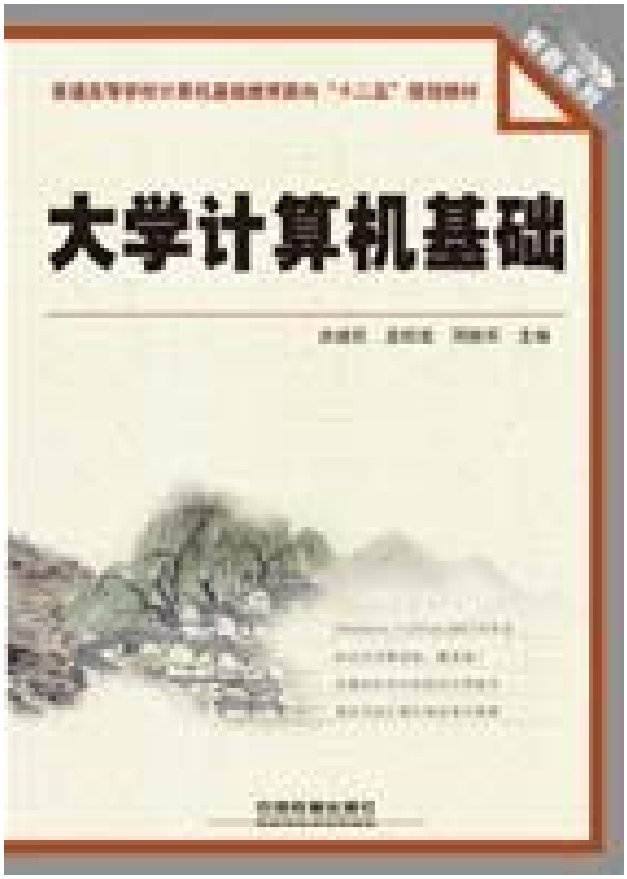
目录

1 大学计算机

1.1 教材封面



1.2 出版信息页

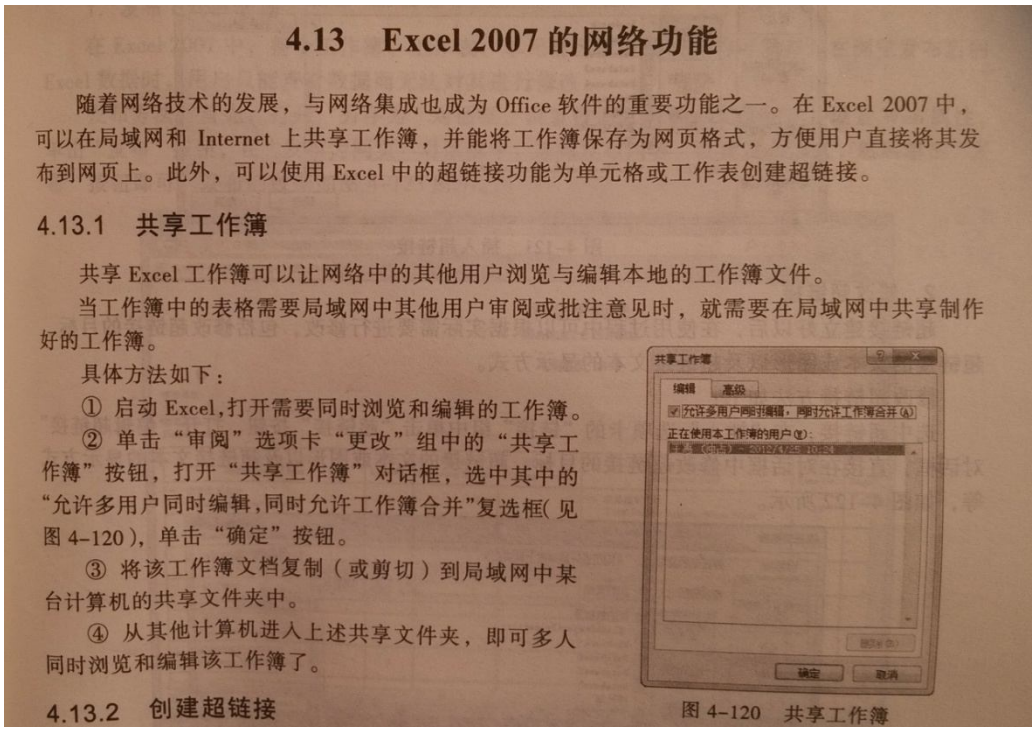
作者: 曲建民 周晓军 孟昭宽 主编 高雅荣 刘畅 副主编
书号:978-7-113-14470-8
译者:
丛书名: 普通高等学校计算机基础教育面向“十二五”规划教材·创新系列
版次: B1
定价: 38.0 元
印次: Y1
装帧: 胶订
开本: 16 开
页数: 316
成品尺寸: 184*260
出版时间: 2012-4-30
出版时间: 2012-4-30

1.3 目录

1.4 特色及精选内容

本书的特色在于紧跟计算机软件的更新，内容主要包含了 Windows 7 操作系统，Office 2007 办公自动化软件的内容。此外，在课本的第一章计算机基础知识中阐述了计算机的组成及工作原理、数据处理、多媒体技术等内容。第六章计算机网络与因特网基础，介绍了网络的分类、结构及协议，其中因特网的接入方式与应用是非常实用的内容。第七章程序设计基础为后续要学习的程序设计做了较好的铺垫。

本书的精选内容如下：



4.13　Excel 2007 的网络功能

随着网络技术的发展，与网络集成也成为 Office 软件的重要功能之一。在 Excel 2007 中，可以在局域网和 Internet 上共享工作簿，并能将工作簿保存为网页格式，方便用户直接将其发布到网页上。此外，可以使用 Excel 中的超链接功能为单元格或工作表创建超链接。

4.13.1　共享工作簿

共享 Excel 工作簿可以让网络中的其他用户浏览与编辑本地的工作簿文件。

当工作簿中的表格需要局域网中其他用户审阅或批注意见时，就需要在局域网中共享制作好的工作簿。

具体方法如下：

① 启动 Excel，打开需要同时浏览和编辑的工作簿。

② 单击“审阅”选项卡“更改”组中的“共享工作簿”按钮，打开“共享工作簿”对话框，选中其中的“允许多用户同时编辑，同时允许工作簿合并”复选框（见图 4-120），单击“确定”按钮。

③ 将该工作簿文档复制（或剪切）到局域网中某台计算机的共享文件夹中。

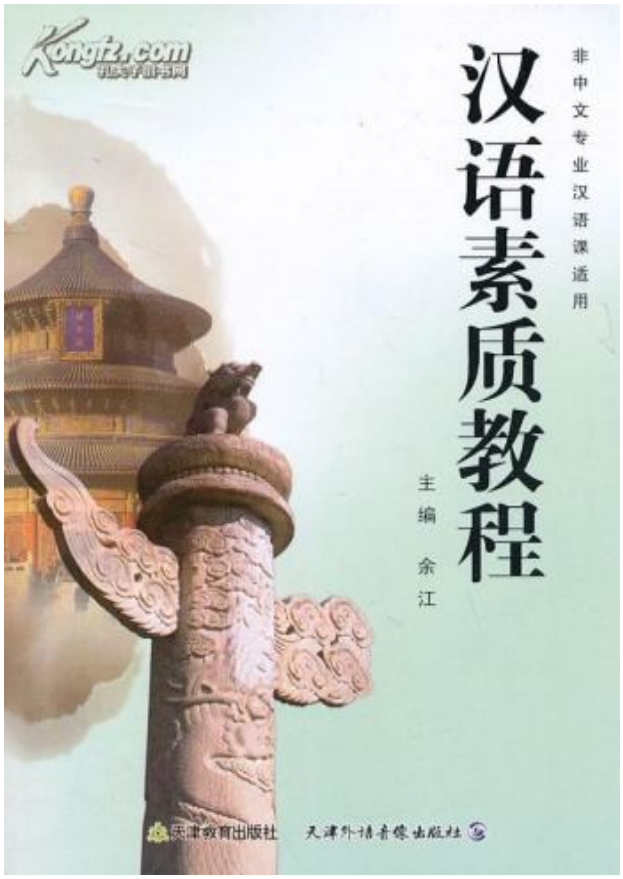
④ 从其他计算机进入上述共享文件夹，即可多人同时浏览和编辑该工作簿了。

4.13.2　创建超链接

图 4-120　共享工作簿

2 汉语素质教程

2.1 教材封面



2.2 出版信息页

作者：余江主编

出版社: 天津教育出版社; 第 1 版 (2010 年 1 月 1 日)

平装: 250 页

语种：简体中文

开本: 32

ISBN: 9787530959473

条形码: 9787530959473

商品尺寸: 21 x 16 x 2.6 cm

商品重量: 340 g

ASIN: B0037VBC4W

2.3 目录

第六节 音变
第七节 语调
第三章 汉字
第一节 文字与汉字
第二节 汉字历史
第三节 “六书”
第四节 汉字字形
第五节 汉字的书写
第六节 汉字信息处理
第四章 基本语法
第一节 语法概说
第二节 词的构造
第三节 词类
第四节 词组
第五节 单句
第六节 复句
第七节 标点符号
第五章 修辞
第一节 什么是修辞
第二节 修辞与语境
第三节 修辞与修辞者
第四节 修辞与语体
第五节 修辞与语言要素
第六节 修辞与修辞格

2.4 特色及精选内容

《汉语素质教程》主要有以下三个特点：1.针对性强。非中文专业“现代汉语”类公共课适用，目的是为了提高全体大学生的汉语基本素质。2.体现素质教育理念。不固守中文专业传统现代汉语教材的编写体例，不贪多求全，不强调理论讲述，多进行举例说明，内容上适当突出一个大学生应知应会的汉语基本知识及其运用。3.形式新颖。本教材根据公共课多使用多媒体教学手段的特点，采用“书+光盘”的形式，将教材内容和补充的“思考与练习”题目及参考答案都制作成 CD—ROM 光盘，既便于老师教学参考，又利于学生预习、复习和学练结合，以便获得更好的教学效果。

3 计算机网络

3.1 教材封面

3.2 出版信息页

出版社：西安电子科技大学出版社

出版数量：4000 册

3.3 目录

3.4 特色及精选内容

5.5 交换式以太网

在传统的以太网中，采用 CSMA/CD 介质访问控制协议。这种传统的介质访问控制方法

使众多的站点处于一个冲突域中，冲突域中的各站点共享一个公共传输介质，在任何给定时间内，共享介质局域网只允许一个工作站在一个方向上发送数据（即半双工传输模式），各站点共享网络固定的带宽(如 10Mb/s)。如果网上共连接了 *n* 个站点，那么每个站点平均分享到的带宽只有总带宽的 1/*n* 当存在冲突时还要低，网络系统的效率会随着节点数的增加和应用的增多而大大降低。另外，由于冲突域的限制，传统的以太网也难以构建较大规模的网络。为了克服网络规模与网络性能之间的矛盾，人们提出将共享介质方式改为交换方式，将半双工传输模式改为全双工传输，这就导致了交换式局域网和全双工以太网发展。

20 世纪 90 年代初出现了以太网交换机或称交换器(switch)，也称为交换式集线器(switching hub)。由交换机连接的交换式以太网能增加传统以太网的带宽，同时又能与传统的电缆线和网络适配器协调工作，因而可以保留已有的网络基本设施。

