第 1 章　Web 应用程序概述

在掌握 Java 基础知识、理解面向对象编程的思想以后，就可以开始逐步学习开发 Web 应用程序。本章将一步一步地深入讲解进行 Web 应用开发所使用的技术结构——B/S 结构，以及 B/S 架构所具有的优势。

【技能目标】 掌握 Tomcat 服务器的发布与运行方法；掌握静态页面的设计。

【知识目标】 B/S 结构的基本概念；B/S 结构与 C/S 结构的区别；静态网页的设计；Tomcat 服务器的发布与运行方法。

【关键词】 超文本(hypertext)　传输(transfer)　协议(protocol)　资源(resource)
浏览器(browser)　服务器(server)　客户(client)　部署(deploy)

1.1　Web 相关概念

在信息管理系统的应用程序中有两种模式，一种模式是在客户端安装相应的应用程序；另一种模式则不需要在客户端安装应用程序，直接利用浏览器访问服务器就可以了，这就是所谓的 C/S 结构和 B/S 结构。

1.1.1　C/S 结构与 B/S 结构

C/S 结构与 B/S 结构如图 1-1 所示。

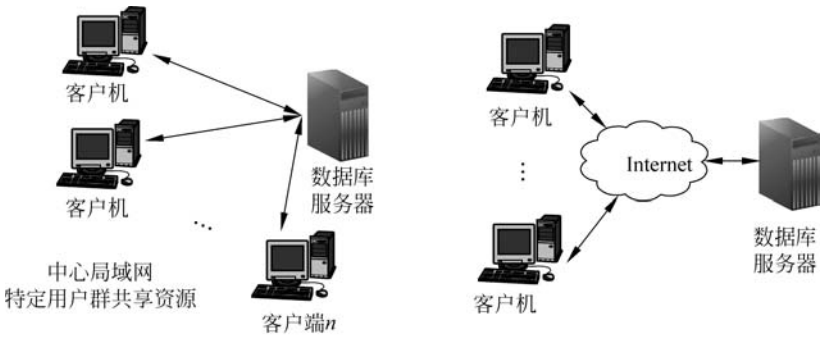

图 1-1　C/S 结构与 B/S 结构

C/S(client/server，客户/服务器)结构下的计算工作分别由服务器和客户机完成。服务器主要负责管理数据库，为多个客户程序管理数据，对数据库进行检索和排序等工作。客户机主要负责与用户的交互，收集用户信息，通过网络向服务器请求数据库、电子表格等信息的处理工作。在 C/S 结构下，资源明显不对等，是一种“胖客户机(fat client)”或“瘦服务器(thin server)”结构。

B/S(browser/server,浏览器/服务器)结构下,客户端不需要开发任何用户界面,而统一采用如 IE 之类的浏览器,通过 Web 浏览器向 Web 服务器提出请求,由 Web 服务器对数据库进行操作,并将结果逐级传回客户端。

B/S 结构简化了客户机的工作,客户机上只需配置少量的客户端软件。服务器将担负更多的工作,对数据库的访问和应用程序的执行在服务器上完成。浏览器发出请求,而其余如数据请求、加工、结果返回以及动态网页生成等工作全部由 Web 服务器完成。

1.1.2 静态网页与动态网页

网页一般又称 HTML 文档,是一种可以在 WWW 上传输、能被浏览器识别和翻译成页面并显示出来的文件。网页是构成网站的基本元素,是承载各种网站应用的平台。通常看到的网页,大都是以. htm 或. html 为扩展名的文件,除此之外网页文件还有以. cgi、. asp、. php 和. jsp 为扩展名的。目前网页根据生成方式,大致可以分为静态网页和动态网页两种。

1. 静态网页

静态网站是最初的建站方式,浏览者所看到的每个页面是建站者上传到服务器上的一个 HTML 文件(静态网页)。这种网站每增加、删除、修改一个页面,都必须重新对服务器上的文件进行一次下载上传。其特点如下。

