21世纪高等学校规划教材 | 计算机科学与技术

分布式对象技术及其应用（第2版）

孟宪福　编著

清华大学出版社

分布式对象技术及其应用（第2版）

孟宪福 编著

清华大学出版社
北京

内 容 简 介

分布式对象技术是在面向对象技术的基础上发展起来的，它要解决的主要问题是位于不同进程中的对象之间的调用问题。在中间件系统、Web 服务以及 SOA 等需要多程序协作的许多领域，分布式对象技术都发挥着重要作用。本书分 8 章，按照循序渐进的原则，从理论到实践逐步介绍分布式对象技术的典型代表 CORBA 和 Java RMI 的基本概念与程序设计规则。特别是，为了使读者能够尽快运用分布式对象技术来解决实际问题，在本书的最后两章完整地给出了基于 CORBA 和 Java RMI 的多个应用实例及其程序开发过程。

本书是作者根据多年的教学经验和实践体会编写而成的，在内容编排上尽量体现易学的特点，在文字叙述上力求条理清晰、简洁、便于读者阅读。

本书可以作为大专院校计算机专业研究生或高年级本科生的教材，也可以作为非计算机专业学生或软件开发人员的参考书或自学用书。

图书在版编目(CIP)数据

分布式对象技术及其应用/孟宪福编著. —2 版. —北京：清华大学出版社，2015
21 世纪高等学校规划教材 · 计算机科学与技术
ISBN 978-7-302-38596-7

Ⅰ. ①分… Ⅱ. ①孟… Ⅲ. ①分布式计算机系统－高等学校－教材 Ⅳ. ①TP338.8

中国版本图书馆 CIP 数据核字(2014)第 274530 号

责任编辑：梁　颖　薛　阳
封面设计：常雪影
责任校对：白　蕾
责任印制：杨　艳

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn，010-62795954

印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：12.75　　**字　　数**：309 千字
版　　次：2008 年 11 月第 1 版　2015 年 1 月第 2 版　　**印　　次**：2015 年 1 月第1 次印刷
印　　数：1～2000
定　　价：35.00 元

产品编号：062071-01

出版说明

随着我国改革开放的进一步深化，高等教育也得到了快速发展，各地高校紧密结合地方经济建设发展需要，科学运用市场调节机制，加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度，通过教育改革合理调整和配置了教育资源，优化了传统学科专业，积极为地方经济建设输送人才，为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是，高等教育质量还需要进一步提高以适应经济社会发展的需要，不少高校的专业设置和结构不尽合理，教师队伍整体素质亟待提高，人才培养模式、教学内容和方法需要进一步转变，学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007 年 1 月，教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》，计划实施“高等学校本科教学质量与教学改革工程”（简称“质量工程”），通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容，进一步深化高等学校教学改革，提高人才培养的能力和水平，更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中，各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势，对其特色专业及特色课程（群）加以规划、整理和总结，更新教学内容、改革课程体系，建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上，经教育部相关教学指导委员会专家的指导和建议，清华大学出版社在多个领域精选各高校的特色课程，分别规划出版系列教材，以配合“质量工程”的实施，满足各高校教学质量和教学改革的需要。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作，提高教学质量的若干意见》精神，紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”，在有关专家、教授的倡议和有关部门的大力支持下，我们组织并成立了“清华大学出版社教材编审委员会”（以下简称“编委会”），旨在配合教育部制定精品课程教材的出版规划，讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师，其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求，“编委会”一致认为，精品课程的建设工作从开始就要坚持高标准、严要求，处于一个比较高的起点上。精品课程教材应该能够反映各高校教学改革与课程建设的需要，要有特色风格、有创新性（新体系、新内容、新手段、新思路，教材的内容体系有较高的科学创新、技术创新和理念创新的含量）、先进性（对原有的学科体系有实质性的改革和发展，顺应并符合 21 世纪教学发展的规律，代表并引领课程发展的趋势和方向）、示范性（教材所体现的课程体系具有较广泛的辐射性和示范性）和一定的前瞻性。教材由个人申报或各校推荐（通过所在高校的“编委会”成员推荐），经“编委会”认真评审，最后由清华大学出版

社审定出版。

目前，针对计算机类和电子信息类相关专业成立了两个“编委会”，即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。推出的特色精品教材包括：