交换机接收并暂存传入的帧，根据目的地址和一个端口-MAC 地址表转发到另一个输出端口。交换机是由网桥(bridge)发展而来的，技术上非常类似网桥，不少专家认为，使用新的名字主要是市场的原因。1984 年网桥就开始进入市场，用于连接扩展局域网。早期的网桥一般只有两个端口，连接两个局域网网段，就像一座桥连接两段路一样。而交换机有多个端口，而且交换机的功能由网桥的基于软件转向使用先进的专用集成电路 ASIC 硬件，使转发速度大大加快，交换机本质上是一个高速的多口网桥(multiport bridge)，它们都工作在数据链路层的 MAC 子层，每个端口都包含一个 MAC 实体，但没有 MAC 地址。

1997 年制定的 IEEE802.3x 标准定义了全双工以太网，其主要特点如下：

①全双工以太网能够同时发送和接收数据，因此它可以提供半双工模式两倍的带宽。为此，需要使用能够同时进行数据发送和接收的媒体类型，10BaseT、10BaseFL、100BaseT4、100BaseTX、100BaseFX、100BaseT2、1000BaseX 的媒体系统支持全双工模式。

②全双工以太网使用交换机通过点对点链路连接计算机组成，在点对点媒体段上只能连接一对站点。计算机的网络接口和交换机必须支持全双工模式。

③使用和半双工以太网同样的帧格式、最小帧长、帧间隙 IFG 和 CRC 校验等。

④不再使用 CSMA/CD 媒体接入控制方式，是无冲突的(collision free)。网络长度也就不受 CSMA/CD 时槽的限制。基于多模光纤的 100BasetFX 的网段长度由半双工模式的 412m 扩大到 2000m，其他 100 兆及以下的以太网网段长度没有变化，这与电缆的信号传输特性有关。1000BaseX 多模和单模光纤在全双工模式下，网段跨距分别可达到 550m 和 5000m。

⑤定义了显式的流量控制。为了进行流量控制，定义了 MAC Control 帧。

全双工模式可以用于以下场合：

①交换机到交换机的连接，它们之间往往有较长的距离。

②交换机到服务器的连接，可以加倍服务器的链路带宽。

③长距离计算机设备的连接。

5.5.1 网桥

1.网桥的工作原理

网桥(bridge)属于数据链路层互联的设备，它在网络互联中起到数据接收、地址过滤与数据转发的作用，它用来实现两个或多个局域网之间的数据交换。

网桥一般有两个端口，桥接两个网段（局域网）。每个端口有一块网卡，有自己的 MAC 子层和物理层。图 5-23 中所示的网桥，其端口 1 与网段 A 相连，而端口 2 则连接到网段 B。

网桥工作在数据链路层的 MAC 子层，其基本功能是在不同局域网网段之间转发帧，转发中不修改帧的源地址。这里的网段可以是单个网段或是由中继器连接的多个网段组成的大网段。网桥从端口接收该接口所连接的网段上传输的帧，每当收到一个帧，就先存于缓冲区中。若此帧未出现传输差错而且目的站属于其他网段，则根据目的地址通过查找存有端口一

MAC 地址映射的桥接表，找到对应的转发端口，将它从该端口发送出去；否则，就丢弃此帧。在同一个网段中通信的帧，网桥不进行转发。

以图 5-23 为例，设网段 A 的三个站①、②和③的 MAC 地址分别为 MAC-1、MAC-2 和 MAC-3，而网段 B 的两个站④和⑤的 MAC 地址分别为 MAC-4 和 MAC-5。若网桥的端口 1 收到站①发给站②的帧，目的地址为 MAC-2，根据桥接表，知道 MAC-2 也连接在端口 1，此帧属于同一网段上传输的帧，因此网桥不转发，将它丢弃。若端口 1 收到站①发给站④的帧，目的地址为 MAC-4，查找桥接表后知道 MAC-4 所在网段连接在端口 2，属于向不同的网段上传输的帧，若此帧没有传输差错，就将它经端口 2 转发到网段 B。

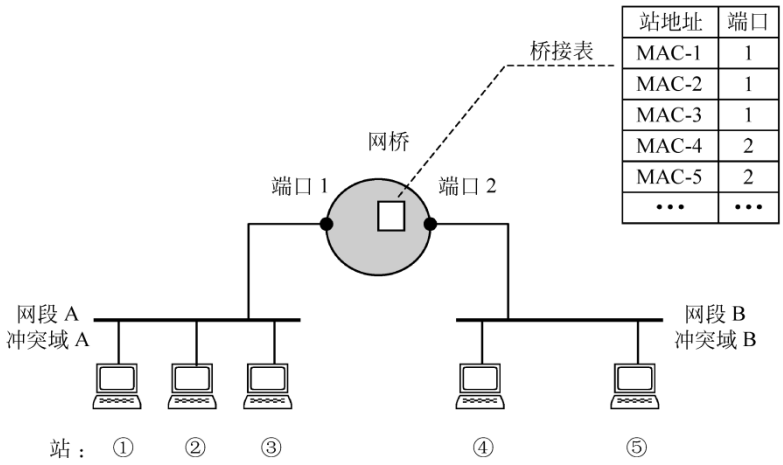


图 5-23 网桥的工作原理

虽然网桥和中继器、集线器都能扩展局域网，但它工作在更高的层次，有更强的功能，当然也有更大的处理时延，其主要特点如下：

（1）工作在 MAC 子层

网桥要检查帧的 MAC 地址，并据此查找桥接表，进行帧的转发。

（2）进行帧过滤减少了通信量

网桥使同一个网段上各工作站之间的通信量不会经过网桥传到其他网段上去，仅局限于本网段的范围之内。但网桥不限制广播帧。由于网桥这种过滤(filtering)作用，减少了无谓的传输，减轻了局域网上总的负荷。而中继器没有这种过滤功能，它对所有的帧，包括无效帧，都不加选择地一律转发。但网桥的转发时延要比中继器大。

（3）隔离冲突域扩大了网络跨距

帧过滤功能使得由网桥连接的以太网的不同网段上同时传送数据时，不会产生冲突。例如，图 5-23 网段 A 上的站①和网段 B 上的站④，它们同时发送数据给其他同一网段或另一网段上的站点，如站①发给站③，站④发给站②，由于它们的帧分别在不同的网段上传送，因此不会冲突。可见，网桥每个端口所连接的网段各属于一个冲突域，如图 5-23 中的网段 A 和网段 B，被分隔为两个独立的冲突域 A 和冲突域 B，使整个网络跨度不受单个以太网冲突域的限制。

（4）可连接不同类型的局域网

中继器和集线器只能连接同一类型的局域网，而网桥则可以连接不同类型的局域网。如通过网桥可以把以太网、令牌总线网和令牌环网连接在一起。当然这种网桥要比连接同类局域网的网桥要复杂一些，它需要进行帧格式的转换。

2.网桥的层次结构

下面以一个网桥连接 802.3 协议的 Ethernet 与 802.5 协议的 Token Ring 为例，进一步讨论网桥的协议结构问题。图 5-24 给出了网桥的协议结构与工作原理示意图。

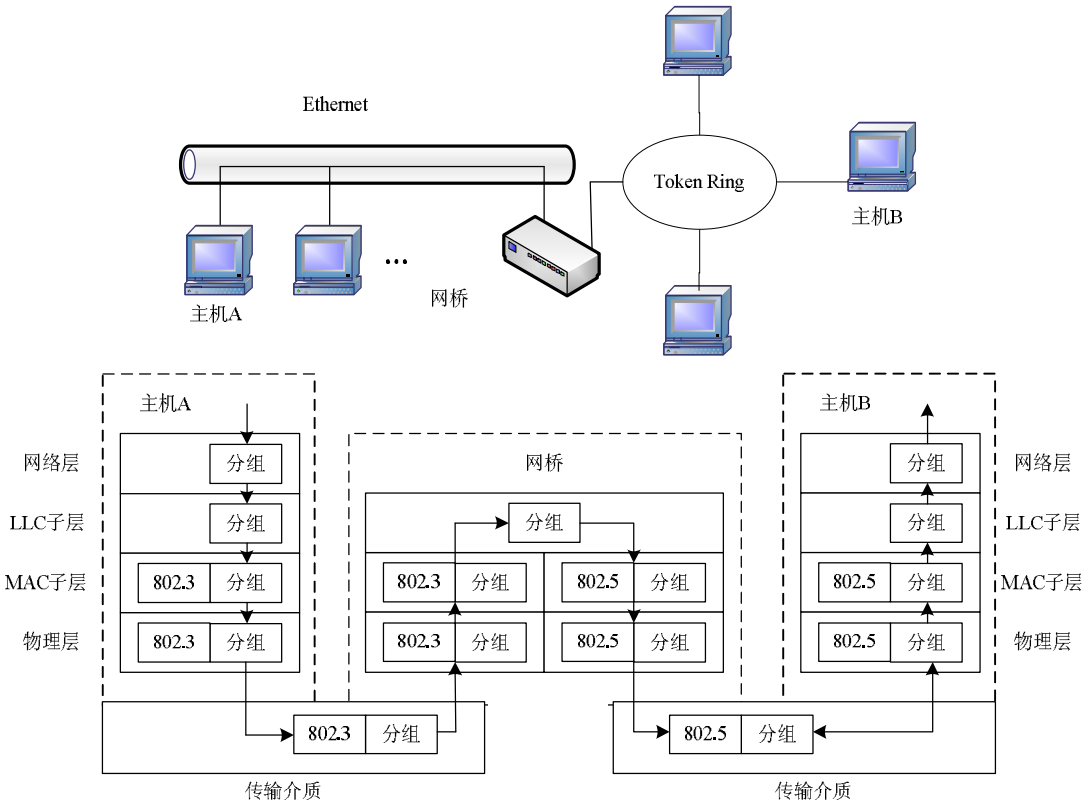


图 5-24　网桥的协议结构与工作原理示意图

网桥的两个端口分别通过 Ethernet 网卡和 Token Ring 网卡连人 Ethernet 与 Token Ring 网中。如果 Ethernet 网中主机 A 有一个分组要发送给 Token Ring 网中主机 B，那么主机 A 的网络层将分组传送给逻辑链路 LLC 子层，逻辑链路子层在分组前加上一个 LLC 分组头后，传送到介质访问控制 MAC 子层；介质访问控制子层按照 802.3 协议的规定，组装成 Ethernet 帧，帧的目的地址为主机 B 的地址；主机 A 的物理层将帧发送到 Ethernet 网中。网桥的一个端口通过 Ethernet 网卡连入主机 A 所在的 Ethernet 网中。网桥在接收到该帧后，由介质访问控制 MAC 子层检查该帧接收是否正确；如果接收正确，将 LLC 分组送给 LLC 子层；网桥软件对 LLC 分组进行处理，它将根据路径选择算法与目的 MAC 地址、源 MAC 地址，确定是否应向 Token Ring 网转发。如果不需要转发，则丢弃该帧；如果需要转发，则将 LLC 分组送到 Token Ring 的介质访问控制子层；介质访问控制子层按照 802.5 协议的规定，组装成 Token Ring 帧；该帧将通过物理层发送到 Token Ring 网中。Token Ring 网中的主机 B 在物理层在接收到该帧后，将它传送给主机 B 的介质访问控制子层、逻辑链路子层，然后将接收分组传送到主机 B 的高层。

从网桥的协议结构的分析中可以看到，网桥的端口分别通过使用不同的物理层和介质访问控制子层协议，分别连接到不同的局域网中。网桥通过路由选择软件完成是否转发，以及通过哪一个端口转发的判断。网桥在转发过程中不更改接收帧数据字段的内容与格式。

3.网桥的路由选择策略

（1）地址学习

常用的网桥是透明网桥(transparent bridge)。透明网桥一般用在两个使用同样的 MAC 层协议的网段之间的互联，例如连接两个 Ethernet 网，或两个令牌环网。透明网桥的标准是 1990

