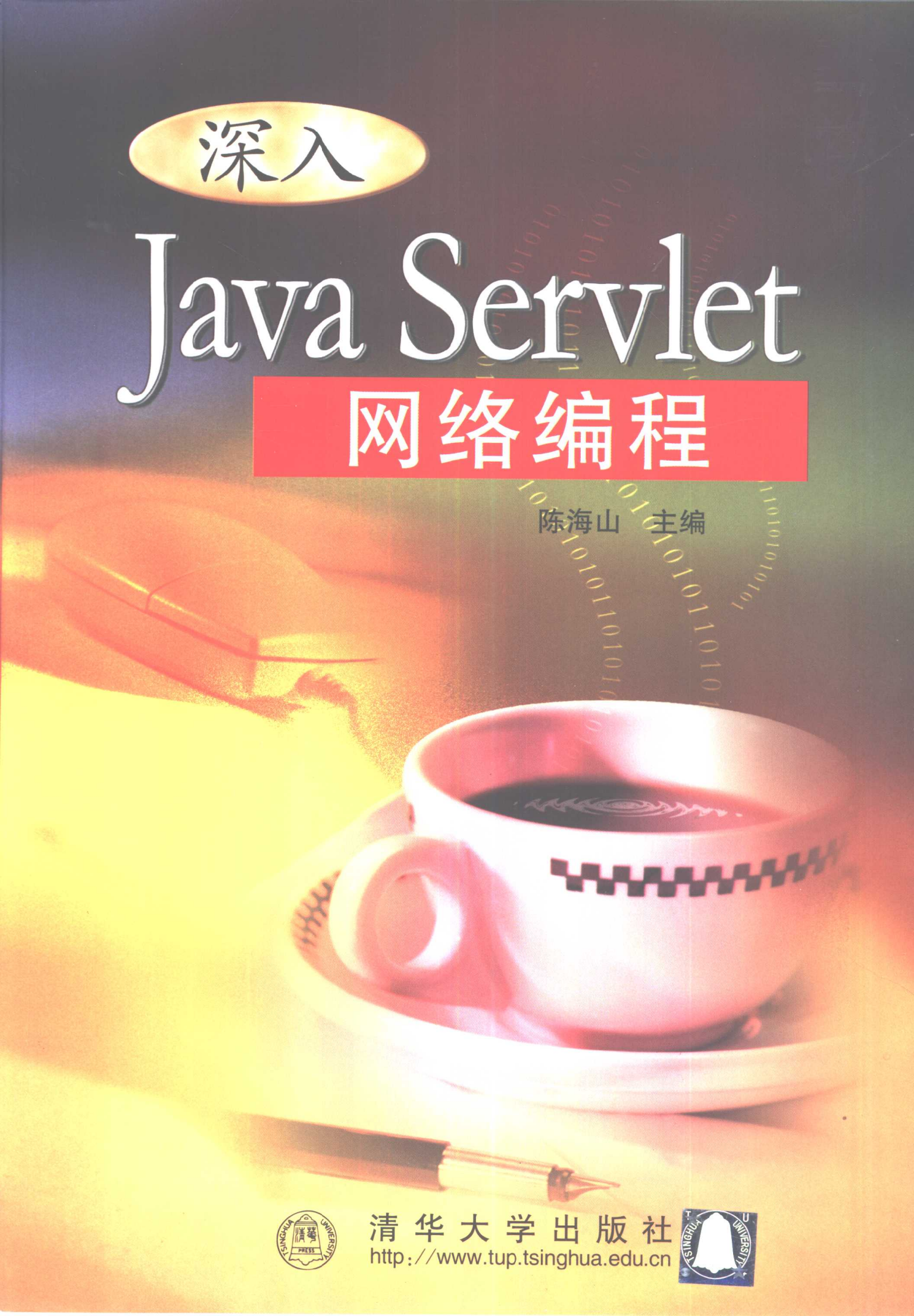
深入 Java Servlet 网络编程

陈海山 主编



清华大学出版社
http://www.tup.tsinghua.edu.cn

深入 Java Servlet 网络编程

陈海山　主编

清华大学出版社

(京)新登字 158 号

内 容 简 介

Java Servlet 是用 Java 语言进行 Web 服务器编程的强大利器，利用 Java Servlet 可以方便、高效地构建各种基于 B/S 结构的解决方案。本书详细介绍了在开发 B/S 结构的系统时，使用 Java Servlet 技巧解决常见问题的编程方法，并从协议细节的角度进行详细的讲述。全书由 16 章和 2 个附录构成，理论结合实例，分门别类地讲述了 Servlet 编程方法、会话管理、服务器端图形生成、网络编程、上传和下载文件、使用数据库、在数据库中存取图像、Servlet 链、Servlet 服务器端包含、Applet 和 Servlet 通信、在 Servlet 中发送和接收邮件、在 Servlet 中利用 XML、在 Servlet 中利用 EJB 实现 Internet 搜索引擎等内容。

本书内容丰富、结构合理、重点突出、实例讲解，易学易会，既可供 Java 应用编程人员学习借鉴，也可以作为高等院校师生的教学参考用书。

书　　名：深入 Java Servlet 网络编程
作　　者：陈海山 主编
出 版 者：清华大学出版社(北京清华大学学研大厦 A 座，邮编 100084)
责任编辑：宋　韬
印 刷 者：北京市清华园胶印厂
发 行 者：新华书店总店北京科技发行所
开　　本：787×1092　1/16　印张：20.25　字数：483 千字
版　　次：2002 年 2 月第 1 版　2002 年 2 月第 1 次印刷
书　　号：ISBN 7-302-05223-9/TP·3069
印　　数：0001-5000 册