（1）21世纪高等学校规划教材·计算机应用——高等学校各类专业，特别是非计算机专业的计算机应用类教材。

（2）21世纪高等学校规划教材·计算机科学与技术——高等学校计算机相关专业的教材。

（3）21世纪高等学校规划教材·电子信息——高等学校电子信息相关专业的教材。

（4）21世纪高等学校规划教材·软件工程——高等学校软件工程相关专业的教材。

（5）21世纪高等学校规划教材·信息管理与信息系统。

（6）21世纪高等学校规划教材·财经管理与应用。

（7）21世纪高等学校规划教材·电子商务。

（8）21世纪高等学校规划教材·物联网。

清华大学出版社经过三十多年的努力，在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌，为我国的高等教育事业做出了重要贡献。清华版教材形成了技术准确、内容严谨的独特风格，这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会

联系人：魏江江

E-mail：weijj@tup.tsinghua.edu.cn

再版前言

随着软件应用领域的不断拓展，对程序设计技术的需求也越来越高，进而出现了分布式对象技术。分布式对象技术是在面向对象技术的基础上发展起来的，它要解决的主要问题是位于不同进程中的对象之间的调用问题。在中间件系统、Web 服务以及 SOA(Service Oriented Architecture：面向服务架构)的研究与开发等许多重要领域，分布式对象技术都发挥着不可替代的作用。本书共由 8 章组成，按照循序渐进的原则，从理论到实践逐步介绍分布式对象技术的典型代表——CORBA 的基本概念与程序设计规则，主要包括 CORBA 的组成与处理流程、IDL 接口定义语言、CORBA 客户端程序设计与服务器程序设计以及动态接口等内容。特别是，为了使读者能够尽快运用分布式对象技术来解决实际问题，本书利用两章的篇幅详细地介绍了基于 CORBA 的程序设计实例和基于 Java RMI 的程序设计实例，所给出的几个例子都是具有代表性的并具有实用价值的，通过对这些实例的学习，能够使读者进一步掌握分布式对象程序设计的要点，并能达到举一反三的目的。所给出的程序都是在实际的环境下调试完成的，以使读者能够尽快掌握分布式对象这门应用技术。

本书详细地介绍了分布式对象的基本内容，从理论到实践完整而系统地介绍了 CORBA 分布式对象系统设计规范和程序设计过程。本书的前 6 章主要是从理论的角度来介绍分布式对象系统的设计过程，后 2 章则从应用的角度来介绍分布式对象系统的实现过程。第 1 章简要介绍 Java 语言的基本内容，这是为阅读后续章节中的程序做准备的，所介绍的内容也仅局限在其他章节中需要使用的基本内容，包括基本语句、类的定义和接口等；第 2 章介绍分布式对象的基本概念以及 CORBA 的基本构成和处理过程，这一章的内容将为准确理解后续章节的内容打下基础；第 3 章主要介绍分布式对象系统的开发流程、IDL 语言以及从 IDL 到 Java 语言的映射，IDL 语言是基于 CORBA 的程序设计基础，而语言映射则为实现客户端和服务器提供了必需代码，这些代码包括 Stub 类、Skeleton 类、Holder 类和 Helper 类；第 4 章介绍 CORBA 客户端程序设计过程，主要包括 ORB 的初始化、分布式对象引用的获取以及分布式方法的调用等；第 5 章介绍 CORBA 服务器程序设计过程，主要包括 BOA 与 POA 的基本内容、分布式对象实现以及服务器进程实现等；第 6 章介绍动态接口技术，主要内容包括 any 类型的处理、接口仓库、动态启动接口 DII 和动态骨架接口 DSI 等；第 7 章介绍几个典型的 CORBA 实例，通过对这些实例的学习，能够准确了解 CORBA 应用系统的完整实现过程；第 8 章介绍 Java RMI 远程对象技术，详细说明了基于回调技术的分布式对象系统设计过程。

作者认为，分布式对象作为一门应用技术，要想学好它，除了掌握基本理论之外，还必须要加强实践环节。读者可以边学习边上机，刚开始时可以在给定的环境下调试本书中的例题，待学习一段时间之后，就可以调试自己编写的程序了。只有这样，才能加快学习进度，提高学习效率，真正掌握这门应用技术。

