21世纪高职高专规划教材
高等职业教育规划教材编委会专家审定

Android KAIFA SHILIHUA JIAOCHENG

Android 开发实例化教程

主　编　李　莉　路永涛
副主编　吴蓬勃　张　静

北京邮电大学出版社
www.buptpress.com

21世纪高职高专规划教材

高等职业教育规划教材编委会专家审定

Android开发实例化教程

主　编　李　莉　路永涛

副主编　吴蓬勃　张　静

北京邮电大学出版社
www.buptpress.com

内 容 简 介

本书以 Android 手机应用开发为主题，通过多个简单实用的实例全面地整合了 Android 手机开发所需的界面布局、基本控件、数据存储等技术，可以很好地帮助读者学习 Android 开发技术、提高自身的程序设计能力，更好地适应移动互联时代下的技术发展。

本书内容共分为 4 章。第 1 章讲解 Java 的基础知识；第 2 章讲解搭建 Android 开发环境的方法；第 3 章讲解 Android 中开发基本界面的方法；第 4 章讲解 Android 的高级开发，包括数据存储技术、多媒体播放器的开发。

本书内容丰富，实例经典，讲述语言简洁、由浅入深，既可作为各级各类高职院校学生的程序设计教材，也可作为软件开发人员的参考书或 Android 爱好者的自学参考书。

图书在版编目(CIP)数据

Android 开发实例化教程 / 李莉，路永涛主编. --北京：北京邮电大学出版社，2015.8
ISBN 978-7-5635-4411-0

Ⅰ. ①A… Ⅱ. ①李… ②路… Ⅲ. ①移动终端—应用程序—程序设计—教材 Ⅳ. ①TN929.53

中国版本图书馆 CIP 数据核字(2015)第 143297 号

书　　名：Android 开发实例化教程
著作责任者：李　莉　路永涛　主编
责 任 编 辑：张珊珊
出 版 发 行：北京邮电大学出版社
社　　址：北京市海淀区西土城路 10 号(邮编：100876)
发 行 部：电话：010-62282185　传真：010-62283578
E-mail：publish@bupt.edu.cn
经　　销：各地新华书店
印　　刷：北京睿和名扬印刷有限公司
开　　本：787 mm×1 092 mm　1/16
印　　张：13.5
字　　数：347 千字
版　　次：2015 年 8 月第 1 版　2015 年 8 月第 1 次印刷

ISBN 978-7-5635-4411-0　　定价：29.00 元

前　言

Android操作系统是由Google公司和开放手机联盟共同开发并发展的移动设备操作系统，目前已经成为炙手可热的智能手机操作系统。Android系统易学、易用、功能强大，极大地降低了开发嵌入式应用程序的难度，大大地提高了程序开发的效率。为了帮助各院校师生全面、系统地学习这门课程，熟练地使用Android进行软件开发，笔者结合自己近几年的Android教学及开发经验，编写了这本《Android开发实例化教程》，希望读者在本书的引领下进入Android开发世界，并成为一名合格的Android开发人员。

Android SDK使用业界常用的Java语言开发，熟悉Java语言是开发的基础，所以我们首先从Android开发所必备的Java基础知识开始讲解。接着讲解Android开发环境的搭建，涵盖了Android程序的界面布局、控件使用、数据存储、多媒体播放器的开发等。书中的知识点都是通过一个个实际的应用实例来讲解的。

本书的特色在于：内容安排上从实际应用出发，以实例带动技术讲解，实用性强，且容易上手；实例的选择上注重由浅入深，突出重点，让涵盖的技术要点一目了然，明确直观，容易掌握；实例的讲解上配以源代码和效果图，让读者快速入门、理解；文字的叙述上注重言简意赅、重点突出。