(1) 网页内容不会发生变化,除非网页设计者修改了网页的内容。

(2) 不能实现和浏览网页的用户之间的交互。信息流向是单向的,即从服务器到浏览器。服务器不能根据用户的选择调整返回给用户内容。

(3) 网页的内容相对稳定,因此容易被搜索引擎检索。

(4) 网页没有数据库的支持,在网站制作和维护方面工作量较大,因此当网站信息量很大时完全依靠静态网页制作方式比较困难。

(5) 网页的交互性较差,在功能方面有较大的限制。

2. 动态网页

所谓“动态”,并不是指网页上简单的 GIF 动态图片或是 Flash 动画,动态网站的概念现在还没有统一标准,但都具备以下几个基本特点。

(1) 交互性。网页会根据用户的需求和选择而动态地改变和响应,浏览器作为客户端,成为一个动态交流的桥梁,动态网页的交互性也是当前 Web 发展的潮流。

(2) 自动更新。站点管理者无须手动更新 HTML 文档,便会自动生成新页面,可以大大节省工作量。

(3) 因时因人而变。浏览者在不同时间、不同用户访问同一网址时会出现不同页面。

3. 静态网页与动态网页的区别

动态与静态最根本的区别是网页在服务器端运行状态的不同,在服务器端运行的程序、网页、组件,属于动态内容,它们会随不同客户、不同时间,返回不同的网页,例如

ASP、PHP、JSP、ASP. NET、CGI 等。运行于客户端的程序、网页、插件、组件，属于静态内容，例如 HTML、Flash、JavaScript、VBScript 等，它们是不变的。

静态网页和动态网页各有特点，网站采用动态网页还是静态网页主要取决于网站的功能需求和网站内容的多少，如果网站功能比较简单，内容更新量不是很大，采用纯静态网页的方式会更简单；反之一般要采用动态网页技术来实现。

静态网页是网站建设的基础，静态网页和动态网页之间也并不矛盾，为了网站适应搜索引擎检索的需要，即使采用动态网站技术，也可以将网页内容转化为静态网页发布。

动态网站也可以采用静动结合的原则，适合采用动态网页的地方用动态网页，如果必须使用静态网页，则可以考虑用静态网页的方法来实现。在同一个网站上，动态网页内容和静态网页内容同时存在也是很常见的事情。

4. 动态网页实现的手段

动态网页大多是由网页编程语言写成的网页程序生成的，访问者浏览的只是其生成的客户端代码，而且动态网页要实现其功能大多还必须与数据库相连。目前比较常见的互动式网页编程语言有 ASP、PHP、JSP、ASP. NET。

(1) HTML 网页适用于所有环境，它本身也相当简单。

(2) ASP 及 ASP. NET 网页主流环境为 Windows Server 的 IIS＋Access/SQL Server。

(3) PHP 网页主流环境为：Linux/UNIX＋Apache＋MySQL＋PHP4＋Dreamweaver。

(4) JSP 网页环境为：JDK＋Tomcat＋Eclipse＋MyEclipse。

1.1.3　Web 运行环境

在了解如何开发 Web 应用程序之前，要首先了解一下这些应用程序的运行平台和环境，包括 Web 访问基本原理、HTTP 协议、Web 服务器以及 Web 浏览器。

1. Web 访问基本原理

图 1-2 显示了浏览器访问 Web 服务器的整个过程。

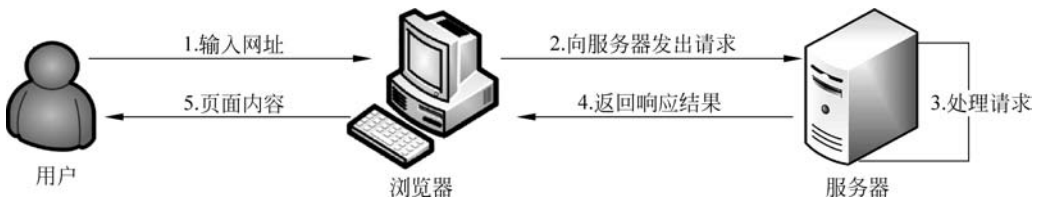

图 1-2　浏览器访问 Web 过程

(1) 用户在浏览器中输入网站的 URL。

(2) 浏览器寻找到指定的主机之后，向 Web 服务器发出请求(request)。

(3) Web 服务器接收请求并做出相应的处理，生成处理结果。

(4) 服务器把响应的结果返回给浏览器。

(5) 浏览器接收到响应结果后，在浏览器中显示相应的内容。

2. HTTP 协议

超文本传送协议(hypertext transfer protocol,HTTP)是一种用于分布式、协作式和超媒体信息系统的应用层协议。HTTP 是一个客户端终端(用户)和服务器端(网站)请求和应答的标准(TCP),是万维网数据通信的基础。

