

普通高等教育“十二五”规划教材

(动漫游戏类)



# Game



## Java手机基础教程



房晓溪◎编著



中国水利水电出版社  
www.waterpub.com.cn

普通高等教育“十二五”规划教材

(动漫游戏类)



# Java手机基础教程



房晓溪◎编著



中国水利水电出版社  
www.waterpub.com.cn

## 内 容 提 要

本书由14章内容构成,包括程序设计概述、Java编程环境、Java语言基础、流程控制、方法、类和对象、继承和多态、异常处理、Java基本类库介绍、Java的集合类等内容。本书内容丰富,讲解精细,通俗易懂,边讲解边操作,大大降低了学习的难度,激发了学习的兴趣和动手的欲望。全书从始至终以讲解Java基础为重点,任务明确,步骤清晰,操作方便。每章均有学习重点,学习难点与学习目标,方便读者抓住每章的重难点。

本书适合全国高等院校的本科生、专科生或高职生相应课程的计算机教材,也可作为游戏编程人员,各类程序开发者和爱好者的学习用书。

## 图书在版编目(CIP)数据

Java手机基础教程 / 房晓溪编著. -- 北京: 中国水利水电出版社, 2011.9  
普通高等教育“十二五”规划教材. 动漫游戏类  
ISBN 978-7-5084-8963-6

I. ①J… II. ①房… III. ①  
JAVA语言—应用—移动电话机—游戏程序—程序设计—高等学校—教材 IV. ①TN929.53②TP311.5

中国版本图书馆CIP数据核字(2011)第177303号

|      |   |
|------|---|
| 书 名  | 普通高等教育“十二五”规划教材(动漫游戏类)<br>Java手机基础教程  |
| 作 者  | 房晓溪 编著  |
| 出版发行 | 中国水利水电出版社<br>(北京市海淀区玉渊潭南路1号D座 100038)<br>网址: www.waterpub.com.cn<br>E-mail: sales@waterpub.com.cn |
| 经 售  | 电话: (010) 68367658 (发行部)<br>北京科水图书销售中心(零售)<br>电话: (010) 88383994、63202643<br>全国各地新华书店和相关出版物销售网点   |
| 排 版  | 北京英宇世纪信息技术有限责任公司  |
| 印 刷  | 北京市兴怀印刷厂  |
| 规 格  | 210mm×285mm 16开本 19.5印张 590千字   |
| 版 次  | 2011年9月第1版 2011年9月第1次印刷   |
| 印 数  | 0001—3000册  |
| 定 价  | 45.00元  |

凡购买我社图书,如有缺页、倒页、脱页的,本社发行部负责调换

版权所有·侵权必究

# 丛书序

本系列教材是一套应用于游戏教学的专业教材，作者在游戏行业拥有多年的教学经验。本系列教材全面系统地对游戏的策划、程序、美术、运营等各个方面进行了详细的阐述、讲解和实训，并增加了案例分析，以使广大读者更深刻地体会到整个体系。

本系列教材包括以下20本：

- 《游戏概论教程》
- 《游戏策划教程》
- 《游戏架构教程》
- 《游戏运营教程》
- 《游戏美术基础教程》
- 《游戏像素图与界面制作教程》
- 《游戏色彩教程》
- 《游戏渲染教程》
- 《游戏角色建模教程》
- 《三维游戏设计与制作教程》
- 《Java手机基础教程》
- 《J2ME手机游戏项目实战教程》
- 《J2ME手机游戏开发教程》
- 《Symbian手机开发教程》
- 《网络游戏C++程序设计教程》
- 《网络游戏DX程序设计教程》
- 《网络游戏HLSL程序设计教程》
- 《网络游戏Windows程序设计教程》
- 《网络游戏引擎程序设计教程》
- 《游戏电子竞技教程》

本系列教材内容丰富，结构完整，通俗易懂，步骤明确，讲解详尽，是一套系统学习游戏的好书。本系列教材的编写目标是：努力追求“一读就懂，学了能用，一用就灵”的学习效果，可以作为全国各高等院校游戏专业课程的教材，也可作为广大游戏从业人员和管理者的学习用书。

希望本系列教材能为广大读者带来方便，欢迎广大读者提出宝贵的意见，以便我们在以后的版本中不断改进。联系E-mail: [fangxiaoxi2002cn@yahoo.com.cn](mailto:fangxiaoxi2002cn@yahoo.com.cn)。

作者

2011年1月

# 前 言

Java语言具有面向对象、与平台无关、安全、稳定和多线程等优良特性，是目前软件设计中极为强大的编程语言。Java语言不仅可以用来开发大型的应用程序，而且特别适合Internet的应用开发。尤其是Java Swing推出之后，不仅使Java的功能更加强大，而且使Java具备了“处处可用”的特点，Java已成为网络时代最重要的语言之一。

本书的作者是程序语言教学的权威人士，多年来一直从事高等教育工作，并有丰富的实践工作经验。作者将自己在教学 and 实际开发过程中的一些经验与体会进行整理和总结，最终完成了这本书的编写，希望与广大读者分享。