本书由李莉、路永涛、吴蓬勃、张静共同编写。李莉负责全书的编排及统稿，并编写了第4章；张静编写了第1章；路永涛编写了第2章；吴蓬勃编写了第3章。

在本书的编写过程中，编者得到了石家庄邮电职业技术学院电信工程系领导、老师的关心和支持，在此表示衷心的感谢！

由于Android本身技术发展日新月异，而编者经验和水平有限，书中不妥及疏漏之处在所难免，敬请读者给予批评指正。

编　者

目　录

第1章　Java编程基础

【内容简介】

本章主要介绍了Java基础知识，包括Java的环境搭建；Java的基本语法，其中包括Java的基本数据类型、Java变量和运算符、Java的流程控制等；类与对象的概念，包括对象与类的关系、面向对象和面向过程的区别等。

【重点难点】

重点：Java的基本语法；类与对象的概念。

难点：Java的环境搭建；对象与类的关系及抽象方法；掌握面向对象的编程方法。

1.1　Java程序设计概述

Java是一种高级的面向对象的程序设计语言，它提供了一个同时用于程序开发、应用和部署的环境。Java语言主要定位于网络编程，使得程序可以最大限度地利用网络资源。

1.1.1　什么是Java语言

Java是在1995年由Sun公司(Sun公司在2009年被Oracle公司收购)推出的一种极富创造力的面向对象的程序设计语言，它由Java之父James Gosling亲手设计，并完成了Java技术的原始编译器和虚拟机。Java最初的名字是OAK，在1995年被重命名为Java。

Java是一种通过解释方式来执行的语言，其语法规则和C++类似。同时，Java也是一种跨平台的程序设计语言。用Java语言编写的程序，可以运行在任何平台和设备上，例如跨越IBM个人电脑、MAC苹果系统、各种微处理器硬件平台，以及Windows、UNIX、OS/2、MAC OS等系统平台，从真正意义上实现了“一次编写、到处运行”。Java非常适合于企业网络和Internet环境，并且已成为Internet中最有影响力、最受欢迎的编程语言之一。

与目前常用的C++相比，Java语言简洁得多，而且提高了可靠性，除去了最大的程序错误根源，此外它还有较高的安全性，可以说它是有史以来最为卓越的编程语言。

Java语言编写的程序既是编译型的，又是解释型的。程序代码经过编译之后转换为一种称为Java字节码的中间语言，Java虚拟机JVM将对字节码进行解释和运行。编译只进行一次，而解释在每次运行程序时都会进行。编译后的字节码采用一种针对JVM优化过的机器码形式保存，虚拟机将字节码解释为机器码，然后在计算机上运行。Java语言程序代码的编译和运行过程如图1-1所示。

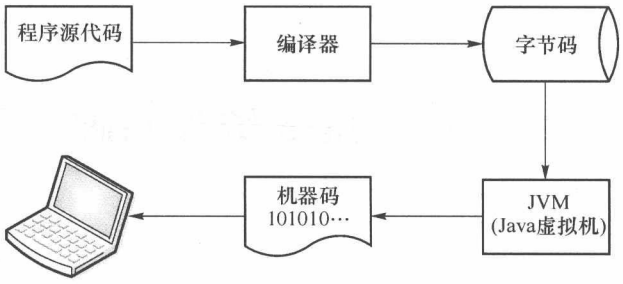

图 1-1　Java 程序的编译和运行过程

1.1.2　Java 的应用领域

Java 能做的事情很多，涉及编程领域的各个方面。

桌面级应用：尤其是需要跨平台的桌面级应用程序。先解释一下桌面级应用，简单地说就是主要功能都在本机上运行的程序，比如 Word、Excel 等运行在本机上的应用就属于桌面应用。

企业级应用：先解释一下企业级应用，简单地说就是大规模的应用，一般使用人数较多，数据量较大，对系统的稳定性、安全性、可扩展性和可装配性等都有比较高的要求。这是目前 Java 应用最广泛的一个领域，几乎一枝独秀。包括各种行业应用、企业信息化，也包括电子政务等，领域涉及办公自动化 OA、客户关系管理 CRM、人力资源 HR、企业资源计划 ERP、知识管理 KM、供应链管理 SCM、企业设备管理系统 EAM、产品生命周期管理 PLM、面向服务体系架构 SOA、商业智能 BI、项目管理 PM、营销管理、流程管理 WorkFlow、财务管理等几乎所有你能想到的应用。

