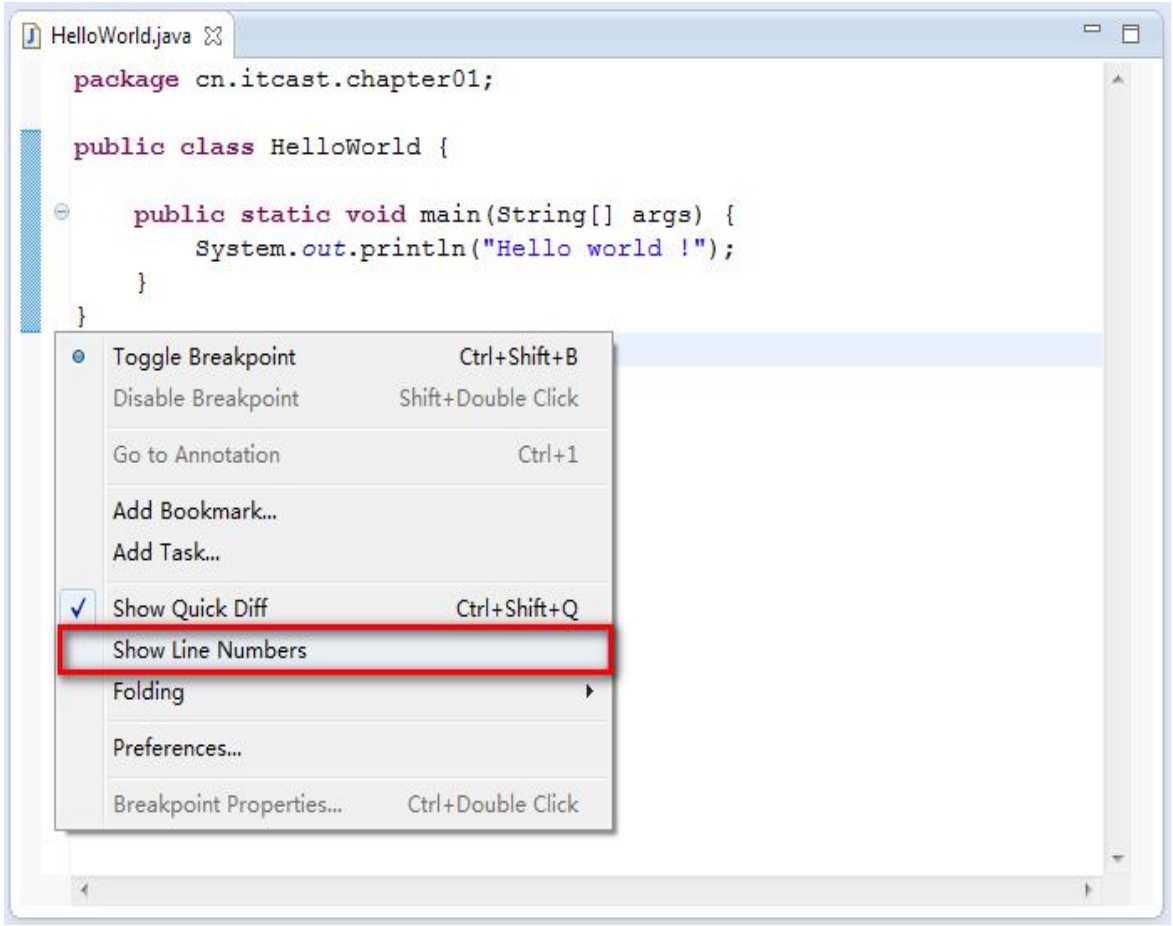
1.6.3 Eclipse的下载与启动

也可以选中要运行的文件，直接单击工具栏上的 按钮运行程序。程序运行完毕后，在Console视图中可以看到运行结果，如下图。



多学一招：Eclipse中显示代码行号

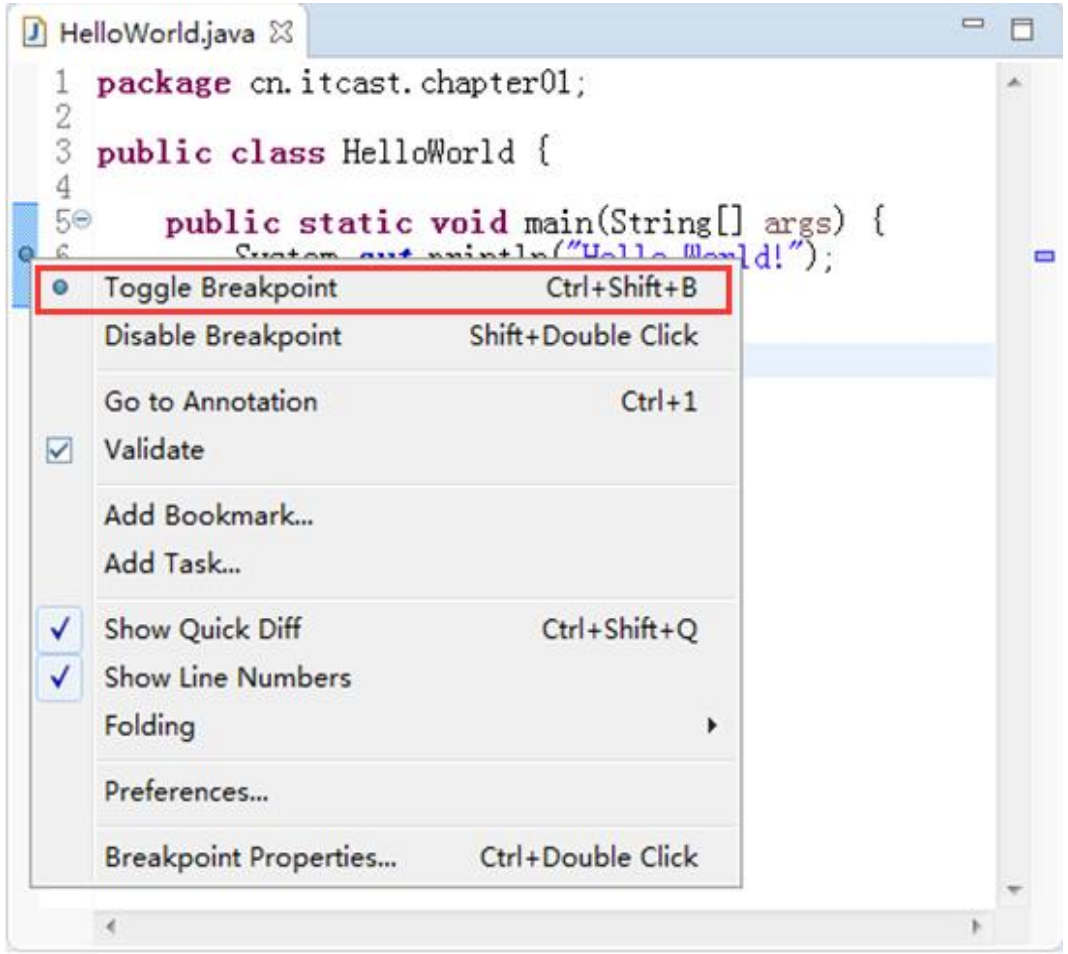
Eclipse提供了显示代码行号的功能，使用鼠标右键单击文本编辑器中左侧的空白处，在弹出的窗口中选择【Show Line Numbers】，即可显示出行号，如右图。