HTTP 协议定义客户端如何从 Web 服务器请求页面,以及服务器如何把页面传送给客户端。HTTP 采用了请求/响应模型。客户端向服务器发送一个请求报文,服务器以一个状态行作为响应。HTTP 请求/响应的步骤如下。

(1) 客户端连接到 Web 服务器。一个 HTTP 客户端,通常是浏览器,与 Web 服务器的 HTTP 端口(默认为 80)建立一个 TCP 套接字连接。

(2) 发送 HTTP 请求。通过 TCP 套接字,客户端向 Web 服务器发送一个文本的请求报文,一个请求报文由请求行、请求头部、空行和请求数据 4 部分组成。

(3) 服务器接收请求并返回 HTTP 响应。Web 服务器解析请求,定位请求的资源。服务器将资源副本写入 TCP 套接字,由客户端读取。一个响应由状态行、响应头部、空行和响应数据四部分组成。

(4) 释放 TCP 连接。若连接模式为 close,则服务器主动关闭 TCP 连接,客户端被动关闭连接,释放 TCP 连接;若连接模式为 keepalive,则该连接会保持一段时间,在该时间内可以继续接收请求。

(5) 客户端浏览器解析 HTML 内容。客户端浏览器首先解析状态行,查看表明请求是否成功的状态代码。然后解析每一个响应头部,响应头部告知以下为若干字节的 HTML 文档和文档的字符集。客户端浏览器读取响应数据 HTML,根据 HTML 的语法对其进行格式化,并在浏览器窗口中显示。

3. Web 服务器

Web 服务器一般指网站服务器,与通信相关的处理都是由服务器软件负责,开发人员只需要把功能代码部署在 Web 服务器中,客户端就可以通过浏览器访问这些功能代码,从而实现向客户提供的服务。目前主流的 Web 服务器有 IIS、Apache、Tomcat 等。

(1) IIS 服务器是微软提供的一种 Web 服务器,提供对 ASP 语言的良好支持,通过插件的安装,也可以提供对 PHP 语言的支持。

(2) Apache 服务器是由 Apache 基金组织提供的一种 Web 服务器,其特长是处理静态页面,对于静态页面的处理效率非常高。

(3) Tomcat 服务器也是由 Apache 基金组织提供的一种 Web 服务器,提供对 JSP 和 Servlet 的支持,通过插件的安装,Tomcat 是一个小型的轻量级 Web 服务器,是开发和调试 JSP 程序的首选。

(4) JBoss 服务器是一个开源的重量级的 Java Web 服务器,在 JBoss 中,提供对 J2EE 各种规范的良好支持,而且 JBoss 通过了 Sun 公司的 J2EE 认证,是 Sun 公司认可的 J2EE 容器。

(5) WebLogic 是 BEA 公司的产品,支持 J2EE 规范,而且不断地完善以适应新的开

发要求。WebLogic Server 支持企业级、分布式的 Web 应用，支持包括 JSP、Servlet、EJB 在内的 J2EE 体系，并提供必要的应用服务，如事务处理，支持集群技术。WebLogic Server 功能特别强大，配置操作简单、界面友好，在电子商务应用中被大量采用。

（6）WebSphere 是 IBM 公司的产品，支持 J2EE 规范。IBM 的 WebSphere Application Server 可运行于 Sun Solaris 等多种操作系统平台上；除可以使用 Servlet 和 JSP 之外，还可以补充用 EJB 编写的业务逻辑。几种技术结合起来，以开放的 Java 标准为基础，提供一种完整的编程模型，以实施各种 Web 站点。

（7）Nginx（Engine x）是一个高性能的 HTTP 和反向代理 Web 服务器，也提供 IMAP/POP3/SMTP 服务。其特点是占用内存少，并发能力强，事实上 Nginx 的并发能力在同类型的网页服务器中表现较好。