嵌入式设备及消费类电子产品：包括无线手持设备、智能卡、通信终端、医疗设备、信息家电（如数字电视、机顶盒、电冰箱）、汽车电子设备等都是近年来热门的 Java 应用领域，尤其是手机上的 Java 应用程序和 Java 游戏，更是普及。

除了上面提到的，Java 还有很多功能，如进行数学运算、显示图形界面、进行网络操作、进行数据库操作、进行文件操作等。

Java 无处不在，它可应用于任何地方、任何领域，并且已拥有成百上千万个用户，其发展速度要快于在它之前的任何一种计算机语言。Java 能够给企业和最终用户带来数不尽的好处。

1.1.3　Java 的版本

自从 Sun 推出 Java 以来，就力图使之无所不能。Java 发展至今，可以按照应用范围不同分成 3 种版本，分别是 Java 标准版（JSE）、Java 企业版（JEE）和 Java 微缩版（JME），每一种版本都有自己的功能和应用方向。本节将分别介绍这 3 个 Java 版本。

1. Java 标准版：JSE（Java Standard Edition）

Java SE 就是 Java 的标准版，是 Sun 公司针对桌面开发以及低端商务计算解决方案而开发的版本，例如我们平常熟悉的 Application 桌面应用程序。Java 标准版是个基础版本，它包含 Java 语言基础、JDBC 数据库操作、I/O 输入输出、网络通信、多线程等技术。本教材主要讲

的就是 JSE。

2. Java 企业版:JEE(Java Enterprise Edition)

Java EE 是 Java2 的企业版,是一种利用 Java 平台来简化企业解决方案的开发、部署以及管理相关复杂问题的体系结构。JEE 技术的基础就是核心 Java 平台或 Java 平台的标准版,JEE 不仅巩固了标准版中的许多优点,例如"编写一次、随处运行"的特性、方便存取数据库的 JDBC API、CORBA 技术以及能够在 Internet 应用中保护数据的安全模式等,同时还提供了对 EJB(Enterprise Java Beans)、Java Servlets API、JSP(Java Server Pages)以及 XML 技术的全面支持。其最终目的就是成为一个能够使企业开发者大幅缩短投放市场时间的体系结构。主要用于开发企业级分布式的网络程序,如电子商务网站和 ERP 系统。

3. Java 微缩版:JME(Java Micro Edition)

Java ME 是对标准版 JSE 进行功能缩减后的版本,主要应用于嵌入式系统开发,如寻呼机、移动电话等移动通信电子设备。JME 在开发面向内存有限的移动终端(例如寻呼机、移动电话)的应用时,显得尤其实用。因为它是建立在操作系统之上的,使得应用的开发无须考虑太多特殊的硬件配置类型或操作系统。因此,开发商也无须为不同的终端建立特殊的应用,制造商也只需要简单地使它们的操作平台可以支持 JME 便可。现在大部分手机厂商所生产的手机都支持 Java 技术。

1.1.4　Java 程序设计环境

Java 编程的初学者会经常听到老师强调"工欲善其事,必先利其器"这句话。在学习 Java 语言之前,必须先搭建好它所需要的开发环境。要编译和执行 Java 程序,JDK(Java Developers Kits)是必备的。下面将具体介绍下载并安装 JDK 和配置环境变量的方法。

1. JDK 下载

JDK 是整个 Java 的核心,包括了 Java 运行环境 JRE(Java Runtime Envirnment)、一些 Java 工具和 Java 基础的类库(rt.jar)。

JDK 的一个常用版本 JSE(Java SDK Standard Edition)可以从 Oracle 的 Java 网站上下载到:http://www.oracle.com/technetwork/java/javase/downloads/index.html,我们建议下载最新版本的,本教材使用的是 Java 7。