本书可分为四个部分：

第一部分，包括1~11章的内容，主要讲解Java语言的基本知识，包括运行的环境、编程环境、流程控制，以及方法、类、对象、继承和多态等内容。

第二部分，是12章的内容，主要讲解Applet基础和多媒体技术，以及对图像的处理。

第三部分，是13章的内容，主要讲解对多线程，包括多线程的概念，基本结构和使用方法，线程的管理等。

第四部分，是14章的内容，主要讲解JSP技术基础，包括网络应用的系统结构，J2EE技术的简介，JSP技术的概述和基本语法，JSP标准动作、内置对象等。

书中各章节都附有习题，这些内容不仅是为了便于学生复习思考，更主要的是作为课堂教学的一种延续。由于Java平台和程序开发技术的不断进步，希望本书能够起到抛砖引玉的作用。

由于作者水平有限，加之时间仓促，书中错误之处在所难免，敬请广大读者批评指正。

作者  
2011年5月

# 目 录

## 丛书序

## 前言

## 第1章 程序设计概述 ..... 1

- 1.1 程序设计语言的分类 ..... 2
  - 1.1.1 按发展过程分类 ..... 2
  - 1.1.2 按执行方式分类 ..... 3
  - 1.1.3 按思维模式分类 ..... 3
- 1.2 Java的简介 ..... 4
  - 1.2.1 Java的历史和主要特点 ..... 4
  - 1.2.2 面向对象的几个基本概念 ..... 7
  - 1.2.3 Java的运行及运行环境 ..... 8
  - 1.2.4 常见的Java集成开发环境 ..... 10
- 1.3 Java的发展 ..... 10
  - 1.3.1 JDK前传 ..... 10
  - 1.3.2 JDK的幼年时期 ..... 11
  - 1.3.3 JDK的青少年时期 ..... 11
  - 1.3.4 JDK的壮年时期 ..... 12
  - 1.3.5 JDK各版的发布时间表 ..... 13
  - 1.3.6 Java的未来10年 ..... 13
  - 1.3.7 Java发展动态 ..... 14
  - 1.3.8 Java机制和类库 ..... 17
- 1.4 Java的特点 ..... 18
  - 1.4.1 平台无关性 ..... 18
  - 1.4.2 安全性 ..... 18
  - 1.4.3 面向对象 ..... 19
  - 1.4.4 简单性 ..... 19
  - 1.4.5 动态特性 ..... 19
  - 1.4.6 分布性 ..... 19
  - 1.4.7 多线程 ..... 19

练习题 ..... 19

## 第2章 Java编程环境 ..... 21

- 2.1 安装JDK ..... 22
  - 2.1.1 通过官方网站下载JDK ..... 22
  - 2.1.2 安装JDK的步骤 ..... 22
  - 2.1.3 安装后的目录结构介绍 ..... 24
- 2.2 Java环境变量设定 ..... 24
- 2.3 JDK工具的使用 ..... 25
- 2.4 JDK的命令行工具 ..... 27
  - 2.4.1 javac ..... 27
  - 2.4.2 java命令 ..... 30
- 2.5 Java集成开发环境 ..... 33
  - 2.5.1 eclipse ..... 33
  - 2.5.2 JBuilder ..... 36
  - 2.5.3 Forte for Java ..... 36
- 练习题 ..... 37

## 第3章 Java语言基础 ..... 38

- 3.1 注释 ..... 39
  - 3.1.1 单行注释 ..... 39
  - 3.1.2 块注释 ..... 39
  - 3.1.3 文档注释 ..... 40
- 3.2 Java的标识符 ..... 40
- 3.3 Java的关键字 ..... 41
- 3.4 变量与常量 ..... 43
  - 3.4.1 变量的声明 ..... 43
  - 3.4.2 赋值和初始化 ..... 44

|       |          |    |
|-------|----------|----|
| 3.5   | 基本数据类型   | 44 |
| 3.5.1 | 布尔型      | 45 |
| 3.5.2 | 字符型      | 46 |
| 3.5.3 | 整数类型     | 47 |
| 3.5.4 | 浮点变量     | 47 |
| 3.5.5 | 数据类型转换   | 48 |
| 3.6   | 字符串      | 50 |
| 3.6.1 | 定义字符串    | 50 |
| 3.6.2 | 字符串的长度   | 50 |
| 3.6.3 | 子串       | 50 |
| 3.6.4 | 字符串的连接   | 51 |
| 3.6.5 | 字符串的比较   | 51 |
| 3.6.6 | 其他常用方法   | 51 |
| 3.6.7 | 转义字符     | 51 |
| 3.7   | 运算符      | 52 |
| 3.7.1 | 算术运算符    | 52 |
| 3.7.2 | 关系运算符    | 54 |
| 3.7.3 | 逻辑运算符    | 55 |
| 3.7.4 | 位运算符     | 56 |
| 3.7.5 | 位移运算符    | 58 |
| 3.7.6 | 赋值运算符    | 59 |
| 3.7.7 | 自增/自减运算符 | 60 |
| 3.7.8 | 三元运算符    | 60 |
| 3.7.9 | 运算符的优先级  | 60 |
| 3.8   | 数组       | 61 |
| 3.8.1 | 数组的定义    | 61 |
| 3.8.2 | 数组的初始化   | 61 |
| 3.8.3 | 多维数组     | 62 |
| 3.9   | 代码段和作用域  | 63 |
| 3.10  | 编程注意事项   | 64 |