1.6.4 Eclipse调试工具

1.设置断点

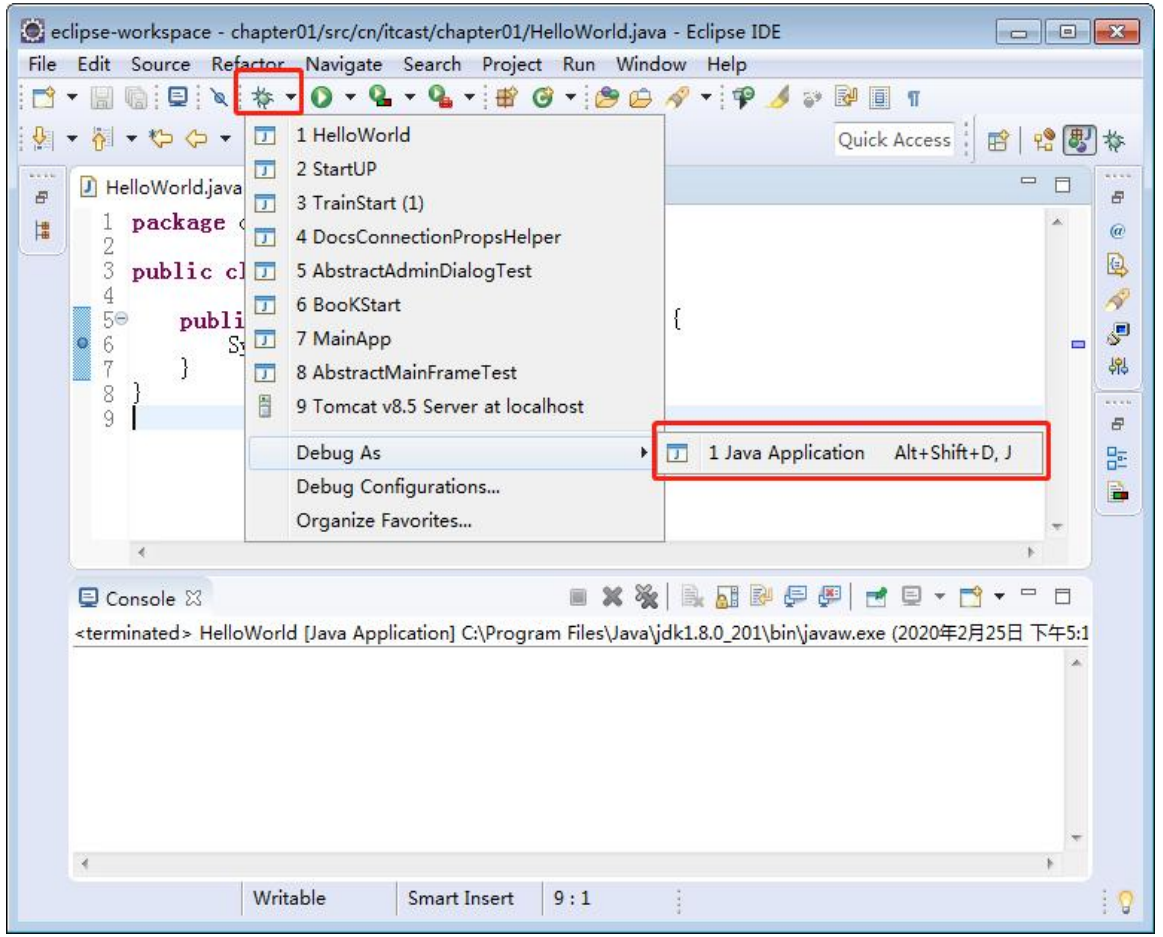
在需要调试的代码行前，单击右键，在弹出的对话框中选择“Toggle Breakpoint”选项。例如，在HelloWorld.java文件的第6行代码前设置断点，如右图。



1.6.4 Eclipse调试工具

2.设置Debug模式

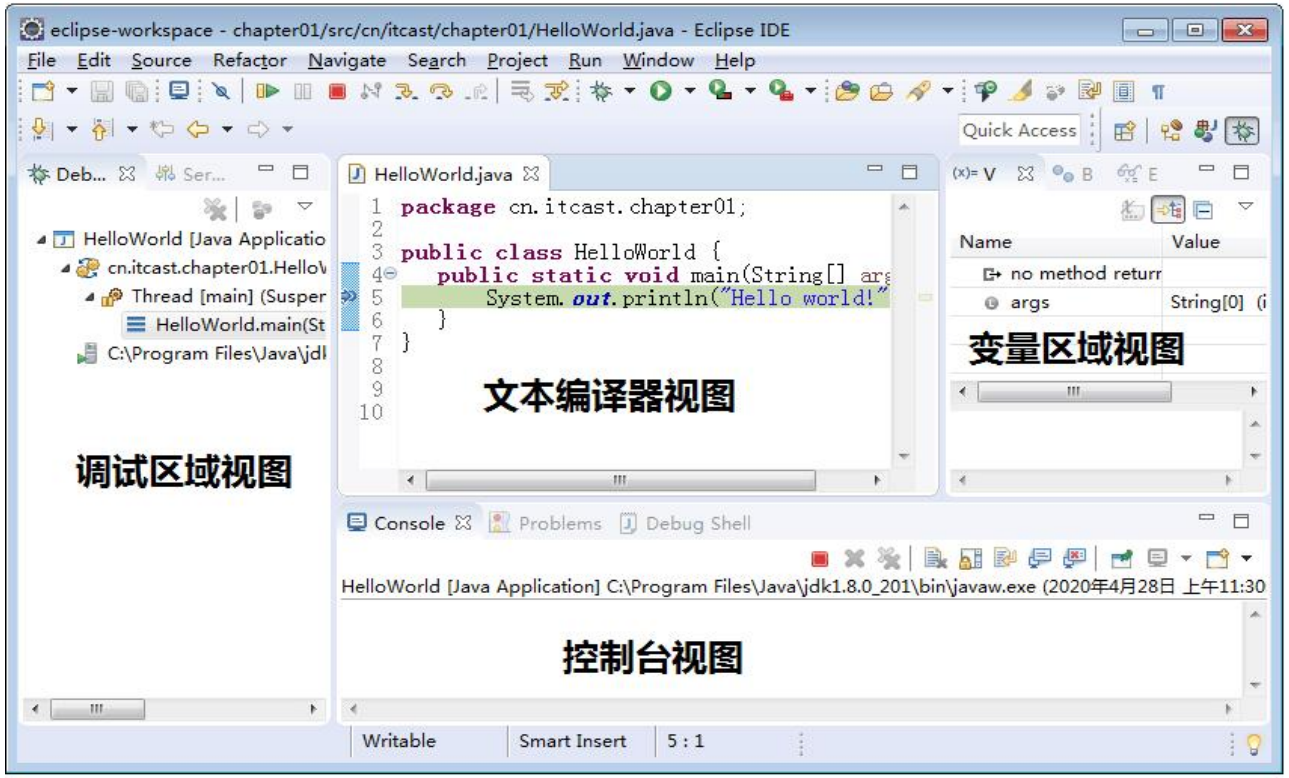
设置断点之后，单击工具栏中“”按钮的下拉菜单，选择【Debug As】→【Java Application】，进入Debug模式，如右图。