本书的出版得到了大连理工大学研究生院教改基金经费资助。

由于作者水平有限，经验不足，书中一定有不少缺点和错误，敬请有关老师、计算机工作者和广大读者批评指正。

作　者

2014 年 10 月

目 录

第 1 章　Java 语言基础 …… 1

1.1　Java 语言的特点及其程序开发过程 …… 1
1.1.1　Java 语言的特点 …… 1
1.1.2　Java 程序的开发过程 …… 2
1.2　数据类型、变量、运算符和基本语句 …… 3
1.2.1　数据类型 …… 4
1.2.2　变量与常量 …… 4
1.2.3　运算符 …… 6
1.2.4　运算符的优先级 …… 7
1.2.5　数组 …… 7
1.2.6　字符串 …… 9
1.2.7　基本数据类型包装类 …… 10
1.2.8　基本语句 …… 11
1.3　类 …… 17
1.3.1　对象的生成与引用 …… 17
1.3.2　this 与 super …… 17
1.3.3　类的定义 …… 19
1.3.4　成员变量的定义 …… 19
1.3.5　方法的定义 …… 20
1.3.6　构造方法的定义与使用 …… 21
1.3.7　static 块 …… 22
1.3.8　对象的释放 …… 22
1.4　接口与异常处理 …… 22
1.4.1　接口 …… 22
1.4.2　异常处理 …… 23
1.4.3　包 …… 25
1.4.4　命令行参数 …… 26
1.5　多态性的实现 …… 27
1.6　委托处理与功能继承 …… 28

第 2 章　分布式对象与 CORBA …… 32

2.1　CORBA 与 OMG …… 32

2.2 CORBA 的发展历程 …… 32
2.3 分布式对象的定义与特点 …… 34
2.3.1 分布式对象的定义 …… 34
2.3.2 分布式对象系统的透明性 …… 35
2.3.3 分布式对象系统的复杂性 …… 35
2.4 CORBA 系统的基本构成 …… 37
2.5 CORBA 分布式对象环境 …… 41
2.6 分布式对象系统的处理过程 …… 44

第 3 章 分布式对象系统设计与 IDL 定义 …… 49

3.1 分布式对象系统的开发流程 …… 49
3.2 基于 CORBA 分布式对象系统设计 …… 50
3.3 IDL 接口定义语言 …… 52
3.3.1 IDL 的作用 …… 52
3.3.2 数据类型 …… 53
3.3.3 类型定义 …… 54
3.3.4 常量定义 …… 56
3.3.5 异常定义 …… 57
3.3.6 属性定义 …… 57
3.3.7 操作定义 …… 58
3.3.8 接口定义 …… 60
3.3.9 模块定义 …… 61
3.3.10 预处理器 …… 62
3.4 从 IDL 到 Java 的映射 …… 63
3.4.1 接口定义的映射 …… 63
3.4.2 实现引用传递的 Holder 类 …… 66
3.4.3 提供各种实用功能的 Helper 类 …… 68
3.4.4 其他 IDL 定义的映射 …… 69
3.4.5 IDL 映射后的使用 …… 78

第 4 章 CORBA 客户端程序设计 …… 81

4.1 问题描述与 IDL 定义 …… 81
4.2 CORBA 客户端的组成 …… 82
4.3 ORB 的初始化 …… 83
4.4 ORB 接口的功能 …… 84
4.5 分布式对象引用的获取 …… 85
4.5.1 利用文件的方法获取对象引用 …… 85
4.5.2 利用 Binding 服务的方法获取对象引用 …… 87
4.5.3 利用命名服务的方法获取对象引用 …… 88

4.5.4 利用 factory 对象的方法获取对象引用 …… 89
4.6 Stub 类的构造 …… 89
4.6.1 代理对象的概念 …… 89
4.6.2 分布式对象引用与本地对象引用的区别 …… 90
4.6.3 Stub 类的构造 …… 90
4.7 org.omg.CORBA.Object 接口 …… 93
4.8 分布式对象方法的启动 …… 95
4.9 Java Applet 中的 CORBA 客户端结构 …… 97
4.9.1 Java Applet 中的 CORBA 客户端结构 …… 97
4.9.2 ORB 的初始化 …… 98
4.9.3 分布式对象引用的获取 …… 98
4.9.4 在 HTML 文件中使用 Applet …… 99