定　　价：30.00 元

前　言

如今，越来越多的公司和企业应用 Java Servlet 技术来构建高效的电子商务平台及各种中间交易系统，创建高水平的大型网站。Java Servlet 是 Java 企业应用编程的一部分。它基于强大的 Java 语言，完全支持面向对象的编程思想，具有良好的可移植性和可伸缩性，与 Java Enteprise API 紧密集成在一起，在开发使用 HTTP 协议通信的企业级应用及电子商务方面具有得天独厚的优势。

Java Servlet 与其他的服务器端技术相比，最大的特点有两点：

首先，作为 Java 语言的一个扩展组成部分，Java Servlet 编程是一种先进的面向对象程序设计。各种流行的服务器端技术（例如，ASP、PHP、CGI 等）在这一点上无法与之相媲美。

其次，Java Servlet 最强大的地方是与 HTTP 协议的紧密结合。在这一方面，Java Servlet 是做得最好的。从面向对象的角度来讲，Servlet API 中的底层类库的设计是与协议无关的，而较高一层的类库则和 HTTP 协议紧密结合。可以这么说，Servlet API 几乎可以控制 HTTP 协议的所有方面。因为 Servlet 和 HTTP 结合得如此紧密，所以在本书附录 A 专门介绍 HTTP 的内容。如果 Servlet 和属于同一家族的 Applet 一起使用，或者和 DHTML 技术以及 XML 技术相结合时，可以在 B/S 结构的系统上实现与传统的 C/S 系统一样强大的界面功能。

本书讲述 Servlet 的底层网络编程知识，以及如何利用各种常用的 Internet 协议来扩展应用程序功能。特别详细地介绍了 Servlet 如何与 HTTP 协议紧密结合，来开发功能强大的 B/S 应用系统。当然，单纯使用 Servlet API 在一个完整的系统中是远远不够的，本书还涉及了如 JDBC、JavaMail、RMI、XML、EJB 等与之相关的其他 Java 技术。