1.6.4 Eclipse调试工具

3.运行程序

程序启动调试运行后，会在设置的断点位置停下来，并且断点行代码底色会高亮显示，如右图。



1.6.4 Eclipse调试工具

Dubug模式的界面由调试区域视图、文本编译器视图、变量区域视图和控制台视图等多个部分组成。文本编译器视图和控制台视图我们已经有所了解，下面介绍一下其他两个视图的作用：

- 调试区域视图：又称为Debug调试区域视图，用于显示正在调试的代码。
- 变量区域视图：又称为Vsriables变量区域，用于显示调试过程中变量的值。

1.6.4 Eclipse调试工具

Eclipse在Debug模式下定义了很多快捷键用于更方便的调试程序，这些快捷键及含义如下表。

快捷键	操作名称
F5	单步跳入
F6	单步跳过
F7	单步返回
F8	继续
Ctrl+Shift+D	显示变量的值
Ctrl+Shift+D	在当前行设置或者去掉断点
Ctrl+R	直接运行所选行（也会跳过断点）

多学一招：包的定义与使用

为了便于对硬盘上的文件进行管理，通常会将文件分目录存放。同理，在程序开发中，也需要将编写的类在项目中分目录存放，以便于文件管理。为此，Java引入了包(package)机制，程序可以通过声明包的方式对Java类分目录管理。

📖多学一招：包的定义与使用

Java中的包是专门用来存放类的，通常功能相同的类存放在同一个包中。

包通过package关键字声明，示例代码如下：

```
package cn.itcast.chapter01; // 使用package关键字声明包
public class Example01{…}
```

需要注意的是，包的声明只能位于Java源文件的第一行。

多学一招：包的定义与使用

在开发时，一个项目中可能会使用很多包，当一个包中的类需要调用另一个包中的类时，需要使用import关键字引入需要的类。使用import关键字可以在程序中导入某个指定包下的类，这样就不必在每次用到该类时都书写完整类名，简化了代码量。使用import关键字导入某个包中的类的具体格式如下所示：

```
import 包名.类名;
```

多学一招：包的定义与使用

import通常出现在package语句之后，类定义之前。如果需要用到一个包中的多个类，则可以使用“import 包名.*;”导入该包下所有类。

📖多学一招：包的定义与使用

在JDK中，不同功能的类都放在不同的包中，其中Java的核心类主要放在java包及其子包下，Java扩展的大部分类都放在javax包及其子包下。Java语言中的常用包。

- java.util:包含Java中大量工具类、集合类等，如Arrays、List、Set等。
- java.net:包含Java网络编程相关的类和接口。
- java.io:包含了Java输入、输出有关的类和接口。
- java.awt:包含用于构建图形界面(GUI)的相关类和接口。

1.7.1 IDEA概述

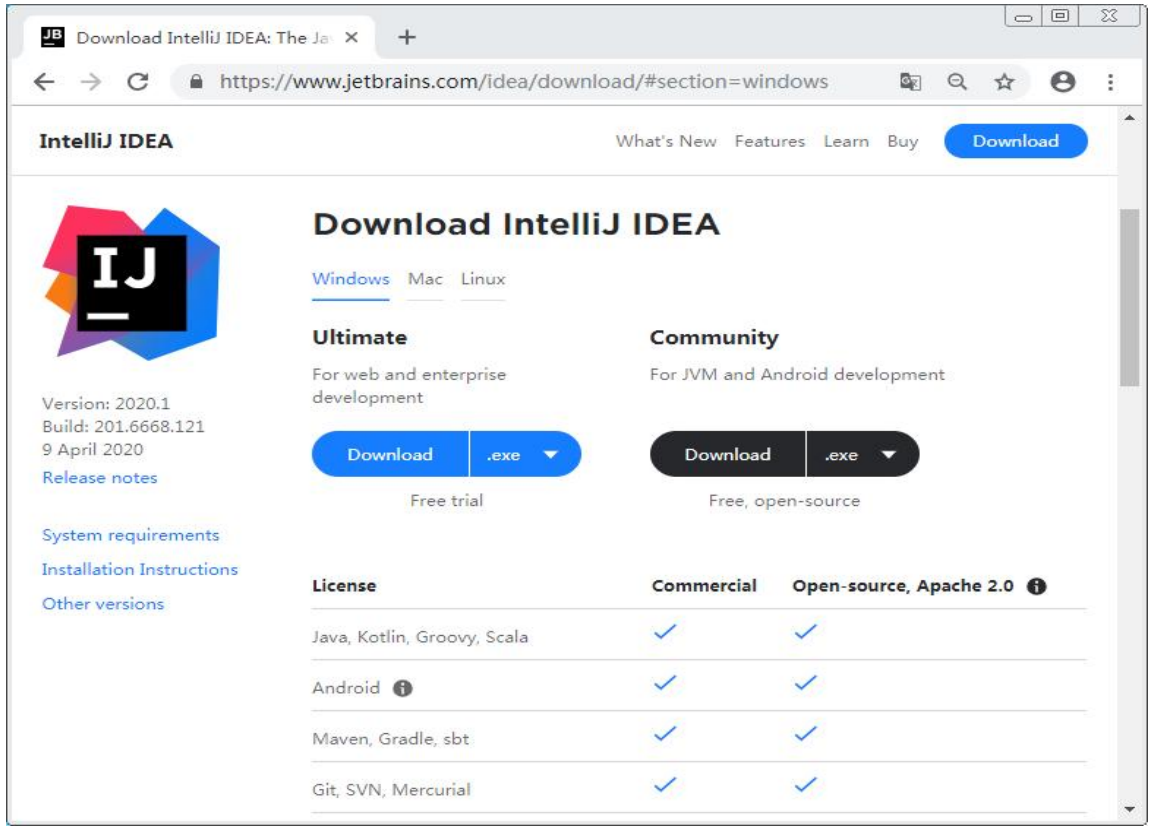
IDEA（全称IntelliJ IDEA）是用于Java程序开发的集成环境（也可用于其他语言），它在业界被公认是最好的Java开发工具之一，尤其在智能代码助手、代码自动提示、重构、J2EE支持、Ant、JUnit、CVS整合、代码审查、创新的GUI设计等方面的功能可以说是超常的。