第 5 章 CORBA 服务器程序设计 …… 101

5.1 CORBA 服务器的构造 …… 101
5.2 对象适配器的作用 …… 102
5.2.1 对象适配器的作用 …… 102
5.2.2 BOA 与 POA …… 103
5.2.3 伪对象 …… 103
5.3 BOA 的功能 …… 103
5.4 分布式对象实现 …… 105
5.4.1 Skeleton 继承方式 …… 105
5.4.2 Skeleton 类的构造 …… 106
5.4.3 Tie 机制方式 …… 110
5.5 分布式对象的生成 …… 113
5.6 分布式对象的登录 …… 113
5.7 接收请求开始 …… 113
5.8 POA 基础 …… 114
5.8.1 POA 中的 CORBA 对象与 Servant 的关系 …… 114
5.8.2 POA 与策略 …… 114
5.8.3 POA 的生成 …… 115
5.8.4 POA 策略简介 …… 116
5.8.5 POA 管理器 …… 122
5.8.6 Servant 管理器 …… 122
5.8.7 默认 Servant …… 123
5.8.8 基于 POA 的服务器程序设计 …… 123

第 6 章 动态接口 …… 127

6.1 通用伪接口的定义 …… 127

6.1.1 TypeCode 接口 …… 127
6.1.2 NamedValue 接口 …… 130
6.1.3 NVList 接口 …… 131
6.2 Any 类型数据的处理 …… 132
6.2.1 Any 的功能与数据构造 …… 132
6.2.2 Any 类型的 Java 映射 …… 133
6.2.3 Any 对象的生成 …… 133
6.2.4 Any 对基本类型数据的存取 …… 133
6.2.5 Any 对用户定义类型数据的存取 …… 134
6.2.6 DynAny 接口 …… 135
6.3 接口仓库 …… 135
6.3.1 接口仓库的构造 …… 135
6.3.2 接口仓库的接口 …… 136
6.3.3 对接口仓库的访问 …… 141
6.3.4 仓库 ID …… 143
6.4 动态启动接口 DII …… 144
6.4.1 DII 程序设计过程 …… 144
6.4.2 Request 对象 …… 144
6.4.3 动态启动调用请求 …… 147
6.4.4 返回值的取出 …… 148
6.5 动态骨架接口 DSI …… 149
6.5.1 DynamicImplementation 类 …… 149
6.5.2 ServerRequest 接口 …… 150

第 7 章 CORBA 实例 …… 151

7.1 Java IDL 及其应用系统开发过程 …… 151
7.2 环境配置 …… 152
7.3 CORBA 实例 1：一般属性和操作的定义与使用 …… 152
7.3.1 问题描述与 IDL 接口定义 …… 153
7.3.2 IDL 到 Java 语言的映射 …… 153
7.3.3 服务器端的 Java 语言程序设计 …… 154
7.3.4 客户端的 Java 语言程序设计 …… 157
7.3.5 Java 类的编译 …… 158
7.3.6 启动 orbd …… 158
7.3.7 服务器端程序的执行 …… 158
7.3.8 客户端程序的执行 …… 159
7.4 CORBA 实例 2：本地方法与 Holder 类的使用 …… 159
7.4.1 问题描述与 IDL 接口定义 …… 159
7.4.2 IDL 到 Java 语言的映射 …… 160

7.4.3 服务器端的 Java 语言程序设计 …… 161
7.4.4 客户端的 Java 语言程序设计 …… 163
7.4.5 Java 类的编译 …… 165
7.4.6 启动 orbd …… 165
7.4.7 服务器端程序的执行 …… 165
7.4.8 客户端程序的执行 …… 165
7.5 CORBA 实例 3:Factory 对象的定义与使用 …… 166
7.5.1 问题描述与 IDL 接口定义 …… 166
7.5.2 服务器程序设计 …… 167
7.5.3 客户端程序设计 …… 169
7.5.4 语言映射、编译与运行 …… 170
7.6 CORBA 实例 4:利用文件方式获取分布式对象引用的程序实现过程 …… 170
7.6.1 IDL 接口定义 …… 171
7.6.2 服务器程序设计 …… 171
7.6.3 客户端程序设计 …… 172
7.6.4 语言映射、编译与运行 …… 173
7.7 简便的程序调试方法 …… 173

