Web 程序设计

顾春华　张雪芹　付　歌　编著

华东理工大学出版社

图书在版编目(CIP)数据

Web 程序设计/顾春华,张雪芹,付歌编著. —上海:华东理工大学出版社,2006.2
ISBN 7-5628-1848-7

Ⅰ.W... Ⅱ.①顾...②张...③付... Ⅲ.计算机网络—程序设计—高等学校—教材 Ⅳ.TP393.09

中国版本图书馆 CIP 数据核字(2006)第 001263 号

Web 程序设计

.....

编　　著 / 顾春华　张雪芹　付　歌
责任编辑 / 张　波
封面设计 / 王晓迪
责任校对 / 徐　群
出版发行 / 华东理工大学出版社
地　址:上海市梅陇路 130 号,200237
电　话:(021)64250306(营销部)
传　真:(021)64252707
网　址:www.hdlgpress.com.cn
印　　刷 / 江苏通州市印刷总厂有限公司
开　　本 / 787×1092　1/16
印　　张 / 20.75
字　　数 / 504 千字
版　　次 / 2006 年 2 月第 1 版
印　　次 / 2006 年 2 月第 1 次
印　　数 / 1—4050 册
书　　号 / ISBN 7-5628-1848-7/TP·144
定　　价 / 32.50 元

内 容 提 要

Internet 的普及使 Web 应用越来越广泛，Web 程序设计也日益成为计算机应用人员的重要开发技术之一。

本书以程序设计为主线索，介绍了各种 Web 程序设计技术，共包括程序设计基础、Internet 和 Web 技术基础、网站设计基础、Web 客户端程序设计、Web 服务器端程序设计、ASP 程序设计、PHP 程序设计、JSP 程序设计和 Web 数据库程序设计实例等 9 章，覆盖了静态 Web 页面设计和动态 Web 应用设计，涉及客户端脚本、CGI、ASP、PHP 和 JSP 等 Web 程序设计技术。全书以 ASP 为重点，以丰富的实例介绍了 Web 应用的设计方法。

本书适用于作为高等学校各专业计算机教学的教材，也可供从事软件开发与应用的工程技术人员用作工具参考书。

前　言

随着 Internet 的不断发展，基于 Internet 的应用也日趋成熟。World Wide Web(简称 Web)是建立在 Internet 基础上的一门主要应用技术。在过去的十多年中，Web 的应用从静态信息的发布发展到简单的动态应用，再发展到企业关键应用。目前，基于 Web 的计算机应用系统已经成为一种主要的应用类型，Web 程序设计也和其他程序设计一样，成为高等学校学生应该掌握的计算机程序设计工具之一。

将 Web 作为程序设计技术来推广到高校是一种尝试。就像 VB 的推出让很多大学生学会了编程但不用忍受“让人头痛”的代码一样，Web 程序设计为学生提供了又一种快乐编程的技术。现在很多高校都开设了 Java、C、VB 和 Web 等不同的程序设计课程，让不同专业的学生在学习程序设计时有所选择。

Web 程序设计和其他高级语言程序设计不同，它本身不是一种程序设计语言，而是一种应用架构。在一个简单的架构上，学生可以一步步地增加功能，逐步构造成一个应用程序，体验现代编程技术的特点。

Web 程序设计涉及的内容相当广泛，本书在介绍程序设计基础知识的基础上，以程序设计为主线，由浅入深地介绍了 Web 的基本概念、超文本标记语言 HTML、Web 客户端程序设计、Web 服务器端程序设计和 Web 数据库程序设计等知识。其中，Web 客户端程序设计部分讲述了如何用客户端脚本语言 JavaScript 和 VBScript 编制程序，以及浏览器脚本对象的功能和用法；Web 服务器端程序设计部分在介绍了服务器端程序的基本原理的基础上，从第 5 章到第 8 章分别对 CGI、ASP、PHP 和 JSP 等四种 Web 服务器端编程技术进行了介绍，并以 ASP 为重点介绍了服务器端对象的功能和使用方法；最后一章首先介绍了数据库访问的基本知识，然后用两个实例给出了用 ASP 编写 Web 应用程序的方法。