2. Windows 系统的 JDK 环境

(1) JDK 安装

下载 Windows 平台的 JDK 安装文件"jdk-7u7-windows-i586.exe"后,运行安装,期间选择安装路径(本教材的安装路径为 C:\Program Files\Java)。

(2) 配置环境变量

在 Windows 系统中配置环境变量的步骤如下。

① 在"计算机"图标上右击,选择"属性"命令,在弹出的对话框中选择"高级系统设置"选项卡,然后单击"环境变量"按钮,将弹出"环境变量"对话框,如图 1-2 所示。单击"系统变量"栏中的"新建"按钮,创建新的系统变量。

② 在如图 1-3 所示的"新建系统变量"对话框中,分别输入变量名"JAVA_HOME"和变量值"C:\Program Files\Java\jdk1.7.0_07",其中变量值是笔者的 JDK 安装路径,读者需要根据自己的计算机环境进行修改。单击"确定"按钮,关闭"新建系统变量"对话框。

③ 在如图 1-2 所示的"环境变量"对话框中双击 Path 变量对其进行修改,在原变量值之前

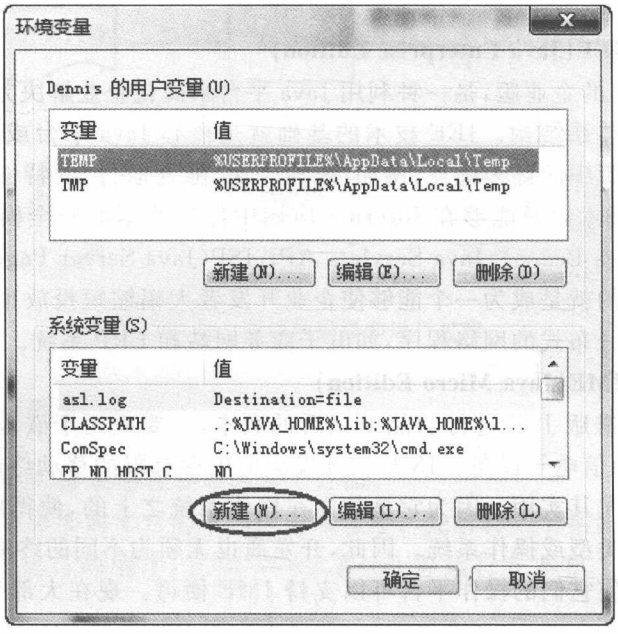

图 1-2 “环境变量”对话框

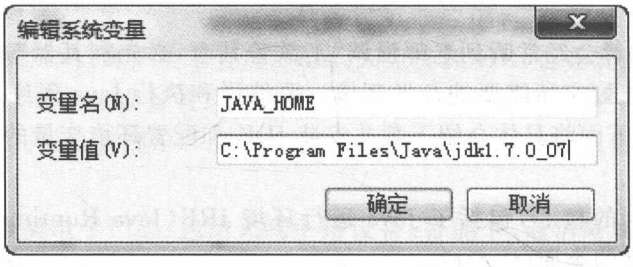

图 1-3 “新建系统变量”对话框

添加“.;%JAVA_HOME%\bin;”变量值(注意:最后的“;”不要丢掉,它用于分割不同的变量值)。单击“确定”按钮完成环境变量的设置。

④ JDK 安装成功之后必须确认环境变量配置是否正确。在 Windows 系统中测试 JDK 环境需要选择“开始”/“运行”命令,然后在“运行”对话框中输入“cmd”并单击“确定”按钮启动控制台。在控制台中输入“java-version”命令,按 Enter 键,将输出 JDK 的版本,如图 1-4 所示,这说明 JDK 环境搭建成功。

