第 1 章 Java 概述

学习目标

【应用】理解 Java 概述、HelloWorld 案例

(1)【了解】Java 语言的发展史。

(2)【理解】JVM、JRE、JDK 分别是什么？有什么作用？有什么关系？

(3)【应用】能够操作常用的 DOS 命令。

(4)【应用】独立下载安装 JDK。

(5)【应用】独立编写 HelloWorld 案例，并能够调试问题，使程序正常运行。

1.1 Java 语言发展史和平台概述

1.1.1 Java 语言发展史

詹姆斯·高斯林(James Gosling)，1977 年获得了加拿大卡尔加里大学计算机科学学士学位，1983 年获得了美国卡内基梅隆大学计算机科学博士学位，毕业后到 IBM 工作，但不受重视。后来转至 SUM(Stanford University Network，斯坦福大学网络)公司，于 1991 年，和其他几位工程师一起合作参与了“Green 计划”(绿色计划)，并开发了一种称为 Oak(橡树)的面向对象语言。1995 年 5 月 23 日，Oak 语言改名为 Java，Java 语言由此诞生。

1.1.2 Java 语言版本

Java 语言版本有三个：标准版、缩减版和企业版。三个版本的应用领域见如图 1-1 所示。

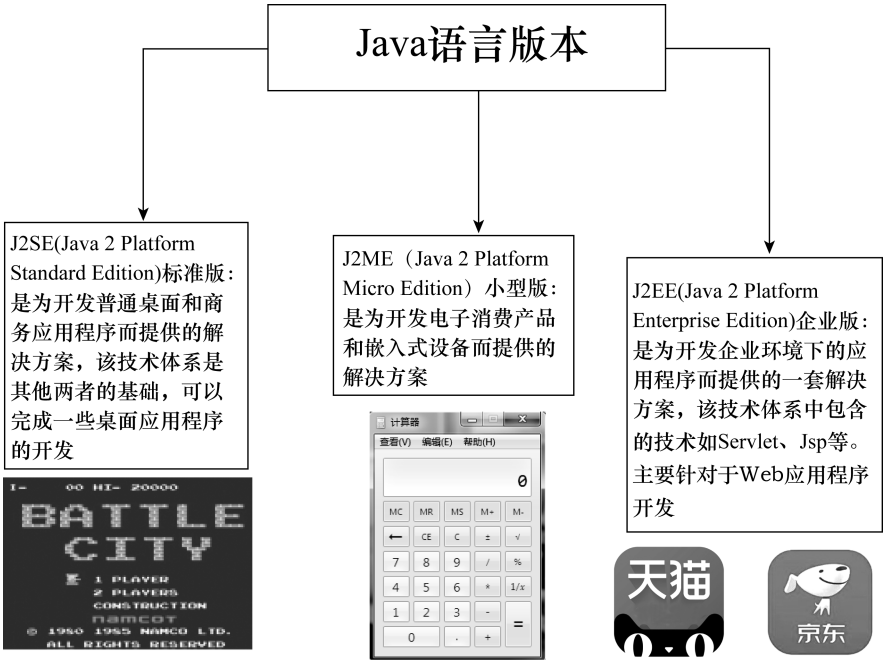


图 1-1　Java 语言版本

1.2　JVM、JRE 和 JDK 的概述

1.2.1　跨平台

平台：指的是计算机硬件或软件的操作环境(如 Windows、Linux、Mac 等)。

跨平台：Java 程序可以在任意操作系统上运行，如图 1-2 所示。一次编写到处运行。

原理：实现跨平台需要依赖 Java 的虚拟机 JVM(Java Virtual Machine)。

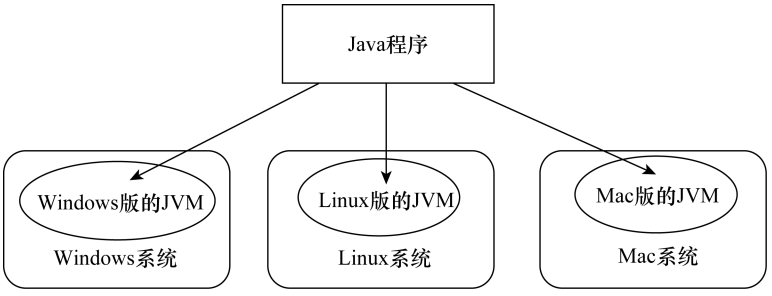


图 1-2　Java 程序可运行的操作系统

1.2.2　JVM、JRE 和 JDK 的说明

1. JVM

JVM 是 Java 虚拟机，Java 程序需要运行在虚拟机上，每个平台都有自己的虚拟机，因此 Java 语言可以跨平台。

2. JRE

JRE 包括 JVM 和 Java 程序所需的核心类库等，如果想要运行一个开发好的 Java 程序，计算机中只需要安装 JRE 即可。

JRE：JVM+类库。

3. JDK

JDK 是提供给 Java 开发人员使用的，它既包含 Java 的开发工具，也包括 JRE。读者安装 JDK，则不用再单独安装 JRE。Java 的开发工具包括编译工具(javac. exe)、打包工具(jar. exe)等。

JDK：JRE+Java 的开发工具。

4. JVM、JRE 和 JDK 的作用和关系

实际开发中程序员利用 JDK(调用 Java API)开发 Java 程序，然后通过 JDK 中的编译程序(javac)将 Java 文件编译成 Java 字节码文件，在 JRE 上运行这些 Java 字节码，JVM 解析这些字节码，映射到 CPU 指令集或 OS 的系统调用。简单来说就是 JDK 包含 JRE，JRE 又包含 JVM 的关系，三者关系如图 1-3 所示。

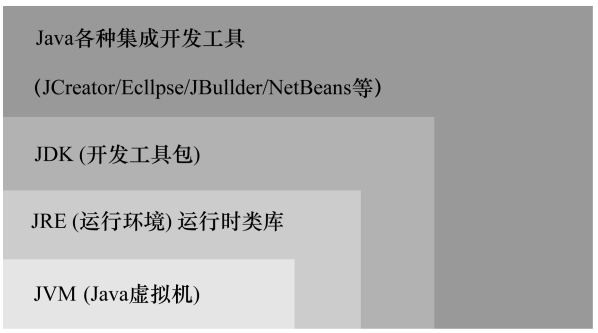


图 1-3　JVM、JRE、JDK 三者关系

1.3　常用 DOS 命令

1.3.1　打开控制台

打开控制台的方法为：“win+R”，然后输入 cmd，敲回车键。

1.3.2 常用命令

常用命令介绍如下。

(1) d：在 DOS 字符界面输入“d:”，敲回车键，表示盘符切换。

(2) dir(directory)：列出当前目录下的文件以及文件夹。

(3) cd(change directory)：改变指定目录(进入指定目录)。

(4) cls：(clear screen)清屏。

(5) exit：退出 DOS 命令行。

1.4 下载安装 JDK

目前最新的是 JDK 12，它于 2019 年 3 月 19 日发布，JDK 12 已处于 Rampdown Phase One 阶段，所有的新特性已经冻结，不会再针对其他。新特效如下。

编号	名称	说明
189	Shenandoah：A Low-Pause-Time Garbage Collector (Experimental)	新增名为 Shenandoah 的、低暂停时间的实验性垃圾收集器
230	Microbenchmark Suite	新增微基准测试套件(Microbenchmark Suite)，使开发人员可以轻松运行现有的微基准测试并创建新的基准测试
325	Switch Expressions (Preview)	对现有 Switch 表达式改进，使其可以用作语句或表达式
334	JVM Constants API	引入 API 来对关键类文件(Key Class-File)和运行时工件(Run-Time Artifacts)的名义描述(Nominal Descriptions)建模，特别是可从常量池加载的常量
340	One AArch64 Port，Not Two	在保留 32 位 ARM 端口和 64 位 AArch64 端口的同时，删除与 ARM64 端口相关的所有源码
341	Default CDS Archives	在 64 位平台上使用默认类列表增强 JDK 构建过程，从而生成类数据共享(CDS)归档
344	Abortable Mixed Collections for G1	如果 G1 混合回收超过暂停目标时，使其可中止
346	Promptly Return Unused Committed Memory from G1	增强 G1 垃圾收集器，以便在空闲时自动将 Java 堆内存返回给操作系统