本书的内容安排以各种解决方案为主，除第 1 章外，本书的每一章都包括一个完整的实例，非常易于读者学习掌握。

本书适用于各种层次的 Servlet 程序员。在阅读本书之前，最好熟悉面向对象的程序设计及 Java 编程语言。

另外，欧阳劲、高小龙、董保东、刘杰、贺涛、景小荣、袁连海、陈曙东、安建伟、肖晖、王学吉、杨吉利、王福生、张忠军、乔善磊、于大鹏、吴林朋、吕化为、王小健、张晓辉、张大全、胡珂、丁乐春、陈海蓉、王进新、孙金轩、郑新、高忠民、付莉、何谓、刘洁、刘昕和樊志立等同志也参加了本书的编写、排版和资料的收集工作，在此对他们表示感谢。

延伸服务：如果读者愿意参加“深入 Java Servlet 网络编程”的学习培训，或是在学习过程中发现问题，或有更好的建议，欢迎致电。联系电话：(028)5404228；网址：E - mail：bojiakeji@163.net；通讯地址：成都四川大学（西区）建筑学院成都博嘉科技资讯有限公司；邮编：610065。

作　者

目　录

第1章　分布式Web应用程序

本章要点：

本章主要介绍分布式Web应用程序和n－层应用相关概念，并且详细讲述了Java Servlets与各种服务器端应用技术的比较，及Java Servlet在n－层应用中的地位和作用。

本章主要内容：

- ☐ 分布式Web应用程序
- ☐ 企业级n－层应用
- ☐ 典型Web应用程序的处理过程
- ☐ 各种服务器端技术的比较
- ☐ 基于Java解决方案的特点

最近几年,计算机领域特别是计算机网络取得了令人难以置信的飞速发展。“网络就是计算机”不再仅仅是一个口号,现在正在逐步接近现实。随着 Internet 网的迅速普及,网络已经延伸到世界的各个角落。现在人们在世界各地都可以共享信息,进行电子商务交易,甚至进入公司的内部网进行办公,因此 Internet 网是如此重要。很多企业内部也开始逐渐利用 B/S(浏览器/服务器)结构的应用程序,来取代传统的 C/S(客户机/服务器)结构的应用程序。这就促进了基于分布式 Web 应用程序的发展。

1.1 分布式 Web 应用程序

一个分布式 Web 应用程序的各个部分分布在网络的不同计算机上,按照基本的设计模式可将应用程序分成“模型—视图—控制器”(Model-View-Controller,MVC);按照面向对象的设计规则:不要在一个对象中封装过多的功能,而要按照应用程序的逻辑结构将对象的功能划分开。其目的是保证一个对象尽可能地可重复利用,以及发挥应用程序的可伸缩性能。

- 模型(Model):是应用程序的数据模型,一般用来保存应用程序的数据内容。
- 视图(View):是用户界面,用来显示内容。
- 控制器(Controller):用来控制模型和视图两层之间的通信。

另外,大多数针对企业级解决方案的应用程序还需要一个稳定、持久能满足容量要求和查询要求的数据存储,该存储可以是一个文件系统或数据库。对于一个分布式 Web 应用程序,将用户界面的定义存储在服务器上,在各个不同的客户机通过 HTTP 协议发出请求时,再下载到客户机上由客户机的浏览器解释执行,共享运行在服务器上的服务程序。数据模型可以用 XML 来定义,由服务器端脚本或客户端脚本来控制与用户界面的通信。数据库和文件系统可分布在不同的服务器上。

分布式 Web 应用程序基于一些开放的以及被广泛接受的标准。它运行在一个基于 TCP/IP 协议的网络上,使用 Web 浏览器作为其用户界面,使用 HTTP 协议与服务器通信。这意味着服务器与客户端传递数据时,使用 HTML 或 XML 与其他的标准的 MIME 类型。

1.2 企业级 n-层应用

传统的 C/S(客户机/服务器)结构大多属于 2-层结构,整个应用程序都分布在客户端计算机上(即胖客户),数据库和文件系统则放在服务器上。这种结构虽然在一定程度上能够满足应用的需要,但存在着很多缺点。