本书第 1 章、第 5 章和第 8 章由顾春华编写，第 2 章、第 3 章、第 6 章第 1,3,4 节和第 7 章由张雪芹编写，第 4 章、第 6 章的第 2 节和第 9 章由付歌编写。全书由顾春华统一修改定稿。上海市高等学校计算机等级考试的多位专家，特别是邵志清教授对本书的编写和出版给予了大力支持；葛旼阅读了全部书稿并提出了很多有益的建议。对此编者表示由衷的感谢。

由于 Web 程序设计覆盖面宽，发展又很迅速，编者水平有限，疏漏难免，诚恳希望读者不吝指正。

编　者

2005 年 12 月于上海

目　　录

第 1 章　程序设计基础

Internet 为我们的学习和生活带来很多便利，通过 Internet，可以和远在他乡的朋友聊天，也可以获取最新的科学研究资料。准备去一个地方旅游前，可以先在网上查寻该地区的天气情况，等等。那么，为什么 Internet 能给我们提供这些服务信息呢？为什么仅仅通过一个浏览器，我们就可以完成这么多工作呢？答案是程序，因为 Internet 上有不计其数的 Web 程序。本章将介绍程序的概念、程序设计方法以及 Web 程序设计等内容。

1.1　程序的基本概念

程序随计算机而来，自从有了计算机就有了程序。无论现代的计算机能处理多复杂的事务，无一例外都是通过执行程序来完成的，所谓“一切行动听程序指挥”。

1.1.1　什么是程序

在日常生活中，我们也经常会用到“程序”这个词，如“评审程序”、“工作程序”等，这里程序的意思是“做事情的工作步骤”，计算机中的程序也是指安排计算机工作的步骤。程序(Program)是指由人根据某一特定的需要而编写的控制计算机工作的有限命令序列，是对所要解决问题的各个对象和处理规则的描述。

从本质上讲，能正常工作的计算机是一台能高速执行程序的机器，它由处理器、存储器以及输入输出设备组成，其中处理器用于执行程序，程序和执行程序所需的数据保存在存储器中，输入设备用于将需要处理的外部世界的数据传递给计算机，而输出设备则将处理的结果信息传递给外部世界。没有程序的计算机，不能做任何事情，即便是最简单的操作也不会做。计算机解决问题的基本过程是这样的：首先把能解决问题的程序存入计算机，然后由计算机执行该程序，输入解决该问题必要的数据，计算机按照程序规定的步骤进行处理，最后输出结果。计算机网络出现后，多台计算机可以协同工作，来解决同一个问题，相应地，为了解决一个问题，需要多个程序一起工作才能达到目的。例如，我们要从 Internet 上下载一首歌曲，则需要如下步骤：

(1) 在一台 Internet 服务器上安装下载服务程序，如 FTP Server；在用户工作计算机上装上下载客户端程序，如浏览器；
(2) 启动下载服务程序；在工作计算机上启动浏览器；
(3) 输入关于所需下载的歌曲及地址信息；
(4) 下载服务程序找到该歌曲，经过网络发送到工作计算机；
(5) 用户工作计算机接收到该歌曲。

如果需要播放该歌曲,则需要启动另外一个专门的歌曲播放程序,按照相同的步骤来工作。通常,为了解决不同的问题,就需要编写不同的程序。程序种类繁多,有些通用(例如,文字处理程序),有些专用(例如,企业的生产过程控制程序);运行和开发这些程序的环境可以是 PC 机,也可以是网络。但不管怎样变化多端,任何应用总是离不开程序,离不开程序设计。

与程序相关的另外一个概念是软件,软件相当于"程序+文档",是程序上升为独立产品后的产物。例如,我们通常称购买来的杀毒程序为"杀毒软件",因为购买的除了杀毒程序外,还有相关的操作说明书、技术资料和病毒特征数据。

那么,计算机程序是怎么得到的呢?首先,要知道程序要做什么(What to do?),在此基础上考虑怎么做(How to do?),然后根据考虑结果开始编写程序(Do it.),最后是编译和运行程序。图 1-1 表示了计算机解决问题的过程。