本书案例的运行环境是 64 位版 Win7，使用的 JDK 版本是 jdk1.8，具体安装步骤如下。

(1) 首先到官网根据操作系统环境下载对应的 jdk1.8 版本，如图 1-4 所示。

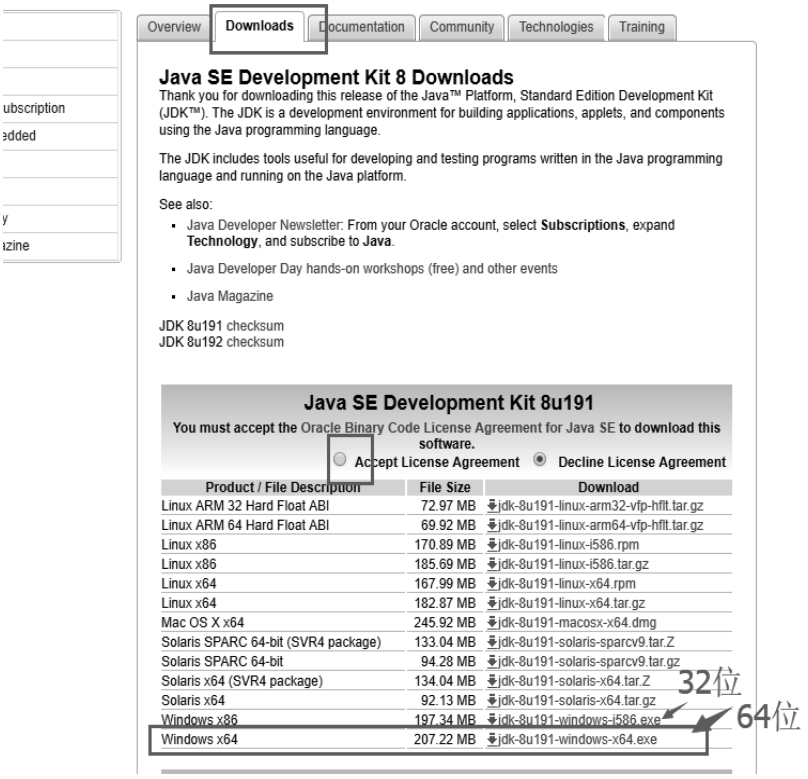


图 1-4　下载页面

（2）接下来只需根据提示进行安装，这里不更换任何配置，直接“下一步”到安装结束，如图 1-5、1-6、1-7 所示。

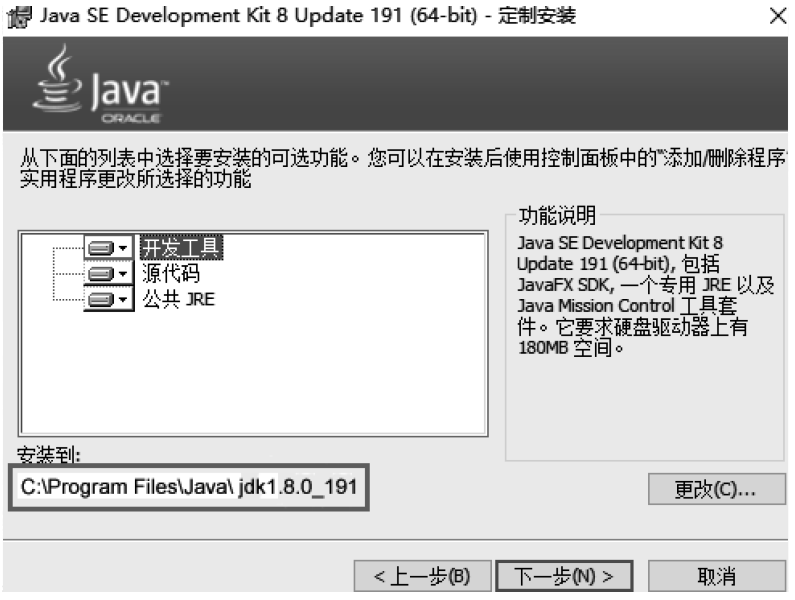


图 1-5　安装界面(1)

图 1-6　安装界面(2)

图 1-7　安装界面(3)

1.5　HelloWorld 案例

1.5.1　Java 程序执行流程

Java 程序执行流程如图 1-8 所示。

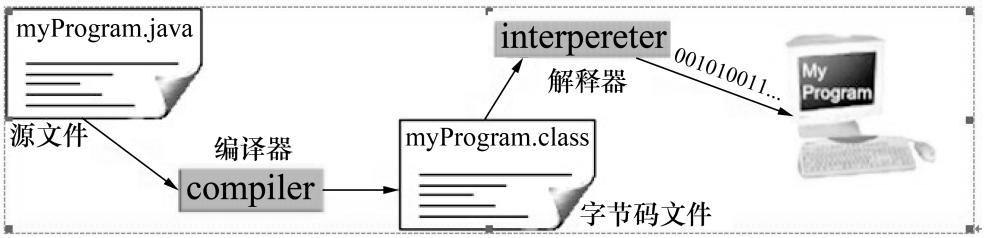


图 1-8　Java 程序执行流程

1.5.2　编写代码步骤

1. 首先定义一个类

即：public class 类名

2. 在类定义后加上一对大括号

即：{ }

3. 在大括号中间添加一个主(main)方法/函数

即：public static void main(String[ ] args){ }

4. 在主方法的大括号中间添加一行输出语句

即：System. out. println("HelloWorld");

文件 1-1　HelloWorld. Java

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

1.5.3　运行代码步骤

(1) 在命令行模式中，输入 javac 命令对源代码进行编译，生成字节码文件，即：

```
javac 源文件名 . java
```

(2) 编译完成后，如果没有报错信息，输入 Java 命令对 class 字节码文件进行解释运行，执行时不需要添加 . class 扩展名，即：

java 源文件名

1.5.4 HelloWorld 案例常见问题

1. 找不到文件

（1）文件扩展名隐藏导致编译失败。

（2）文件名写错了。

2. 单词拼写问题

（1）class 写成 Class。

（2）String 写成 string。

（3）System 写成 system。

（4）main 写成 mian。

3. 括号匹配问题

（1）把类体的那对大括号弄掉一个。

（2）把方法体的那对大括号弄掉一个。

（3）把输出语句的那对小括号弄掉一个。

4. 中英文问题

中英文问题主要出现在提示信息，例如，“错误：非法字符：\ ???? 的格式、错误”“编码 GBK 的不可映射字符”，等等。

注意：Java 编程中需要的基本上都是英文字符。

本章小结

本章主要讲解了 Java 语言的发展史，并介绍了 JVM、JRE、JDK 三者之间的关系，JDK 安装的步骤，经典的 HelloWorld 案例。通过本章的学习，结合 HelloWorld 案例，学生需要掌握如何编写程序和调试程序。

习　题

一、简答题

1. 简述 Java 语言的发展史。

2. JVM、JRE、JDK 分别是什么。

3. 编写一个程序，在控制台输出“欢迎来到 Java 世界”。

第 2 章 环境配置与 Eclipse 工具介绍

学习目标

【理解】理解配置环境变量、开发工具使用

(1)【应用】独立配置 path 环境变量。

(2)【应用】熟练使用 Eclipse 开发工具。

2.1 环境变量配置

2.1.1 环境变量配置基本原理

程序的编译和执行需要使用到 javac 和 java 命令，如果不配置环境变量，则应用程序只能在 jdk 安装目录的 bin 目录下编写程序，而实际开发中，不可能把程序写到 bin 目录下，所以为了能让 javac 和 java 命令在任意目录下都能够被访问，需要配置环境变量。

2.1.2 环境变量配置基本步骤

环境变量配置基本步骤如图 2-1 所示。

(1) 创建新的变量名称：JAVA_HOME。

其方法为：计算机→右键属性→高级系统设置→高级→环境变量→系统变量。

(2) 为 JAVA_HOME 添加变量值，即：JDK 安装目录。

本书建议安装目录为：“C:\Program Files\Java\jdk1.8.0_191”。

(3) 在 path 环境变量最前面添加如下内容。

%JAVA_HOME%\bin;