在 2-层结构的应用中,应用程序的大部分运行在客户机上,而服务器只存放数据库和文件系统。一些公用的商务逻辑部分要分别在每个客户机上运行,每个客户机都要求较高的资源,且浪费比较严重。而且,与数据库打交道的代码在客户机上运行,势必造成网络流量的增加。如果多个客户机同时进行数据库操作,容易造成网络阻塞。另外,2-

层结构的应用程序维护特别困难。每一次应用程序的升级都需要更新所有的客户端，在一些大型的企业应用中，由此所造成的成本是惊人的。

为了解决上述问题，人们提出了 3－层应用程序的解决方案。即将应用程序划分为 3 层，每一层都通过一些定义好的接口与其他各层通信。这 3 层一般来讲在物理上和逻辑上都是可以相互分离的。第 1 层是表示层，主要向用户展现图形界面(UI)；第 2 层是封装应用程序的具体商务逻辑，负责接收表示层的数据，按照业务规则进行一定的处理后传送到数据层，再将结果返回给客户端；第 3 层是数据层，主要是存储数据，可以是数据库、文件系统或是目录服务。

3－层应用程序的出现早于 Web，但 Web 应用程序极大地丰富了 3 层应用程序的内容。Web 应用程序基于一些被广泛接受的运行于 HTTP 协议之上的标准。Web 应用程序是基于浏览器的，因此应用程序可以做成瘦客户端，应用程序的所有逻辑都可以放在服务器上。应用程序升级是只需在服务器端一次更新，这就解决了 2－层结构中的升级和维护问题。而且，因为只需在第 2 层和第 3 层之间传递针对数据库的操作，大大减小了网络流量。

在一些企业应用中，客户分成不同的群体，有的进行数据库操作，有的进行订单查询，还有的可能是发送电子邮件。数据层的情况也比较复杂，有可能包括数据库、XML 文档、目录服务等。相应的中间层也进行了划分，单一的中间层划分成多个对象或多个组件，由统一的接口进行控制。它可以在物理上和逻辑上进行多层分布，这形成了 n－层结构。在一个面向对象的 n－层结构中，最主要的是对象之间通过接口而不是通过对象本身进行通信。方法调用实现与对象所处的位置无关，并通过统一划分事务边界来实现企业级的事务处理。图 1－1 是一种基于组件的 n－层 Web 应用程序的结构示意图。

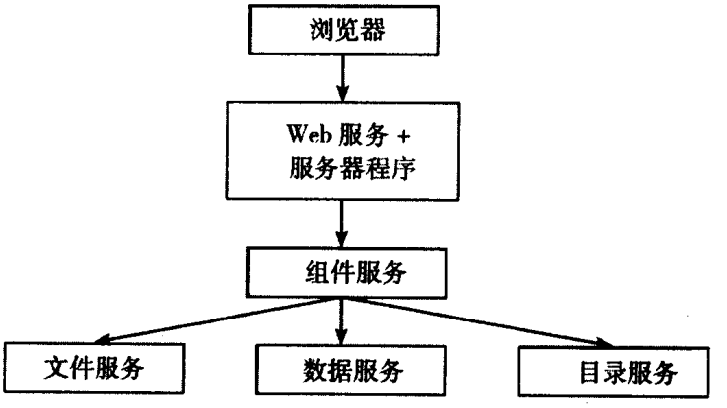


图 1－1　一种基于组件的 n－层 Web 应用程序结构示意图

注意：每层之间是通过什么协议来通信的。在实际应用中，也可能不基于组件，而是自己来编写远程调用对象。每一层也可能根据不同情况，再做划分。

1.3 典型 Web 应用程序的处理过程

一个 Web 应用程序的处理程序是:前端(表示层),一般是浏览器,从用户那里收集数据后,发送请求,将数据通过 HTTP 协议的 Get 或 Post 方法发送到服务器端。服务器端(第 2 层和第 3 层)用相应的方法处理数据,将结果返回给客户端。近年来,随着 XML 技术的不断成熟,浏览器和服务器端交换数据可定义成标准的 XML 格式。

