



动态网站 应用设计与开发

辛向丽 著



北京理工大学出版社

BEIJING INSTITUTE OF TECHNOLOGY PRESS

动态网站应用设计与开发

辛向丽 著

 **北京理工大学出版社**
BEIJING INSTITUTE OF TECHNOLOGY PRESS

版权专有 侵权必究

图书在版编目 (CIP) 数据

动态网站应用设计与开发 / 辛向丽著. —北京: 北京理工大学出版社, 2018. 8

ISBN 978 - 7 - 5682 - 6221 - 7

I. ①动… II. ①辛… III. ①网页制作工具 - 程序设计 IV. ①TP393.092

中国版本图书馆 CIP 数据核字 (2018) 第 198173 号

出版发行 / 北京理工大学出版社有限责任公司

社 址 / 北京市海淀区中关村南大街 5 号

邮 编 / 100081

电 话 / (010) 68914775 (总编室)

(010) 82562903 (教材售后服务热线)

(010) 68948351 (其他图书服务热线)

网 址 / <http://www.bitpress.com.cn>

经 销 / 全国各地新华书店

印 刷 / 北京富达印务有限公司

开 本 / 787 毫米 × 1092 毫米 1/16

印 张 / 11.5

字 数 / 250 千字

版 次 / 2018 年 8 月第 1 版 2018 年 8 月第 1 次印刷

定 价 / 38.00 元



责任编辑 / 杜春英

文案编辑 / 郭贵娟

责任校对 / 周瑞红

责任印制 / 施胜娟

图书出现印装质量问题, 请拨打售后服务热线, 本社负责调换

前 言

高等职业教育是针对岗位一线培养应用性技能型人才的，但目前高等职业教育普遍存在的问题是结合岗位的针对性不强，技能训练的实践性不够。为了解决这些问题，本书编写思路采用应用程序开发新模式：结合本岗位的题目（任务）驱动，掌握课程知识内容，再由掌握的课程知识点，做课程题目设计练习；通过题目设计实践，提高动态网站应用开发的专业能力。采用这种设计模式编写，在实际开发过程中，能够更好地适应高等职业教育人才的培养目标。本书的特点是将主要精力集中在所要解决的问题上，把动态网站应用程序开发的方法融入实际案例环节，并且在编排内容顺序方面保持与对应的网站开发体系内容相吻合，做到循序渐进、系统学习、广泛实践，便于读者接受。本书不但适用于高等职业教育，也可用作计算机岗位培训资料，还可作为网站开发爱好者的学习参考书。

本书编写目的是进行网站开发和解决实际问题。本书在保证了完整的 Java/JSP 知识体系的基础上，用大量的实例帮助读者掌握网站开发与程序设计思想，学会动态网站应用程序设计的方法，达到初步解决实际问题的要求。全书注重原理与实践结合，配有大量的实验例题和应用系统实践开发题目，实用性强。

本书共分 8 章，按照三个学习阶段。第 1 章入门阶段，介绍 JSP 概念和运行环境，熟悉 Eclipse 集成编程环境；通过上机操作进行问题解决。读者基本掌握 JSP 网站开发的过程和原理并激发学生学习的浓厚兴趣。第 2 ~ 5 章基础阶段，学习 JSP 基本语法和内置对象的使用，掌握 JSP 开发的基本流程和特点，使读者打下网站应用开发的坚实基础。第 6 ~ 8 章为提高阶段，其中第 6 章 Servlet，介绍 Servlet 的功能和特点，使读者掌握应用 Servlet 进行网站设计开发的能力；第 7 章，介绍如何运用实用组件进行网站开发；第 8 章学习使用 JDBC 数据库的访问。每章结合基础知识附有综合应用程序设计与开发案例，给出一些典型题目，根据实际问题，有针对性地进行程序设计和解决，培养网站应用程序设计和开发综合能力。