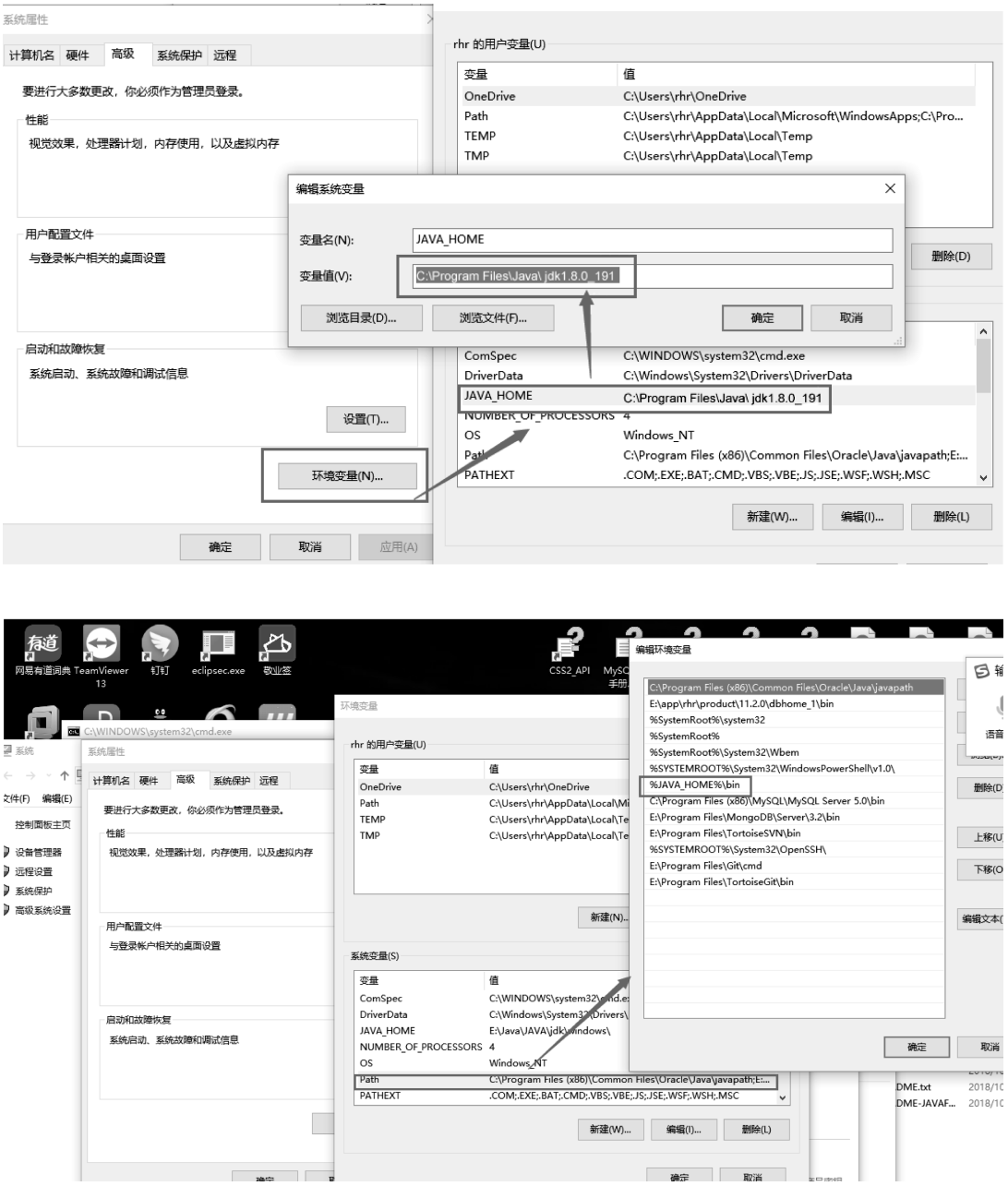


图 2-1　环境变量配置基本步骤

2.2　Eclipse 概述和安装

Eclipse 是一个 IDE(Integrated Development Environment)集成开发环境，是一款集成代码编写功能、分析功能、编译功能、调试功能等于一体的开发软件。

Eclipse 的特点描述如下：免费、纯 Java 语言编写、免安装、扩展性强。

Eclipse 下载地址：http://eclipse.org/

Eclipse 安装：绿色版解压就可以使用 Eclipse，如图 2-2 所示。

图 2-2　Eclipse 安装

2.3　Eclipse 的基本使用

2.3.1　Eclipse 基本操作

1. 选择工作空间

工作空间——程序员编写应用程序源代码所在的目录(如图 2-3 所示)。

图 2-3　Eclipse 工作空间

2. 用 Eclipse 来完成一个 HelloWorld 案例

（1）创建 Java 项目：点击 File 或者在最左侧空白处，选择 Java Project（如图 2-4 所示），在 Project name 中输入项目名称“chapter02”，然后点击 Finish 即可（如图 2-5 所示）。

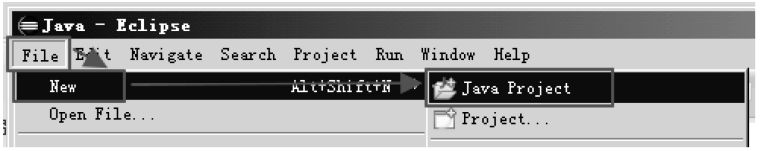


图 2-4　创建 Java 项目

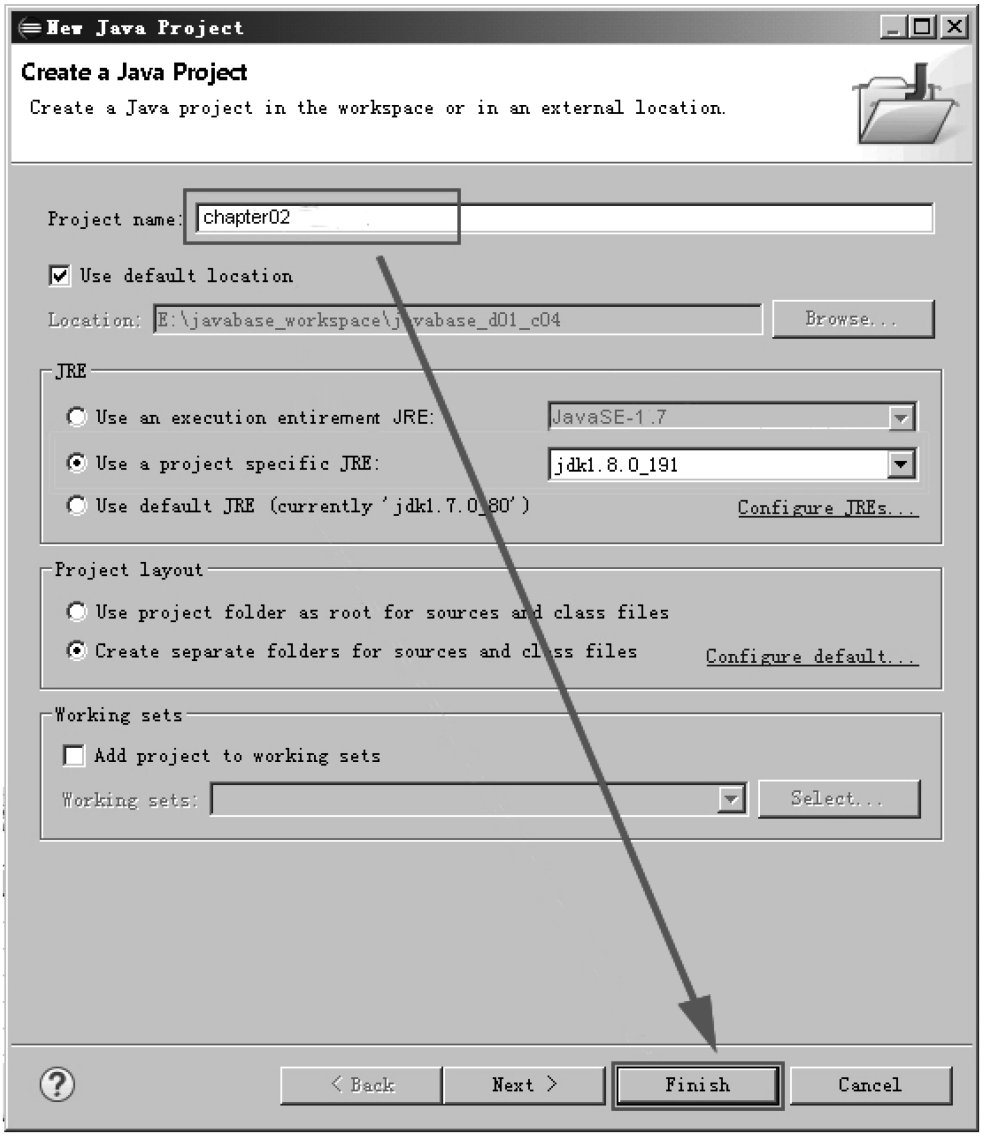


图 2-5　创建新的 Java Project

（2）创建包：展开项目，在 src 目录中新建一个 Package，Package 类似于 Windows 的目录，Package 的 Name 值为：com. igeek_02(如图 2-6 所示)，有关 Package 包的概念和使用方法将在第 6 章介绍。