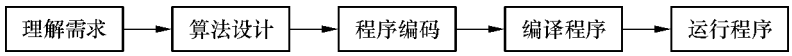

图 1-1　计算机解决问题的过程

下面,将简单介绍每个环节的概念和相关活动。

1.1.2　需求

在动手编写程序前,首先要了解需要解决的问题是什么,然后确定通过怎样的处理过程可以完成所需的任务。需要解决的问题就称为需求(Requirement),而这个处理过程就是算法(Algorithm)。

弄清需求是计算机解决问题的第一步,也是计算机正确解决问题的基础。需求就是所要解决的问题是什么,通常包括以下几个方面:

(1) 问题的范围;

(2) 问题的具体内容;

(3) 解决问题的限制条件;

(4) 解决问题的时间及成本要求;

(5) 问题解决的标准。

描述需求的形式没有固定的格式,关键在于对需求真正的理解,简单的问题甚至不用书面记录,复杂问题通常要编写专门的需求规格说明书来统一问题相关各方的理解。例如,我们想利用计算机程序来对某门课程成绩进行排序,这个需求很清楚,一句话就可以让人理解了;如果要用计算机程序来实现一个网上购物系统,则需要编写完整的需求说明书,才能明确这个网上购物软件究竟要完成什么。

1.1.3　算法

算法的概念由来已久,简而言之,算法就是对解决问题过程的具体描述。算法设计是程序设计的先导,算法的优劣直接影响程序的质量,因此,算法被称为"程序的灵魂"。为了方便说明,请先看两个例子。

[例 1]　需求是:已知两个正整数 m 和 n,求它们的最大公约数。

解:根据数学家欧几里得提供的方法,通过反复执行以下三步操作来求解。即:

第一步:求余。以较小的数(n)除较大数(m),求得余数 r ($0 \leqslant r < n$);

第二步:判断。若 $r=0$,n 即为所求数,否则执行第三步;

第三步:置换。令 $n \rightarrow m$, $r \rightarrow n$, 然后返回第一步。

这就是欧几里得算法,按上述步骤可以计算两个正整数的最大公约数。例如,两个正整数 $m=159$,$n=106$,则:

第一步:$m=159$,$n=106$,$r=53$;

第二步:$r \neq 0$,执行置换;

第三步:$m=106$,$n=53$,进入第二轮;

第一步:$m=106$,$n=53$, $r=0$;

第二步:$r=0$,最大公约数$=n=53$,计算结束。

推而广之,也可以利用这个算法计算三个正整数的最大公约数,方法是先计算出前两个数的最大公约数,再用同样的方法算出它和第三个数的最大公约数,即可获得所求结果。

[例 2]　需求是:垃圾邮件是 Internet 普及后的副产品,现要求设计一个方法来自动识别一个新的电子邮件是否是垃圾邮件。采用计算邮件标题出现概率的方法(贝叶斯算法)来解决这个问题,则可以通过执行以下步骤来实现。

第一步:收集大量的垃圾邮件和非垃圾邮件,建立垃圾邮件集合和非垃圾邮件集合;

第二步:提取邮件标题,例如“Play Game”,“飞机票”等,并统计该标题出现的次数。按照上述的方法分别处理垃圾邮件集和非垃圾邮件集中的所有邮件;

第三步:按第二步方法从第一步的两个邮件集合中分别产生一个标题频率表,table _ good 对应非垃圾邮件集,而 table _ bad 对应垃圾邮件集,表中存储邮件标题到出现频率的映射关系,如“飞机票”出现 78 次;

第四步:计算每个标题频率表中各标题出现的概率 $P=$(某标题的出现次数)/(对应标题频率表的长度);

第五步:综合考虑 table _ good 和 table _ bad,根据邮件标题来推断出新邮件为垃圾邮件的概率。数学表达式为:

设 t 代表标题串

则 $P(t)$表示邮件标题为 t 的该新邮件为垃圾邮件的概率。

设$Pg(t)=$(t 在 table _ good 中的值)