|     |    |
|-----|----|
| 练习题 | 65 |
|-----|----|

## 第4章 流程控制 67

|       |              |    |
|-------|--------------|----|
| 4.1   | 结构化程序设计      | 68 |
| 4.2   | 分支语句         | 68 |
| 4.2.1 | if语句         | 68 |
| 4.2.2 | switch语句     | 70 |
| 4.3   | 循环语句         | 72 |
| 4.3.1 | while语句      | 72 |
| 4.3.2 | do...while语句 | 73 |
| 4.3.3 | for语句        | 73 |
| 4.4   | 跳转语句         | 74 |
| 4.4.1 | break语句      | 75 |
| 4.4.2 | continue语句   | 75 |
| 4.4.3 | return语句     | 75 |
| 练习题   | 76           |    |

## 第5章 方法 78

|       |             |    |
|-------|-------------|----|
| 5.1   | 方法的概念及作用    | 79 |
| 5.2   | 方法中变量的可见范围  | 80 |
| 5.3   | 参数传递和返回值    | 81 |
| 5.3.1 | 方法的调用及参数的传递 | 82 |
| 5.3.2 | 方法的返回值      | 83 |
| 5.4   | 递归方法        | 84 |
| 练习题   | 85          |    |

## 第6章 类和对象 86

|     |          |    |
|-----|----------|----|
| 6.1 | 面向对象编程思想 | 87 |
|-----|----------|----|

|        |             |     |
|--------|-------------|-----|
| 6.2    | 对象的实现       | 87  |
| 6.2.1  | 对象的结构       | 87  |
| 6.2.2  | 对象和类的关系     | 88  |
| 6.2.3  | 类的定义        | 88  |
| 6.2.4  | 对象的创建       | 90  |
| 6.2.5  | 程序结构及使用对象成员 | 91  |
| 6.2.6  | 修饰符         | 92  |
| 6.2.7  | 再谈方法        | 95  |
| 6.2.8  | 对象间的赋值      | 97  |
| 6.2.9  | 构造函数        | 98  |
| 6.2.10 | 对象有效范围和废物回收 | 100 |
| 6.3    | Java应用程序的结构 | 100 |
| 6.3.1  | 应用程序结构      | 100 |
| 6.3.2  | 命令行参数       | 101 |
|        | 练习题         | 102 |

## 第7章 继承和多态 104

|       |               |     |
|-------|---------------|-----|
| 7.1   | 继承和多态的概念      | 105 |
| 7.1.1 | 继承的概念         | 105 |
| 7.1.2 | 多态的概念         | 106 |
| 7.2   | 类成员的继承        | 107 |
| 7.2.1 | 属性与方法的继承      | 108 |
| 7.2.2 | this和super的使用 | 111 |
| 7.2.3 | 构造函数的重载和继承    | 112 |
| 7.2.4 | 最终类           | 113 |
| 7.2.5 | 对象在继承过程中的改变   | 114 |
| 7.2.6 | 抽象类及在继承中的特点   | 116 |
| 7.3   | 接口            | 118 |

|       |       |     |
|-------|-------|-----|
| 7.3.1 | 接口的定义 | 119 |
| 7.3.2 | 接口的实现 | 120 |
| 7.3.3 | 适配器   | 121 |
|       | 练习题   | 122 |

## 第8章 异常处理 124

|       |                       |     |
|-------|-----------------------|-----|
| 8.1   | 异常处理机制                | 125 |
| 8.2   | 异常类的层次                | 126 |
| 8.3   | 异常处理语句及使用             | 127 |
| 8.3.1 | try...catch语句         | 127 |
| 8.3.2 | try...catch...finally | 127 |
| 8.3.3 | throws语句              | 128 |
| 8.3.4 | throw语句               | 128 |
| 8.4   | 异常处理举例                | 128 |
| 8.4.1 | 异常类结构                 | 129 |
| 8.4.2 | 异常处理                  | 131 |
| 8.4.3 | 自定义异常类                | 134 |
|       | 练习题                   | 136 |