1.3.1 Web 浏览器发送请求

Web 浏览器是一种应用程序,其基本功能是把客户通过图形界面的请求转换为标准的 HTTP 请求,并把 HTTP 响应转换为客户能够接受的图形界面。在典型的 Web 应用程序中,一般是通过浏览器收集数据。通过客户端脚本验证后,转化成标准的 XML 格式(可选)发送到服务器端。发送数据一般有 Get 和 Post 两种方法。

Get 方法是所有 Web 请求的默认方法。当用户向 Web 页面发出请求时,浏览器就会执行一个 Get 请求。因为 Get 请求把所有表单数据打包,附加在请求的 URL 后面,浏览器会把它当成一个 URL 放在 HTTP 的请求报头中。因此,可以从浏览器的缓存中查到请求串的内容。另外,将要传递的数据附加在 URL 之后,传递的数据量会受到一定的限制。

Post 方法同样把表单数据打包成请求实体的一部分,但不是附加在请求的 URL 后面而是单独存放。服务端程序能够用"关键字/值"的方式进行解析。Post 方法没有长度的限制,而且可以传送多种类型的数据。例如,图像、文件等。

1.3.2 执行服务器端程序

Web 服务器的一个重要功能就是向特定的脚本、程序传递需要处理的请求。Web 服务器首先检查所请求文件的扩展名(或文件所在目录),以此来决定需要装入什么样的运行环境。例如,以.jsp 为扩展名的文件按 JavaServer Page 进行处理,在 cgi-bin 目录下的文件作为 CGI 脚本进行处理。

一旦 Web 服务器确定了所请求文件的类型,它就装入执行该文件在所需要运行时的(Runtime)环境。例如,对于 JSP 文件将装入相应的 Java 解释环境;对于用 Perl 写的 CGI 程序,Web 服务器会创建一个新进程并装入 Perl 解释器。

创建了执行该文件所需的环境后,执行文件。处理浏览器用 Get 方法或 Post 方法传递数据,再将运算的结果返回给浏览器,运算的结果以 HTML、XML 或二进制数据表示,浏览器进行相应的处理。

1.3.3　将结果返回给浏览器

一般情况下，将结果返回给浏览器时，要由服务器端应用程序指定响应的内容类型(Content-Type)、内容长度(Content-Length)，然后把响应内容写入到输出流中。浏览器收到响应后，首先查看响应头的内容类型，确定输入流中响应实体的 MIME 类型，再来确定如何处理数据。返回的内容可以是 HTML、文本、XML、图像或音频视频流等。

1.4　服务器端技术

服务器端程序和 Web 服务器是紧密结合在一起的，一般的服务器端程序具有下面几个特点：

- 程序必须能被 Web 服务器激活。当收到从浏览器发出的请求时，Web 服务器应该能够定位并装入相应的运行环境执行该程序。
- 能够有一种机制把浏览的表单数据传送给该程序。
- 程序被激活后，要有一个标准的入口点来开始执行，并进行一些初始化操作。
- 程序可以把输出写入到 HTTP 的响应中，传回给浏览器。

现在比较流行的服务器端技术有：CGI、ISAPI、ASP、PHP、Java Servlet 和 JSP，下面将对这几种流行的技术进行详细的说明和比较。

1.4.1　各种服务器端技术的比较

1. CGI(通用网关接口)

作为一种最古老的服务器端 Web 技术，CGI 被大多数 Web 服务器所支持，CGI 程序可以用大多数高级语言来编写，包括：C/C++、Perl、VB 和 Java。

CGI 编程的最大缺点是效率不高。每当 Web 服务器接收到一个关联到 CGI 程序的请求时，就会创建一个完整的新进程。每个这样的新进程，包括它自己的环境变量集、运行时环境要求程序的一个实例和程序的一个拷贝及运行程序所需的内存。当服务器接收到大量请求时，会很快耗尽服务器的资源。