IDEA是JetBrains公司开发的产品，开发人员是以严谨著称的东欧程序员为主。

1.7.2 IDEA安装与启动

1．安装IDEA开发工具

读者可以登录IDEA官网下载IDEA安装包，登录IDEA官网，可以看到IDEA有两个版本，分别是旗舰版和社区版，如右图。

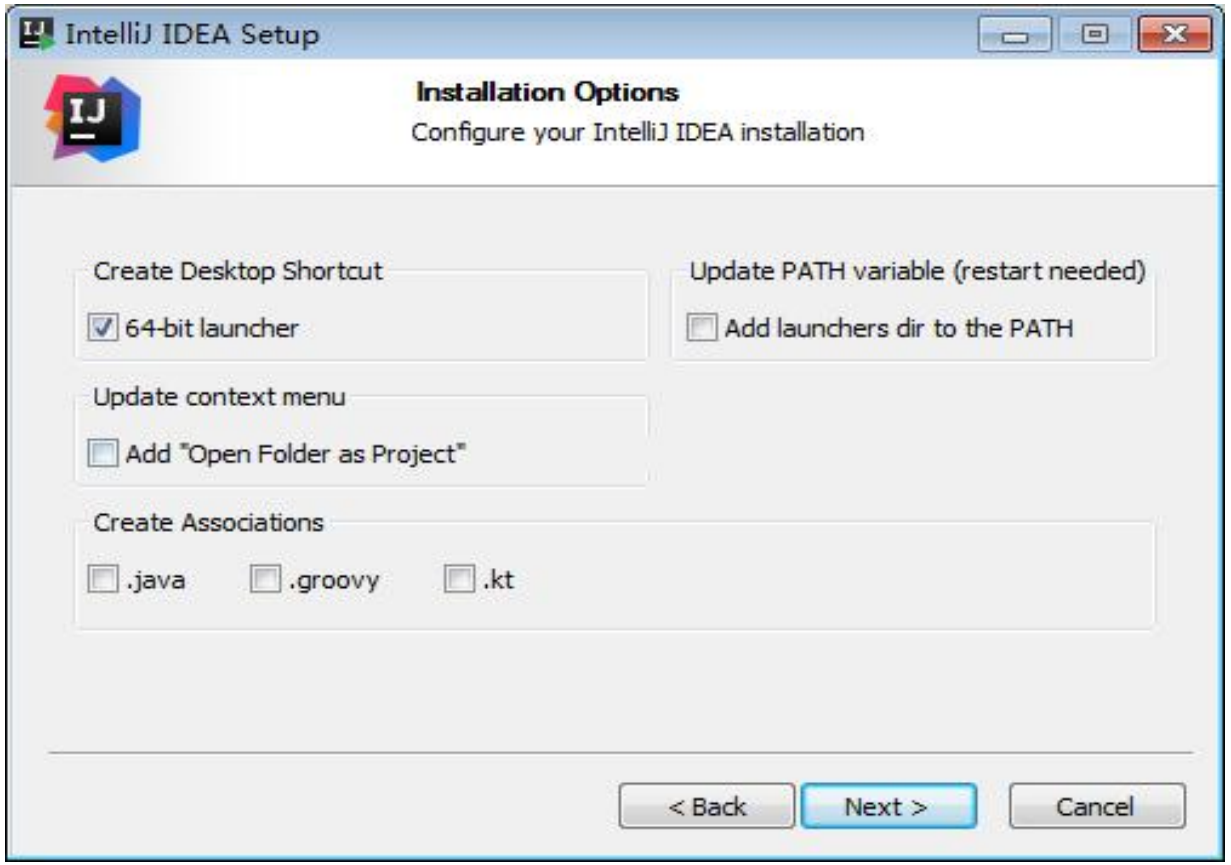


1.7.2 IDEA安装与启动

旗舰版比社区版的组件更全面，所以这里我们选择使用旗舰版。单击旗舰版下面“Download”链接进行下载。下载完成后，双击安装包，弹出IDEA安装欢迎界面，如右图。