## 第9章 Java基本类库介绍 139

|       |                      |     |
|-------|----------------------|-----|
| 9.1   | 包的概念                 | 140 |
| 9.1.1 | 构建包                  | 140 |
| 9.1.2 | 包的引用                 | 141 |
| 9.2   | Java语言类库的结构          | 141 |
| 9.3   | java.lang包中的常用类介绍    | 142 |
| 9.3.1 | Object类              | 142 |
| 9.3.2 | Class类               | 143 |
| 9.3.3 | Math类                | 145 |
| 9.3.4 | String与StringBuffer类 | 146 |



|             |                            |            |             |   |            |
|-------------|----------------------------|------------|-------------|---|------------|
| 9.3.5       | System类                    | 151        | 11.5.1      | 文件类   | 197        |
| 9.3.6       | 数据类型类                      | 155        | 11.5.2      | 文件输入/输出   | 201        |
| 9.4         | 关于Java的技术文档                | 161        | 11.5.3      | 随机访问文件  | 203        |
|             | 练习题                        | 162        | 11.6        | 连接多个流   | 205        |
|             |                            |            | 11.7        | 过滤流   | 207        |
| <b>第10章</b> | <b>Java的集合类</b>            | <b>164</b> | 11.7.1      | 缓冲流BufferedInputStream<br>和BufferedOutputStream | 207        |
| 10.1        | 集合类概述                      | 165        | 11.7.2      | LineNumberInputStream流                          | 208        |
| 10.2        | 原集合类                       | 165        | 11.7.3      | DataInputStream和<br>DataOutputStream            | 208        |
| 10.2.1      | 数组                         | 165        | 11.7.4      | PushbackInputStream流                            | 209        |
| 10.2.2      | Vector类                    | 169        | 11.8        | PrintStream流                                    | 210        |
| 10.2.3      | BitSet类                    | 172        |             | 练习题   | 210        |
| 10.2.4      | Stack类                     | 174        |             |   |            |
| 10.2.5      | Hashtable类                 | 175        | <b>第12章</b> | <b>Applet基础和多媒体技术</b>                           | <b>212</b> |
| 10.2.6      | Random类                    | 176        |             |   |            |
| 10.2.7      | Date类                      | 178        | 12.1        | 第一个Applet小应用程序                                  | 213        |
| 10.3        | 新集合类                       | 180        | 12.2        | Applet的运行平台浏览器                                  | 214        |
| 10.3.1      | Collection                 | 180        | 12.2.1      | 浏览器   | 214        |
| 10.3.2      | Set                        | 182        | 12.2.2      | 超文本中的Applet标签                                   | 215        |
| 10.3.3      | Map                        | 183        | 12.2.3      | 用getParameter (String<br>paramname) 方法取得参数      | 216        |
| 10.3.4      | Utilities                  | 186        | 12.3        | Applet程序的运行机理                                   | 217        |
|             | 练习题                        | 190        | 12.4        | Applet程序的结构                                     | 218        |
| <b>第11章</b> | <b>输入/输出系统</b>             | <b>192</b> | 12.5        | Applet程序设计                                      | 221        |
| 11.1        | 流的层次结构                     | 193        | 12.5.1      | 基本Applet程序                                      | 221        |
| 11.2        | InputStream流和OutputStream流 | 193        |             |   |            |
| 11.3        | 内存读写                       | 194        |             |   |            |
| 11.4        | 管道流                        | 195        |             |   |            |
| 11.5        | 文件输入/输出流                   | 197        |             |   |            |

|         |                                    |     |
|---------|------------------------------------|-----|
| 12.5.2  | 多线程Applet设计                        | 228 |
| 12.6    | Applet的安全机制                        | 229 |
| 12.7    | 多媒体设计类的继承关系                        | 229 |
| 12.8    | 对象的属性                              | 229 |
| 12.9    | 设置字体                               | 230 |
| 12.9.1  | Font类                              | 231 |
| 12.9.2  | 字体对象的创建                            | 231 |
| 12.9.3  | 设置字体                               | 232 |
| 12.10   | 设置颜色                               | 233 |
| 12.10.1 | Color类                             | 233 |
| 12.10.2 | 颜色对象的创建                            | 234 |
| 12.10.3 | 颜色的设置                              | 234 |
| 12.11   | Graphics类                          | 236 |
| 12.11.1 | Graphics类的定义及<br>主要方法              | 236 |
| 12.11.2 | 绘图方法的具体使用                          | 238 |
| 12.11.3 | 利用Graphics封装的<br>方法进行绘图            | 238 |
| 12.12   | 图像处理                               | 239 |
| 12.12.1 | 普通图像的装载                            | 239 |
| 12.12.2 | 使用缓冲技术绘制图像                         | 240 |
| 12.12.3 | 使用图像监控器<br>ImageObserver监视图像<br>装载 | 242 |
| 12.13   | 动画技术                               | 244 |
| 12.13.1 | 缓冲技术创建动画                           | 244 |
| 12.13.2 | 使用MediaTracker播放动画                 | 245 |
|         | 练习题                                | 247 |