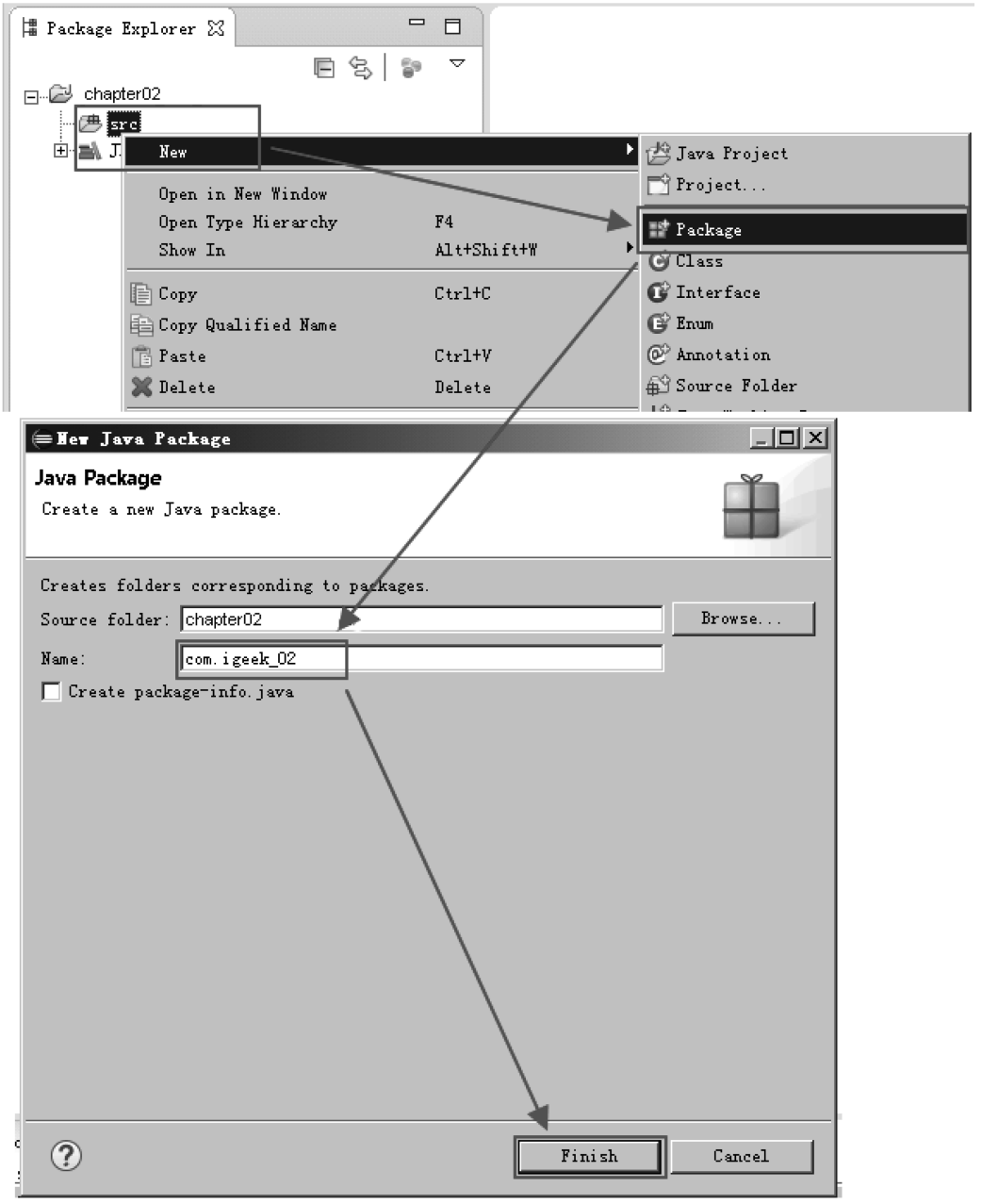


图 2-6　创建 Java 包

（3）创建类：在 com. igeek_02 包中新建一个 Class，Class 的 Name 值为：HelloWorld，然后点击 Finish 即可。(如图 2-7 所示)

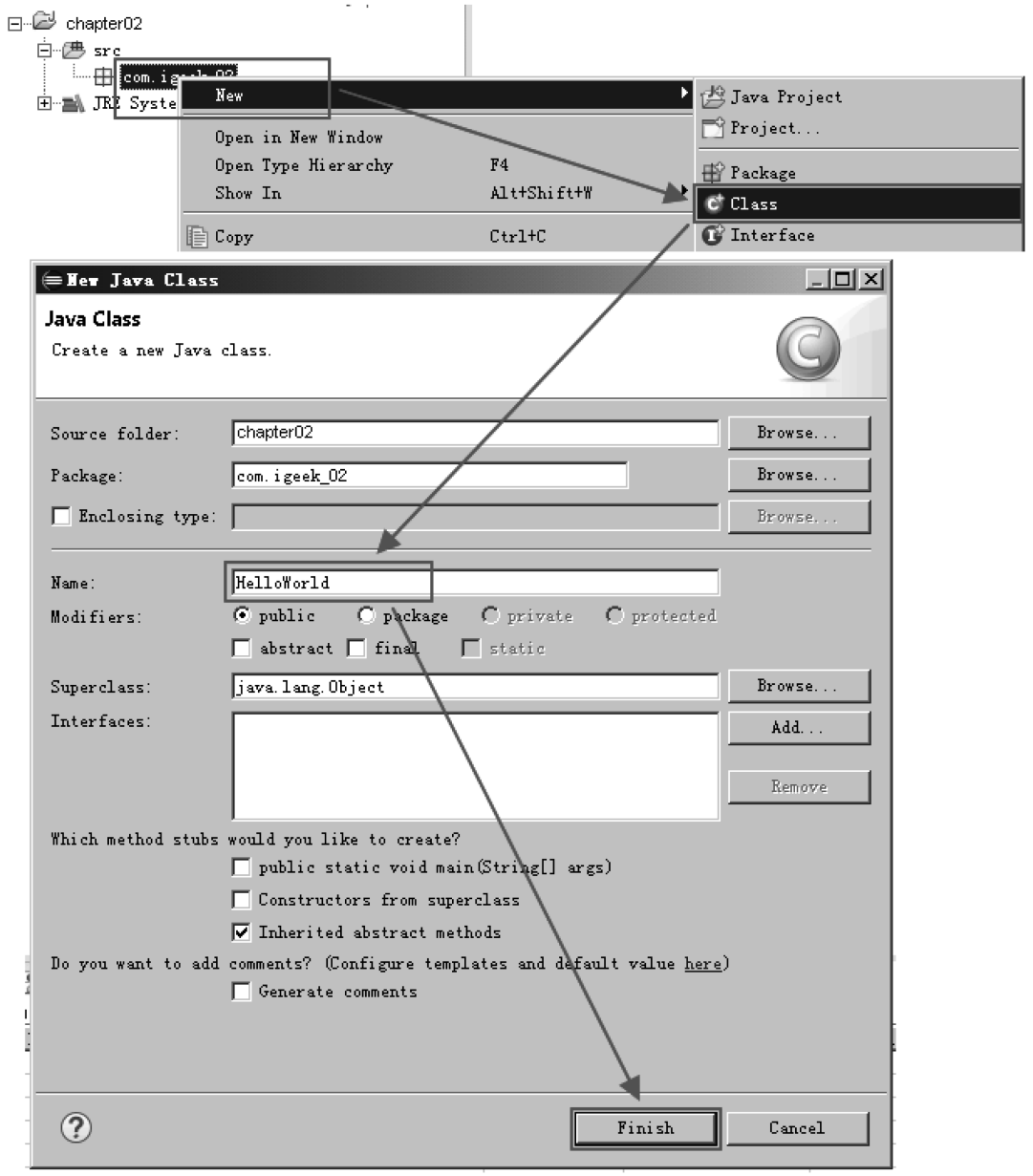


图 2-7　创建 Java 类

（4）编写代码：在 HelloWorld 类写 main 方法，在 main 方法中写一条输出语句：我是极客营程序员，我骄傲，我自豪。

（5）编译：自动编译，在保存的那一刻就已经编译了。

（6）运行：选择要运行的文件或者在要运行的文件内容中，即点击鼠标右键→Run as-Java Application 即可，在控制台可以看到运行结果(如图 2-8 所示)。

图 2-8　运行 Java 程序

2.3.2　在 Eclipse 开发平台中编写第一个应用程序

文件 2-1 HelloWorld. Java

```
package com. igeek_02;

public class HelloWorld {
    public static void main(String[ ] args) {
        System. out. println("我是极客营程序员,我骄傲,我自豪。");
    }
}
```

程序运行结果如下。

2.4 Eclipse 工作空间的基本配置

1. 行号的显示和隐藏(如图 2-9 所示)

显示：在代码区域的最左边的空白区域，点击鼠标右键→Show Line Numbers 即可。

隐藏：把上面的动作再做一次。

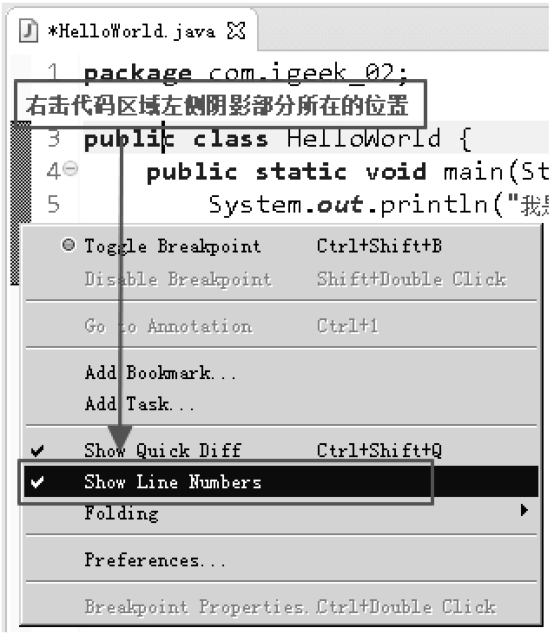


图 2-9 行号的显示和隐藏

2. 字体大小及颜色设置

(1) Java 代码区域的字体大小和颜色设置如图 2-10 所示，即：

Window→Preferences→General→Appearance→Colors And Fonts→Java Edit or Text Font→Edit…