4. Web 浏览器

网页浏览器用于显示网页服务器或档案系统内的文件，并实现用户与这些文件进行交互操作。目前，有很多 Web 浏览器，但是比较普及和流行的为 Microsoft Internet Explorer（IE）、Mozilla Firefox 和 Google Chrome，其他的浏览器还有傲游浏览器（Maxthon）、腾讯 TT 浏览器、Opera 等。

1.2　Java Web 开发环境的安装与配置

开发 Java Web 应用程序需要搭建 Java Web 的开发和运行环境。

（1）JDK（Java development kit）：它是 Sun 官方的 Java 开发和运行环境。

（2）Eclipse 和 MyEclipse：它是 Java Web 集成开发环境（integrated develop environment，IDE）。

（3）Tomcat：它是开源 Web 应用服务器。

1.2.1　开发工具包 JDK

要运行 Java 的程序，首先要安装 Java 运行环境，即 JRE（Java runtime environment）。作为开发人员，需要安装 Java 的开发环境 JDK，JDK 包含开发 Java 程序的所有工具。

1. JDK 下载与安装

JDK 可以在 Sun 公司的主页下载，直接进入官网 https://www.oracle.com/，找到下载页后，选择相应的版本直接下载即可。通常 32 位的系统只支持 32 位的 JDK，64 位系统可以兼容 32 位和 64 位的 JDK。

下载完成后就可以运行 JDK 安装程序进行安装，安装过程中所有选项保持默认值即可。

2. JDK 配置

安装结束后进行环境变量的配置，基本步骤如下。

（1）依次单击“控制面板”→“高级系统设置”→“高级”，或右击“我的电脑”再依次单击“属性”→“高级系统设置”→“高级”，如图 1-3 所示。

图 1-3　单击“环境变量”按钮

（2）单击“环境变量”按钮，在系统变量下新建变量 JAVA_HOME，变量值指向 JDK 安装的文件夹，如图 1-4 所示。

（3）选中 Path，单击“编辑”按钮，直接在末尾添加%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin，如图 1-5 所示。

图 1-4　新建环境变量

图 1-5　编辑系统变量 Path

(4) 测试 JDK 环境配置是否成功。

按 Win+R 键在“运行”中输入 cmd,单击“确认”按钮,输入 java -version(java 后空一格)按 Enter 键。如果出现 JDK 版本信息,即 JDK 环境配置成功。如果出现“java 不是内部命令”说明配置失败,如图 1-6 所示。

图 1-6　测试 JDK 环境配置

配置不成功的原因通常是 JAVA_HOME 变量值错误,编辑 Path 时,新建变量输入的值不对,或者输入完成后,未单击“确认”按钮,而是直接关闭。

注意:如果以后要安装诸如 Eclipse、Borland JBuilder、JCreator、IntelliJ IDEA 等集成开发环境,应该在 IDE 中编译运行一个简单的程序进行测试,以保证 IDE 可以识别 JDK 的位置。

1.2.2　Tomcat 服务器

1. Tomcat 的下载与安装

进入 Tomcat 官方安装包下载页面 https://tomcat.apache.org/,在 Download 目录下,找到并下载合适版本的 Tomcat。将下载的压缩包直接解压到 D 盘根目录,然后按照安装提示完成安装。安装时要注意 JDK 的安装路径。

2. Tomcat 的配置

1) 配置 Tomcat 的环境变量

安装好 Tomcat 之后,依次单击“计算机”→“属性”→“高级系统设置”→“高级”→“环境变量”,打开环境变量设置对话框。

(1) 新建变量名 CATALINA_BASE,变量值为 D:\apache-tomcat。

(2) 新建变量名 CATALINA_HOME,变量值为 D:\apache-tomcat。

(3) 为 Path 添加变量值%CATALINA_HOME%\lib;%CATALINA_HOME%\bin,注意要用分号把 Path 的各个变量分开。

Tomcat 的环境配置是否成功:在命令行中,输入 startup,按 Enter 键,启动 Tomcat。设置成功,则能正常启动。

2) 修改 Tomcat 的 JDK 目录

打开 tomcat/bin/catalina.bat 文件,在最后一个 rem 后面增加 set JAVA_HOME=C:\Program Files\Java\jdk1.8.0。