年的 IEEE802.1d 或 ISO8802.1d。透明网桥是由网桥自己来决定路由选择，而局域网上的各个站都不介入路由选择。透明的意思是局域网上的每个站，不需要知道也不知道所发送的帧将经过哪几个网桥，网桥对各站来说是看不见的、透明的。透明网桥完全不需要管理，便于安装运行，通电后就可以工作，不需要管理人员干预，属于即插即用设备。

图 5-25 给出了网桥利用桥接表实现不同网段之间帧转发的过程。网桥最重要的工作是构建和维护桥接表。桥接表中记录了不同结点的物理地址与网桥转发端口关系。没有桥接表，网桥没有办法确定帧是否需要转发，以及如何转发。

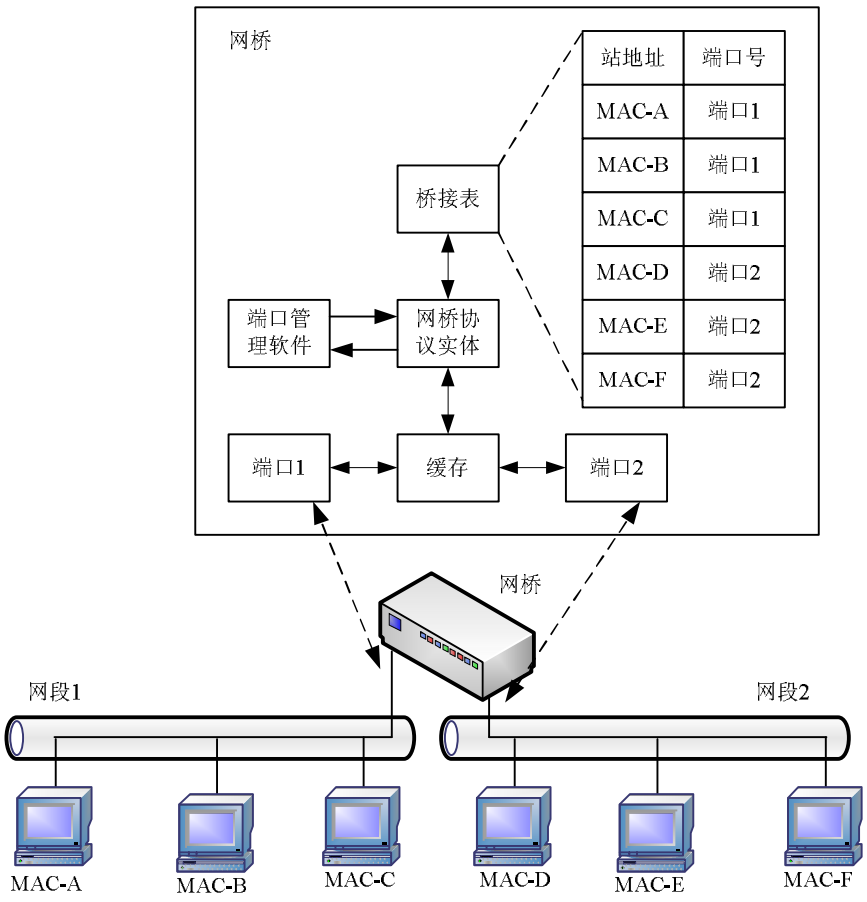


图 5-25 网桥帧转发过程示意图

一个网桥刚刚连接到局域网上时，其桥接表是空的。显然，此时网桥暂时还无法作出转发决策。此时网桥若收到一个帧，就采用洪泛法(flooding)转发它，即向除上游端口(接收此帧的端口)以外的所有端口转发。如果此帧又传输到了另一个网桥且其桥接表也是空的，则该网桥也用洪泛法转发此帧。这样进行下去就一定可以使该帧到达其目的站。

网桥在转发过程中通过学习将其桥接表逐步建立起来，学习的方法是所谓的逆向学习法(backward learning)。例如，对于图 5-25 的情况，假定网桥收到从端口 1 发来的帧，从帧的头部的地址信息中得知源站的地址为 MAC-A。于是，网桥就可以推论出，在相反的方向上，只要以后收到发往目的地址 MAC-A 的帧，就应当由端口 1 转发出去。于是就将地址 MAC-A 和端口 1 作为一个表项(即一行)登记在桥接表中，如图 5-25 中桥接表的第 1 行所示。这样，在转发过程中通过学习就把桥接表逐步建立起来。

局域网的拓扑可能会发生变化。为了使桥接表能动态地反映出网络的最新拓扑，可以在登记一个表项时将帧到达网桥的时间也记录下来。网桥中的软件周期性地扫描桥接表，只要是在规定的时间范围(例如几分钟)之前登记的表项，则予以清除，重新学习，这样就使得桥接表能及时反映网络的变化(为了从简，图 5-25 中的桥接表中没有画出时间)。

(2)生成树算法

在很多实际应用中，一个分布在企业内部或校园网中的，我们很难保证通过网桥互联的系统中不会出现图 5-26 所示的环状结构。环状结构可能使网桥反复地复制和转发同一个帧，从而增加了网络不必要的负荷，降低系统性能。为了防止出现这种现象，透明网桥使用了一个生成树(spanning tree)算法。

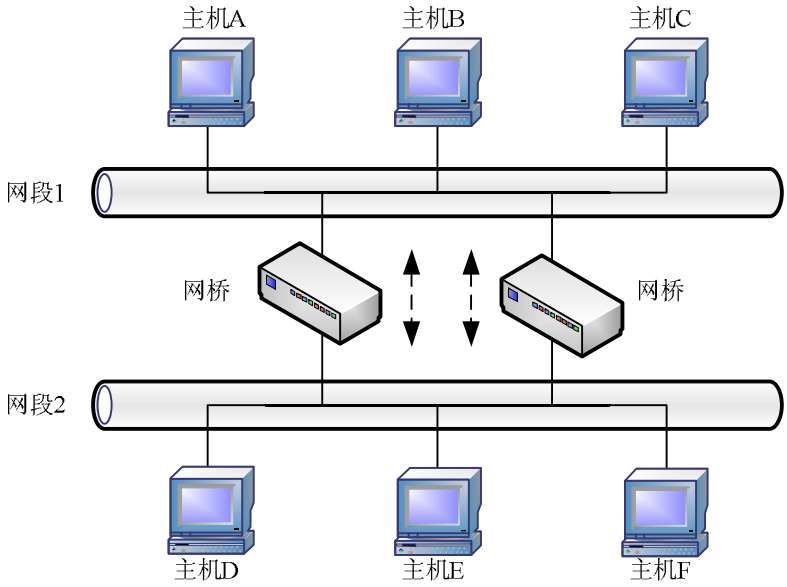
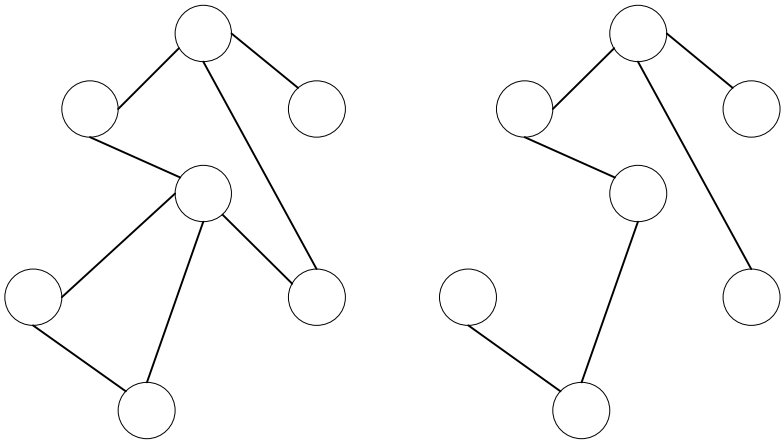


图 5-26 网桥互联的环状结构

为了建造生成树，首先必须选出一个网桥作为生成树的根。实现的方法是每个网桥广播其序列号，该序列号由厂家设置并保证全球唯一，选序列号最小的网桥作为根。接着，按根到每个网桥的最短路径来构造生成树。如果某个网桥或局域网失败，则重新计算。该算法的结果是建立起从每一个局域网到根网桥的唯一路径。该过程由生成树算法软件自动运行产生；拓扑结构改变时将更新计算生成树。图 5-27(a)是一个包含环路的网络拓扑结构，图 5-27(b)是经过计算后产生的一个逻辑上无环路的生成树。



（a）带环图　　（b）对应的生成树

图 5-27 网络拓扑与对应的生成树

生成树算法通过网桥和网桥之间的一系列协商，构造出一个生成树。这些协商的结果是：每个网桥都有一个端口都被置于转发状态，而其他端口则被置于阻塞状态。该过程将保证网络中任何两个设备之间只有一个通路，并可以防止出现任何形式的环路，创建了一个逻辑上无环路的网络拓扑结构。

4.网桥与广播风暴

我们在讨论网桥的工作原理时已经知道，网桥工作在第二层(数据链路层)。网桥通过接收数据帧、地址过滤、存储与转发数据帧的方式，来实现多个局域网系统的互联。网桥根据局域网中数据帧的源地址与目的地址来决定是否将接收和转发数据帧。根据网桥的工作原理，网桥对同一个子网中传输的数据帧不转发，因此可以达到隔离互联的子网的通信量的目的。因为网桥要确定传输到某个目的结点的数据帧要通过哪个连接端口转发出去，就必须在网桥中保存一张“端口一结点地址表”。同时，网桥中保存“端口-结点地址表”的存储器空间是有限的。因此随着网络规模的扩大与用户结点数的增加，会不断出现“端口-结点地址表”中没有的结点地址信息。当带有这一类目的地址的数据帧出现时，网桥无从决定应该从哪个端口转发。那它唯一的办法就是通过所有的端口广播出去，只要这个结点在互联的局域网中，那么广播的数据帧总有可能到达目的结点。这种方法很简单，但是却带来了很大的问题。那就是“盲目地”广播会使网络无用的通信量剧增，造成“广播风暴”。路由器可以从根本上网桥的广播风暴解决这个问题。

10.2 密码学基础

10.2.1 概述

将信息加密后进行传输是传统的保密通信的主要手段，计算机网络安全中的信息加密、数字签名、身份认证等都是以密码学为基础的。

1.密码算法相关的几个概念

①明文（plaintext） 原始的未经加密处理的信息称为明文，它是一段有意义的文字或数据。明文通常用 m 或 P 表示。

②密文（ciphertext） 经过加密处理的信息称为密文，表面上看密文是一串杂乱无序的无意义的符号和数字。密文通常用 C 表示。

③加密（encrypt） 加密的基本思想是伪装明文以隐藏其真实内容，即将明文 X 伪装成密文 Y。伪装明文的操作称为加密。

④解密（decrypt） 将密文恢复成明文的过程称为解密。

⑤加密算法：是将明文变换为密文的变换函数，相应的变换过程称为加密，即编码的过程（通常用 E 表示，即 C=Ek(m)或 C=E (m,k)）。

⑥解密算法：是将密文恢复为明文的变换函数，相应的变换过程称为解密，即解码的过程（通常用 D 表示，即 m=Dk(c)或 m=D (C,k) ）。

⑦密钥（key） 是参与密码变换的参数，加密和解密过程要使用密钥，它控制加密和解密算法。密钥是独立于明文的，同一个明文用同一个算法，但用不同的加密密钥进行加密将产生不同的密文。接收者在收到密文后利用解密算法和解密密钥将密文还原为明文。密钥通常用 K 表示。K 可以是很多数值里的任意值。密钥 K 的可能值的范围叫做密钥空间(key space)。对于同一种加密算法，密钥的位数越长，破译的困难也就越大，安全性也就越好。因为密钥位数越长，密钥空间就越大，攻击者也就越不容易通过蛮力攻击(brute—force attack)来破译。在蛮力攻击中，破译者可以用穷举法对密钥的所有组合进行猜测，直到成功地解密。