## 第13章 多线程处理 249

|        |              |     |
|--------|--------------|-----|
| 13.1   | 线程的基本概念      | 250 |
| 13.1.1 | 程序与进程        | 250 |
| 13.1.2 | 进程与线程        | 250 |
| 13.1.3 | Java的线程模型    | 251 |
| 13.2   | 线程的基本结构与使用方法 | 252 |
| 13.2.1 | 线程的生命周期      | 252 |
| 13.2.2 | 定制run()方法    | 254 |
| 13.3   | 线程的管理        | 256 |
| 13.3.1 | 同步           | 256 |
| 13.3.2 | 优先级          | 260 |
| 13.3.3 | 有关线程的其他概念    | 260 |
| 13.4   | 用于制作动画的线程    | 261 |
| 13.4.1 | 动画程序框架       | 261 |
| 13.4.2 | 帧的画法         | 262 |
| 13.4.3 | 避免闪动         | 262 |
| 13.4.4 | 使用图片         | 263 |
|        | 练习题          | 265 |

## 第14章 JSP技术基础 266

|        |                    |     |
|--------|--------------------|-----|
| 14.1   | 网络应用的系统结构          | 267 |
| 14.1.1 | 两层结构               | 267 |
| 14.1.2 | 三层结构               | 267 |
| 14.1.3 | 网络应用构架中的一些<br>补充概念 | 267 |
| 14.2   | J2EE技术简介           | 269 |
| 14.2.1 | J2EE的结构            | 269 |
| 14.2.2 | J2EE中的主要组件         | 270 |
| 14.3   | JSP技术概述            | 272 |



# 第 1 章

## 程序设计概述

### 主要内容：

- ※ 程序设计语言的分类
- ※ Java的简介
- ※ Java的发展
- ※ Java的特点

### 本章重点：

- ※ 程序设计语言的分类
- ※ Java的特点

### 本章难点：

- ※ 认识Java语言的面向对象概念
- ※ 认识Java语言的分布特性
- ※ 掌握Java语言的多线程特性

### 学习目标：

- ※ 了解程序设计语言的分类
- ※ 了解Java语言的运行环境和集成开发环境
- ※ 掌握Java的类库
- ※ 熟悉Java的特性

# 1.1 程序设计语言的分类

程序设计语言是学习计算机技术的基础，它经历了较长的发展过程，也有许多不同的分类方法，下面将介绍几种对学习 Java 比较有帮助的分类方法。

## 1.1.1 按发展过程分类

按计算机系统发展的历程来看，程序设计语言大致可以分为 4 种。

### 1. 机器语言

先看一段机器语言编写的程序：

```
+1300042774  
+1400593419  
+1200274027
```

这是什么含义？一定看得你一头雾水。那就对了，如果能看懂，那你就是 CPU、是机器了。这就是机器语言，是计算机的“自然语言”，与特定的机器硬件有关。机器语言编写程序费时、繁琐、容易出错，难于记忆、难于理解，人们用起来非常困难。机器语言，曾经折磨了一大批计算机业的先驱。实际上这段代码是想将每天的加班费和基本工资相加，然后存储总的费用。用这种语言编写程序的古董级软件开发人员已经基本不存在了。我们也不需要学习，除非和自己过不去。

### 2. 汇编语言

再来看一段汇编语言编写的代码：

```
LOAD  BASEPAY  
ADD   OVERPAY  
STORE GROSSPAY
```

比起机器语言那种类似天书般的语言，从这段代码中似乎可以看出某种朦胧的含义。与上面的机器语言功能类似，同样将每天的加班费和基本工资相加，然后存储总的费用。这段代码对我们来说，容易理解些，对机器来说却很困难，必须经过汇编器的翻译程序将它们转换为机器语言表示的目标程序，才能被计算机执行。汇编语言的出现使计算机的应用有了进一步的发展，但汇编语言仍然是面向机器的，即使一个简单的任务也需要多行代码，一般用户比较难于掌握，现在部分计算机专业课程中仍然教授，主要面向硬件的程序的开发。而面向应用的软件程序开发人员基本不需要学习。为了方便编程和加快计算机的应用，人们开发出了高级语言。

### 3. 高级语言

先看一段高级语言编写的代码：

```
myIncome=baseSalary+overTimePay;
```

含义是什么？多数人基本可以猜出：我的工资是基本工资与加班费之和。如果还看不出来，就看看我们用 Java 语言的另一种书写格式：

我的收入 = 基本工资 + 加班工资；

怎么样？很亲切了吧（注意，尽管可以像上面的例子那样在 Java 中用中文，因为 Java 是基于 Unicode

编码的，就是说 Java 支持所有的字符、中文、日文、阿拉伯文等，后边的章节中会更详细地介绍，但我们不推荐使用，最好用合适的英文单词字母来编写程序，如果对应英文实在想不起来，不得已就用拼音吧）。

显然，用高级语言编写的代码比较容易理解。易学、易用，所以也就比较有吸引力。C、C++、Java 都是功能强大、应用广泛的高级语言。

要使高级语言编写的代码在计算机上运行，有两种方式：其一必须先由编译器翻译成机器语言目标程序；其二是使用“解释程序”，解释型的高级语言源程序可以直接在相应“解释程序”支持下运行，而不再需要将其编译成目标程序。前一种高级语言编写的程序一般会比后一种高级语言编写的程序执行效率要高很多，而我们将要学习的 Java 语言属于后一种，下面我们详细学习 Java 运行机制的时候会看到解释程序在帮助 Java 实现各种平台的可移植性方面起着非常重要的作用。由于 Java 采取了许多优化措施，所以 Java 的执行效率也非常高。