3) 启动内存参数的配置

打开 tomcat/bin/catalina.bat 文件(如果是 Linux 系统则是 catalina.sh),在 rem 的

后面增加 set JAVA_OPTS= -Xms256m -Xmx256m -XX:MaxPermSize=64m。

4）Tomcat 的端口配置

Tomcat 默认使用 8080 端口，可以通过 server.xml 文件修改 Tomcat 的端口号。

将 port 定义的内容修改即可，如下面将端口号修改为 80 端口。

```
<Connector port="80" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort=
"8443" />
```

这样以后直接输入“http://localhost”即可进行访问，不用再输入端口号。

5）配置虚拟目录

在 Tomcat 服务器的配置中，最重要的就是配置虚拟目录，每个虚拟目录保存了一个完整的 Web 项目。设置虚拟目录为 site，通过 http://localhost:8080/site 访问物理路径 D:\site 目录中的内容。设置过程如下。

（1）复制 Tomcat\webapps\ROOT\WEB-INF 文件夹到 D:\site 目录中。

（2）打开 Tomcat\conf\server.xml 文件，在< Host >和</Host >之间加入以下内容。

```
<Context path="" docBase="ROOT" debug="0" reloadable="true"></Context>
<Context path="/site" docBase="d:\site" reloadable="true"/>
```

其中，path="/site" 是虚拟目录的名称；docBase="d:\site"为物理路径。

（3）打开 Tomcat\conf\web.xml 文件，找到以下内容。

```
<init-param>
  <param-name>listings</param-name>
  <param-value>false</param-value>
</init-param>
```

把 false 改成 true 后保存，重启 Tomcat，就可以应用 http://localhost:8080/site 虚拟目录了。其浏览效果如图 1-7 所示。当系统正式运行时，将< param-value >的值设置为 false。

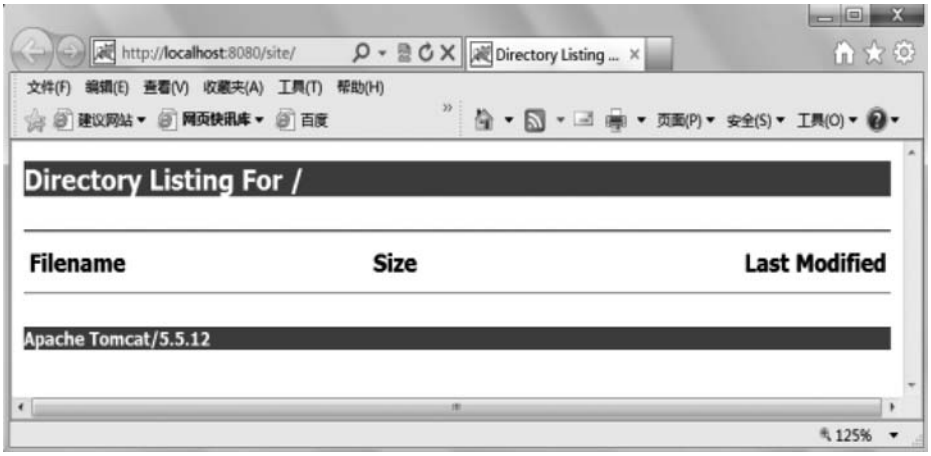

图 1-7　浏览虚拟目录

因为默认情况下，Tomcat 启动过程中配置虚拟目录的时候会从 webapps 目录下查找 webContent 应用。这样配置好了，即使以后从一台服务器移植到另一台服务器，不做任何修改也能运行。

6）解决 GET 方式下 URL 乱码问题

打开 tomcat/conf/server.xml，在最后增加以下代码。

```
<Connector port="80" maxHttpHeaderSize="8192"
URIEncoding="UTF-8" useBodyEncodingForURI="true"/>
```