目 录

第 1 章 JSP 概述	1
1.1 JSP 技术简介	1
1.2 JSP 程序开发模式	1
1.3 JSP 处理过程	3
1.4 JSP 开发环境搭建	4
1.4.1 JDK 的安装与配置	4
1.4.2 Tomcat 的安装与启动	7
1.4.3 MySQL 的安装与启动	10
1.4.4 MyEclipse 的安装与使用	15
案例 1 开发环境搭建与 JSP 程序简单举例	16
1. 案例目的	16
2. 案例内容	16
3. 案例步骤	16
第 2 章 JSP 开发基础	19
2.1 Java 程序基础	19
2.1.1 Java 的程序分类	19
2.1.2 Java 的基本语法	21
2.1.3 Java 的数据类型	27
2.1.4 Java 的基本语句	36
2.1.5 Java 的类和对象	44
2.1.6 Java 的关键字	56
2.2 JavaScript 脚本语言	62
2.2.1 JavaScript 脚本语言简介	62
2.2.2 在 JSP 中应用 JavaScript	63
2.2.3 JavaScript 函数	63
2.2.4 JavaScript 事件	64
2.2.5 JavaScript 对象	65
案例 2 HTML 表单与 JavaScript 验证数据	68
1. 案例目的	68
2. 案例内容	68
3. 案例步骤	68
第 3 章 JSP 语法基础	73
3.1 JSP 页面构成要素	73
3.2 HTML 标签	74



3.3	Java 程序片	75
3.4	JSP 注释	76
3.5	JSP 指令	78
3.6	JSP 标签	82
案例 3	JSP 语法应用	89
1.	案例目的	89
2.	案例内容	89
3.	案例步骤	90
第 4 章	JSP 内置对象	92
4.1	JSP 内置对象简介	92
4.2	request 对象	93
4.2.1	访问请求参数	93
4.2.2	作用域管理属性	93
4.2.3	获取客户信息	93
4.3	response 对象	94
4.3.1	重定向网页	94
4.3.2	设置 HTTP 响应报头	95
4.3.3	缓冲区配置	95
4.4	session 对象	96
4.4.1	创建及获取会话对象	96
4.4.2	移除指定的对象	96
4.4.3	销毁 session 对象	96
4.4.4	会话超时的管理	97
4.5	application 对象	97
4.5.1	访问应用程序初始化参数	97
4.5.2	管理应用程序环境属性	97
4.6	out 对象	98
4.6.1	管理响应缓冲	98
4.6.2	向客户端输出数据	98
4.7	其他内置对象	99
4.7.1	获取会话范围的 pageContext 对象	99
4.7.2	读取 web.xml 配置信息的 config 对象	99
4.7.3	应答或请求的 page 对象	100
4.7.4	获取异常信息的 exception 对象	100
案例 4	JSP 内置对象	101
1.	案例目的	101
2.	案例内容	101
3.	案例步骤	101



第 5 章 JavaBean 技术	105
5.1 JavaBean 概述	105
5.1.1 JavaBean 简介	105
5.1.2 JavaBean 种类	105
5.1.3 JavaBean 规范	106
5.2 创建 JavaBean	106
5.3 JavaBean 应用	107
5.3.1 获取 JavaBean 信息	107
5.3.2 向 JavaBean 中发送信息	107
5.3.3 在 JSP 页面中应用工具 JavaBean	108
案例 5 JavaBean 技术的应用	108
1. 案例目的	108
2. 案例内容	108
3. 案例步骤	109
第 6 章 Servlet	114
6.1 Servlet 概述	114
6.1.1 什么是 Servlet	114
6.1.2 Servlet 的工作原理与生命周期	115
6.1.3 创建第一个 Servlet	116
6.1.4 web.xml 文件	117
6.2 请求与响应	118
6.2.1 处理表单的参数	118
6.2.2 Header 与初始化参数	119
6.2.3 发送非网页文档	120
6.2.4 转发与重定向	123
6.3 会话跟踪	125
6.3.1 Cookie	125
6.3.2 URL 重写	128
6.3.3 session	128
6.3.4 Servlet 的上下文	132
6.4 过滤器	132
6.4.1 什么是过滤器	132
6.4.2 创建过滤器	132
案例 6 Servlet 应用	134
1. 案例目的	134
2. 案例内容	134
3. 案例步骤	134
第 7 章 JSP 实用组件	138
7.1 JSP 文件操作	138