由于 Java 语言有许多其他语言所不具备的优势，如简单易学，功能强大，安全等（在下面的章节会进一步介绍），所以手机游戏开发选择 Java 语言，本课程就是帮助大家学习、领会 Java 手机开发中必备的基础知识，为以后学习手机游戏开发打下扎实的基础。

#### 4. 第四代语言

一种还未成熟的语言。它具有一定的智能，更接近于日常语言，它对语言的概括更为抽象，从而使语言也更为简洁。

### 1.1.2 按执行方式分类

可以把程序按照程序的执行方式分成两大类。

#### 1. 编译执行的语言

编译执行是在编写完程序之后，通过特定的工具软件将源代码经过目标代码转换成机器代码，即可执行程序，然后直接交操作系统执行，也就是说程序是作为一个整体来运行的。这类程序语言的优点是执行速度比较快，另外，编译、链接之后可以独立在操作系统上运行，不需要其他应用程序的支持。缺点是不利于调试，每次修改后都要执行编译、链接等步骤，才能看到其执行结果。当然，有些集成开发环境可以提供单步追踪等工具，方便程序员调试程序。另外，这类程序设计语言的编译器与机器之间存在一定的依赖性，不同操作系统需要的编译器可能不相同，因此，在一个系统上编译的程序到另外一个系统上并不一定能够运行。常见的编译执行的程序语言有 Pascal 和 C 等。

#### 2. 解释执行的语言

解释执行是程序读入一句执行一句，而不需要整体编译、链接，这样的语言与操作系统的相关性相对较小，但运行效率低，而且需要一定的软件环境来做源代码的解释器。当然，有些解释执行的程序并不是使用源代码来执行，而是需要预先编译成一种解释器能够识别的格式，再解释执行。Java 就是这样的一种语言。

### 1.1.3 按思维模式分类

程序设计语言总是需要以某种思维方式进行设计和实现，因此不同的语言可能有不同的思维方式。

#### 1. 面向过程的程序设计语言

面向过程就是以要解决的问题为思考的出发点和核心，并使用计算机逻辑描述需要解决的问题和解决的方法。针对这两个核心目标，面向过程的程序设计语言注重高质量的数据结构和算法，研究采用什么样的数据结构来描述问题，以及采用什么样的算法来高效地解决问题。在 20 世纪 70 ~ 80 年代，大多

数流行的高级语言都是面向过程的程序设计语言，如 Basic、Fortran、Pascal 和 C 等。这类语言面向求解问题的过程，而不依赖于计算机硬件，可移植性相对较好，在计算机所要解决的问题还不是非常复杂、使用的范围还不是非常广泛的条件下，是非常有效的解决问题的方法。但面向过程的程序语言有一个致命的缺点，它极度面向过程，即使需要解决的问题发生微小的变化也会对程序本身产生很大的影响，也就是说需要程序做较大的改动。而且，不同的问题需要不同的程序解决，问题与解决几乎是一对一的，以往的成果很难直接利用。因此，其可维护性和可重用性都比较差。

## 2. 面向对象的程序设计语言

面向对象不仅仅是一种程序设计语言的概念，应该说是一种全新的思维方式。简单地说，面向对象的基本思想就是以一种更接近人类一般思维的方式去看待世界，把世界上的任何一个个体都看成是一个对象，每个对象都有自己的特点，并以自己的方式做事，不同对象之间存在着通信和交互，以此构成世界的运转。用计算机专业的术语来说，对象的特点就是它们的属性，而能做的事就是它们的方法。关于面向对象中的各种概念本书后面会有深入的讲解，需要在这里提醒大家的是，我们学习 Java，其语言规则只占很小的一部分，而要学习的主要内容是如何以面向对象的观点去分析问题和解决问题，这是一个程序员的基本功。常见的面向对象的程序设计语言包括 C++ 和 Java 等。

# 1.2 Java的简介

在所有的面向对象程序设计语言中，Java 是最纯粹、结构最清晰的一种语言，它严格遵守着面向对象的绝大多数思想和理念，是学习面向对象思想最好、最有效的一种语言。同时，Java 又具备构造非常复杂应用的能力，现在以及未来的很多应用都将基于 Java 来开发，它的发展前景是不可估量的。

## 1.2.1 Java的历史和主要特点

Java 诞生于 20 世纪 90 年代，其前身是 Sun 微系统公司开发的一种智能化家电话言 Oak。到了 1993 年，万维网得到了迅速发展，但当时在浏览器中能够看到的页面都是静态的，内容是程序员事先写好的。Sun 公司发现可以利用 Oak 创造动态页面，便开始对 Oak 进行了大规模的改造，于 1995 年推出了 Java。同年，Netscape 公司推出支持 Java 的浏览器 Navigator 2.0，Java 便因此获得了迅速发展的大好契机。如今，Java 本身已经从一种程序设计语言上升成为一类技术，成为网络编程及电子商务系统开发不可缺少的有力工具和平台。Java 之所以能够在计算机和网络技术高度发达的今天占据这样关键性的地位，主要是由以下特点决定的。