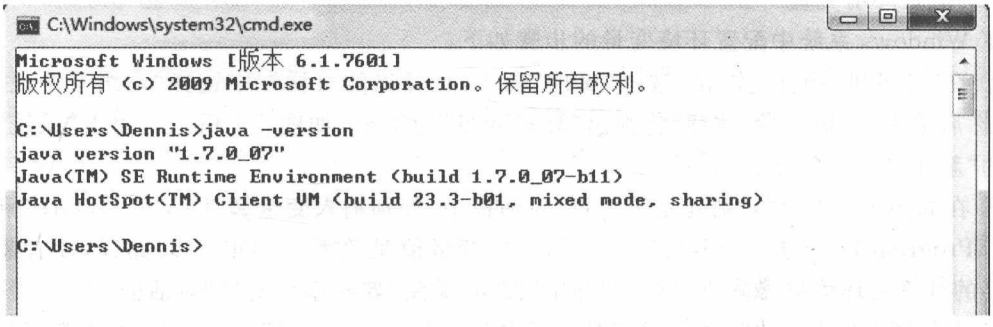

图 1-4 Windows 下测试 JDK 结果

1.1.5　第一个 Java 程序——HelloWorld

编写 Java 应用程序，可以使用任何一个文本编辑器来编写程序的源代码，然后使用 JDK 搭配的工具进行编译(javac. exe)和运行(java. exe)。当然，现在流行的开发工具可以自动完成 Java 程序的编译和运行。本节将介绍使用 UltraEdit 编辑器来开发一个简单的 Java 程序。

实例 1-1　编写一个 Java 程序，它在屏幕上输出“Hello，World”信息。

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

Java 源程序需要编译成字节码才能够被 JVM 识别，还需要使用 JDK 的“javac. exe”命令，假设“HelloWorld. java”(文件名必须与类名 HelloWorld 一致)文件保存在 D 盘 Java Codes 文件夹中。启动控制台，在控制台中输入“d:”命令将当前位置切换到 D 盘根目录，再通过输入“cd 子目录文件名”跳转到 Java 文件保存的子目录中，然后输入“javac　HelloWorld. java”命令编译源程序。源程序被编译后，会在相同的位置生成相应同名的“. class”文件，该文件便是编译后的 Java 字节码文件，然后输入“java　HelloWorld”命令运行程序，屏幕上会输出“Hello，World”信息。

编译与运行 Java 程序的步骤以及运行结果如图 1-5 所示。

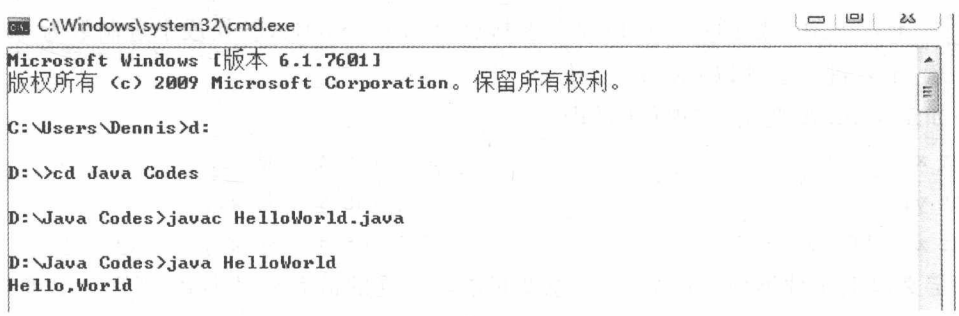

图 1-5　编译与运行 Java 程序的步骤以及运行结果

1.2　基本数据类型

在 Java 中，一共有 8 种基本数据类型，其中有 4 种整型、2 种浮点型、1 种用于表示 Unicode 编码的字符单元的字符型 char 和 1 种用于表示真值的 boolean 类型。

1.2.1　整型

整型用来存储整数数值，即没有小数部分的数值。可以是正数，也可以是负数。整型变量根据它在内存中所占大小的不同，可分为 byte、short、int 和 long 4 种类型。它们具有不同的