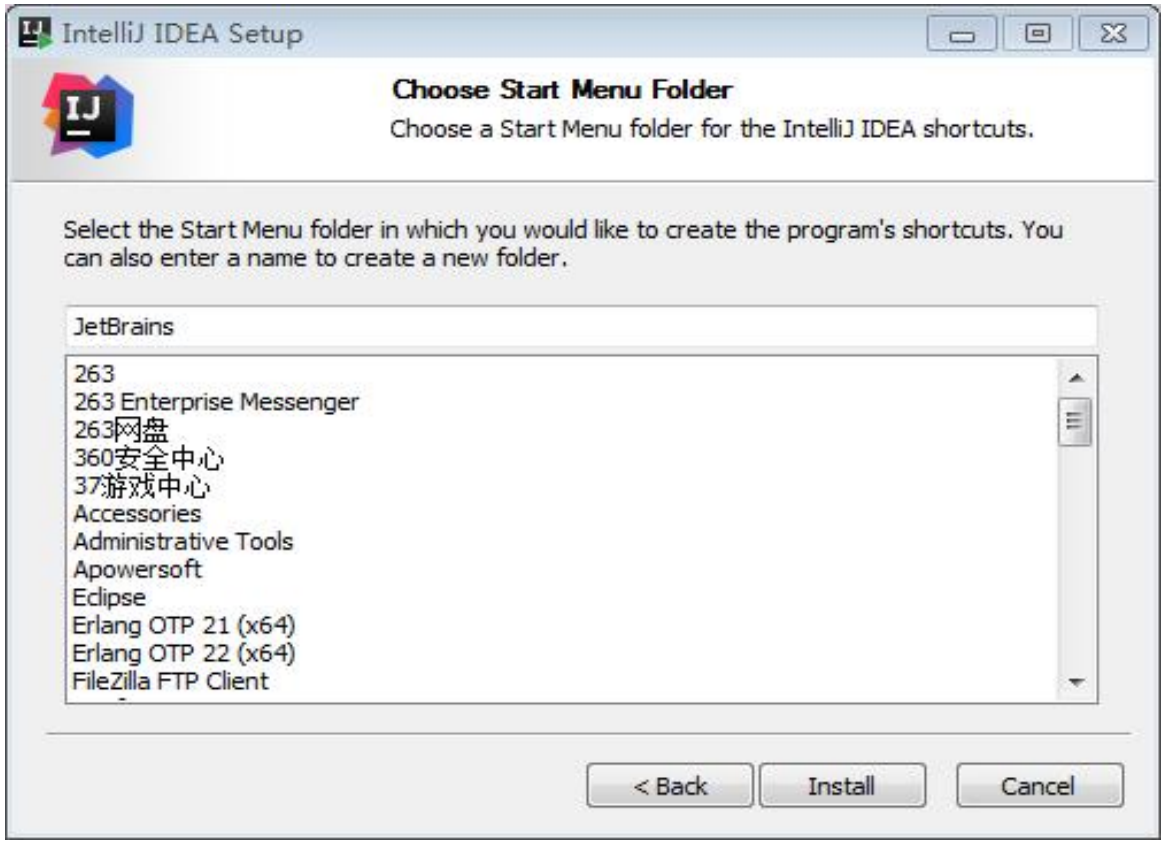
1.7.2 IDEA安装与启动

设置完安装路径之后，单击【Next】按钮，弹出基本安装选项配置界面，如右图。



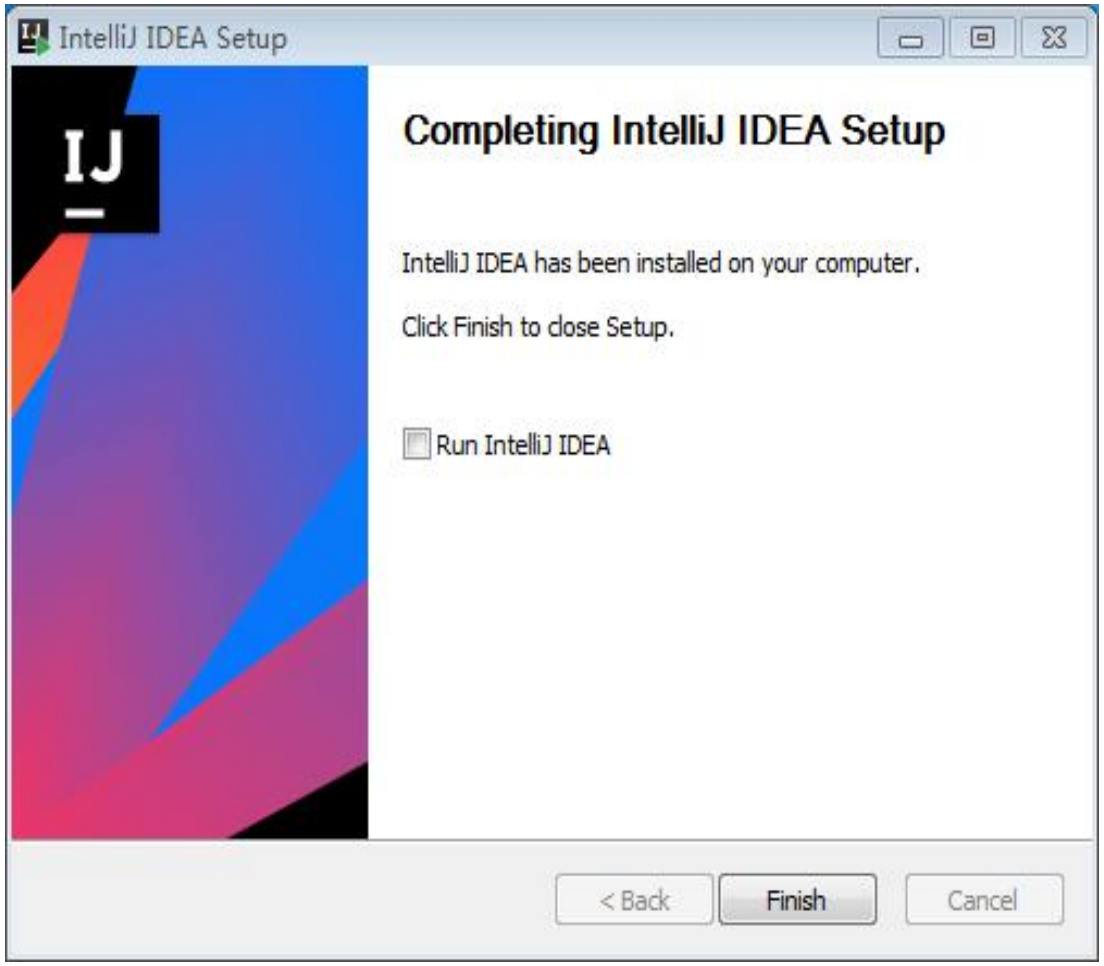
1.7.2 IDEA安装与启动

在上图中，勾选 “64-bit launcher”复选框。勾选了该复选框，IDEA在安装完成后会生成桌面快捷方式。勾选之后，单击【Next】按钮，弹出选择开始菜单界面，如右图。



1.7.2 IDEA安装与启动

在上图中，单击【Install】按钮安装IDEA。安装完成界面，如右图。



1.7.2 IDEA安装与启动

2．启动IDEA开发工具

IDEA安装完成之后，双击桌面快捷方式进行启动，启动界面如右图。

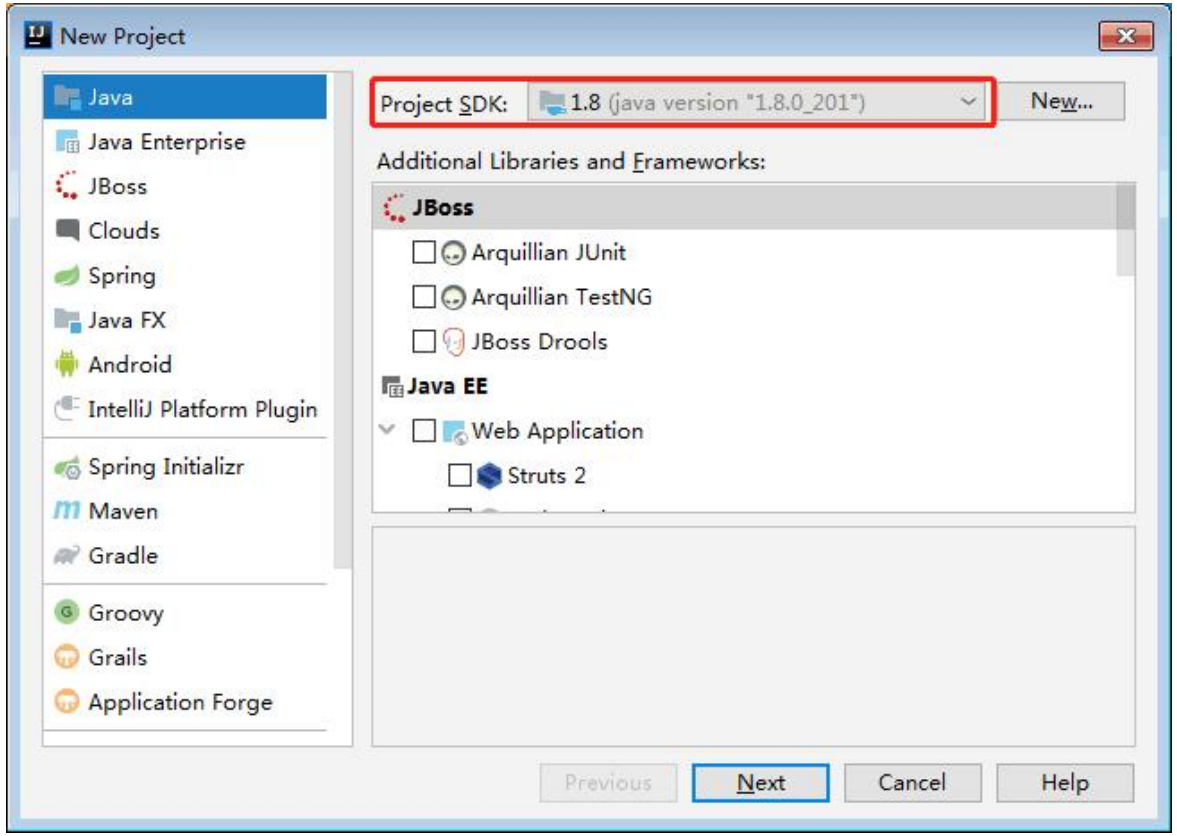
1.7.2 IDEA安装与启动

IDEA启动完成后会弹出一个对话框，提示需要购买IDEA。由于IDEA旗舰版有30天免费试用期，因此这里我们可以免费使用。直接进入IDEA主界面，IDEA主界面如右图。