7.1.1	jspSmartUpload 组件安装与配置	138
7.1.2	jspSmartUpload 组件常用类	139
7.2	发送 E-mail	141
7.2.1	Java Mail 组件简介	141
7.2.2	Java Mail 核心类简介	142
7.2.3	搭建 Java Mail 开发环境	146
案例 7	JSP 组件应用 - jspSmartUpload	146
1.	案例目的	146
2.	案例内容	147
3.	案例步骤	147
第 8 章	JSP 数据库应用开发	152
8.1	JDBC 概述	152
8.1.1	JDBC 技术介绍	152
8.1.2	JDBC 驱动程序	153
8.2	JDBC 常用接口	153
8.2.1	驱动程序接口 Driver	153
8.2.2	驱动程序管理器 DriverManager	154
8.2.3	数据库连接接口 Connection	154
8.2.4	执行 SQL 语句接口 Statement	155
8.2.5	执行动态 SQL 语句接口 PreparedStatement	156
8.2.6	执行存储过程接口 CallableStatement	157
8.2.7	访问结果集接口 ResultSet	157
8.3	数据库操作	158
8.3.1	加载 JDBC 驱动程序	158
8.3.2	创建数据库连接	159
8.3.3	执行 SQL 语句	159
8.3.4	获得查询结果	160
8.3.5	关闭连接	161
8.4	数据库常用操作	161
8.4.1	查询操作	161
8.4.2	添加操作	161
8.4.3	修改操作	162
8.4.4	删除操作	163
案例 8	JSP 数据库应用开发	163
1.	案例目的	163
2.	案例内容	163
3.	案例步骤	163

第 1 章

JSP 概述

1.1 JSP 技术简介

在介绍 JSP 技术之前，首先需要了解一些与 JSP 技术相关的技术。

Java 语言：Java 是一门面向对象编程语言，吸收 C/C++ 语言的优点，摒弃 C/C++ 语言中的指针等概念，极好地实现了面向对象理论，允许程序员以优雅的思维方式进行复杂的编程。Java 可以编写多种应用程序，如 Web 应用程序、分布式系统和嵌入式系统应用程序。

Servlet 技术：Servlet (Server Applet) 是 Java Servlet 的简称，称为小服务程序或服务连接器，是用 Java 编写的服务器端程序，主要功能在于交互式地浏览和修改数据，生成动态 Web 内容。从 Servlet 的命名可以看出 Java 命名的特点，如 Applet 表示小应用程序；Scriptlet = Script + Applet，表示小脚本程序；同样 Servlet = Service + Applet，表示小服务程序。

JavaBean 技术：JavaBean 是一种 Java 语言写成的可重用组件。JavaBean 类必须是具体的和公共的，并且具有无参数的构造器。JavaBean 通过提供符合一致性设计模式的公共方法，将内部成员属性通过 set 和 get 方法获取。属性名称符合这种模式，其他 Java 类可以通过自省机制（反射机制）发现和操作这些 JavaBean 的属性。JavaBean 可分为两种：一种是有用户界面（UI, User Interface）的 JavaBean；还有一种是没有用户界面，主要负责处理事务（如数据运算、操纵数据库）的 JavaBean。JSP 通常访问的是后一种 JavaBean。