取值范围，如表 1-1 所示。

表 1-1　整型数据类型

数据类型	位数	取值范围
字节型	8	$-2^7 \sim 2^7-1$
短整型	16	$-2^{15} \sim 2^{15}-1$
整型	32	$-2^{31} \sim 2^{31}-1$
长整型	64	$-2^{63} \sim 2^{63}-1$

在通常情况下，int 类型最常用。但如果表示地球上的居住人数，就需要使用 long 类型了。byte 和 short 类型主要用于特定的应用场景，例如，底层的文件处理或者需要控制占用存储空间量的大数据。

在 Java 中，整型的范围与运行 Java 代码的机器无关。这就解决了软件从一个平台移植到另一个平台，或者在同一个平台中的不同操作系统之间进行移植时给程序员带来的诸多问题。由于 Java 程序必须保证在所有机器上都能够得到相同的运行结果，所以每一种数据类型的取值范围必须固定。

整数常量在 Java 程序中有 3 种表示形式，分别为十进制、八进制和十六进制。

☑ 十进制。十进制的表现形式大家都很熟悉，如 120、0、－127。

注意：不能以 0 作为十进制数的开头(0 除外)。

☑ 八进制。如 0123(转换成十进制数为 83)、－0123(转换成十进制数为－83)。

注意：八进制必须以 0 开头。

☑ 十六进制。如：0x25(转换成十进制数为 37)、0Xb01e(转换成十进制数为 45086)。

注意：十六进制必须以 0X 或 0x 开头。

例如定义 int 型变量，实现代码如下：

```
int x;                                  //定义 int 型变量 x
int y;                                  //定义 int 型变量 x,y
int x = 350, y = -762;                  //定义 int 型变量 x,y 并赋给初值
```

在定义以上 4 种类型变量时，要注意变量能够接受的最大与最小值，否则会出现错误。对于 long 型值，若赋给的值大于 int 型的最大值或小于 int 型的最小值，则需要在数字后加 L 或 l，表示该数值为长整数，例如 long num = 3241593732L。

实例 1-2　在项目中创建类 Number，在主方法中创建不同数值型变量，并将这些变量相加，将和输出。

```
public class Number
{                    //创建类
   public static void main(String[] args)
   {                 //主方法
      byte mybyte = 123;                        //声明 byte 型变量并赋值
      short myshort = 23456;                    //声明 short 型变量并赋值
      int myint = 12345678;                     //声明 int 型变量并赋值
      long mylong = 34567891;                   //声明 long 型变量并赋值
```

```
        long result = mybyte + myshort + myint + mylong;  //获得各数相加后的结果
        System.out.println("结果为:" + "Java\u2122");  //将以上变量相加的结果输出
    }
}
```

运行结果如图 1-6 所示。

```
D:\Java Codes>javac Number.java

D:\Java Codes>java Number
结果为: 46937148
```

图 1-6　实例 1-2 运行结果

1.2.2　浮点类型

浮点类型表示有小数部分的数值。Java 语言中浮点类型分为单精度浮点类型(float)和双精度浮点类型(double)。它们具有不同的取值范围,如表 1-2 所示。

表 1-2　浮点型数据类型

类型	位数	取值范围
单精度浮点类型	32	1.4e−45～3.4e+38
双精度浮点类型	64	4.9e−324～1.7e+308

double 表示这种类型的数据精度是 float 类型的两倍。绝大多数应用程序都采用 double 类型。在多数情况下,float 类型的精度很难满足需求。在默认情况下小数都被看作 double 型,若想使用 float 型小数,则需要在小数后面添加 F 或 f。可以使用后缀 d 或 D 来明确表明这是一个 double 类型数据。但加不加"d"没有硬性规定,可以加也可以不加。而声明 float 型变量时如果不加"f",系统会认为是 double 类型而出错。例如定义浮点型变量,实例代码如下:

```
float f1 = 13.23f;
double d1 = 4562.12d;
double d2 = 45678.1564;
```