2.传统密码技术

使用密码技术古已有之。早期的密钥密码体制中有两种常用加解密形式，一种是替代密码(substitution cipher)，另一种是变位密码(transposition cipher)。

（1）替代密码

替代密码是将明文中每个(或每组)字母，由另一个(或另一组)字母所替代。最古老的一种替代密码是凯撒密码(Caesar cipher)。将英文的26个字母a，b，…，z分别替代成g，h，…，f，相差6个字符，于是明文 timebombwillblowupatfive 就变成了密文 zoskhushcorrhrucavgzloyk，这里密钥为6。

（2）变位密码

变位密码是按照某一规则重新排列报文中的字符顺序，一种常用的变位密码是列变位密码，下面是一个简单的例子，使用密钥 bridge 对明文 timebombwillblowupatfive 进行加密：

密钥：b r i d g e

顺序：1 6 5 2 4 3　　（根据密钥字母在字母表中的顺序确定列顺序）

明文：t i m e b o

m b w l l l

b l o w u p

a t f i v e

根据密钥给出的顺序，按列的顺序由小到大重新安排明文字符的位置，加密变换得到的密文是：tmbaeiwiolpebluvmwofiblt。

3.现代密码体制

早期的密码技术已经过时，庞培也许确实不能破译凯撒密码，但现代的科学和计算机技术可以解决非常复杂的问题，人们设计了相当复杂的密码技术。现代密码学中有两种密码体制，即对称密钥密码体制(symmetric key cryptography)和公开密钥密码体制(public key cryptography)，前者的主要算法是 DES，后者的主要算法是 RSA，本节将分别进行介绍。

在现代密码学研究中，加密和解密算法是要经过极大的努力进行设计、测试和安装的，难以频繁地改变，一般要经过几年才会更新，将加密和解密算法本身进行保密的做法在现实中是不可行的。而密钥是相对较短的字符串，可以容易地频繁改变。密码技术中的一个原则是：加密和解密算法是公开的，而密钥是保密的。这称为 Kerckoff 原则(Kerckoff's principle)，军事密码学家 Augueste Kerckoff 在 1833 年首先提出了这一思想。

保守密钥的秘密无疑是防止攻击的关键。对于攻击者来说，密钥的穷举搜索(exhaustive search)是一种重要攻击手段。但当密钥足够长且随机分布时，以当时的计算水平，穷举搜索实际上难以实现。比如，如果密钥长度为二进制 128b，则密钥空间为 2^{128}。，约 3.4×10^{38}（见表 10-1），即使计算机对密钥空间的搜索速度可以达到每微秒 100 万次，那么完成密钥空间全部搜索的时间也将超过 10^{11} 亿年，我们称这样的密钥是计算上不可破译的。实用的密码体制一般都是计算上不可破译的，而不是理论上不可破译的。

表 10-1　密钥长度与密钥个数

密钥长度（位）	组合个数
40	2^{40}=1 099 511 627 776
56	2^{56}=7.205 759 403 793×10^{16}
64	2^{64}=1.844 674 407 371×10^{19}
112	2^{112}=5.192 296 858 535×10^{33}

128	$2^{128}=3.402\ 823\ 669\ 209\times 10^{38}$

10.2.2 对称密钥密码体制

1.对称密钥密码体制及其特点

对称密钥密码体制也称为传统密码体制。对称密钥密码体制的加密和解密变换过程如图10-1 所示，加密和解密过程都包括一个算法和一个密钥。对称密钥密码体制的特点是，解密时使用的解密密钥和加密时使用的加密密钥是通信双方共享的同一密钥，它称为共享密钥(shared key)。

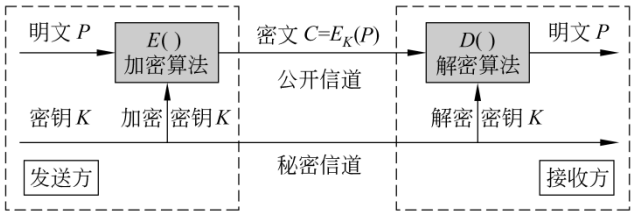


图 10-1 对称密钥密码体制

图 10-1 中，用 $C=E_K(P)$表示使用加密算法 $E(\ )$和密钥 K 对明文 P 加密得到密文 C，类似地，用 $P=D_K(C)$表示使用解密算法 $D(\ )$和密钥 K 对密文 C 解密得到明文 P，那么：$D_K(E_K(P))=P$。

密文 C 在公开信道中传输时，可能受到攻击者的攻击，比如截取和篡改等。

2.数据加密标准 DES

对称密钥密码体制中最重要的算法是数据加密标准(Data Encryption Standard，DES)。DES 是 IBM 开发的，1977 年被美国政府采纳为非机密信息的加密标准，在工业界广泛使用。

DES 算法是一种块密码(block cipher)算法，明文被分成 64b 的块，一块一块地进行加密。DES 算法的密钥长 64b，但每个字节的第 8b 是奇校验位，所以密钥的有效长度实际上是 56b。

DES 算法工作原理如图 10-2 所示，对 64b 明文的处理有 19 个步骤，过程如下：

首先，对 64b 的明文 P 进行初始变位(Initial Permutation，IP)。初始变位得出 P_0，其左半边 32b 和右半边 32b 分别记为 L_0 和 R_0。

然后，对 P_0 再进行 16 次迭代。如果用 P_i 表示第 i 次的迭代结果，其左半边 32b 和右半边 32b 分别记为 L_i 和 R_i，则变换算式为

$$L_i=R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

式中 i=1，2，…，16，K_i 是 48b 密钥，它从原来的 64b 密钥经过若干次变换而得出。每次迭代要进行函数 f 的变换、模 2 加运算(相当于异或运算)以及左右半边交换。16 次迭代后得到 $L_{16}R_{16}$。

最后，左右半边再交换为 $R_{16}L_{16}$，这是为了使算法既能加密又能解密，然后进行 IP 的逆变换 IP^{-1}，其输入是 $R_{16}L_{16}$，输出则为 DES 算法加密的密文 C。

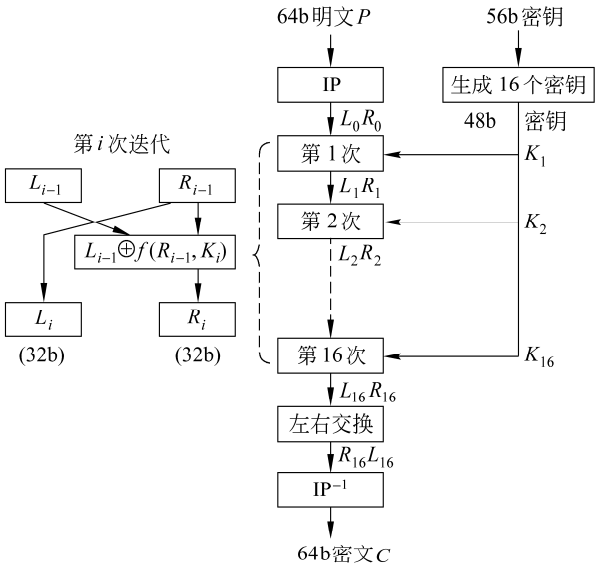


图 10-2 DES 加密算法

上述 DES 算法中的函数 f 起着关键的作用，它是一个非常复杂的变换，下面对它作一简单的介绍。$f(R_{i-1}, K_i)$ 先将 32b 的 R_{i-1} 进行变换，扩展为 48b，再与 48b 的 K_i 按比特进行模 2 加，所得结果顺序地划分为 8 个 6b 长的组 B_1，B_2，…，B_8。然后将 B_1，B_2，…，B_8 经过称为“S 变换”的替代转换为 4b 的组，可以记为 $B_j \rightarrow S_j(B_j)$（j=1，2，…，8），这里使用了 8 个不同的 S 函数 S_1( )，S_2( )，…，S_8( )(S 也是一个复杂的函数)。将所得的 8 个 4b 长的 $S_j(B_j)$ 按顺序排好，再进行一次置换，就得出 32b 的 $f(R_{i-1}, K_i)$。

解密过程和加密过程相似，但生成 16 个密钥的顺序正好相反。

上述的 DES 实际上就是一种长度为 64b 的单字符替代。它有一个明显的缺点，就是对相同的明文生成相同的密文，这增加了破译的机会。

为了提高 DES 的安全性，可采用密码块链接(Cipher Block Chaining，CBC)技术，构成 DES-CBC。在加密过程中，64b 的明文块 P_0 先和一个随机选择的初始向量(Initialization Vector，IV)逐比特进行异或运算，然后进行加密操作，得到密文 C_0；再将 C_0 和下一个 64b 的明文块 P_1 进行异或运算，然后进行再加密，得出密文 C_1；以后各块都用上述相同的方法进行操作。上述 DES-CBC 加密过程可以形式化表示为：

$C_0=E(P_0 \oplus \mathrm{IV})$

$C_1=E(P_1 \oplus C_0)$

$C_2=E(P_2 \oplus C_1)$

…

这样，采用了 DES-CBC，即使明文相同的块，加密后得到的密文也是不一样的。

DES-CBC 的解密过程是：密文块 C_0 先解密，再与 IV 进行异或运算，得出明文块 P_0；下一个密文块 C_1 解密后，和密文 C_0 进行异或得出第二个明文块 P_1；以后各密文块都用上述相同的方法重复进行操作。DES-CBC 解密过程可以形式化表示为：

$P_0=D(C_0) \oplus \mathrm{IV}$

$P_1=D(C_1) \oplus C_0$

$P_2=D(C_2) \oplus C_1$

…

DES 是世界上第一个公认的实用密码算法标准，它对密码学的发展作出了重大贡献。目前 DES 较为严重的问题是它的密钥的长度较短。56b 长的密钥意味着密钥空间为 2^{56}，约有 7.2×10^{16} 种密钥。假若一台计算机 1 微秒可执行 100 次 DES 算法，同时假定平均只需

搜索密钥空间的一半即可找到密钥，那么破译 DES 需要 11.42 年。

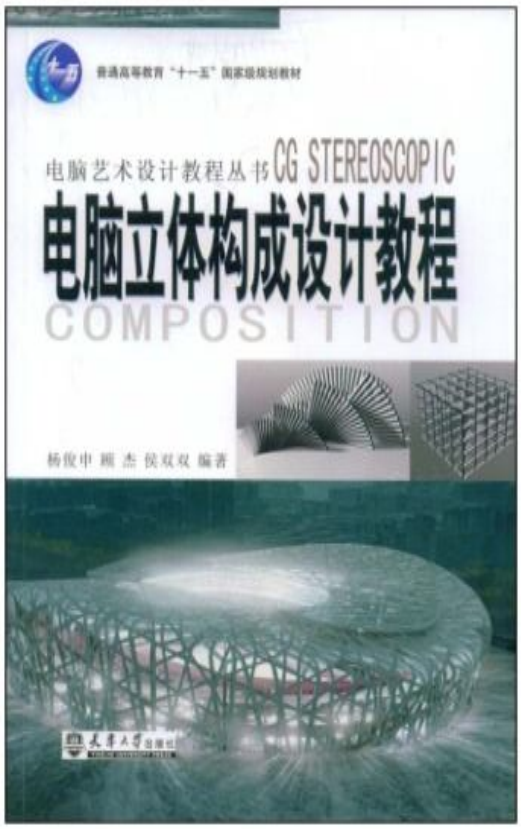
1997 年美国 RSA 数据安全公司在 RSA 安全年会上，公布了一项“秘密密钥挑战”竞赛，悬赏 1 万美元破译 56b 的 DES 密钥。1997 年一些在 Internet 上合作的人，用了 96 天破译了 DES 密钥。现在已经设计出搜索 DES 密钥的专用芯片，对 DES 构成了威胁。1998 年 7 月，电子边境基金会使用一台 25 万美元的专用计算机，花了 56 小时就破译了 DES 密钥。1999 年 1 月 RSA 数据安全会议期间，他们又把破译的时间缩短到 22 小时 15 分。