JSP 技术：JSP 全名为 Java Server Pages，其根本是一个简化的 Servlet 设计，JSP 技术有点类似 ASP 技术，它是在传统的网页 HTML（标准通用标记语言的子集）文件（*.htm, *.html）中插入 Java 程序段（Scriptlet）和 JSP 标记（tag），从而形成 JSP 文件，后缀名为（*.jsp），实现了 Html 语法中的 Java 扩展（以 <%,%> 形式）。JSP 与 Servlet 一样，是在服务器端执行的。通常返回给客户端的就是一个 HTML 文本，因此客户端只要有浏览器就能浏览。JSP 技术使用 Java 编程语言编写类 XML 的 tags 和 scriptlets，来封装产生动态网页的处理逻辑。网页还能通过 tags 和 scriptlets 访问存在于服务端的资源的应用逻辑。JSP 将网页逻辑与网页设计的显示分离，支持可重用的基于组件的设计，使基于 Web 的应用程序的开发变得迅速和容易。

1.2 JSP 程序开发模式

1. 单纯的 JSP 页面编程模式

在该模式下，通过应用 JSP 中的脚本标志，可直接在 JSP 页面中实现各种功能。虽然这种模式很容易实现，但是其缺点也非常明显。因为将大部分的 Java 代码与 HTML 代码混淆



在一起，会给程序维护和调试带来很多困难，而且对于整个程序的结构更是无从谈起。这就好比规划管理一个大的企业，如果将负责不同任务的所有员工都安排在一起工作，势必会造成公司秩序混乱、不易管理等隐患。所以说，单纯的 JSP 页面编程模式是无法应用到大型、中型甚至小型的 JSP Web 应用程序开发中的。

2. JSP + JavaBean 编程模式

该模式是 JSP 程序开发经典设计模式之一，适合小型或中型网站的开发。利用 JavaBean 技术，可以很容易地完成一些业务逻辑上的操作。JavaBean 是一个遵循一定规则的 Java 类，在 JSP 页面中通过动作标签来调用这个类，从而执行这个业务逻辑。此时的 JSP 除了负责部分流程的控制外，大部分用来进行页面的显示，JavaBean 则负责业务逻辑的处理。可以看出，该模式具有一个比较清晰的程序结构，在 JSP 技术的起步阶段，JSP + JavaBean 设计模式曾被广泛应用。下面将通过一个流程图来说明该模式对客户端的请求进行处理的过程，如图 1-1 所示。

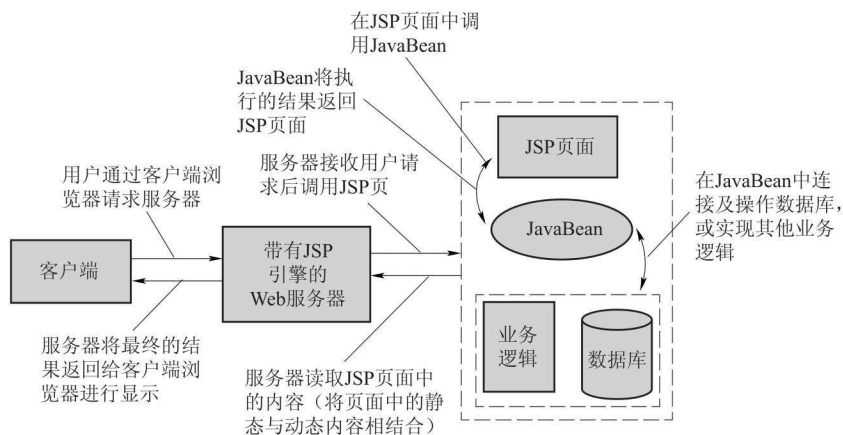


图 1-1 JSP + JavaBean 编程模式

3. JSP + Servlet + JavaBean 编程模式

JSP + JavaBean 编程模式虽然已经将网站的业务逻辑和显示页面进行分离，但这种模式下的 JSP 不但要进行程序中大部分的流程控制，而且要负责页面的显示，所以仍然不是一种理想的设计模式。在 JSP + JavaBean 编程模式的基础上加入 Servlet 来实现程序中的控制层，是一个很好的选择。在这种模式中，由 Servlet 来执行业务逻辑并负责程序的流程控制，JavaBean 组件实现业务逻辑，充当着模型的角色，JSP 用于页面的显示。可以看出，这种模式使得程序中的层次关系更明显，各组件的分工也非常明确。下面将通过一个流程图说明该模式对客户端的请求进行处理的过程，如图 1-2 所示。