1.2.3　字符类型

1. char 类型

char 类型用于表示单个字符,占用 16 位(两个字节)的内存空间,通常用来表示字符常量。在定义字符型变量时,要以单引号表示,例如'A'表示编码为 65 所对应的字符常量。而"A"则表示一个包含字符 A 的字符串,虽然其只有一个字符,但由于使用双引号,所以它仍然表示字符串,而不是字符。例如:

```
char x = 'a';          // 声明字符型变量
```

由于字符 a 在 Unicode 表中的排序位置是 97,因此允许将上面的语句写成:

```
char x = 97;
```

2. 转义字符

转义字符是一种特殊的字符变量。转义字符以反斜线“\”开头，后跟一个或多个字符。转义字符具有特定的含义，不同于字符原有的意义，故称“转义”。例如，printf 函数的格式串中用到的“\n”就是一个转义字符，意思是“回车换行”。Java 中转义字符如表 1-3 所示。

表 1-3 转义字符

转义字符	说明	转义字符	说明
\'	单引号	\n	换行
\"	双引号	\f	换页
\\	斜杠	\t	跳格
\r	回车	\b	退格

1.2.4 boolean 类型

boolean(布尔)类型又称逻辑类型，只有两个值 true 和 false，用来判定逻辑条件。布尔值和整型值之间不能进行相互转换。布尔类型通常被用在流程控制中作为判断条件。通过关键字 boolean 定义布尔类型变量，实例代码如下：

```
boolean b;                    //定义布尔型变量 b
boolean b1, b2;               //定义布尔型变量 b1,b2
boolean b = true;             //定义布尔型变量 b,并赋给初值 true
```

1.2.5 数据类型转换

数据类型转换是将一个数值从一种数据类型更改为另一种数据类型的过程。例如，可以将 String 类型数据“678”转换为一个数值型。而且，可以将任意类型的数据转换为 String 类型。

如果从低精度数据类型向高精度数据类型转换，则永远不会溢出出错，并且总是成功的；而把高精度数据类型向低精度数据类型转换则必然会有信息的丢失，有可能失败。

数据类型转换有两种方式，即隐式类型转换与显式类型转换。

1. 隐式类型转换

从低级类型向高级类型的转换，系统将自动执行，程序员无须进行任何操作。这种类型的转换称为隐式转换。

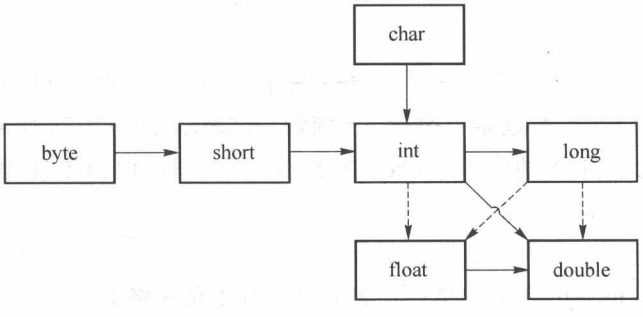

图 1-7 数值类型之间的合法转换

如图 1-7 所示，其中有 6 个实心箭头，表示无信息丢失的数据类型转换；有 3 个虚心箭头，表示有可能有精度丢失的数据类型转换。当使用上面两个数值进行二元操作时（例如 a－b），先要将两个操作数转换（默认自动完成转换）为同一种类型，然后进行计算。

（1）如果两个操作数中有一个是 double 类型的，另一个操作数就会被默认转换为 double 类型。

（2）否则，如果其中一个操作数是 float 类型，另一个操作数将会被默认转换为 float 类型。

（3）否则，如果其中一个操作数是 long 类型，另一个操作数将会被默认转换为 long 类型。

（4）否则，两个操作数都将会被转换成 int 类型。

实例 1-3　使用 int 型变量为 float 型变量赋值，此时 int 型变量将隐式转换成 float 型变量。实例代码如下：

```
int x = 50;                    //声明 int 型变量 x
float y = x;                   //将 x 赋值给 y
```