$Pb(t)=$(t 在 table _ bad 中的值)

则 $P(t)=Pg(t)/[(Pg(t)+Pb(t)]$;

第六步:建立新的垃圾邮件概率表 table _ probability,存储邮件标题 t 到垃圾邮件概率 $P(t)$的映射;

第七步:根据建立的垃圾邮件概率表 table _ probability,可以估计一封新到的邮件为垃圾邮件的可能性。当新到一封邮件时,查询 table _ probability,得到该邮件标题对应的该邮件为垃圾邮件的概率。当 $P(t)$超过预定阈值时,就可以判断邮件为垃圾邮件。

上述例子给出了识别垃圾邮件的步骤,当然,这只是其中一种方法,我们也可以设计其他算法来解决这个问题。自古以来,人们为了解决各种数学问题(包括以智力游戏形式出现的问题)创造过许多算法,闪耀着人类智慧的光辉。上述垃圾邮件识别算法就利用了著名的贝叶斯算法。

一个算法应该具有以下特性:

- **有效性**
  算法中的每一个操作都应该是计算机可以执行的,并能得到确定的结果。
- **确定性**
  算法中的每一个步骤必须有清楚的定义,而不应是含糊的、模棱两可的。
- **有穷性**
  一个算法必须在执行有穷步骤之后结束,不能无限执行下去。
- **输入**
  算法执行前或执行时可以有零个或多个输入量。
- **输出**
  算法执行完毕,至少要有一个输出。

1.1.4 编码和编译

算法代表了对问题的解,而程序则是算法在计算机上的特定的实现。计算机程序是需要人来编制的,将算法转变为由某种语言编写成程序的过程就是程序编码。程序编码也可以简称为编码(Coding),负责程序编码的人员称为程序员,由他选择某种程序设计语言来根据算法来编写程序。

程序编码时,程序员不仅需要一点不差地实现算法所表示的逻辑,而且必须严格按照程序设计语言的语法来编写每一句语句,即所编写的每个语句命令的拼写都必须正确,所有的标点符号都要完全对号入座。

例如,对于前面提到的计算两个数的公约数的算法,如果我们采用 Visual Basic 语言来编码,则可以得到如下程序:

```
Dim m As Integer, n As Integer
m=InputBox("m=")
n=InputBox("n=")
If m < n Then
        a = m: m = n: n = a
End If
Do
        r = m Mod n
        If r <> 0 Then
           m = n: n= r
        Else
           Exit Do
        End If
Loop
MsgBox" g.c.d.=" + Str(n)
End
```

经验丰富的程序员有时可以将算法设计和程序编码结合起来,一步就完成这两项任务,但这不是一个好习惯,而且只适用于设计和编写一些非常简单的程序,就像我们可以一步就设计并写好一个通知一样。然而,对于大部分复杂的程序就不可能这样做了,就像我们不可能一步就写好一部小说一样。

由于计算机只懂一种只包含 0 和 1 组成的机器语言写成的程序,因此,程序员编好的程序一般不能直接运行,只有经过一个翻译器将程序员用的高级语言程序转化为机器(低级)语言程序,计算机才能理解程序并且运行程序。有两种不同翻译过程:编译和解释。图 1-2 表示了这两种翻译过程:

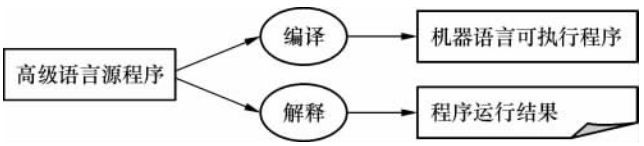

图 1-2　编译和解释

执行经过编译后得到的机器语言可执行程序,就可以得到运行结果,而解释则是翻译并执行每一句高级语言程序中的语句,直接得到运行结果。

1.1.5　程序设计语言

前面提到的机器语言和高级语言都是程序设计语言,程序设计语言种类繁多,总数已不下千种。从程序的运行方式上区分,可以把它们分为解释程序设计语言和编译程序设计语言;根据程序设计方法的不同,可以分为结构化程序设计语言和面向对象程序设计语言。