1.7.3 使用IDEA进行程序开发

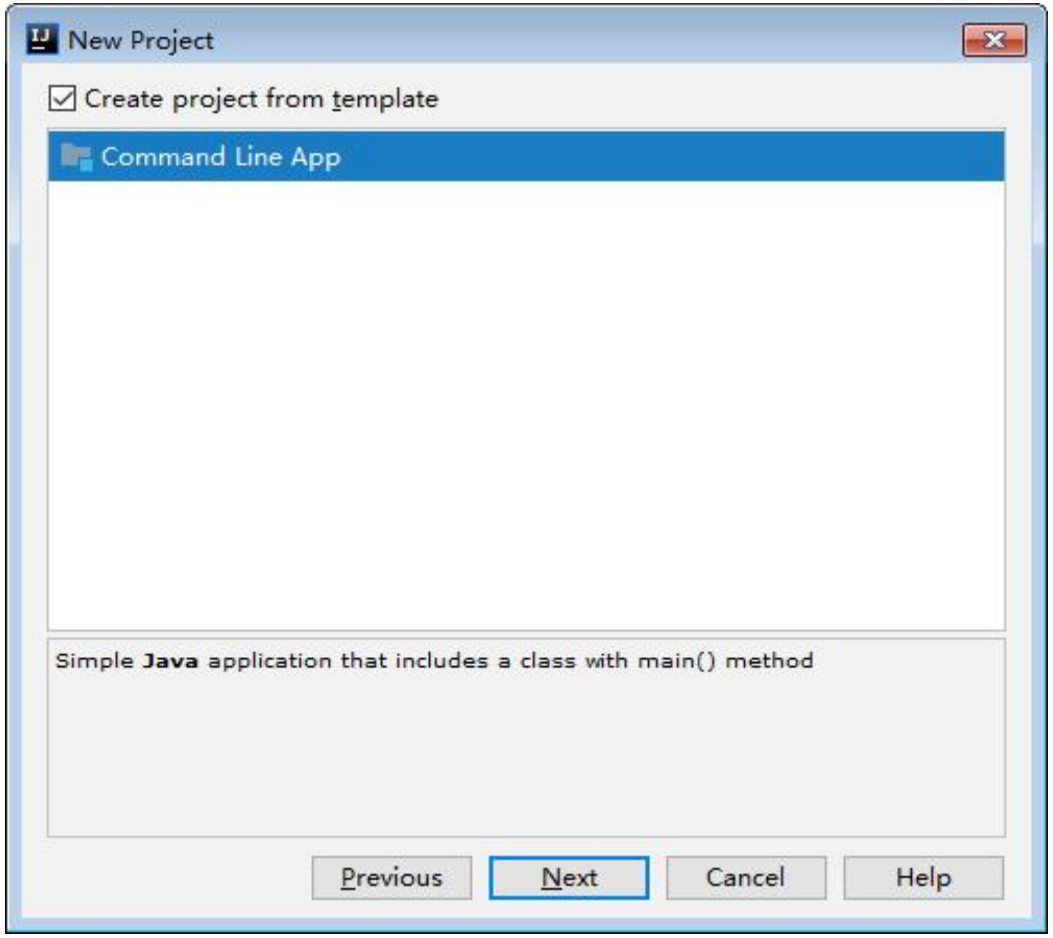
1．创建Java项目

在上图中单击“Create New Project”选项创建新项目，单击之后进入New Project页面，如右图。



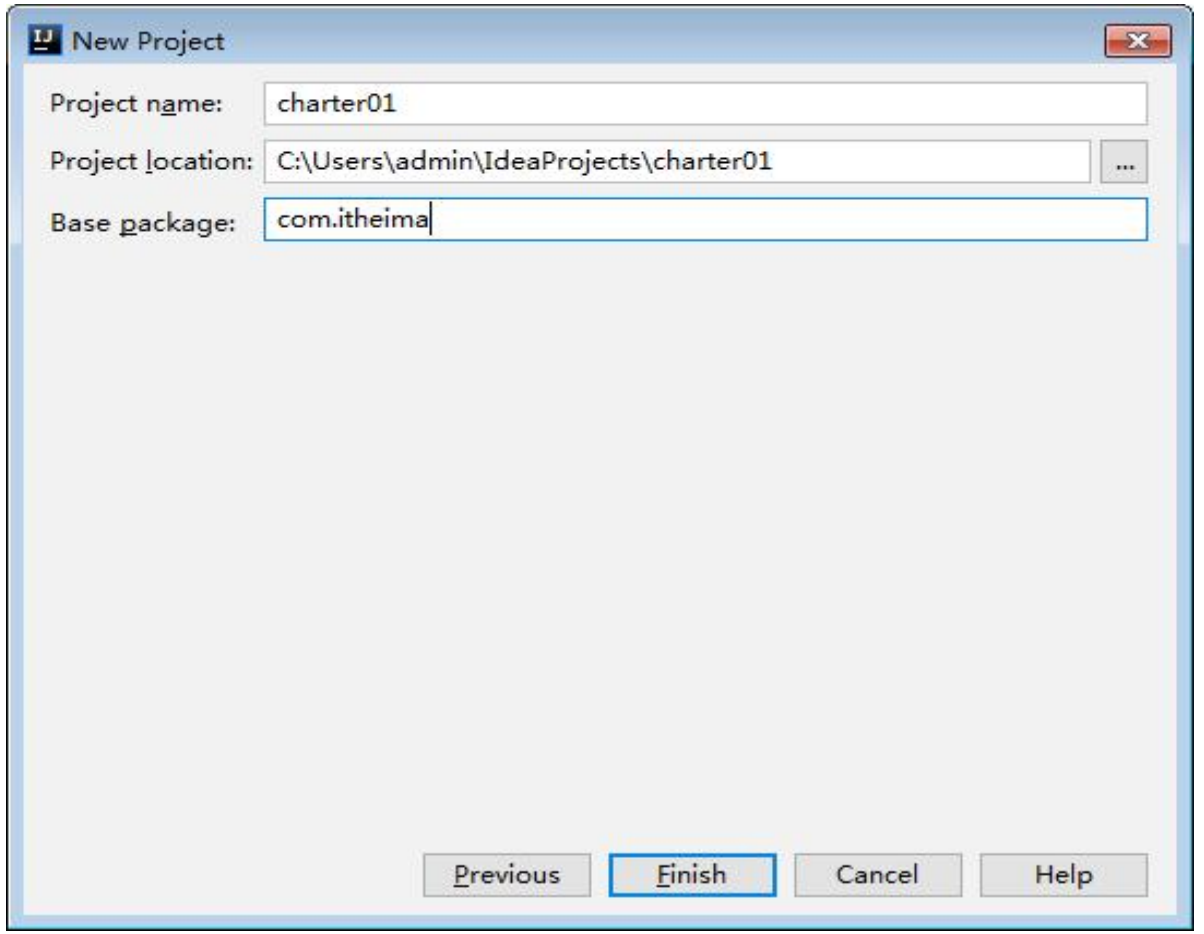
1.7.3 使用IDEA进行程序开发

在上图中，需要设置Java程序开发所需要的JDK。在左侧栏选中“Java”，在右侧栏顶部“Project SDK”后面选择下载好的JDK，然后单击【Next】按钮进入选择模板创建项目界面，如右图。