Java 是最具吸引力的语言之一，全球有 200 万~300 万的 Java 程序员，2001 年 6 月 JavaOne™ 大会展示美国有 56% 的大学将 Java 作为必修课程，2003 年，大学委员会将其作为高级计算机科学课程标准化。

Java 的应用正在各个领域迅猛发展：具有图形用户界面的软件；可以处理图形、图像、动画、音频、视频等多媒体的软件；在 Internet 上运行，可以与他应用程序通信的软件；可以充分利用“多线程”的灵活性（“多线程”让程序员可以安排多个动作并行进行）的软件；可以方便地处理文件的软件；可以方便地移植而无须修改的软件等，而 Java 可以轻易地满足这些要求，其他语言却很难实现。

如今，Java 在全球广泛流行，各种报刊杂志，到处都是 Java 的信息，招聘 Java 程序的比例更在招聘软件开发人员中占明显优势。

Java 的一个显著优点就是运行时环境（Java Runtime Environment）提供了平台无关性：即你可以在 Windows、Solaris、Linux 或其他操作系统上使用完全一样的代码。也就是一次编写，随处运行（Write Once, Run Anywhere），这对于在各种不同平台上运行从 Internet 上下载的程序来说，非常必要，而其他语言却难以做到这一点。

Java 还是完全面向对象的语言，除了数字之类的几个基本类型，Java 中的一切都是“对象”（如果你对“面向对象”的概念不了解，也没有关系，在后边相关章节会详细介绍。但 Java 的面向对象的特性太突出，我们不得不提及）。

Java 到底是一种什么样的语言呢？概括地说，Java 是一种简单的、面向对象的、分布式的、解释型的、健壮的、安全的、结构中立的、可移植的、性能优异的、多线程的、动态的编程语言。以上 11 点，是 Java 白皮书列出的并引以为自豪的特性。有兴趣的读者可以从 <http://Java.sun.com/docs/white/langenv/> 网址得到有关的信息。

### 1. 简单

Java 最初是为了对家用电器进行集成控制而设计的一种语言，这类设备硬件能力有限，因此它必须简单明了。如果你了解 C++ 语言（本教材总会有意无意地将 Java 与 C++ 语言对比，许多 Java 书籍都有这样的习惯，之所以会出现这种情况，是因为在 Java 之前，C++ 语言太流行，许多 Java 程序员都是从 C++ 转过来的，我们尽量避免这样的对比，如果你没有学习过 C++，许多对比的地方就可以跳过，这并不会影响你对 Java 的学习），与 C++ 相比，Java 语言的简单性主要体现在以下两个方面。

(1) Java 的风格类似于 C++，在此之前有大量的 C++ 程序员，因此，C++ 程序员可以很快就掌握 Java 编程技术。从某种意义上讲，Java 语言是 C 及 C++ 语言的一个变种。

(2) Java 摒弃了 C++ 中很少使用、难于理解、易于混淆、容易出错的特性，如指针和内存管理。这些特性带来的方便，远远多于带来的麻烦，在学习网游编程课程时，我们会碰到这些令人头疼的特性，如头文件、指针算法（甚至指针语法）、结构、联合、操作符重载等。

简单的另一个方面就是小，Java 的一个目标就是支持能够在小型设备上独立运行的软件，Java 的运行环境可以小到几十 KB，一个 Java 应用程序也可以小到几十 KB。

当然，与有些可视化编程语言（一种可以通过拖拽图形化的组件，就可以自动生成代码框架的开发语言）相比，Java 也许并不简单，你必须编写大量的代码，现在已经有少数公司提供了类似与拖拽方式开发 Java 程序的开发工具，不过还不是很成功，不是很流行。

### 2. 面向对象

如果你没有一点面向对象编程的概念和经验，那么，需要仔细阅读第 5 章的有关概念和知识。

面向对象可以说是 Java 最重要的特性。也就是编程处理问题的过程如同在生活中解决实际问题的过程，如需要在程序中移动飞机或一辆赛车，我们就编程创建一架飞机或一辆赛车，然后改变飞机或赛车的状态。Java 语言的设计完全是面向对象的，它不支持类似 C 语言那样的面向过程的程序设计技术。Java 支持静态和动态风格的代码继承及重用。