(2) 控制台设置(类似(1)的过程)。

Window→Preferences→General→Appearance→Colors And Fonts→Debug→Console font

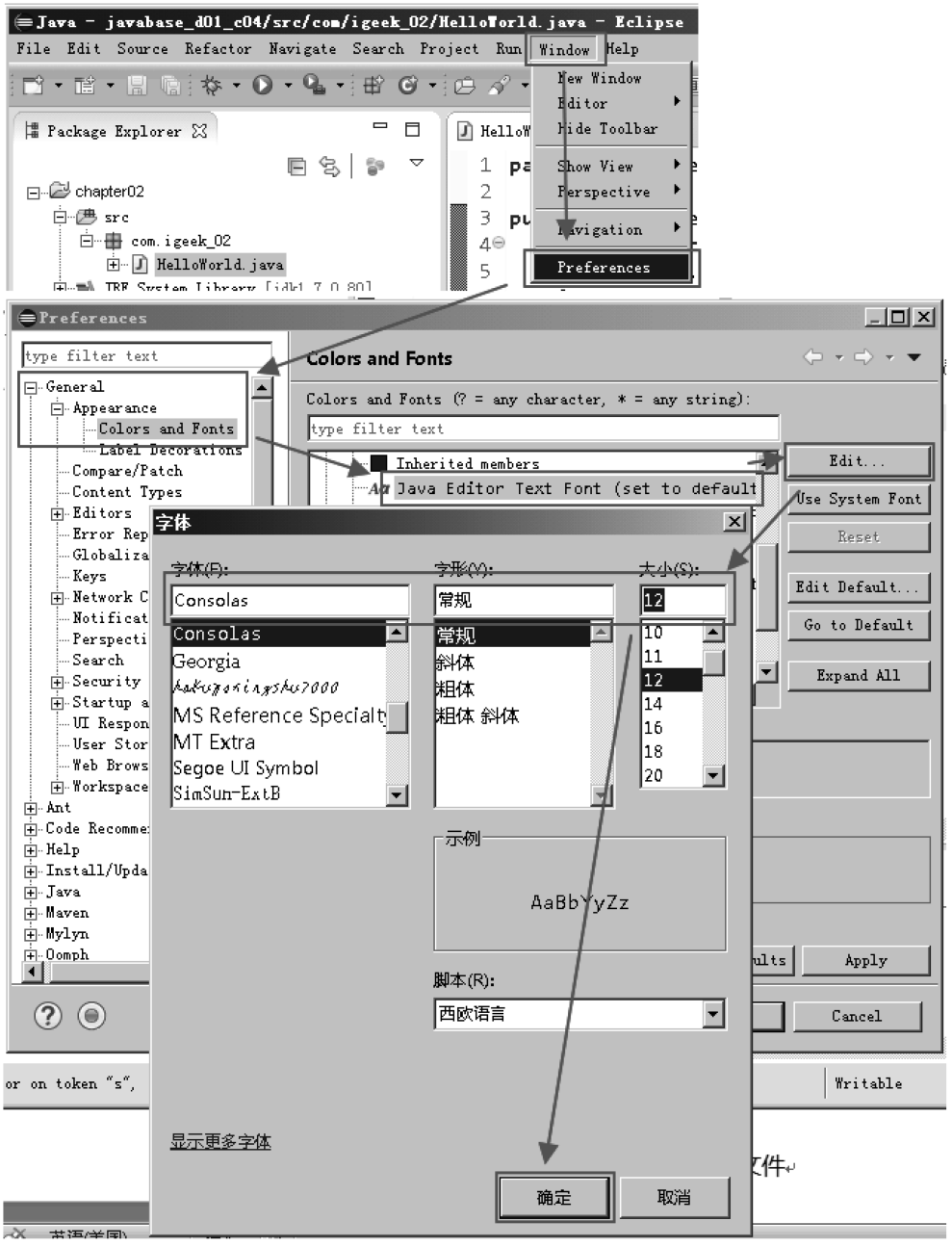


图 2-10　设置 Java 代码区域的字体大小和颜色

(3) 其他文件设置(类似(1)的过程)。

Window→Preferences→General→Appearance→Colors And Fonts→Basic→Text Font

3. 重置窗体布局(如图 2-11 所示)

即：Window→Perspective→Reset Perspective

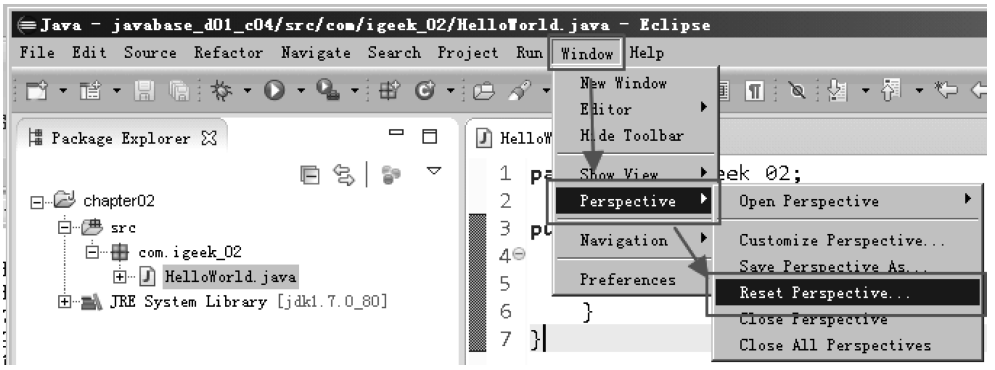


图 2-11　重置窗体布局

4. 打开控制台窗体(如图 2-12 所示)

即：Window→Show View→Console

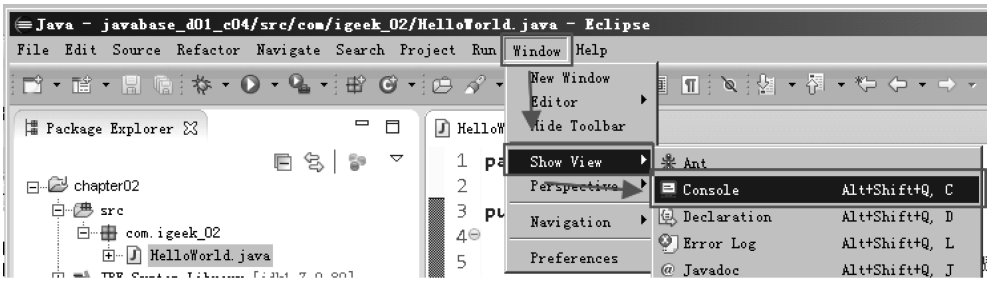


图 2-12　控制台窗体

2.5　Eclipse 中辅助键和快捷键的使用

2.5.1　常用辅助键和快捷键概述

1. 内容辅助键(Alt+/)

(1) 输入语句 main，即 Alt+/：main 方法。

(2) 输出语句 sysout，即 Alt+/：System.out.println()。

2. 快捷键

(1) 单行注释：选中内容，Ctrl+/，如需要取消单行注释，只需重新操作一遍。

(2) 多行注释：选中内容，Ctrl+Shift+/，如需要取消多行注释，Ctrl+Shift+\ 。

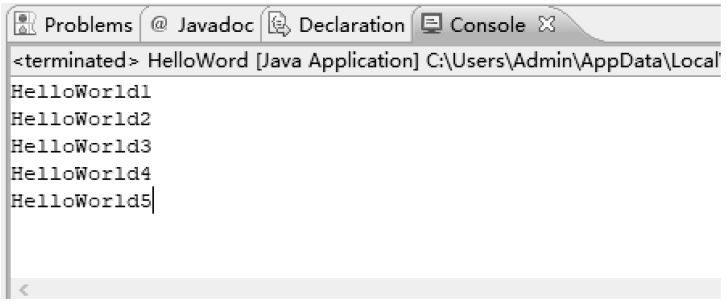
(3) 格式化：Ctrl+Shift+F。

2.5.2　案例代码

文件 2-2　HelloWorld. java

```
package com. igeek_03;

/**
 * 内容辅助键:Alt+/
 * (1) 输入语句 main,即 Alt+/:main,回车
 * (2) 输出语句 sysout,即 Alt+/:System. out. println( ); 回车
 *
 * 快捷键:
 * (1) 注释
 * 单行 选中内容,Ctrl+/, 再来一次就是取消
 * 多行 选择内容,Ctrl+Shift+/(选中), Ctrl+Shift+\(取消)
 * (2) 格式化
 * Ctrl+Shift+F
 */
public class HelloWorld {
    public static void main(String[ ] args) {
        System. out. println("HelloWorld1");
        System. out. println("HelloWorld2");
        System. out. println("HelloWorld3");
        System. out. println("HelloWorld4");
        System. out. println("HelloWorld5");
    }
}
```