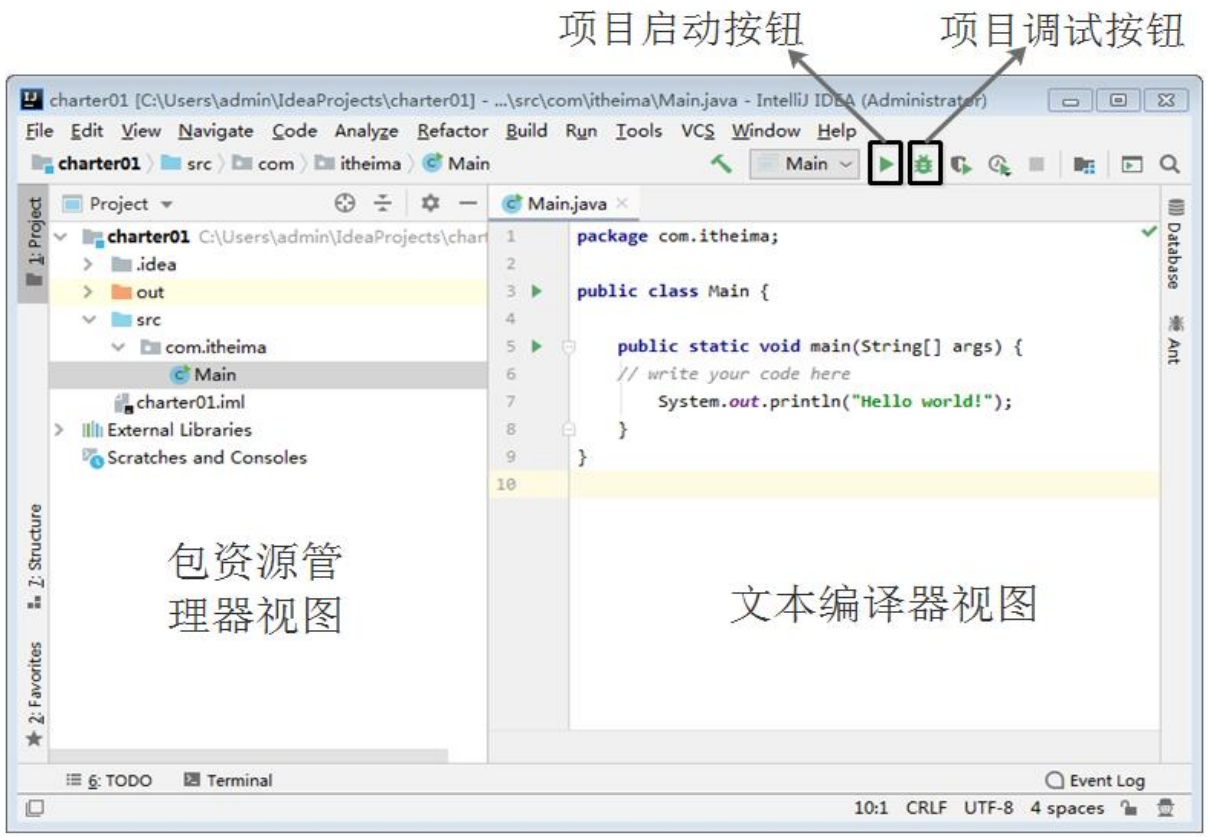
1.7.3 使用IDEA进行程序开发

在上图中，单击【Next】按钮进入项目设置界面，如右图。



1.7.3 使用IDEA进行程序开发

在上图中，设置完成项目名称、项目路径和包名之后，单击【Finish】按钮进入IDEA开发界面，如右图。



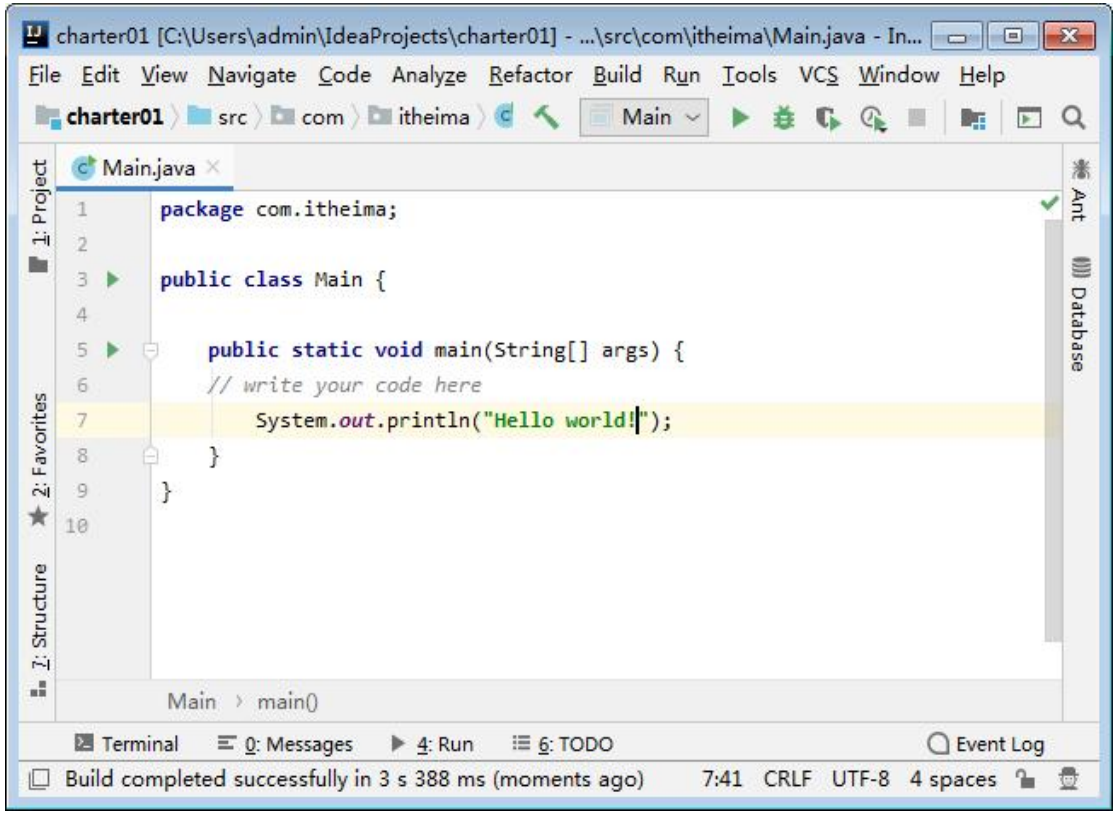
1.7.3 使用IDEA进行程序开发

由上图可以看到，IDEA开发界面上有包资源管理器视图、文本编辑器视图等多个视图。与Eclipse类似，Intellij IDEA视图可以单独出现，也可以和其他视图叠放在一起，并且可以通过拖动随意改变视图布局和位置。