解释执行的程序设计语言因为解释器不需要直接同机器语言程序打交道,所以实现起来较为简单,而且便于在不同的平台上面移植,用于设计动态 Web 页面的脚本语言都属于这类,如 VBScript、JavaScript、PHP、Perl 等,还有一些常见的高级语言如 Java、Visual Basic、Visual Foxpro、Power Builder 等都属于这类语言。编译执行的程序设计语言因为要直接同计算机的 CPU 的指令集打交道,不同的硬件平台有不同的指令集,因而具有很强的指令依赖性和系统依赖性,但编译后的程序执行效率要比解释语言高得多,像 Visual C/C++、Delphi 等都是编译语言。

常见的结构化程序设计语言有 Pascal、C 等,而 C++和 Java 是典型的面向对象的程序设计语言。有关结构化程序和面向对象程序设计的内容,在下一节作详细介绍。

1.2　程序设计方法

一个程序就像一部电影,编得好与不好相差很大,一部好电影可以让人看了还想看,而一部差的电影很少会被人看完。好的程序不但能解决问题,而且还有容易被看懂、运行效率高、容易修改等特点。那么,如何才能编写一个好的程序?这是程序员在进行程序编码之前必须考虑的问题,就是需要选取合适的程序设计方法。

1.2.1　两次飞跃

随着计算机硬件技术的不断发展,程序设计方法也随之不断发展。最早的程序设计方

法称为个体化方法,每个程序员都按照自己的习惯和风格来编写程序,其结果是程序很难维护,特别是规模较大的程序。个体化方法导致程序的寿命缩短,成本提高,不能满足需要。经过计算机工作者的努力,程序设计方法持续改进,在过去的几十年中,程序设计方法主要经历了两次革命性的飞跃:结构化方法和面向对象方法,相对应地产生了面向过程程序设计和面向对象程序设计。

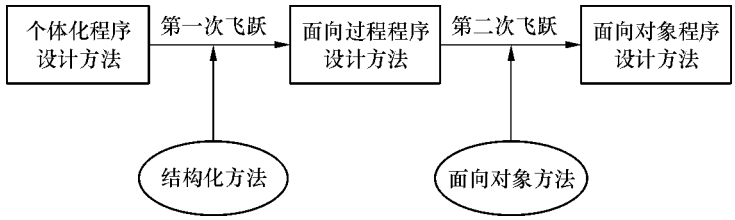

图 1-3　程序设计的两次飞跃

下面介绍结构化方法(Structured Method)和面向对象方法(Object-Oriented Method)。

1. 结构化方法

曾被称誉为“程序设计方法的革命”的结构化程序设计,使程序设计从主要依赖于程序员个人的自由活动变成为有章可循的一门科学。它的主要贡献,是推动了程序设计风格从“追求技巧与效率”到“清晰第一、效率第二”的转变,从而提高了程序的易读性和可靠性。

1976 年,瑞士的沃斯(Nikiaus Wirth)教授在其经典名著《算法+数据结构=程序》的序言中指出:“程序就是在数据的某些特定表示形式和结构的基础上,对于抽象算法的具体描述。”、“不了解施加于数据上的算法,就无法决定如何构成数据;反之,算法的结构和选择,却常常在很大程度上依赖于作为基础的数据结构。简而言之,程序的构成与数据结构是两个不可分割地联系在一起的问题。”

沃斯的论述,精辟地概括了算法与数据结构在当时程序设计中的地位与相互关系。“算法+数据结构=程序”,影响和指导了一代又一代的程序员。从早期的程序设计语言 Fortran、Cobol、ADA、Pascal 到现代的 C 语言,全是面向过程的,都有一个特点:当对软件进行分析或设计时,开发人员总是遵循“程序=数据结构+算法”的思路,把程序理解为由一组被动的数据和一组能动的过程所构成,这样的程序设计方法就称为面向过程的程序设计方法。