第 8 章 Java RMI 技术 …… 174

8.1 Java RMI 远程对象调用过程 …… 174
8.2 远程对象 …… 174
8.2.1 远程接口 …… 174
8.2.2 远程接口的实现类 …… 175
8.2.3 远程对象的生成 …… 175
8.3 Stub 与 Skeleton …… 175
8.4 启动 RMI 注册器 …… 175
8.5 RMI 程序设计过程 …… 176
8.5.1 远程接口的定义 …… 176
8.5.2 服务器程序的实现 …… 176
8.5.3 客户端程序的实现 …… 177
8.5.4 类文件的编译 …… 178
8.5.5 启动 RMIregistry …… 178
8.5.6 运行服务器程序 …… 179
8.5.7 运行客户端程序 …… 179
8.6 基于回调技术的 RMI 程序设计 …… 180
8.6.1 服务器的远程接口 …… 180
8.6.2 服务器的远程接口的实现类 …… 181
8.6.3 客户端的远程接口 …… 184
8.6.4 客户端的远程接口的实现类 …… 184

8.6.5 异常类的定义 …… 185
8.6.6 Applet 程序与 HTML 文件的定义 …… 185
8.6.7 定义 java.policy 文件 …… 187
8.6.8 编译与运行 …… 188

参考文献 …… 189

第1章 Java语言基础

Java语言是目前应用最广泛的面向对象程序设计语言之一，它具有面向对象、与平台无关、安全、稳定和多线程等优良特性。Java语言不仅可以用来开发大型的应用程序，而且特别适合于包括Internet应用等网络程序的开发。由于本书是以Java语言为基础来描述分布式对象技术的，因此，本章将对后续章节中需要使用的Java语言的基本内容进行简单的介绍。

1.1 Java语言的特点及其程序开发过程

Java语言的魅力主要体现在以下三个方面：

(1) 不管使用何种机器环境，只要有Java运行环境，Java的程序就可以执行。

(2) Java是一种拥有图形用户接口(GUI)和图像处理能力的新型的面向对象程序设计语言。

(3) Java语言程序可以作为Web页面的一部分来使用，这不仅体现在能使Web页面具有动态性的特点，而且体现在能够将Java语言程序从一台机器上快速下载到另一台机器上并运行这一强有力的功能上。

下面从程序设计语言方面来介绍Java语言的特点，同时简要介绍一下Java语言程序的开发过程。

1.1.1 Java语言的特点

从程序设计语言的角度来看，Java语言主要有如下一些主要特点。

1. 与C++语言相似

Java语言是不具有C++语言中的结构体、联合(共用体)、指针、预处理器等功能的非常简单的程序设计语言，如果熟悉C++语言的话，就可以比较容易地学会Java语言。

2. 以类为基础的面向对象程序设计语言

Java是以面向对象为基础而设计的语言，具有面向对象程序设计语言所具有的一切特点，如模块化(抽象)、数据隐藏、继承和多态等。同时，与C++语言不同的是，Java语言中的方法(包括main()方法)和全局变量等都只能作为类的成员来定义，而不能定义在类的外

部，这更体现出以类为基础的面向对象程序设计语言的特点。

3. 强类型检查功能

程序中所使用的字符串和数组等变量必须在使用之前进行类型定义。在编译时进行类型检查，以便在运行时将可能出现的问题降到最低。

4. 解释执行

Java 程序被编译为不依赖于任何机器的二进制形式的命令集，即字节码，其执行过程则是利用解释器来完成的。

5. 提供了丰富的类库

除了基本类库之外，Java 语言还提供了开发应用程序所不可缺少的 GUI 和网络功能等大量类库。

6. 可并发执行的语言

利用 Java 语言可以开发出同时由多个任务执行的多任务系统。通过利用多线程，可以实现并行处理。

7. 重视鲁棒性与安全性

Java 语言中没有指针类型，在编译时进行强类型检查。除此之外，在执行时还由解释器来检查字节码的正确性，其中，对字节码是否访问受限制的内存区域进行检查，从而能够保证系统运行的安全性和稳定性。

8. 性能

Java 程序的执行速度对于一般的实用程序来讲是足够快的，但一般认为比 C 语言慢，这主要是由于解释执行的原因。如果不对字节码进行解释执行，而是将其翻译为机器语言，那么就可以实现与 C 语言相同的执行速度。

9. 内存管理

程序设计者不需要明确地申请内存和释放内存，对内存的管理是自动进行的，使用完了的内存将被自动释放掉。

10. 与其他语言程序的接口

利用 C 语言等其他语言编写的程序可以作为动态装入库在 Java 程序中进行使用。Java 程序的一部分（即方法）可以利用其他语言来编写，这样的方法被称为 Native 方法。