一种称为三重 DES(triple DES)的加密算法在 1985 年成为美国的一个商用加密标准。三重 DES 使用两个密钥，执行三次 DES 算法，加密过程是：$C=E_{K1}(D_{K2}(E_{K1}(P)))$，相应的解密过程是：$P= D_{K1}(E_{K2}(D_{K1}(C)))$。两个密钥合起来的长度有 112b，使密钥空间增加到 2^{112}，目前的计算技术很难破译，至今还没有攻破三重 DES 的报道。三重 DES 加密过程采用 E-D-E 而不是 E-E-E，这是为了与现有的 DES 系统向后兼容。加密和解密过程都是两个 64b 数之间的一种映射，从密码角度上来看，这两种映射的作用是一样的。

在 DES 之后又出现了著名的国际数据加密算法(International Data Encryption Algorithm，IDEA)。IDEA 使用长达 128b 的密钥，更不易被攻破。IDEA 和 DES 相似，也是先将明文划分为一个个 64b 长的数据块，然后经过 8 次迭代和一次变换，得出 64b 的密文。由两位年轻比利时密码学家 Rijmen 和 Daemen 提出的 Rijndael，也是非常优秀的对称密码算法，已成为美国联邦信息处理标准，它使用 128～256 位的密钥。

4 电脑立体设计构成教程

4.1 教材封面



4.2 出版信息页

作者: 杨俊申、顾杰、侯双双
书号：9787561829325
译者：
丛书名：普通高等教育“十一五”国家级规划教材·电脑艺术设计教材丛书
版次: B1
定价: 55.0 元
印次: Y1
装帧: 胶订
开本: 16 开
页数: 163
成品尺寸: 25.8 x 18.2 x 1.2 cm
出版时间: 2009-3-1

4.3 目录

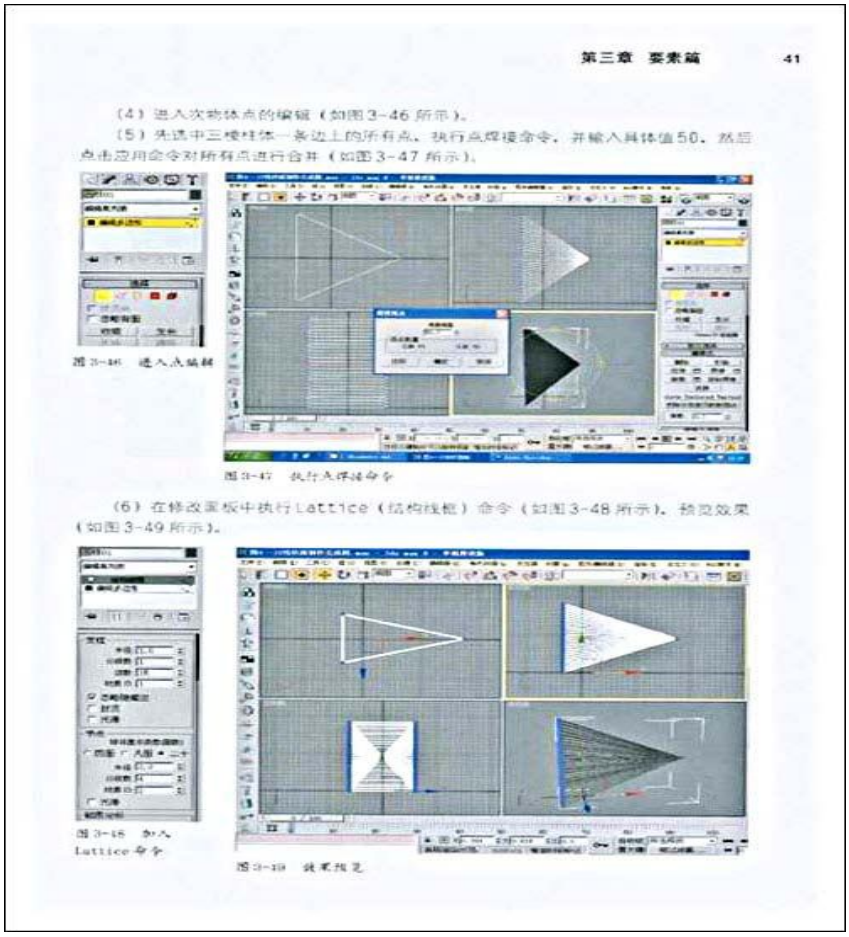
(二)体的类别
(三)由面到体的电脑制作
(四)体的立体造型构成及电脑制作表现
五、点、线、面、体的综合造型
(一)点、线、面、体的综合造型的意义
(二)综合造型在三维软件中的制作
第四章 法则篇
课题四 电脑立体构成的形式法则
一、变化与统一
(一)变化与统一是形式美的基本原理
(二)电脑立体构成中立体形态的对比变化因素
二、稳定与均衡
(一)稳定
(二)均衡
三、张力与运动
(一)张力
(二)运动
四、节奏与韵律
(一)节奏
(二)韵律
五、比例与尺度
(一)比例
(二)尺度
第五章 实践篇
课题五电脑立体构成的制作实践
(一)同体发射形体的制作
(二)组合体的制作
(三)使用 Haya 中 NurbS 曲面创建跑车座位
(四)人头制作
第六章 赏析篇
课题六优秀作品赏析
一、产品设计部分
二、人物部分
三、机器人部分
四、卡通部分
五、动画场景部分
六、静物部分
七、建筑设计部分
参考文献
后记

4.4 特色及精选内容

“电脑艺术设计教程丛书”之《电脑平面构成设计教程》《电脑色彩构成设计教程》《电脑立体构成设计教程》的编著是几位多年在教学第一线从事艺术设计教育教学的老师教学实践经验和成果的总结。该丛书依照艺术设计创作规律，突出实践教学内容，在全面、系统、

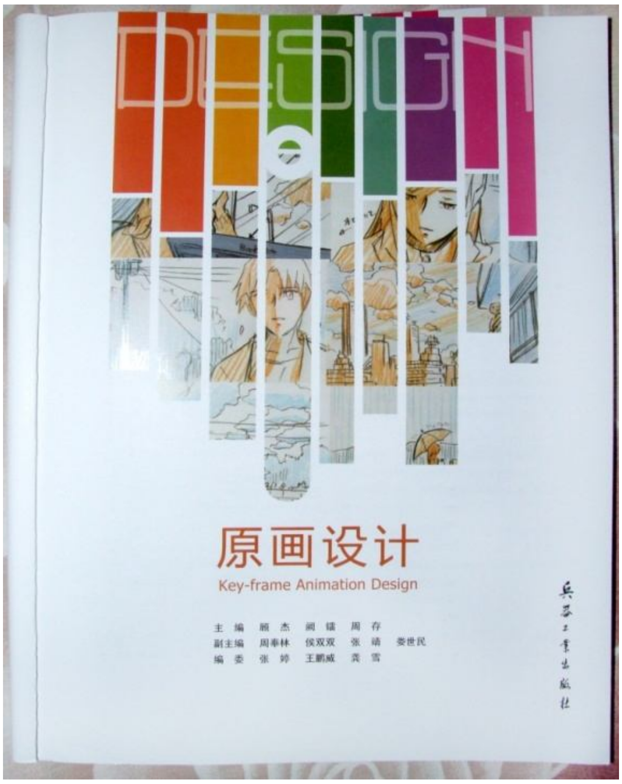
深入浅出地介绍现代设计构成理论的同时，又和电脑应用操作的学习结合一起，一举解决艺术设计与表现两方面的教学问题，使艺术设计学习事半功倍，是对艺术设计教育教学有益的探索。

本书的精选内容如下：

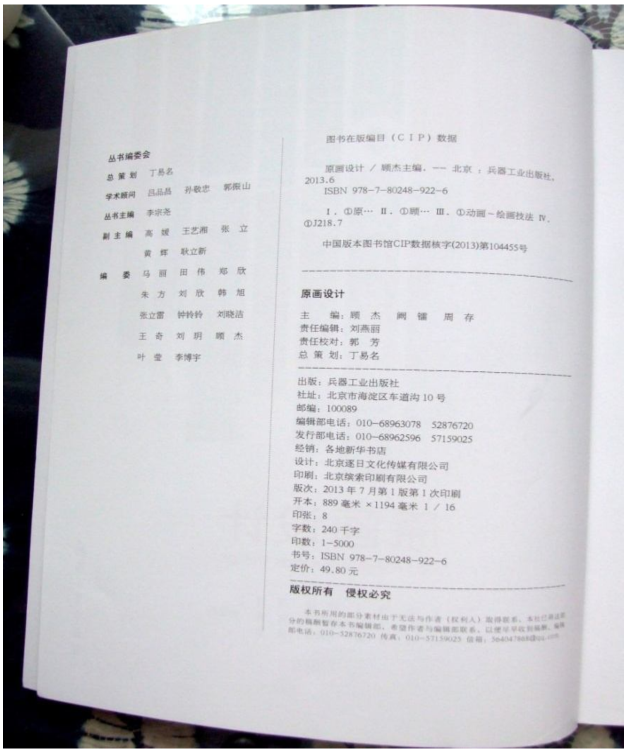


5 原画设计

5.1 教材封面



5.2　出版信息页



丛书编委会
总 策 划　丁易名
学术顾问　吕品晶　孙敬忠　郭振山
丛书主编　李宗尧
副 主 编　高　媛　王艺湘　张　立
黄　辉　耿立新
编　　委　马　丽　田　伟　郑　欣
朱　方　刘　欣　韩　旭
张立雷　钟铃铃　刘晓洁
王　奇　刘　玥　顾　杰
叶　莹　李博宇

图书在版编目（CIP）数据

原画设计 / 顾杰主编. —— 北京 ：兵器工业出版社，2013.6
ISBN 978-7-80248-922-6

Ⅰ. ①原… Ⅱ. ①顾… Ⅲ. ①动画－绘画技法 Ⅳ. ①J218.7

中国版本图书馆CIP数据核字(2013)第104455号

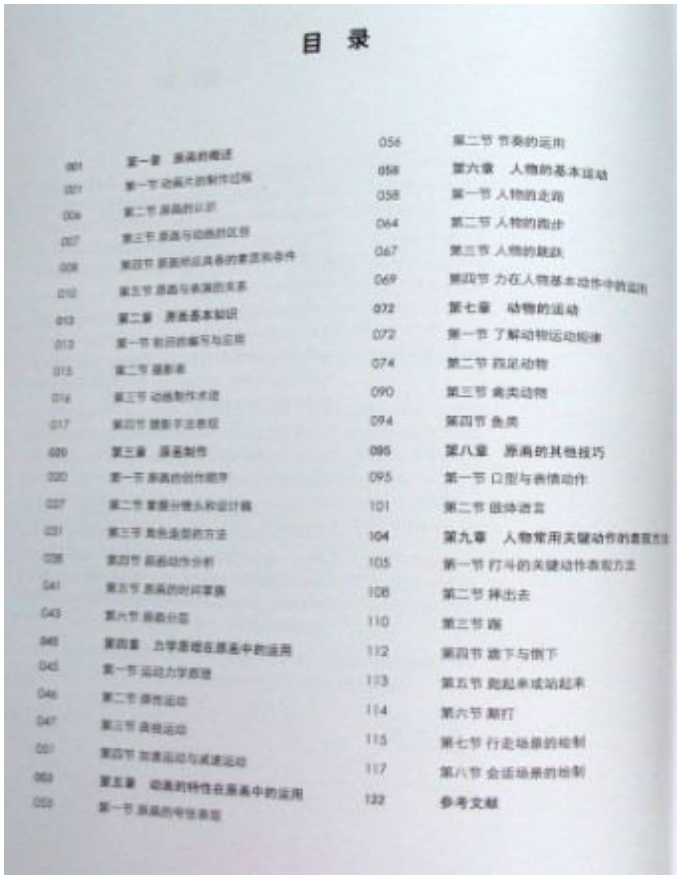
原画设计

主　　编：顾　杰　阚　镭　周　存
责任编辑：刘燕丽
责任校对：郭　芳
总 策 划：丁易名

出版：兵器工业出版社
社址：北京市海淀区车道沟 10 号
邮编：100089
编辑部电话：010-68963078　52876720
发行部电话：010-68962596　57159025
经销：各地新华书店
设计：北京逐日文化传媒有限公司
印刷：北京缤索印刷有限公司
版次：2013 年 7 月第 1 版第 1 次印刷
开本：889 毫米 ×1194 毫米 1 / 16
印张：8
字数：240 千字
印数：1-5000
书号：ISBN 978-7-80248-922-6
定价：49.80 元