4. MVC 模式

MVC (Model - View - Controller, 模型—视图—控制器) 是一种程序设计概念，它同时适用于简单的和复杂的程序。使用该模式可将待开发的应用程序分解为 3 个独立的部分：模型、视图和控制器。提出这种设计模式主要是因为应用程序中用来完成任务的代码——模型（也称为“业务逻辑”）通常是程序中相对稳定的部分，并且会被重复使用，而程序与用户进行交互的页面——视图却是经常改变的。如果因需要更新页面而不得不对业务逻辑代码进行改动，或者要在不同的模块中应用到相同的功能而重复地编写业务逻辑代码，那么不仅会

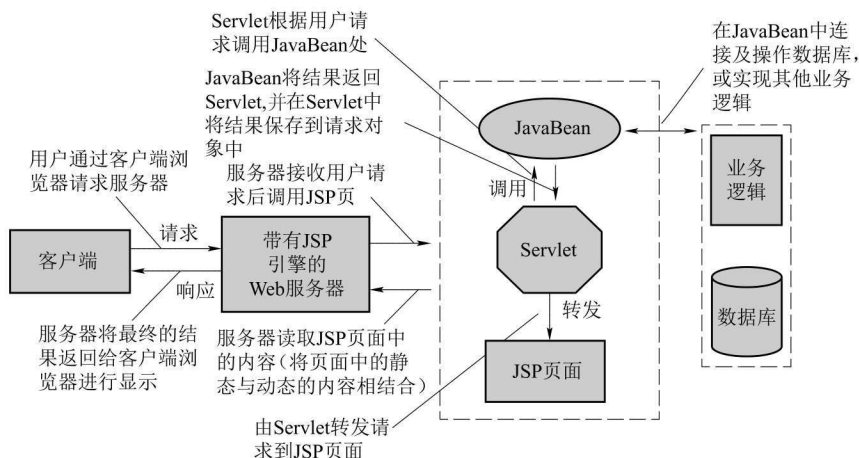


图 1-2 JSP + Servlet + JavaBean 编程模式

降低整体程序的开发进程，而且会使程序变得难以维护。因此，将业务逻辑代码与外观呈现分离，将会更加容易根据需求的改变来改进行程。

MVC 模式中的 Model（模型）指的是业务逻辑的代码，是应用程序中真正用来完成任务的部分。

View（视图）实际上就是程序与用户进行交互的界面，用户可以看到它的存在。视图可以具备一定的功能并应遵守对其的约束，在视图中不应包含对数据处理的代码，即业务逻辑代码。

Controller（控制器）主要控制用户请求并做出响应，它根据用户的请求选择模型或修改模型，并决定返回怎样的视图。

1.3 JSP 处理过程

当客户端浏览器向服务器发出请求访问一个 JSP 页面时，服务器根据该请求加载相应的 JSP 页面，并对该页面进行编译，然后执行。JSP 的具体处理过程如图 1-3 所示。