结构化程序设计的普及促进了软件生产的工业化,也缓解了当时的软件危机,然而它的面向过程的程序设计思路,不适合日益复杂应用系统。实践表明,用结构化技术处理 5000 行以下代码的软件的确是十分有效的,但面对当今的大规模软件产品的复杂性,却仍旧无能为力。可是在客观事物中,实体的内部“状态”(可用数据表示)和“运动”(加于数据的操作)却是结合在一起的,这就使采用传统范型开发的软件模型(称为“解空间”,Solution domain)被人为地偏离客观实体本身的模型(称为“问题空间”,Problem domain)。于是,面向对象的方法就应运而生了。

2. 面向对象方法

在面向对象的程序设计中,数据及其操作被封装在一个个称为“对象”(Object)的统一体中,对象之间则通过“消息”(Message)相互联系,“对象+消息”的机制取代了“数据结构+算法”的思路,因而较好地实现了“解空间”与“问题空间”的一致性,为解决软件危机带来了新的希

望。从结构化程序设计到面向对象的程序设计,是程序设计方法的又一次飞跃,在软件开发和维护中正日益显露其优越性。

面向对象的思想最初起源于 20 世纪 60 年代中期的仿真程序设计语言 Simula 67。20 世纪 80 年代初出现的 Smalltalk 语言和 90 年代推出的 C++、Java 语言及其程序设计环境,先后成为面向对象技术发展的重要里程碑。从 80 年代末开始,面向对象的程序设计和面向对象的需求分析都得到快速发展,特别是 90 年代中期,由 Booch, Rumbaugh 和 Jacoson 共同提出了统一建模语言(Unified Modeling Language, UML),把众多面向对象方法综合成一种标准,使面向对象的方法成为主流的程序设计方法。

面向对象思想的最重要特征,是在解题空间中引入了“对象”的概念,使之逼真地模拟问题空间中的客观实体,从而达到与人类的思维习惯相一致。面向对象方法包容了以下核心概念。

(1) 对象(Object)。对象是现实世界中个体或事物的抽象表示,是它的属性和相关操作的统一封装体。属性表示对象的性质,属性值规定了对象所有可能的状态。对象的操作是指该对象可以展现的外部服务。例如,若将卡车视为对象,则它具有位置、速度、颜色、容量等属性,对于该对象可施行启动、停车、加速、维修等操作,这些操作将或多或少地改变卡车的属性值(状态)。

(2) 类(Class)。类用于表示某些对象的共同特征(属性和操作),对象是类的实例。例如,汽车类可包含位置、速度、颜色等属性,以及启动、停车、加速等操作。卡车是汽车类的一个实例。

(3) 继承(Inheritance)。类之间可以存在继承关系。它是现实世界中遗传关系的直接模拟,可用来表示类之间的内在联系以及对属性和操作的共享。子类可以沿用父类(被继承类)的某些特征,同时子类也可以具有自己独有的属性和操作;对于一个类的修改能自动反映到它的所有子类中。例如,飞行器、汽车和轮船都是交通工具类的子类,它们都可以继承交通工具类的某些属性和操作。

除继承关系外,现实世界中还大量存在着“部分-整体”关系。例如,飞机可由发动机、机身、机械控制系统、电子控制系统等构成。这种关系在面向对象方法学中可表示为类之间的聚集(Aggregation)关系。在聚集关系下,“部分”类的对象是“整体”类对象的一个组成部分。

(4) 消息(Message)。消息传递是对象与其外部世界相互关联的唯一途径。对象可以向其他对象发送消息以请求服务,也可以响应其他对象传来的消息,完成自身固有的某些操作,从而服务于其他对象。例如,直升飞机可以响应轮船的海难急救信号,起飞,加速,飞赴出事地点并实施救援作业。

对象的操作主要是用来响应外来消息并为其他对象提供服务的,所以它们也被称作“外部服务”。Coad 和 Yourdon 认为,采用上述 4 种概念进行开发的软件系统可以认为是面向对象的。为此,他们把面向对象方法归结为一个简单的公式,即

面向对象=对象+分类(Classification)+继承+消息通信(Communication with messages)

1.2.2　结构化程序设计

结构化程序设计(Structured Programming, SP)是由荷兰学者戴克斯特拉(E.W. Dijkstra)