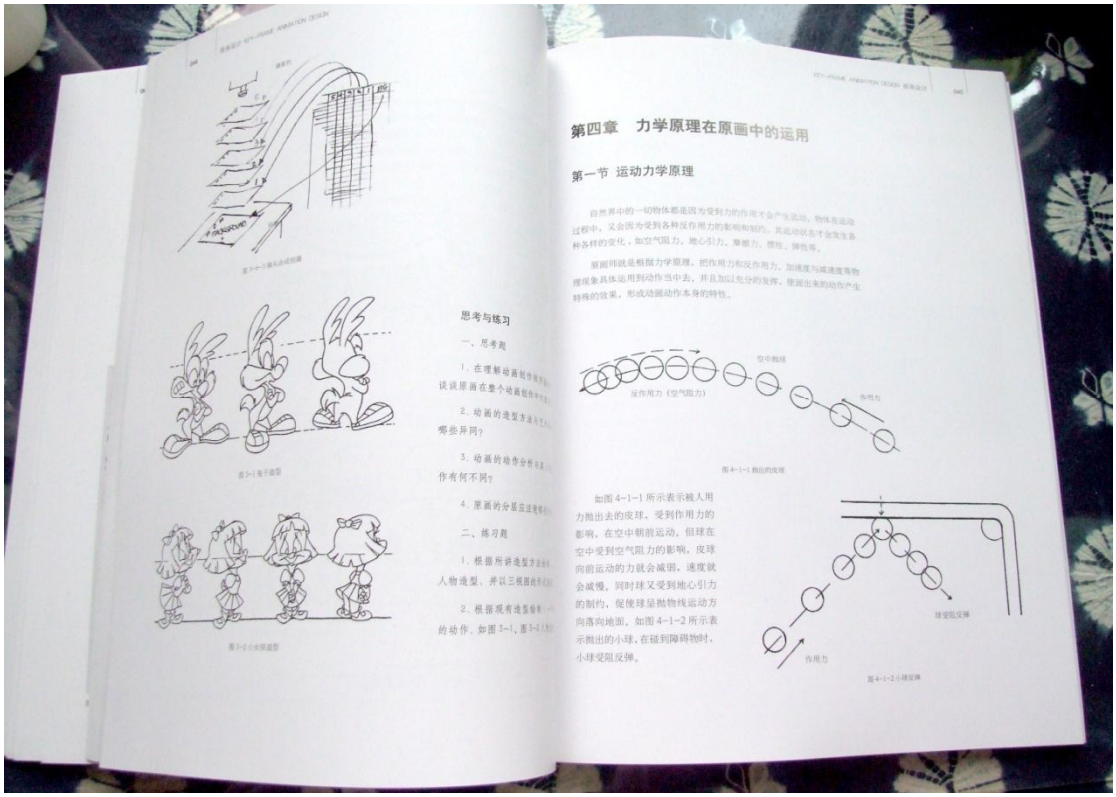
版权所有　侵权必究

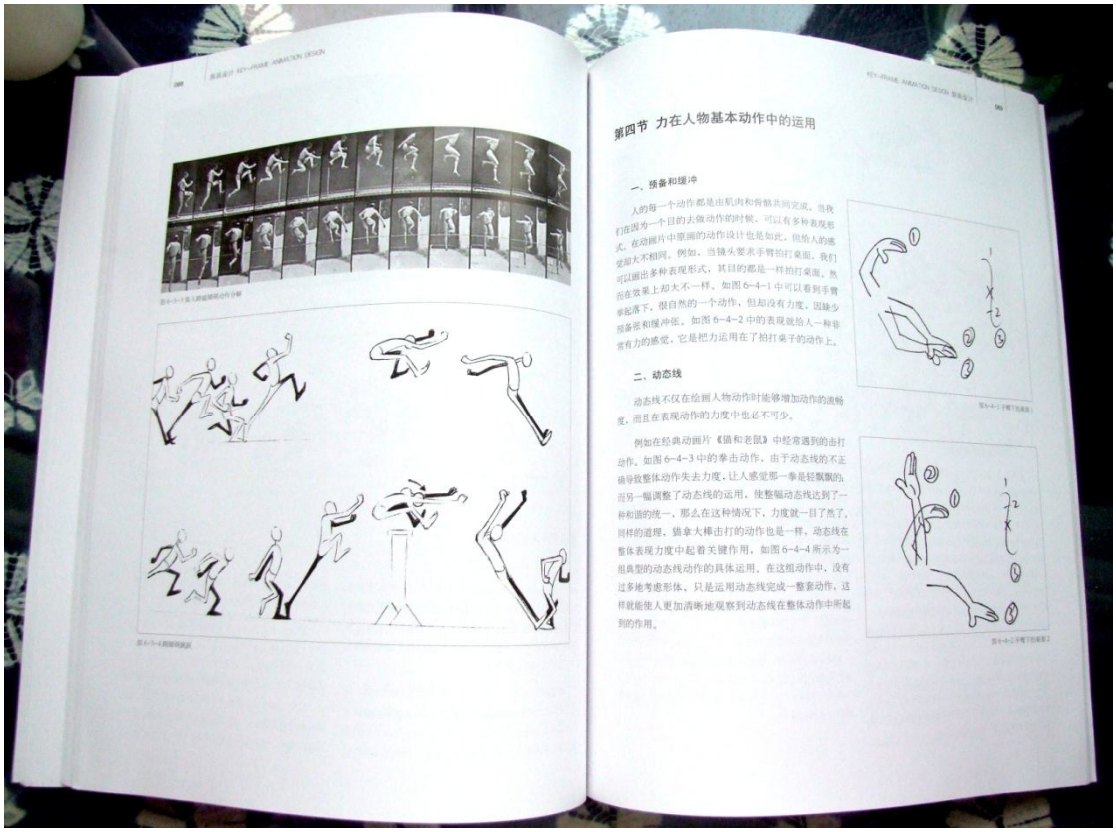
5.3　目录



目 录

5.4 特色及精选内容





第四节 力在人物基本动作中的运用

一、预备和缓冲

人的每一个动作都是由肌肉和骨骼共同完成。当我们在因为一个目的去做动作的时候，可以有多种表现形式。在动画片中原画的动作设计也是如此，但给人的感受却大不相同。例如，当镜头要求手臂拍打桌面，我们可以画出多种表现形式，其目的都是一样拍打桌面。然而在效果上却大不一样。如图 6-4-1 中可以看到手臂举起落下，很自然的一个动作，但却没有力度。因缺少预备张和缓冲张。如图 6-4-2 中的表现就给人一种非常有力的感觉，它是把力运用在了拍打桌子的动作上。

二、动态线

动态线不仅在绘画人物动作时能够增加动作的流畅度，而且在表现动作的力度中也必不可少。

例如在经典动画片《猫和老鼠》中经常遇到的击打动作，如图 6-4-3 中的拳击动作，由于动态线的不正确导致整体动作失去力度，让人感觉那一拳是轻飘飘的。而另一幅调整了动态线的运用，使整幅动态线达到了一种和谐的统一，那么在这种情况下，力度就一目了然了。同样的道理，锤拿大棒击打的动作也是一样，动态线在整体表现力度中起着关键作用，如图 6-4-4 所示为一组典型的动态线动作的具体运用。在这组动作中，没有过多地考虑形体，只是运用动态线完成一整套动作，这样就能使人更加清晰地观察到动态线在整体动作中所起到的作用。

思考与练习

一、思考题

1. 什么是曲线运动，如何运动在原画绘制中的重要性。

2. 力的传递过程在实际中的表现？

二、练习题

根据所学曲线运动规律完成以下练习

1. 花瓣动

2. 飘带的飘动

3. 龙的游动

4. 烟上升轨迹

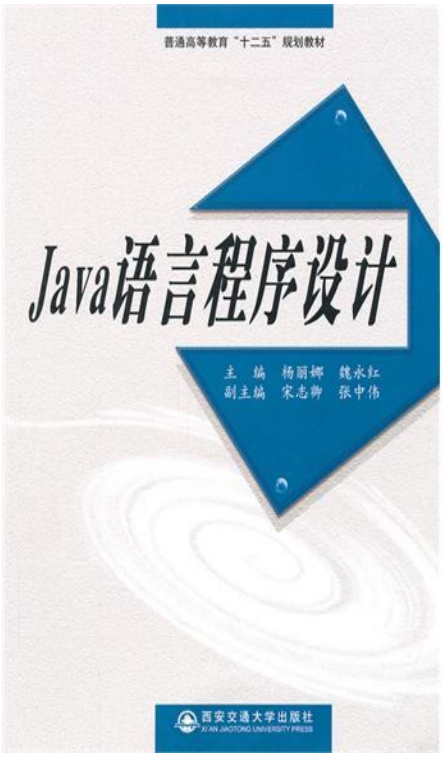
第五章　动画的特性在原画中的运用

第一节 原画的夸张表现

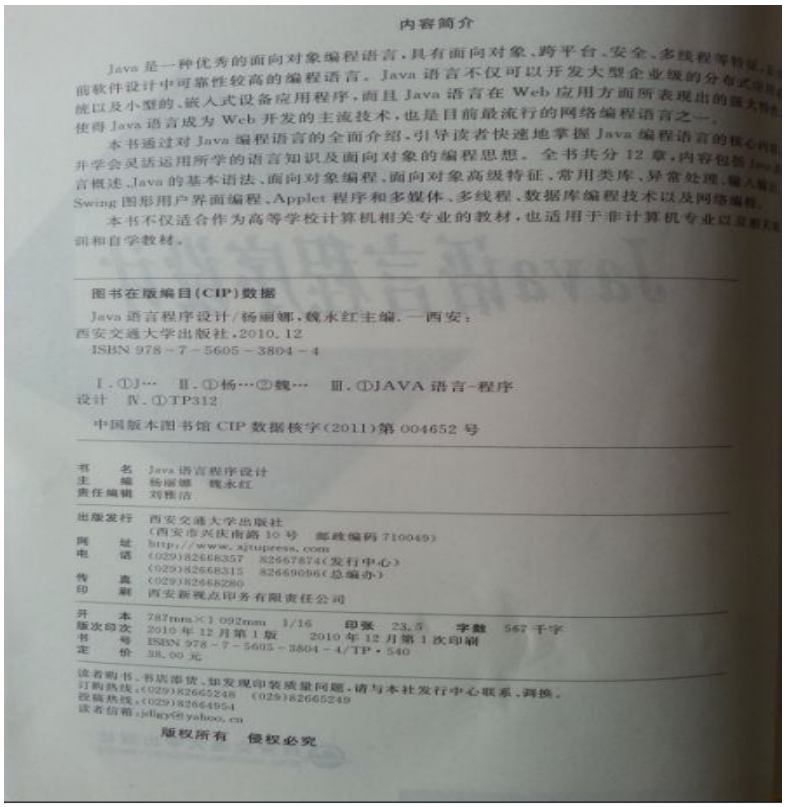
动画艺术最大的特点就是能够表现现实生活中无法满足的视觉图像，动画片中的夸张与趣味是动画艺术的生命与灵魂。原画师作为影片的动作设计者和主要绘制者，应当懂得夸张和想象在动画片中所起的作用，掌握并运用夸张这一技巧，动作将设计得更有特色。