运行结果如下。



2.6 Eclipse 中项目的删除和导入

1. 删除项目

选中项目→点击鼠标右键→点击删除，弹出一个对话框，有一个复选框：从项目区域中删除或从硬盘上删除，读者可以根据实际情况选择操作。（如图 2-13 所示）

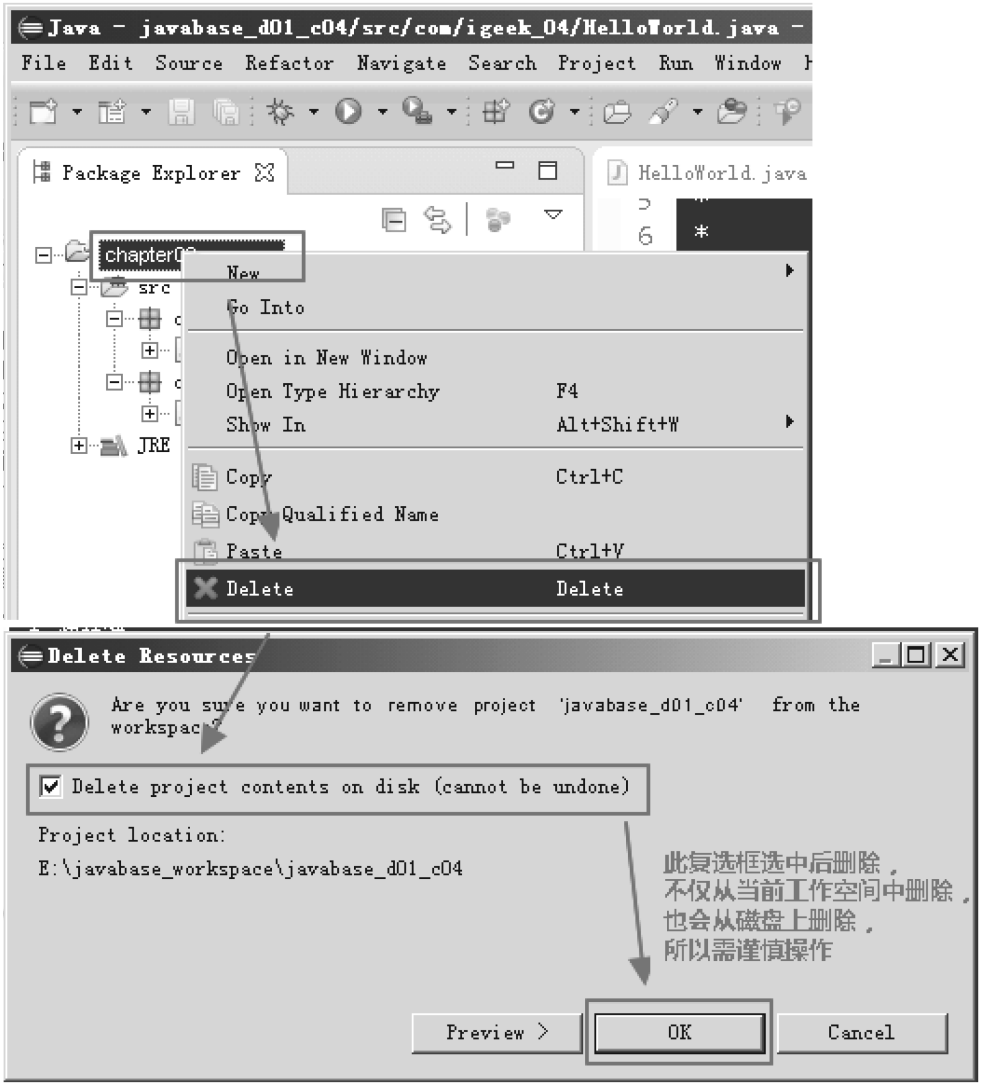


图 2-13　删除项目

2. 导入项目

（1）在项目区域点击鼠标右键，找到 Import 菜单。

（2）找到 General，展开，并找到 Existing Projects into Workspace。

（3）点击 Next，然后选择你要导入的项目。

注意：这里选择的是项目名称(如图 2-14 所示)。

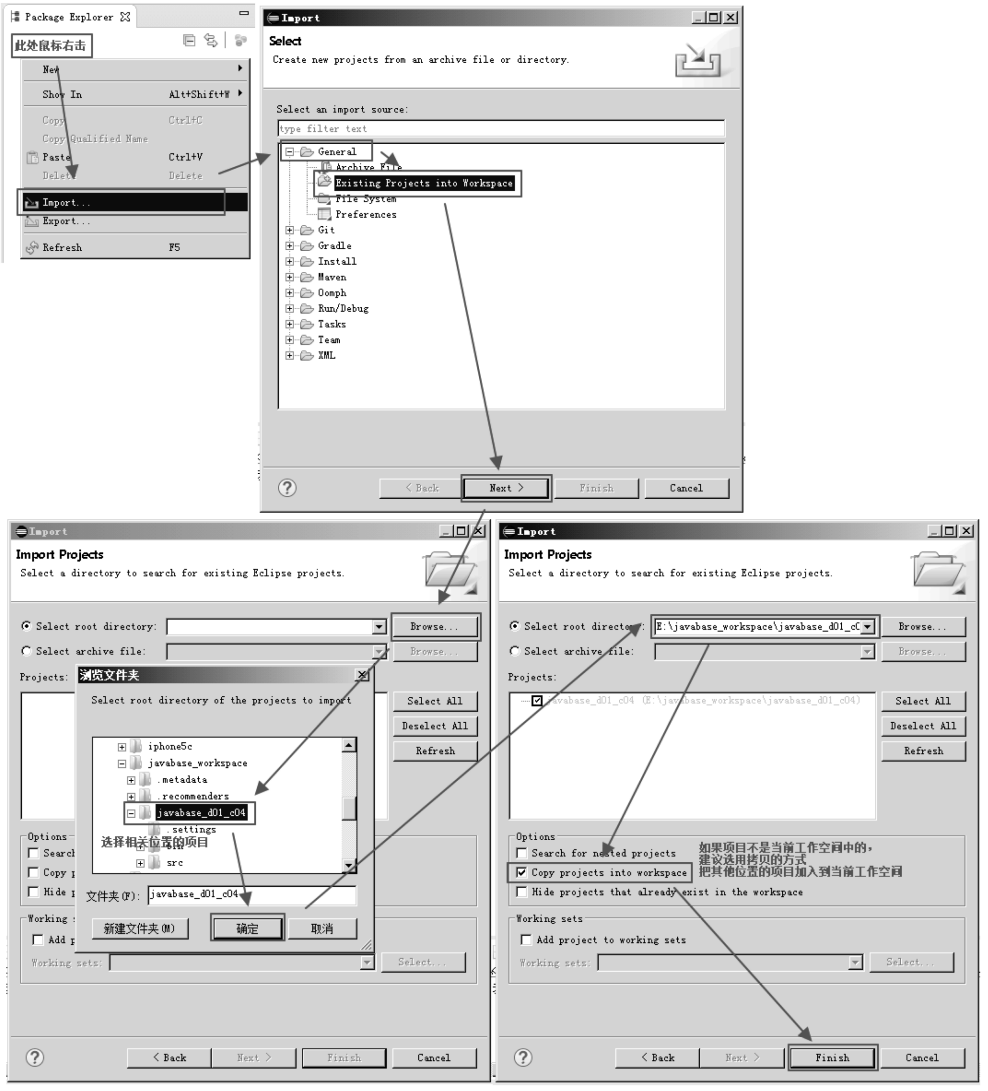


图 2-14　导入项目

本章小结

本章主要讲解了如何配置 path 环境变量以及 Eclipse 开发工具的使用。通过本章的学习，结合一些案例进一步掌握 Eclipse 的基本操作。

习　题

在 Eclipse 中编写一个输入个人基本信息(如所在学校、所属专业、姓名和年龄)的 Java 应用程序。

第 3 章 Java 基础语法

学习目标

【理解】注释、关键字、常量、变量、数据类型、标识符

(1)【理解】阐述注释、关键字的作用及分类、特点。

(2)【理解】阐述常量的特点及分类。

(3)【理解】阐述变量的作用及定义格式。

(4)【理解】阐述数据类型分类。

(5)【理解】阐述标识符的组成规则及其注意事项。

(6)【应用】能够定义变量，使用变量。

(7)【理解】阐述数据类型转换之隐式数据类型转换和强制数据类型转换的方式及其注意事项。

3.1 注 释

3.1.1 注释概述

在应用程序中注释是用于解释或说明程序的文字，注释内容不会被编译器编译，它的主要作用是提高程序的阅读性。Java 中包含以下三种注释方式。

1. 单行注释

格式：//注释文字

2. 多行注释

格式：/* 注释文字 */

3. 文档注释

格式：/** 文档注释 */

修改原有的 HelloWorld. java 程序，添加注释，加以说明。

文件 3-1　HelloWorld. java

```
/**
   注释:用于解释说明程序的文字
   分类:
        (1) 单行
        (2) 多行
        (3) 文档注释
   作用:解释说明程序,提高程序的阅读性
*/
//这是我的 HelloWorld 案例
public class HelloWorld {
   /*
        这是 main 方法
        main 是程序的入口方法
        所有代码的执行都是从 main 方法开始的
   */
   public static void main(String[ ] args) {
        //这是输出语句
        System. out. println("HelloWorld");
   }
}
```

3. 2　关键字

文件 3-2　HelloWorld. java

```
/*
     关键字:被 Java 语言赋予特定含义的单词

     特点:
          (1) 组成关键字的字母全部小写
          (2) 常见的代码编辑器,针对关键字有特殊的颜色标记
*/
public class HelloWorld {
     public static void main(String[ ]args) {
          System. out. println("HelloWorld");
     }
}
```

3.2.1　关键字概述

在 Java 应用程序中有一些特殊的符号，如上述文件中的 public、class、static、void 等，这些符号被 Java 语言赋予特定含义，它们被称为关键字。表 3-1 列出了 Java 编程语言的关键字。

表 3-1　Java 编程语言的关键字

用于定义数据类型的关键字				
dlass	interface	byte	short	int
long	float	double	char	boolean
void				
用于定义数据类型值的关键字				
true	false	null		
用于定义流程控制的关键字				
if	else	switch	case	default
while	do	for	break	continue
return				
用于定义访问权限修饰符的关键字				
private	protected	public		
用于定义类、函数、变量修饰符的关键字				
abstract	final	static	synchronized	
用于定义类与类(接口)之间关系的关键字				
extends	implements			
用于定义建立实例及引用实例，判断实例的关键字				
new	this	super	instanceof	
用于导常处理的关键字				
try	catch	finally	throw	throws
用于包的关键字				
package	import			
其他修饰符关键字				
native	strictfp	transient	volatile	assert