在第 1 章就介绍这些生涩的概念，理解起来确实很困难，这里先有个印象。

### 3. 分布式

Java 包括一个支持 HTTP 和 FTP 等基于 TCP/IP 协议的子库（这里提到的这些协议就是我们登录互联网、浏览网页、传输文件等常用的一些协议）。因此，Java 应用程序可凭借 URL 打开并访问网络上的对象，其访问方式和便利程度与访问本地文件系统几乎完全相同。为分布环境尤其是 Internet 提供动态内容无疑是一项非常宏伟的任务，但 Java 的语法特性却使我们很容易地实现这项目标。Java 的网络能力强大而易于使用，尤其是一种称为 servlet 的优秀机制使得服务器端的 Java 处理变得非常高效，很多流行的 Web 服务器都支持 servlet。一种称为“远程方法调用”（RMI）的机制，可以使 Java 程序非常方便地进行分布式对象间的通信（分布式是指一个大的应用程序可以由多个部分组成，各部分可以位于不同的地方、不同的服务器上；“通信”，通俗地讲就是彼此可以传递数据、信息）。这一特性在企业，或电子商务、电子政务等大型应用领域使用很广。

### 4. 健壮

Java 的设计目标之一就是设计出的 Java 程序在多方面都可靠。它花费大量的精力进行早期的问题检



查（编译时），后期动态检查（运行时），以及消除可能导致出错的状态，从而避免了其他语言重写内存、数据崩溃的可能性，这样就可以检查出其他语言在运行时才能发现的错误。

## 5. 体系结构中立

Java 编译器生成体系结构中立的目标文件格式，也就是说这种文件格式不依赖某种特定的硬件系统，只要有“Java 运行系统”的机器都能执行这种中间代码。现在，Java 运行系统有针对 Solaris 2.4 (SPARC)、Win32 系统 (Windows 95 和 Windows NT) 等不同平台的版本。Java 源程序被编译成一种高层次的与机器无关的 byte-code 格式语言，这种语言被设计在虚拟机 (VM) 上运行。

## 6. 安全

Java 主要的目标是设计为用于网络 / 分布式环境，为了达到这个目标，在安全方面投入了很大的精力，用 Java 可以构建防病毒、防篡改的系统。

Java 安全机制禁止 Java 程序进行一些操作。下边是被限制的一些操作：

- (1) 禁止运行时堆栈溢出，就像我们常遇到的蠕虫病毒作怪方式。
- (2) 禁止在自己的处理空间外破坏内存。
- (3) 禁止通过安全控制类装载器来读写本地文件，就像 Web 浏览器禁止此类访问一样。

Java 的确是当前最安全的语言，但任何语言都难以保证百分之百的安全，好在 Sun 公司公布了 Java 解释器工作机制的内部规范，这使得人们可以更容易地发现 Java 安全机制中的 bug，从根本上为外界团体发现非常隐蔽的 bug 提供支持。Java 中不断加入许多安全特性。总之，Java 的安全机制越来越强，发现的 bug 越来越微小，数目也越来越少。

在代码传输机制上，微软的安全方案仅仅依赖于数字签名（一种可以验证软件是哪个厂商开发的数字签名技术），这难以保证真正的安全，即便是来自知名厂商的软件也可能会引起系统崩溃，对系统造成危害。而 Java 程序不仅有数字签名的验证机制，并且在运行时也对程序进行控制，禁止程序进行破坏操作。所以说 Java 安全机制比微软的 ActiveX 强大得多。

## 7. 可移植的

这句话一直是 Java 程序设计师们的精神指标，也是 Java 之所以能够受到程序设计师们喜爱的原因之一，最大的功臣就是 JVM (Java 虚拟机) 技术。大多数编译器产生的目标代码只能运行在一种 CPU 上（如 Intel 的 x86 系列），即使那些能支持多种 CPU 的编译器也不能同时产生适合多种 CPU 的目标代码。如果你需要在 3 种 CPU（如 x86、SPARC 和 MIPS）上运行同一程序，就必须编译 3 次。

但 Java 编译器就不同了。Java 编译器产生的目标代码 (J-Code) 是针对一种由代码实现的“CPU”——Java 虚拟机 (Java Virtual Machine)，而不是某一实际的 CPU。Java 虚拟机掩盖不同 CPU 之间的差别，使 J-Code 能运行于任何具有 Java 虚拟机的机器上。

同体系结构无关的特性使得 Java 应用程序可以在配备了 Java 解释器和运行环境的任何计算机系统上运行，这成为 Java 应用软件便于移植的良好基础。但仅仅如此还不够。如果基本数据类型设计依赖于特定计算机系统，通过定义独立于平台的基本数据类型及其运算，Java 数据得以在任何硬件平台上保持一致。这样 Java 语言编写的程序编译后无需修改就可以在各种系统平台上运行，实现了“一次编写，随处运行”的目标。

## 8. 解释型

Java 解释器 (JVM) 能直接运行 Java 字节码。链接程序通常比编译程序所需资源少，所以程序员可以在创建源程序上花上更多的时间。

## 9. 高性能

与其他解释型的语言相比，Java 的运行速度要快得多。例如，对简单的 Java 程序，在工作站或高性能的 PC 上，每秒钟可以创建 119 000 个对象；对于带有多个方法的对象，每秒钟可以创建 89 000 个，可