1.1.2 Java 程序的开发过程

Java 语言程序的开发过程是由如下几部分组成的：

（1）编辑 Java 源文件，并将其保存在以 java 为扩展名的文件中。

(2) 编译 Java 源文件,以生成扩展名为.class 的字节码文件。Java 编译器程序是 javac.exe。

(3) 运行 Java 程序。Java 程序可分为两类:一类是 Java 应用程序(即 Application),一类是 Java 小程序(即 Applet)。Java 应用程序一般需要通过 Java 解释器(java.exe)来解释执行其字节码文件;Java 小程序则必须通过支持 Java 标准的浏览器来解释执行。

下面的程序用于说明一个简单的 Java 应用程序的开发及运行过程。

```
//HelloJava.java
public class HelloJava
{
    public static void main(String args[ ])
    {
        System.out.println("Hello Java!");
    }
}
```

说明:

(1) 一个 Java 源程序是由若干个类组成的。在该程序中只有一个类,其类名为 HelloJava。

(2) 在一个 Java 应用程序中,必须有且只能有一个类含有 main()方法,包含 main()方法的类被称为主类,程序是从 main()方法开始执行的。

(3) 假设该程序被保存在 E:\Hello 目录下,则其编译过程如下:

```
E:\Hello\>javac HelloJava.java
```

编译完成后将生成一个 HelloJava.class 字节码文件,该字节码文件也被保存在 E:\Hello 目录下。

(4) 经编译之后,该程序的运行过程如下:

```
E:\Hello\>java HelloJava
```

屏幕上将显示如下信息:

```
Hello Java!
```

1.2 数据类型、变量、运算符和基本语句

Java 语言程序中所用到的每一个常量、变量及其函数等都是程序的基本操作对象,它们都隐式地或显式地与一种数据类型相联系,每种数据类型都表示了它的可能取值范围以及能在其上所进行的运算。Java 语言中提供了丰富的数据类型和运算符,运用这些数据类型和运算符能够进行复杂的数学运算。另外,由于程序的执行流程是通过语句来控制的,为此 Java 语言也提供了完善的语句支持。

本节主要讨论 Java 语言中的一些基本概念,如基本数据类型、运算符以及利用这些运算符来构成相应表达式的一些规则等,同时对基本语句也进行详细的介绍。

1.2.1 数据类型

Java 语言中的变量必须被定义为某种数据类型。Java 语言中的数据类型可以分为基本数据类型和引用数据类型两种。基本数据类型包括整型、浮点型和字符类型等；引用数据类型则包括对象和数组等。本节主要介绍基本数据类型。

Java 语言中的基本数据类型及其所占位数等信息如表 1.1 所示。

表 1.1 基本数据类型

基本数据类型	值的种类	位数	默认值
boolean	真、假	1	false
byte	带符号整数	8	0
short	带符号整数	16	0
int	带符号整数	32	0
long	带符号整数	64	0L
char	字符(Unicode)	16	'\u0000'
float	单精度浮点数	32	0.0f
double	双精度浮点数	64	0.0

在表示 float 型数据时，可以在数据的末尾加上 f 或 F，以区别于 double 类型数据。

以类的实例作为其值的变量被称为引用数据类型的变量。对于基本数据类型的变量来讲，其存储区域直接保存着值，而对于引用类型的变量来讲，其存储区域则保存着实际数据的引用。所谓引用，实际上在 C/C++语言中被称为指针，由于 Java 语言没有指针的概念，因此称其为引用。

类、接口和数组都属于引用类型。当定义了新的类或接口时，也就创建了新的引用类型。有关引用类型的基本内容将在后面详细介绍。

1.2.2 变量与常量

1. 变量

变量是指在程序运行过程中其值可以被改变的量。变量被区分为不同的类型，不同类型的变量在内存中占用不同的存储单元，以便用来保存相应变量的值。

变量的定义形式如下：

[访问修饰符][域修饰符] 数据类型名 变量名 [= 初值]；

如果在变量名的后面加上＝和初值，则在定义变量的同时为其初始化。如果没有指定初值，则对于基本数据类型的变量来讲，将被初始化为默认初值，而对于引用类型的变量来讲，将被初始化为 null。

Java 语言中的变量名(包括一般的标识符，如方法名、数组名和类型名等)是由字母、下划线、美元符号($)和数字组成的，并且第一个字符不能是数字。

作为习惯用法，变量名的首字符一般为小写字母，类名的首字符一般为大写字母。