3.2.2　关键字特点

（1）组成关键字的字母全部小写。

（2）常用的代码编辑器，针对关键字有特殊的颜色标记，非常直观，所以我们不需要

去死记硬背，在今后的学习中重要的关键字也会不断地出现。

3.3 常　量

3.3.1 常量概述

在程序执行的过程中，其值不可以发生改变的量称为常量。

3.3.2 常量分类

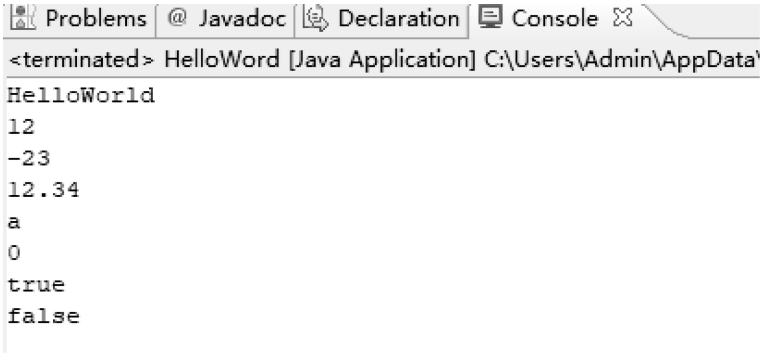
（1）字符串常量是用双引号括起来的内容（如“HelloWorld”）。
（2）整数常量表示所有整数（如 12，-23）。
（3）小数常量表示所有小数（如 12.34）。
（4）字符常量用单引号括起来的内容（如‘a’，‘A’，‘0’）。
（5）布尔常量较为特殊，只有 true 和 false。
（6）空常量 null 表示没有的意思。

文件 3-3　ChangLiang.java

```
/*
    常量:在程序执行的过程中,其值不可以发生改变的量

    常量分类:
        (1)字符串常量"HelloWorld"
        (2)整数常量 12, - 23
        (3)小数常量 12.34
        (4)字符常量'a' , '0'
        (5)布尔常量 true, false
        (6)空常量 null(后面讲解)
*/
public class ChangLiang {
    public static void main(String[ ] args) {
        //字符串常量
        System.out.println("HelloWorld");

        //整数常量
        System.out.println(12);
        System.out.println(- 23);

        //小数常量
```

```
            System. out. println(12. 34);

            //字符常量
            System. out. println('a' );
            System. out. println('0' );

            //布尔常量
            System. out. println(true);
            System. out. println(false);
        }
    }
```

运行结果如下。



3.4　变　量

3.4.1　变量概述

在程序执行的过程中，在某个范围内其值可以发生改变的量称为变量。

从本质上讲，变量其实是内存中的一小块区域。

3.4.2　变量定义格式

数据类型　变量名=初始化值;

注意：格式是固定的，记住格式，以不变应万变。

3.4.3 变量图解(如图 3-1 所示)

变量：是内存中一块区域。在程序的执行过程中，其值是可以在一定范围内发生改变的量

组成：

(1) 对区域要有限定
如何限定呢?用数据类型
(2) 必须对区域起个名字
变量名
(3) 区域内必须有内容
初始化值

定义格式：

数据类型 变量名 = 初始化值;

图 3-1 变量图解

3.5 数据类型

3.5.1 计算机存储单元

变量是内存中的小容器，用来存储数据。那么计算机内存是怎么存储数据的呢？无论是内存还是硬盘，计算机存储设备的最小信息单元叫“位(bit)”，我们又称之为“比特位”，通常用小写的字母 b 表示。而计算机最小的存储单元叫“字节(Byte)”，通常用大写字母 B 表示，字节是由连续的 8 个位组成。

3.5.2 数据类型概述和分类

Java 语言是强类型语言，对于每一种数据都定义了明确的具体数据类型，在内存中分配了不同大小的内存空间。Java 中数据类型分为：基本数据类型和引用数据类型，如图 3-2 所示。基本数据类型介绍如表 3-2 所示。

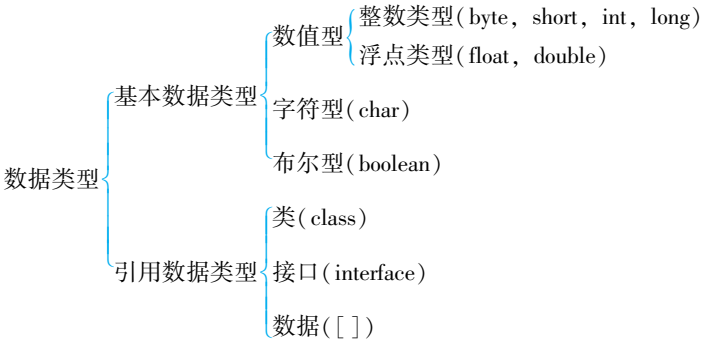


图 3-2 Java 的数据类型

表 3-2　基本数据类型介绍

四类	八种	字节数	数据表示范围
整型	byte	1	-128~127
	short	2	-32768~32767
	int	4	-2147483648~2147483648
	long	8	-2^{63}~$2^{63}-1$
浮点型	float	4	-3.403E38~3.403E38
	double	8	-1.798E308~1.798E308
字符型	char	2	表示一个字符，如('a'，'A'，'0'，'家')
布尔型	boolean	1	只有两个值 true 与 false

3.6　标识符

3.6.1　标识符概述

1. 作用

标识符的作用主要是为包、类、常量、变量、方法等命名，用于标识它们。

2. 组成规则

(1) 由字符(字母、数字)、下划线(_)、美元符($)组成。
(2) 不能以数字开头。
(3) 不能是 Java 中的关键字。
例如：
合法的标识符有 yourname、your_name、_yourname、 $yourname 等；
非法的标识符有 class、67.9、Hello Careers 等。

3.6.2　标识符命名原则

标识符的命名原则主要是“见名知意”。

1. 包

最好是公司域名倒置，要求所有的字母小写，如：公司域名是 igeek.com，则包名为 com.igeek。

2. 类或者接口

如果是一个单词，则单词的首字母大写；如果是多个单词组成，则每个单词的首字母大写(驼峰标识)。

3. 方法或者变量

如果是一个单词，则全部小写；如果是多个单词组成，则从第二个单词起首字母大写

4. 常量

如果是一个单词，则所有字母大写；如果是多个单词组成，则所有的单词大写，然后用下画线连接每个单词。

文件 3-4　BiaoShiFu. java

```
/*
    标识符:就是给包,类,方法,变量起名字的符号

    组成规则:
        (1) Unicode 字符
        数字字符,英文大小写,汉字(不建议使用汉字)
        (2) 下划线_
        (3) 美元符 $

    注意事项:
        (1) 不能以数字开头
        (2) 不能是 Java 中的关键字

    常见命名规则:
        (1) 基本要求
            见名知意
        (2) 常见的命名
            ① 包(其实就是文件夹,用于对类进行管理)
                全部小写,多级包用 . 隔开
                举例:com,com. igeek
            ② 类
                一个单词首字母大写
                    举例:Student, Car
                多个单词每个单词的首字母大写
                    举例:HelloWorld
            ③ 方法和变量
                一个单词全部小写
                    举例:age, show( )
                多个单词从第二个单词开始每个单词的首字母大写
                    举例:maxAge, getAge( )
            ④ 常量
                所有字母大写
*/
public class BiaoShiFu {
```

```
    public static void main(String[ ] args) {
        //定义变量
        //数据类型变量名=初始化值;
        int a=10;

        //正确
        int b2=20;
        //错误
        //int 2b=30;

        //不能是 java 中的关键字
        //错误
        //int public=40;
    }
}
```

3.7　定义变量

3.7.1　基本数据类型变量的定义和使用

变量的定义格式如下。

数据类型　变量名=初始化值;

基本数据类型如下。

byte、short、int、long、float、double、char、boolean

注意：

① 整数默认是 int 类型，定义 long 类型的数据时，要在数据后面加 L;

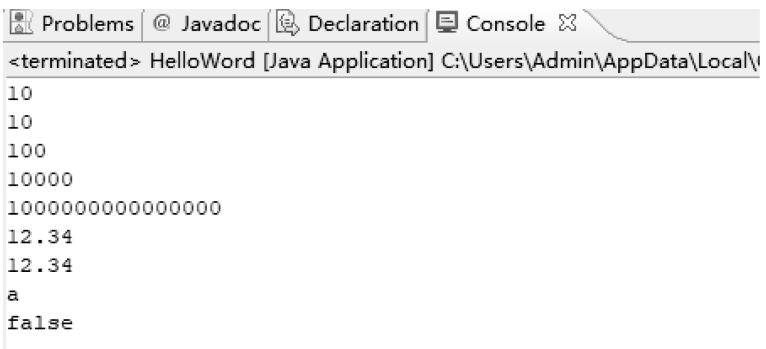
② 浮点数默认是 double 类型，定义 float 类型的数据时，要在数据后面加 F。

文件 3-5　VariableDemo. java

```
public class VariableDemo {
    public static void main(String[ ] args) {
        //定义 byte 类型的变量
        byte b=10;
        System. out. println(10);
        System. out. println(b);

        //定义 short 类型的变量
        short s=100;
```

```
        System. out. println(s);

        //定义 int 类型的变量
        int i=10000;
        System. out. println(i);

        //定义 long 类型的变量
        long l=1000000000000000L;
        System. out. println(l);

        //定义 float 类型的变量
        float f=12. 34F;
        System. out. println(f);

        //定义 double 类型的变量
        double d=12. 34;
        System. out. println(d);

        //定义 char 类型的变量
        char c='a';
        System. out. println(c);

        //定义 boolean 类型的变量
        boolean bb=false;
        System. out. println(bb);
    }
}
```