6 JAVA 语言程序设计

6.1 教材封面



6.2 出版信息页



内容简介

Java是一种优秀的面向对象编程语言，具有面向对象、跨平台、安全、多线程等特征，是前软件设计中可靠性较高的编程语言。Java语言不仅可以开发大型企业级的分布式应用统以及小型的、嵌入式设备应用程序，而且Java语言在Web应用方面所表现出的强大特性使得Java语言成为Web开发的主流技术，也是目前最流行的网络编程语言之一。

本书通过对Java编程语言的全面介绍，引导读者快速地掌握Java编程语言的核心内容并学会灵活运用所学的语言知识及面向对象的编程思想。全书共分12章，内容包括Java语言概述、Java的基本语法、面向对象编程、面向对象高级特征、常用类库、异常处理、输入输出、Swing图形用户界面编程、Applet程序和多媒体、多线程、数据库编程技术以及网络编程。

本书不仅适合作为高等学校计算机相关专业的教材，也适用于非计算机专业以及相关培训和自学教材。

图书在版编目(CIP)数据

Java语言程序设计/杨丽娜，魏永红主编. —西安：西安交通大学出版社，2010.12
ISBN 978-7-5605-3804-4

Ⅰ.①J… Ⅱ.①杨…②魏… Ⅲ.①JAVA语言-程序设计 Ⅳ.①TP312

中国版本图书馆CIP数据核字(2011)第004652号

书 名 Java语言程序设计
主 编 杨丽娜 魏永红
责任编辑 刘雅洁

出版发行 西安交通大学出版社
（西安市兴庆南路10号 邮政编码710049）
网 址 http://www.xjtupress.com
电 话 (029)82668357 82667874(发行中心)
(029)82668315 82669096(总编办)
传 真 (029)82668280
印 刷 西安新观点印务有限责任公司

开 本 787mm×1 092mm 1/16 印张 23.5 字数 567千字
版次印次 2010年12月第1版 2010年12月第1次印刷
书 号 ISBN 978-7-5605-3804-4/TP·540
定 价 38.00元

读者购书、书店排货，如发现印装质量问题，请与本社发行中心联系，调换。
订购热线：(029)82665248 (029)82665249
投稿热线：(029)82664954
读者信箱：jdlgy@yahoo.cn

6.3 目录

目　录

第 12 章　Java 网络编程

6.4 特色及精选内容

第 8 章 基于 Swing 的图形用户界面

图形用户界面（Graphics User Interface，GUI）是应用程序的外观，是用户和程序之间的接口，也是程序设计的最重要的部分之一。友好和美观的图形界面可以帮助用户更好理解程序的功能，也为使用带来了很多便捷。Java 早期版本提供支持 GUI 设计的抽象窗口工具集（Abstract Window Toolkit，AWT）。从 JDK1.2 开始，Java 又提出了一种功能更强大、更加实用的开发 GUI 工具包即 Swing。本章介绍 Swing 概述、Swing 的常用组件、布局管理器以及事件处理机制，并通过大量的实例讲解基于 Swing 的 GUI 设计方法。

8.4 布局管理器

布局管理器负责对添加到容器中的组件进行布局，使设计的界面看起来更专业、更美观。布局管理器类都实现了接口 LayoutManager。常用的布局管理器有 FlowLayout、BorderLayout、GridLayout、CardLayout、GridBagLayout 和 BoxLayout，其中前 5 个布局管理器类在 java.awt 包中，而 BoxLayout 类在 javax.swing 包中。Java 中的每一个容器都有一个相关联的布局管理器来管理容器中的组件。一般调用容器的 setLayout 方法来设置容器所需要的布局管理器。

8.4.2 BorderLayout 布局管理器

BorderLayout 是 JFrame 和 JDialog 默认使用的布局管理器。BorderLayout 是一种简单的布局管理策略。它把容器内空间划分成 5 个区域：东、南、西、北和中。每个方位区域只能放一个组件。“北”占据容器的上方，“东”占据容器的右侧，“南”占据容器的下方，“西”占据容器的左侧，“中间区域”是东、南、西、北都填满后剩下的区域。BorderLayout 类提供了 5 个常量属性值：EAST、WEST、SOUTH、NORTH、CENTER 来表示东、西、南、北和中的位置。

BorderLayout 布局管理器根据组件的最佳尺寸和容器大小的约束条件来对组件进行布局。如果某个方位上无组件时，则其他方位上的组件自动进行缩放占有其位置：NORTH 和 SOUTH 组件可在水平方向进行伸展；EAST 和 WEST 组件可在垂直方向进行伸展；CENTER 组件可在水平和垂直两个方向上伸展，来填充整个剩余空间。

1.常用构造方法

（1）public BorderLayout() 创建一个默认的 BorderLayout 即组件间水平、垂直距离为 0。

（2）public BorderLayout(int hgap,int vgap) 创建行列间距为 hgap，列间距为 vgap 的 BorderLayout。

2. 常用方法

（1）public void setHgap(int hgap) 设置组件之间以及组件与容器见的水平间隔。

（2）public void setVgap(int vgap) 设置组件之间以及组件与容器见的垂直间隔。

3. 将组件加入容器中的方法

若容器的布局管理策略是 BorderLayout，则使用 public void add(Component comp,Object constraints) 方法向容器中添加组件，其中，参数 comp 为添加的组件，constraints 取值为：

- BorderLayout.EAST 或“East ”
- BorderLayout.WEST 或“West”
- BorderLayout.NORTH 或“North”
- BorderLayout.SOUTH 或“South”
- BorderLayout.CENTER 或“Center”（默认）

【例 8.7】创建窗体，并重新为窗体构造组件间隔为 5 的 BorderLayout 布局管理器。然后创建 5 个按钮，放到 5 个不同的区域中。

```
import java.awt.*;
import javax.swing.*;
public class BorderLayoutDemo extends JFrame{
    public BorderLayoutDemo(){
     super("BorderLayout布局管理器");
    BorderLayout layout=new BorderLayout(5,5);//设置组件像素的间距为5
     this.setLayout(layout);//设置窗体布局管理器
     //创建5个按钮对象
    JButton buttonEast=new JButton("东");
    JButton buttonWest=new JButton("西");
    JButton buttonSouth=new JButton("南");
    JButton buttonNorth=new JButton("北");
    JButton buttonCenter=    new JButton("中");
     //将5个按钮按照BorderLayout布局管理器部署到容器指定的位置
     this.add(buttonNorth,BorderLayout.NORTH);
     this.add(buttonSouth,BorderLayout.SOUTH);
     this.add(buttonEast,BorderLayout.EAST);
     this.add(buttonWest,BorderLayout.WEST);
     this.add(buttonCenter,BorderLayout.CENTER);
     this.setSize(300,200);
     this.setVisible(true);
   }
   public static void main(String args[]){
     BorderLayoutDemo application=new BorderLayoutDemo();
     application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }
}
```

上述程序运行结果如图 8.8 所示。

图 8.8 BoderLayout 布局管理器

8.4.3 GridLayout

GridLayout 是网格布局管理器，它以网格形式对容器的组件进行布置。容器被分成大小相等的网格，一个网格中仅放置一个组件，构造该布局管理器时指定网格的行数和列数，添加到 GridLayout 中的组件会依序从左到右、由上至下地填充每个网格。容器中各组件占据

的网格大小相同，所以 GridLayout 布局管理器强制组件根据容器的实际容量来调整它们的大小。

1.常用构造方法

（1） public GridLayout（int rows, int cols）建立一个行数为 rows，列数为 cols 的 GridLayout 布局管理器。

（2）public GridLayout(int rows, int cols, int hgap, int vgap) 创建一个行数为 rows，列数为 cols，且行间距为 hgap，列间距为 vgap 的 GridLayout 布局管理器。

2.常用方法

（1）public void setHgap(int hgap) 设置组件之间的水平间隔。

（2）public void setVgap(int vgap) 设置组件之间的垂直间隔。

（3）public void setRows(int rows) 设置布局中的行数。

（4）public void setColumns(int cols)设置布局中的列数。

3. 将组件加入容器中的方法

若容器的布局管理策略是 GridLayout，则使用 public Component add(Component comp) 方法将指定的组件 comp 添加到容器中。

在设置网格的行列数时需注意：当设置的行数和列数都设置为非零时，指定的列数被忽略；仅当将行数设置为零时，指定的列数才对布局有效。例如，对一个指定了 3 行 2 列的网格来说，如果在布局中添加 9 个组件，则它们将显示为 3 行 3 列。

【例 8.8】JFrame 组件中的 6 个按钮按网格的方式布局： 2 行 2 列。注意，当行数和列数都设置为非零时，指定的列数被忽略。

```
import java.awt.*;
import javax.swing.*;
public class GridLayoutDemo extends JFrame{
   private final String names[]={"one","two","three",
                                 "four","five","six"};
   public GridLayoutDemo(){
     super("GridLayout布局管理器");
     GridLayout grid1=new GridLayout(2,2,5,5);
     this.setLayout(grid1);
    JButton buttons[]=new JButton[names.length];
     for(int count=0;count<names.length;count++){
        buttons[count]=new JButton(names[count]);
        this.add(buttons[count]);
     }
     this.setSize(300,150);
     this.setVisible(true);
   }
   public static void main(String args[]){
     GridLayoutDemo application=new GridLayoutDemo();
     application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }
}
```

上述程序运行结果如图 8.9 所示。

图 8.9 GridLayout 布局管理器

8.4.4 CardLayout 布局管理器

CardLayout 是 JTabbedPane 容器默认的布局管理器。它在布局容器内的组件时，将容器中每个组件看作一张卡片，而容器充当卡片的堆栈。在某一时间，容器只能从这些组件中选择一个来显示，就像一副扑克牌每次只能显示最上面的一张一样，而且可以向前翻阅组件，也可以向后翻阅组件。

1.常用构造方法

（1）public CardLayout( ) 创建一个默认的 CardLayout，即水平、垂直间距均为 0。

（2）public CardLayout(int hgap,int vgap) 创建一个水平、垂直间距分别为 hgap 和 vgap 的 CardLayout。

2.常用方法

容器中组件的位置由组件添加到容器的顺序决定。CardLayout 定义了一组方法，这些方法允许应用程序按顺序地浏览这些卡片（组件），或者显示指定的卡片。

（1）public void first(Container parent) 显示容器中的第一个组件。

（2）public void next(Container parent) 显示容器中下一个组件。如果当前显示的组件是容器中的最后一个组件，调用此方法将显示第一个组件。

（3）public void previous(Container parent) 显示容器中前一个组件。如果当前显示的组件是第一个组件，调用此方法将显示最后一个组件。

（4）public void last(Container parent) 显示容器中最后一个组件。

（5）public void show(Container parent,String name) 显示指定名字的组件。

3.将组件加入容器中的方法

若容器的布局管理策略是 CardLayout，则使用 public Component add(String s1,Component comp)方法将组件添加到容器中。其中参数 s1 是指定的卡片名称，comp 是指定添加的组件。