2. ISAPI

ISAPI 是微软公司为克服 CGI 的低效性，而开发的运行在 Windows 和 IIS 上的一组 API，使开发者能够把服务器应用写成共享库。这些共享库被装入一个进程，并能处理多个请求而不需要重新创建进程。它们可以在 Web 服务器启动时装入，也可以在需要时装入。一旦它们在规定的时间内没被使用，Web 服务器就把它们从内存中卸载。

ISAPI 的缺点主要是一般用 VC 开发，编程复杂，开发效率低。另外一方面，由于 ISAPI 和 Web 服务器运行在同一进程内，如果一个 ISAPI 程序出现了致命的错误，有可

能造成整个 Web 服务器的瘫痪。

3. ASP(Active Server Pages)

ASP 可能是现在应用最广泛的一种服务器端技术。ASP 的优点是编程简单,支持多种脚本语言,并可与 COM 组件结合,功能几乎可以无限扩充。因为 ASP 的输出是标准的 HTML 语言,所以 ASP 的客户端具有平台无关性。

但 ASP 的安全性一直受到人们的怀疑,而且只能工作在微软公司的操作系统下,所以在大型的电子商务应用上受到了限制。另外,ASP 的代码和 HTML 及 JScript 混合在一起,给应用程序的维护带来了很大的麻烦。

4. PHP

PHP 是基于标准 C 语法,可以内嵌于 HTML 的一种脚本语言。近年来发展速度比较快,理论上讲 PHP 通过插件可以运行在多种平台上,包括微软公司的 IIS。但是,在实际应用上,PHP 在 Linux 的 Apache 下应用比较多。

PHP 的缺点是对各种数据库的接口支持不一致,缺乏层次支持,没有形成一套完整的企业级的解决方案,不适于开发大型的电子商务应用。

5. Java Servlet 和 JSP

Java Servlet 是 Sun 公司提出的针对企业级应用的众多 Java API 的关键部分,极大地扩展了 Web 服务器的功能。当用户向一个 Servlet 发起请求时,服务器只是简单地使用一个新线程来处理请求。这种方法能提高执行效率,因为它不会像 CGI 程序那样产生多个进程,线程需要更少的资源。设计良好的 Web 服务器,能够使用线程池来控制用于客户请求的线程数量,来维护负载平衡。使用 Servlet 的另外一个好处是它"继承"了 Java 语言的特性,能够跨平台操作;能够使用完全面象对象的设计思想来进行程序设计;能够利用 Java 语言的"强制性异常处理"和"类型安全性"这类先进的安全特性,防止应用程序的错误转变成灾难。通过在服务器端利用一些与特定平台相关的 Java 语言加速器,能够解决 Java 语言的运行速度问题。

JSP(Java Server Pages)类似于微软公司的 ASP(Active Server Pages)。在 HTML 页中内嵌入 Java 代码,并可利用 JavaBean 组件。当用户请求 JSP 文件时,Web 服务器首先编译产生相应的 Servlet 类。然后,Web 服务器激活 Servlet 并向浏览器返回结果。

1.4.2 基于 Java 解决方案的特点

任何一种技术都不可能十全十美,Java Servlet 不能利用 COM 组件,ASP 也不能利用 JavaBean 和 EJB。但服务器端的 Java 技术是现在最先进和最完善的技术之一,主要具有以下几项优点:

1. 平台无关性

Java Servlet 同 Java 语言一样具有平台无关性。Servlet 代码被编译成字节码,再由服

务器上的与平台相关的 Java 虚拟机(JVM)解释执行。由于 Servlet 是以与平台无关的字节代码组成,所以可以被移植到支持 Java 的其他平台上。

2. 效率

当 Java Servlet 接收请求时,它在相同的进程中创建另一个线程来处理请求,使得成百上千的用户能够同时访问 Servlet 而不影响服务器的性能。另外,Servlet 在第一次装入内存后,以后的请求可在内存中直接执行。这样,大大加快了速度。