运行结果如下。



3.7.2　变量定义的注意事项

（1）变量未赋值，不能直接使用。

（2）变量只在它所属的范围内有效，即变量在哪对大括号内，变量的作用范围就局限在哪对大括号内。

（3）一行上可以定义多个变量，但是不建议这样。

文件 3-6　VariableDemo2. java

```
/*
    变量定义注意事项:
        (1) 变量未赋值,不能直接使用
        (2) 变量只在它所属的范围内有效,即变量属于它所在的那对大括号
        (3) 一行上可以定义多个变量,但是不建议这样
*/
public class VariableDemo2 {
    public static void main(String[ ] args) {
        //定义变量
        int a=10;
        System. out. println(a);

        int b;
        b=20; //变量在使用前赋值都是可以的
        System. out. println(b);

        {
            int c=100;
            System. out. println(c);
        }
        //System. out. println(c);

        /*
        int aa, bb, cc;
        aa=10;
        bb=20;
        cc=30;
        */
```

```
            /*
            int aa=10;
            int bb=20;
            int cc=30;
            */

            int aa=10, bb=20, cc=30;
        }
    }
```

3.8 数据类型转换

3.8.1 隐式数据类型转换

取值范围小的数据类型与取值范围大的数据类型进行运算，会先将小的数据类型提升为大的，再运算。隐式数据类型转换图如图 3-3 所示。

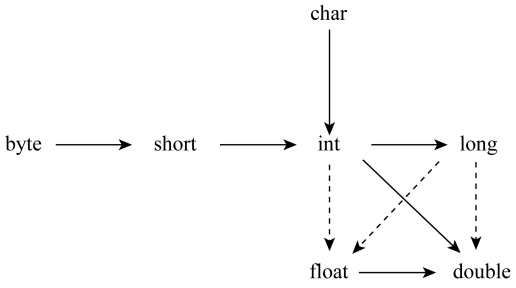


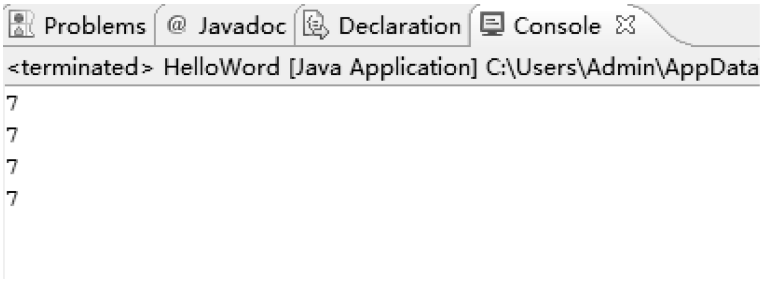
图 3-3 隐式数据类型转换图

文件 3-7 TypeCastDemo.java

```
/*
    +:是一个运算符,做加法运算的
    我们在做运算的时候,一般要求参与运算的数据类型必须一致

    类型转换:
        隐式转换
        强制转换

    隐式转换
        byte, short, char→int→long→float→double
*/
```

```
public class TypeCastDemo {
    public static void main(String[ ] args) {
        //直接输出运算的结果
        System. out. println(3+4);

        //定义两个 int 类型的变量
        int a=3;
        int b=4;
        int c=a+b;
        System. out. println(c);

        //定义一个 byte 类型,定义一个 int 类型
        byte bb=2;
        int cc=5;
        System. out. println(bb+cc);

        //能不能不直接输出,用一个变量接受呢?
        //用变量接受,这个变量应该有类型
        //可能损失精度
        //byte dd=bb+cc;
        int dd=bb+cc;
        System. out. println(dd);
    }
}
```

运行结果如下。



3.8.2　强制类型数据转换

取值范围小的数据类型与取值范围大的数据类型进行运算，如果将取值范围大的数据类型转换成取值范围小的数据类型，则必须进行强制类型转换。其转换格式为：

(类型名)表达式；

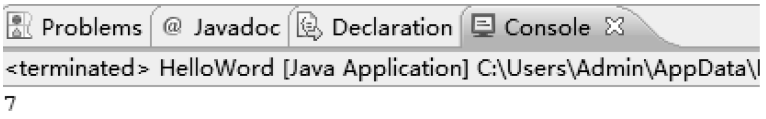
如 int a；byte b=(byte)a；

强制转换的注意事项：强制类型转换会造成数据精度损失。

文件 3-8　TypeCastDemo2. java

```
/*
    强制转换：
        目标类型变量名=(目标类型)(被转换的数据);
    不建议强制转换,因为会有精度的损失
*/
public class TypeCastDemo2 {
    public static void main(String[ ] args) {
        int a=3;
        byte b=4;
        int c=a+b;
        //byte d=a+b;
        byte d=(byte) (a+b);
        System.out.println(d);      }
}
```

运行结果如下。



本章小结

本章主要讲解了注释、关键字、常量及变量。首先讲解了注释、关键字的作用及其分类和特点，其次介绍了常量的特点及分类，变量的作用及定义格式，以及数据类型分类，最后介绍标识符的组成规则及注意事项，定义变量，使用变量，数据类型转换之隐式数据类型转换和强制数据类型转换的方式及注意事项。通过本章的学习，结合不同的案例，让读者进一步掌握常量、变量、数据类型及标识符等的使用方法。

习　题

一、选择题

1. 下列(　　)是合法的标识符。

A. 12class　　B. void　　C. -5　　D. _blank

2. 下列(　　)不是 Java 中的保留字。

A. if　　B. sizeof　　C. private　　D. null

3. 下列(　　)不是合法的标识符。

A. $million　　B. $_million　　C. 2$_million　　D. $2_million

4. 下列选项中，(　　)不属于 Java 语言的基本数据类型。

A. 整数型　　B. 数组　　C. 浮点型　　D. 字符型

5. 下列关于基本数据类型的说法中，不正确的一项是(　　)。

A. boolean 类型变量的值只能取真或假

B. float 是带符号的 32 位浮点数

C. double 是带符号的 64 位浮点数

D. char 是 8 位 Unicode 字符

6. 下列关于基本数据类型的取值范围的描述中，正确的是(　　)。

A. byte 类型的取值范围是-128~128

B. boolean 类型的取值范围是真或假

C. char 类型的取值范围是 0~65536

D. short 类型的取值范围是-32767~32767

7. 下列关于 Java 语言简单数据类型的说法中，正确的一项是(　　)。

A. 以 0 开头的整数代表 8 进制整型常量

B. 以 0x 或 0X 开头的整数代表 8 进制整型常量

C. boolean 类型的数据作为类成员变量的时候，相同默认的初始值为 true

D. double 型数据占计算机存储的 32 位

8. 下列 Java 语句中，不正确的一项是(　　)。

A. $e, a, b=10;　　B. char c, d='a';

C. float e=0.0d; .　　D. double c=0.0f;

9. 设 a、b 为 long 型变量，x、y 为 float 型变量，ch 为 char 类型变量且它们均已被赋值，则下列语句中正确的是(　　)。

A. switch(x+y) {}　　B. switch(ch+1) {}

C. switch ch {}　　D. switch(a+b); {}

10. 下列语句中不正确的是(　　)。

A. float f=1.1f;　　B. byte b=128;

C. double d=1.1/0.0;　　D. char c=(char)1.1f;

二、填空题

1. 变量是 Java 程序的基本存储单元之一，变量的主要类型包括两大类：_____和_____。

2. Java 语言的整数类型变量和常量一样，各自都包括四种类型的数据，它们分别是 byte、_____、_____和 long。

3. _____类型数据不可以做类型转换。

4. 在 Java 语言的基本数据类型中，占存储空间最少的类型是_____，该类型占用的存储空间为_____位。

5. Java 语言中的_____具有特殊意义和作用，不能作为普通标识符使用。

6. 在 Java 语言中，浮点类型数据属于实型数据，可以分为_____和_____两种。

7. char 类型的数据可以表示的字符数共为_____。

8. 定义初始值为 10 的 8 次方的常整型变量 iLong 的语句是_____。

9. Java 语言中的数据类型转换包括_____和_____ 两种。

10. Java 中的字符采用的是 16 位的_____编码。

11. 数据类型中存储空间均为 64 位的两种数据类型是_____和_____。

12. 表达式“9 * 4/-5%5”的值为_____。（十进制表示）

13. 表达式“5&2”的值为_____。（十进制表示）

14. 表达式“42<<4”的值为_____。（十进制表示）

15. 表达式“11010011>>>3”的值为_____。（二进制表示）

16. 表达式“7 | 3”的值为_____。（十进制表示）

17. 表达式“10^2”的值为_____。（十进制表示）

18. Java 语言中的逻辑与(&&)和逻辑或(||)运算采用_____方式进行运算。

19. 若 a、b 为 int 型变量，并且已分别赋值为 5 和 10，则表达式“(a++)+(++b)+a * b”的值为_____。

20. 假设 i=10，j=20，k=30，则表达式“!(i<j+k)||!(i+10<=j)”的值为_____。

三、判断题

1. Java 语言使用的是 Unicode 字符集，每个字符在内存中占 8 位。 (　　)

2. Java 语言中不同数据类型的长度是固定的，不随机器硬件不同而改变。 (　　)

3. 所有的变量在使用前都必须进行初始化。 (　　)

4. 已知“byte i=(byte)127; i=i+1;”这两个语句能被成功编译。 (　　)

5. String str="abcdefghi"; char chr=str. charAt(9); (　　)

6. int i, j; boolean booleanValue=(i==j); (　　)

7. int int Array[]={0, 2, 4, 6, 8}; int length=int Array. length(); (　　)

8. String str="abcedf"; int length=str. length; (　　)

9. short shortValue=220; byte byteValue=shortValue; (　　)

10. int[] int Array[60]; (　　)

四、程序题

输入扇形的半径和角度，在控制台输出扇形的周长和面积。要求：扇形的周长只保留整数部分，舍掉小数部分。