此时执行输出语句，y 的结果将是 50.0。

2. 显式（强制）类型转换

当把高精度变量的值赋给低精度的变量时，必须使用显式类型转换运算（又称强制类型转换）。语法如下：

```
(类型名)要转换的值;
```

例如将不同的数据类型进行显式类型转换，实例代码如下：

```
int a = (int)45.23;            //此时输出 a 的值为 45
long y = (long)456.6F;         //此时输出 y 的值为 456
int b = (int)'d';              //此时输出 b 的值为 100
```

当执行显示类型转换时可能会导致精度损失。只要是 boolean 类型以外的其他基本类型之间的转换，全部都能以显式类型转换的方法达到。

```
byte b = (byte)129;            //此时输出 b 的值为 -127
```

1.2.6　字符串

前面的章节中介绍了 char 类型，它只能表示单个字符，而由多个字符连接而成的称为字符串（String）。从概念上来说，Java 字符串就是 Unicode 字符序列。Java 中没有内置的字符串类型，而是在标准 Java 类库中提供了一个预定义类 String。每个用双引号括起来的字符串都是 String 类的一个实例，例如：

```
String s = "";             //空字符串
String greeting = "Hello";
```

1. 声明字符串

在 Java 语言中字符串必须包含在一对“”（双引号）之内。字符串是由许多个字符连接而成的。例如“23.23”“ABCDE”“你好”，这些都是字符串的常量，字符串常量是系统能够显示的任何文字信息，甚至是单个字符。可以通过如下语法格式来声明字符串的变量：

```
String str = [null];
```

☑　String：指定该变量为字符串类型。

☑ str：任意有效的标识符，表示字符串变量的名称。

☑ null：如果省略 null，表示 str 变量是未初始化的状态，否则表示声明的字符串的值就等于 null。

例如：

```
String s;      //  声明字符串变量 s
```

2. 创建字符串

在 Java 语言中将字符串作为对象来管理，因此可以像创建其他类对象一样来创建字符串对象。创建对象要使用类的构造方法。String 类的常用构造方法如下。

(1) String(char a[])方法

用一个字符数组 a 创建 String 对象。例如：

```
char a[]={'g','o','o','d'};
String s=new String(a);
```

等价于 ⟹ String s=new String("good")

(2) String (char a[], int offset, int length)

提取字符数组 a 中的一部分创建一个字符串对象。参数 offset 表示开始截取字符串的位置，length 表示截取字符串的长度。例如：

```
char a[]={'s','t','u','d','e','n','t'};
String s=new String(a,2,4);
```

等价于 ⟹ Strings=new String ("uden");

(3) String (char[]value)

该构造方法可分配一个新的 String 对象，使其表示字符数组参数中所有元素连接的结果。例如：

```
char a[]={'s','t','u','d','e','n','t'};
String s=new String(a);
```

等价于 ⟹ String s=new String("student");

除通过以上 String 类的构造方法来创建字符串变量以外，还可以通过字符串常量的引用赋值给一个字符串变量，实现代码如下：

```
String str1,str2;
      str1 = "We are students";
      str2 = "We are students";
```

此时 str1 与 str2 引用相同的字符串常量，因此具有相同的实体。内存示意图如图 1-8 所示。

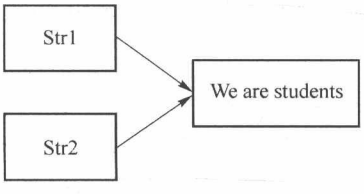

图 1-8　内存示意图

3. 字符串生成器

对于创建成功的字符串对象，它的长度是固定的，内容不能被改变和编译。虽然不使用“+”可以达到附加新字符或字符串的目的，但“+”会产生一个新的 String 实例，会在内存中创建新的字符串对象。如果重复地对字符串进行修改，将极大地增加系统开销。而 J2SE5.0