3. 访问 Enterprise Java API

Java Servlet 是 Java 的整体解决方案的一部分,它能够访问所有的 Java API,利用强大的 Java 语言提供所有功能。例如,它可以利用 Java Mail API 收发邮件,可以利用 Enterprise Java Bean 扩充功能,可以使用 RMI 实现远程方法调用等。

4. 重用性

Java 语言是完全支持面向对象程序设计的高级语言,可以利用面向对象程序设计思想中所提供的所有重用机制。它可以通过将特定功能封装在对象中,及将特定的对象封装在一起形成组件来支持重用。

第 2 章　Java Servlet 简介

本章要点：

本章主要介绍 Java Servlet 的基础应用，包括 Servlet 的生命周期；编写 Servlet 程序应遵守的规则；Java Servlet API 中常用类和方法的使用方法；Servlet 中获得表单数据的方法；Servlet 中处理错误和日志记录；Servlet 在 n－层 Web 应用程序中的具体作用。

本章主要内容：

- ☐ Servlet 生命周期
- ☐ Java Servlet API 简介
- ☐ 处理表单和返回数据的实例
- ☐ 错误处理和日志记录
- ☐ 服务器端日志记录
- ☐ Servlet 在 n－层结构中的作用

Java Servlet 为应用 Web 的企业使用 Java 技术提供了一个核心的方法。Java Servlet 组件由应用程序开发人员构建,通过 HTTP 和 Web 处理特定的商业请求并生成响应。用简单的 Java Servlet API 集合也能够提供丰富的 HTTP 行为。Java Servlet 有时也被称为服务器端小程序(Applet),Java Servlet 确实和 Applet 有相似之处:

- 它们不是独立的应用程序,没有 main()方法。
- 它们不是由用户或程序员调用,而是由另外一个应用程序(容器)调用。
- 它们都有一个生存周期,包含 init()和 destroy()方法。

Java Servlet 在遵循 Java Servlet 规范(http://java.sun.com/product/servlet/2.2)定义的标准容器环境中运行,容器环境在 HTTP 通信和 Web 服务器平台之间实现了一个抽象层。容器环境负责把请求传递给 Servlet,并把结果返回给客户。容器环境也提供了配置 Java Servlet 应用的简单方法,并且也提供了以声明性的方式,使用特殊的基于 Web 应用 XML 布置描述符对 Java Servlet 应用进行各种管理服务。容器环境是如此重要,因此 Java Servlet API 中包括了 ServletContext 接口,实现与容器环境的交互。

2.1 Servlet 生命周期

在用 Java Servlet 进行实际开发时,理解 Servlet 的生命周期是十分必要的。在一个具体的 HTTP 请求处理响应过程中,容器环境中 Java Servlet 的基本生命周期如下:

- 当 Web 客户请求 Servlet 服务或当 Web 服务启动时,容器环境加载一个 Java Servlet 类。
- 容器环境也将根据客户请求创建一个 Servlet 对象实例,或者创建多个 Servlet 对象实例,并把这些实例加入到 Servlet 实例池中。
- 容器环境调用 Servlet 的初始化方法 HttpServlet.init()进行 Servlet 实例化。在调用初始化时,要给 init()方法传入一个 ServletConfig 对象,ServletConfig 对象包含了初始化参数和容器环境的信息。
- 容器环境利用一个 HttpServletRequest 和 HttpServletResponse 对象,封装从 Web 客户接收到的 HTTP 请求和由 Servlet 生成的响应。
- 容器环境把 HttpServletRequest 和 HttpServletResponse 对象传递给 HttpServlet.Ser-vice()方法。这样,一个定制的 Java Servlet 就可访问这种 HTTP 请求和响应接口。
- 定制的 Java Servlet 从 HttpServletRequest 对象读取 HTTP 请求数据,访问来自 HttpSession 或 Cookie 对象的状态信息,进行特定应用的处理,并且用 HttpServletResponse 对象生成 HTTP 响应数据。
- 当 Web 服务器和容器关机时,调用 HttpServlet.destroy()方法关闭任何打开的资源,并进行一些关闭前的处理。