1.7.3 使用IDEA进行程序开发

2．编写程序代码

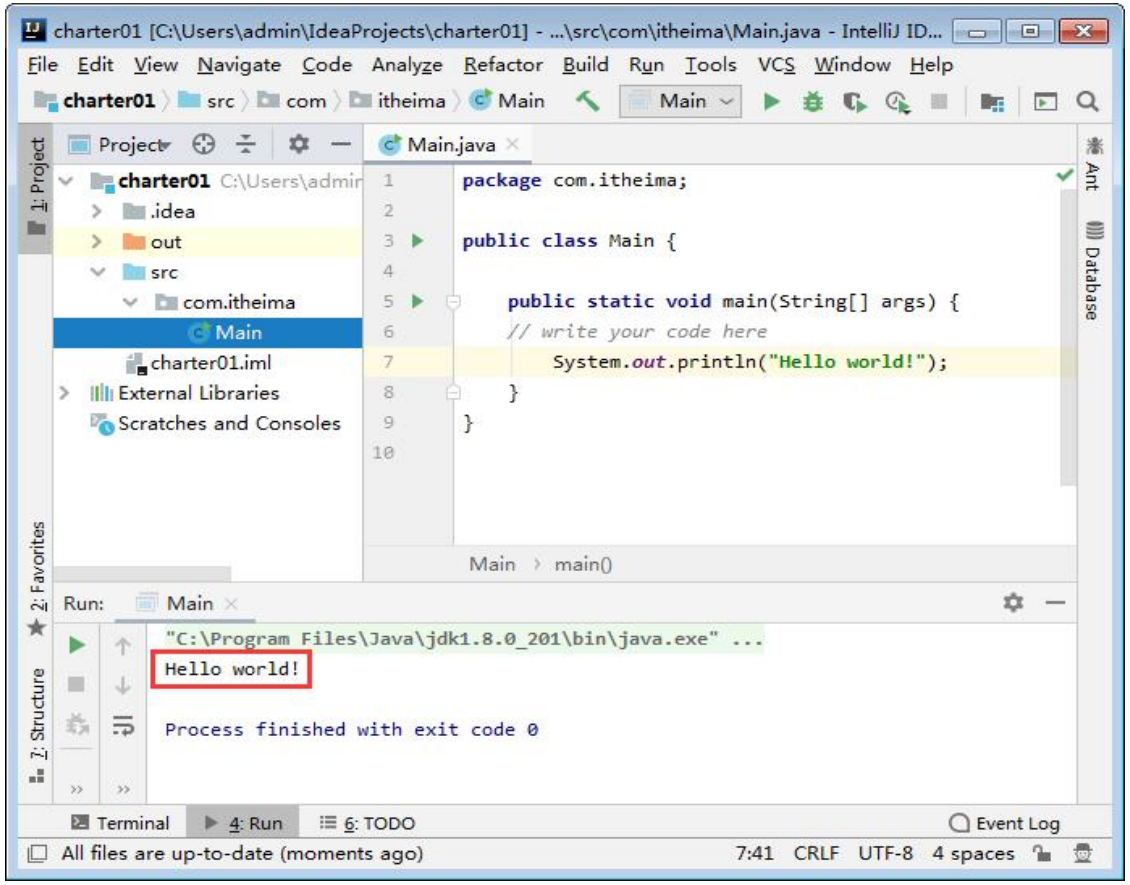
项目新建完成后，系统会自动创建一个名称为Main.java文件，我们可以在该文件中编写Java代码，如右图。



1.7.3 使用IDEA进行程序开发

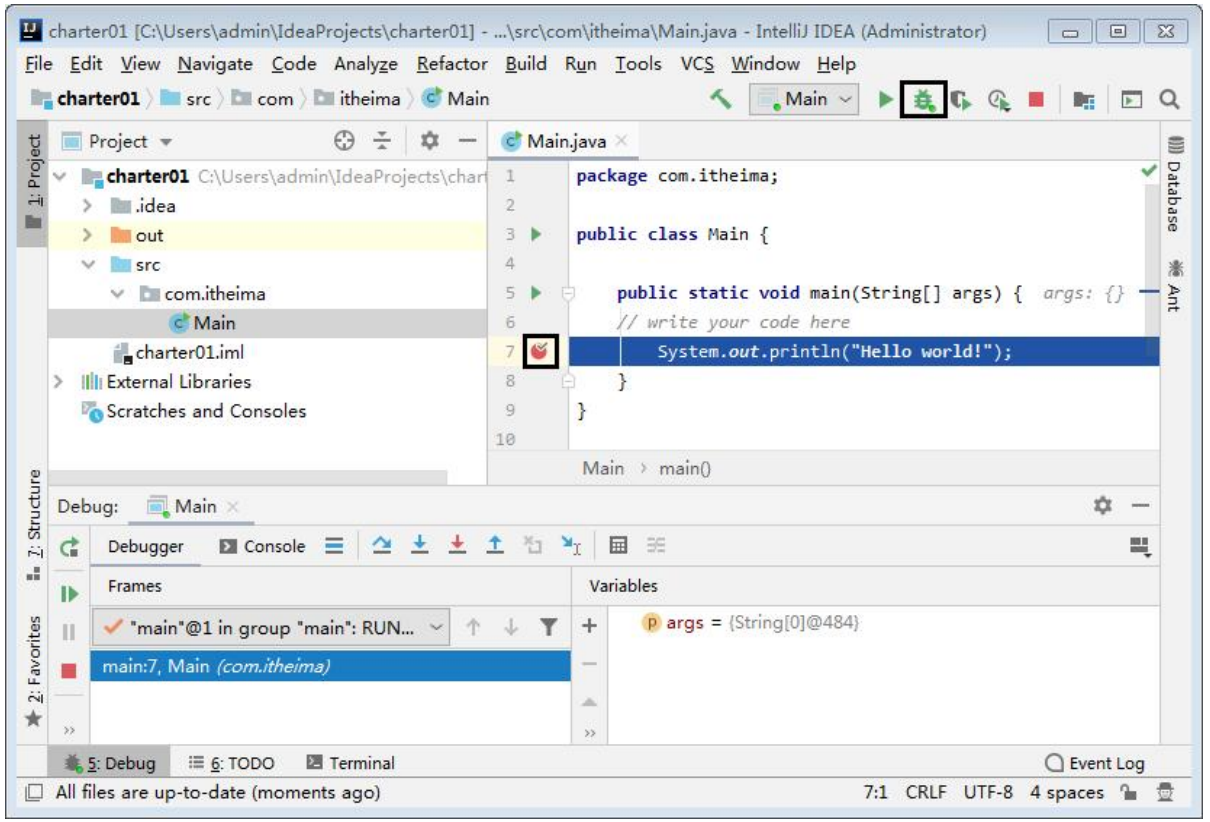
3．运行程序

在上图中，单击工具栏中的“ ▶ 按钮运行程序，控制台会显示运行结果，如右图。



1.7.4 IDEA工具调试程序

IDEA调试的方式与Eclipse类似，首先需要设置断点，然后单击图1-59中的“”按钮进入Dubug模式。



1.7.4 IDEA工具调试程序

IDEA在Dubug模式下也定义了一些快捷键用于调试，这些快捷键的含义如下表。

快捷键	操作名称
F8	单步调试（不进入函数内部）
F7	单步调试（进入函数内部）
Shift+F7	选择要进入的函数
Shift+F8	跳出函数
Alt+F9	运行到断点F7F7
Alt+F8	执行表达式查看结果
F9	继续执行，进入下一个断点或执行完程序

1.8 本章小结

本章首先介绍了Java语言、Java语言的相关特性和Java语言的发展史；其次介绍了JDK的概念，并在Windows7系统中安装JDK；然后带领读者编写了一个简单的Java程序，并讲解了Java程序的运行机制和环境变量的配置；最后为读者介绍了Eclipse和IDEA这两种主流的Java程序开发工具，包括工具的特点、下载、安装以及入门程序的编写和调试。通过本章的学习，读者能够对Java语言有一个基础认识，为后面学习Java知识开启了大门。