其中的 UTF-8 可根据需要自己修改，比如也可以是 GBK。

7）配置虚拟主机文件

```
tomcat/conf/server.xml
<!-- 默认的主机 -->
<Host name="localhost" appBase="webapps" .unpackWARs="true"
autoDeploy="true" xmlValidation="false" xmlNamespaceAware="false">
<Context path="" docBase="ROOT" debug="0" reloadable="true"></Context>
</host>
<!-- 以下是新增的虚拟主机 -->
<Host name="" appBase="webapps" unpackWARs="true" autoDeploy="true" xmlValidation=
"false" xmlNamespaceAware="false">
<Context path="" docBase="d:\" debug="0" reloadable="true"/>
<!-- 虚拟目录 -->
<Context path="/count" docBase="d:\counter.java2000.net" debug="0" reloadable="true"/>
</Host>
<Host name="java2000.net" appBase="webapps" unpackWARs="true" autoDeploy="true"
xmlValidation="false" xmlNamespaceAware="false">
<Context path="" docBase="d:\ " debug="0" reloadable="true"/>
<Context path="/count" docBase="d:\counter.java2000. net" debug="0" reloadable="true"/>
</Host>
```

3. Tomcat 的安全设置

默认安装 Tomcat 时，Tomcat 作为一个系统服务运行。如果没有将其作为系统服务运行，通常会将其以 Administrators 权限运行。这两种方式都允许 Java 运行时访问 Windows 系统下任意文件夹中的任何文件，对系统的安全有极大的威胁。

为了保证系统的安全，应以 System 权限启动。根据权限最小的安全原则，降低脚本所获取的操作本地系统权限。操作步骤如下。

1）新建一个账户

通过操作系统创建一个普通用户账户，设置用户密码，并设置“密码永不过期”被选中。

2）修改 Tomcat 安装目录的访问权限

对 WebApps 目录设置只读权限，如果某些 Web 应用程序需要写权限，则单独为其授予。

3）Tomcat 作为系统服务运行

打开“控制面板”，选择“管理工具”选项卡，然后选择“服务”，找到 Tomcat，比如 Apache Tomcat. exe 等，打开“属性”，选择“登录”，选择“以……登录”（Log ON Using）选项。输入新建的用户名，输入密码，重启机器。

4. Tomcat 服务器的启动

Tomcat 的启动和停止脚本存在于 bin 目录下。其中，各脚本用途如下。

catalina：Tomcat 的主要脚本，它会执行 Java 命令以调 Tomcat 的启动与停止类。

configtest：Tomcat 的配置项检测脚本。

digest：生成 Tomcat 密码的加密摘要值，用于生成加密过的密码。

service：该脚本以 Windows 服务的方式安装和卸载 Tomcat。

setclasspath：这是唯一用于系统内部，以设定 Tomcat 的 classpath 及许多其他环境变量的脚本。

shutdown：运行 catalina. bat stop 可以停止 Tomcat 运行。

startup：运行 catalina. bat start 可以启动 Tomcat。

tool-wrapper：用于 digest 脚本系统内部。这是最常用的 Tomcat 命令行工具，用于封装可用于设置环境变量的脚本，并调用 classpath 中设置的完全符合限定的主要方法。

version：这是运行 Catalina 的版本，会输出 Tomcat 的版本信息。

执行 catalina. bat 时，必须附带一个选项，常用的有 start、run 及 stop。catalina 脚本启动选项含义如下。

-help：输出命令行选项的摘要表。

-nonaming：在 Tomcat 中停用 JNDI。

-security：启用 catalina. policy 文件。

debug：以调试模式启动 Tomcat。

embedded：在嵌入模式中测试 Tomcat，应用程序服务器的开发者通常使用此选项。

jpda start：以 jpda 的调试方式启动 Tomcat。

run：启动 Tomcat，但不会重定向标准输出与错误。

start：启动 Tomcat，并将标准输出与错误送至 tomcat 的日志文件。

stop：停止 Tomcat。

version：输出 Tomcat 的版本信息。

catalina. bat version：打印环境变量和版本信息。

当以 start 选项调用 catalina 时，它会启动 Tomcat，并将标准输出与错误流导出到 $TOMCAT_HOME/logs/catalina. out 文件中。选项 run 会让 Tomcat 保留当前的标准输出与错误流（如控制台窗口）。

如果使用 catalina 及 start 选项，或调用 startup 脚本而非使用参数 run，那么会在控制台上看到前几行 Using ……其余的输出信息则被重定向到 catalina. out 的日志文件中。