【例 8.9】创建两个 JPanel 组件分别为 cardPanel 和 controlPanel，并添加到默认布局管理器的窗体的“Center”和“South”位置。cardPanel 面板的布局管理器设置为 CardLayout，并添加 4 个按钮；controlPanel 面板的布局管理设置为 FlowLayout，并添加两个按钮。程序运行时，通过单击“按钮”，可顺序地浏览这些卡片。

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class CardLayoutDemo extends JFrame{
    JPanel cardPanel=new JPanel();
    CardLayout cardLayout=new CardLayout();
   public CardLayoutDemo() {
       super("CardLayout布局管理器");
```

```
        cardPanel.setLayout(cardLayout);    //将CardLayout放入Panel中
        for(int i =1;i<=4;i++){  //向cardPanel中放入4个button按钮
            cardPanel.add("button"+i,new JButton("button"+i));
        }
        this.add(cardPanel,BorderLayout.CENTER);
        JButton nextButton=new JButton("下一张卡片");
        JButton prevButton=new JButton("前一张卡片");
        //为按钮注册监听器，响应按钮的操作
        nextButton.addActionListener(new ActionListener(){
         public void actionPerformed(ActionEvent e){//定义监听方法
                cardLayout.next(cardPanel);
                }
            });
        prevButton.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
                    cardLayout.previous(cardPanel);
                }
            });
        JPanel controlPanel=new JPanel();
        controlPanel.add(prevButton); //将prevButton放入面板中
        controlPanel.add(nextButton); //将nextButton放入面板中
        this.add(controlPanel,BorderLayout.SOUTH);
        }
        public static void main(String[] args) {
            CardLayoutDemo cardDemo=new CardLayoutDemo(); //创建窗体对象
            cardDemo.setSize(300, 200);
            cardDemo.setVisible(true);
            cardDemo.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
         }
    }
```

上述程序运行结果如图 8.10 所示。当单击“下一张卡片”或“前一张卡片”按钮时，cardPanel 面板将显示不同的按钮。

图 8.10 CardLayout 布局管理器示例

8.4.5 BoxLayout 布局管理器

BoxLayout 布局管理器是按照自上而下（垂直）或从左到右（水平）布置容器中所包含的组件，即使用 BoxLayout 布局的容器将组件排列在一行或排列在一列。在建立 BoxLayout 布局管理器时，需要指定添加到容器中组件是按照水平排列还是垂直排列。默认情况下，组件是按照垂直排列，即自上而下居中对齐。

javax.swing 包中的 Box 类是使用 BoxLayout 作为默认布局管理器的轻量级容器，并提供了静态方法来创建带有水平或垂直方向 BoxLayout 的 Box 对象。在实际应用中，一般情况下都是通过使用 Box 容器来使用 BoxLayout 布局管理器。

1. Box 类常用的静态方法

（1）创建 Box 对象的方法

- public static Box createHorizontalBox()创建一个从左到右布局组件的 Box。
- public static Box createVerticalBox()创建一个从上到下布局组件的 Box。

（2）创建决定组件之间间隔的方法

- public static Component createRigidArea(Dimension d)创建一个总是具有指定大小的不可见 RigidArea 组件。
- public static Component createHorizontalStrut(int width)创建一个固定宽度的不可见的 Strut 组件。
- public static Component createVerticalStrut(int height)创建一个固定高度的不可见的 Strut 组件。
- public static Component createHorizontalGlue() 创建一个横向 Glue 组件。
- public static Component createVerticalGlue() 创建一个纵向 Glue 组件。

（3）影响布局的不可见组件

- Glue 组件：是一种不可见的组件，用于占据其他大小固定的 GUI 组件之间的多余空间。在调整容器大小时，由 Glue 组件分割的 GUI 组件保持原有的尺寸，但 Glue 组件本身将被拉伸或收缩，以占据其他组件之间的多余空间。
- Strut 组件：是一种不可见组件，水平 Strut 具有固定像素的宽度，垂直 Strut 具有固定像素的高度。Strut 用于确保 GUI 组件之间保持固定的间隔。在调整容器的大小时，GUI 组件之间由 Strut 分开的距离保持不变。
- RigedArea 组件：是一种不可见的、具有固定像素高度和宽度的 GUI 组件。在调整容器的大小时，RigidArea 的大小保持不变。

2. 将组件加入 Box 容器中的方法

由于 Box 容器默认的布局管理器是 BoxLayout，所以在向 Box 容器添加组件的方法是 public Component add(Component comp)。

【例 8.10】设置 JFrame 的布局管理器为 FlowLayout，然后向其中添加一个横向 Box 组件和一个纵向 Box 组件，并分别给每一个 Box 添加 3 个 button。

```
import java.awt.*;
import javax.swing.*;
public class BoxLayoutDemo extends JFrame{
   public BoxLayoutDemo(){
     super("Demostrating BoxLayout");
     Box horizontalBox=Box.createHorizontalBox();//创建水平排列组件的Box
     Box verticalBox=Box.createVerticalBox();//创建横向排列组件的Box
     for(int count=1;count<=3;count++){
      horizontalBox.add(new JButton("Button"+count));//创建按钮并加入Box
     //创建宽高为20的RigidArea组件并添加到Box容器中
```

```
        horizontalBox.add(Box.createRigidArea(new Dimension(20,20)));
      }
      for(int count=4;count<=6;count++){
        verticalBox.add(Box.createVerticalStrut(10));
        verticalBox.add(new JButton("Button"+count));
      }
      this.setLayout(new FlowLayout(FlowLayout.LEFT));//设置容器布局管理
      this.add(horizontalBox);
      this.add(verticalBox);
      this.setSize(400,200);
      this.setVisible(true);
    }
    public static void main(String[] args){
     BoxLayoutDemo application=new BoxLayoutDemo();
     application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

上述程序运行结果如图 8.11 所示。

图 8.11 BoxLayout 布局管理器示例

8.4.6 GridBagLayout 布局管理器

GridBagLayout 类似于 GridLayout 的布局管理器，也是将容器分成若干行与列组成的网格单元，但行和列的大小可以不同，且每个组件可占用一个或多个单元网格（称为组件的显示区域）；组件加入容器的顺序可任意。

每个由 GridBagLayout 布局的组件都与一个 GridBagConstraints 对象相关联，该对象指定了如何将组件放置到 GridBagLayout 布局的容器中，又称为约束对象。

1. GridBagConstraints 常量属性

GridBagConstraints 类封装了一组 public 类型的约束变量，设置这组约束变量的值将对布局行为进行约束。常用的一组约束变量定义如下。

（1）gridx，gridy：放置组件的左上方网格单元的列号和行号。GridBagLayout 布局中的最左上角网格单元地址 gridx 为 0，gridy 为 0。gridx 和 gridy 的值也可设置为 GridBagConstraints 类的常量 RELATIVE（缺省值）和 REMAINDER。RELATIVE 定义此组件放在当前行的最后一个组件的右边（gridx 为 RELATIVE）或当前列的最后一个组件的下面（gridy 为 RELATIVE）。REMAINDER 定义此组件是当前行（或列）的最后一个组件。

（2）gridwidth，gridheight：指定组件占有的列网格单元数和行网格单元数。它们缺省

值为 1。也可设定为 GridBagConstraints 类中常量 REMAINDER 和 RELATIVE。

（3）fill：当组件的显示区域大于组件尺寸时，如何改变组件大小。其取值可为 GridBagConstraints 类的常量之一：NONE、HORIZONTAL、VERTICAL 和 BOTH。NONE（默认值）表示不改变组件的大小；HORIZONTAL 使组件的宽度在水平方向上变大以填充它的显示区域，但组件高度不变；VERTICAL 使组件在垂直方向上变大以填充它的显示区域，但组件宽度不变；BOTH 使组件同时在水平和垂直方向变大，以填满它的显示区域。

（4）weightx，weighty：给组件指定一个列权值（weightx）和行权值（weighty），以确定如何将多余空白空间在组件间分配。每一列组件的 weightx 值指定为该列组件的 weightx 的最大值，每一行组件的 weighty 值指定为该行组件的 weighty 最大值。此两变量取值为 0.0~1.0。数值越大表明组件所在的行或列将获得更多的空间。如果组件跨越多行或多列的网格单元，则在跨越方向上的权值应该为 0；另外，同一行或同一列上只需选择一个单元格设置其权值，其他单元格的权值均为 0。

（5）anchor：当组件没有填满显示区域时，确定如何将组件置于显示区域的何处。可赋予下列 GridBagConstraints 类的常量之一：NORTH、SOUTH、WEST、EAST、NORTHWEST、NORTHEAST、SOUTHWEST、SOUTHEAST 和 CENTER（默认值）。

2. 应用 GridBagLayout 布局组件的基本步骤

（1）创建 GridBagLayout 对象，并设置容器的布局策略为 GridBagLayout。

（2）使用 GridBagConstraints 类的构造方法创建一个约束对象。

（3）使用 GridBagConstraints 对象的常量属性设置组件的约束条件。

（4）调用 GridBagLayout 对象的 setConstraints(Component comp, GridBagConstraints constraints) 方法将组件与 GirdBagConstraints 对象相关联。

（5）调用容器的 add（Component comp）方法，将组件加入设置了 GridBagLayout 的容器中，完成该组件在该容器中的布局。

【例 8.11】 使用 GridBagLayout 布局 JFrame 容器中的 10 个按钮。

```
import java.awt.*;
import javax.swing.*;
public class GridBagLayoutDemo extends JFrame {
    public GridBagLayoutDemo() {
        GridBagLayout gridbag = new GridBagLayout();//创建网格包对象
        GridBagConstraints c = new GridBagConstraints();//创建约束条件对象
        this.setLayout(gridbag);//窗体设置布局管理器
        c.fill = GridBagConstraints.BOTH;  //组件充满显示区域
        c.weightx = 1.0;//设置列权值
        makebutton("Button1", gridbag, c);//调用成员方法
        makebutton("Button2", gridbag, c);
        makebutton("Button3", gridbag, c);
        c.gridwidth = GridBagConstraints.REMAINDER; //所在行最后一个组件
        makebutton("Button4", gridbag, c);
        c.weightx = 0.0;     //恢复为缺省值
        makebutton("Button5", gridbag, c); //另外一行,c.gridwidth同Button4
        c.gridwidth = GridBagConstraints.RELATIVE; //所在行最后一个组件右边
        makebutton("Button6", gridbag, c);
        c.gridwidth = GridBagConstraints.REMAINDER; //到行结束
        makebutton("Button7", gridbag, c);
```

```
        c.gridwidth = 1;        //恢复为缺省值
        c.gridheight = 2;
        c.weighty = 1.0;
        makebutton("Button8", gridbag, c);
        c.weighty = 0.0; //恢复为缺省值
        c.gridwidth = GridBagConstraints.REMAINDER; //到行结束
        c.gridheight = 1;                  //恢复为缺省值
        makebutton("Button9", gridbag, c);
        makebutton("Button10", gridbag, c);
    }
    //创建标签为name的button，并使用约束对象c将按钮加入到gridbag布局的容器中。
   private void makebutton(String name,GridBagLayout
        gridbag,GridBagConstraints c) {
        JButton button = new JButton(name);
        gridbag.setConstraints(button, c);//将约束条件附加给按钮组件
        add(button);//将按钮加入容器
        }
    public static void main(String args[]) {
        GridBagLayoutDemo window = new GridBagLayoutDemo();
        window.setTitle("GridBagLayout布局管理器");
        window.pack();
        window.setVisible(true);
        window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

上述程序运行结果如图 8.12 所示。

图 8.12 GridBagLayout 布局界面