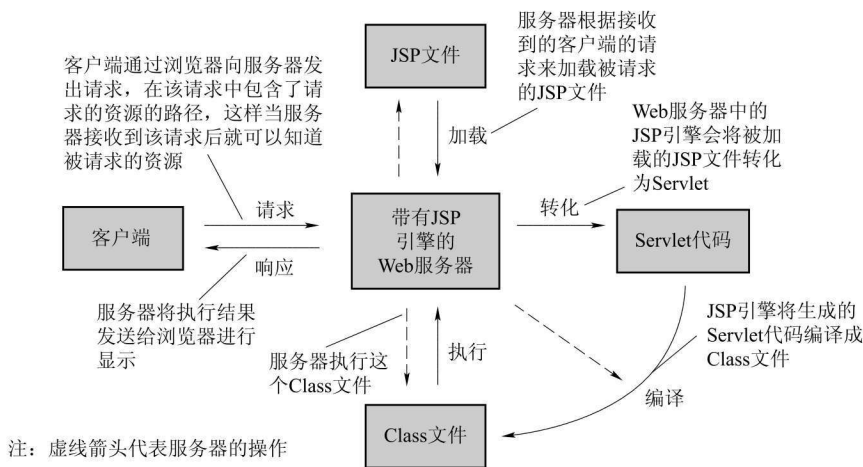


图 1-3 JSP 处理过程



从前面的介绍中可以看到，JSP 文件被 JSP 引擎进行转换后，又被编译成 Class 文件，最终由服务器通过执行这个 Class 文件来对客户端的请求进行响应。其中第 3 步与第 4 步构成了 JSP 处理过程中的翻译阶段，而第 5 步为请求处理阶段。但并不是每次请求都需要重复进行这样的处理。当服务器第一次接收到对某个页面的请求时，JSP 引擎就开始进行上述的处理过程，将被请求的 JSP 文件编译成 Class 文件。在后续对该页面再次进行请求时，若页面没有进行任何改动，那么服务器只需直接调用 Class 文件执行即可。所以当某个 JSP 页面第一次被请求时，会有一些延迟，而再次访问时会快很多。如果被请求的页面经过修改，那么服务器将会重新编译这个文件，然后执行。

1.4 JSP 开发环境搭建

1.4.1 JDK 的安装与配置

Java Development Kit（简称 JDK）是 Oracle 公司针对 Java 开发人员发布的免费软件开发工具包。其版本号也在不断更新中，从 JDK1.5 开始增加了泛型类库等功能，经历 JDK1.6，目前使用最广泛的是 JDK1.8 版本，最新版本是 JDK1.10。JDK 中包含一组用于 Java 程序开发的组件，在开发 Java 程序的过程中主要用到以下几个最常用的工具：

- (1) javac——Java 语言编译器，将后缀名为 .java 的源代码编译成后缀名为 .class 的字节码文件。
- (2) java——Java 语言的运行工具，用于运行 .class 的字节码文件。
- (3) appletViewer——Java 小程序（Applet）浏览工具，用于测试并运行 Java 小程序。
- (4) jar——打包工具，可以将相关的多个 Java 类文件打包成一个归档文件。
- (5) javadoc——Java API 帮助文档生成器，可以直接从源程序生成帮助文档。
- (6) jdb——Java 调试工具。
- (7) javah——可以从 Java 类生成 C 语言头文件及源文件，实现 Java 和 C 语言的交互转换。
- (8) javap——将字节码 .class 文件分解还原成源文件，显示类文件中的可访问数据和方法。

1. JDK 的下载与安装

本书采用的是 JDK1.8 版本，读者可以到 Oracle 公司官方网站进行下载，网址是：<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>，下载后的文件名称为 jdk-8u171-windows-x64.exe，双击该文件即可开始安装。安装步骤如下：

- (1) 双击 jdk-8u171-windows-x64.exe 文件，在弹出的“安装向导”对话框中，单击“下一步”按钮，开始安装，如图 1-4 所示。
- (2) 在“定制安装”对话框中，可以设置 JDK 的安装路径并选择安装组件。这里选择默认即可，单击“下一步”按钮，进行安装，如图 1-5 所示。
- (3) 在“安装进度”对话框中，将打开 JRE 安装路径，设置 JRE 安装路径，如图 1-6 所示。



图 1-4 “安装向导”对话框



图 1-5 “定制安装”对话框



图 1-6 JRE 安装路径设置



(4) 在“目标文件夹”对话框中,单击“下一步”按钮继续完成 JRE 安装。当安装完成时,将打开安装成功提示对话框,单击“完成”按钮,即可完成 JDK 的安装。

2. JDK 的配置与测试

1) JDK 环境变量配置

JDK 安装完成后不能直接使用,必须设置一下环境变量。JDK1.5 以上的版本通常需要设置两个环境变量,一个命名为 path,另一个命名为 classpath。在 Windows 操作系统下,系统环境变量的设置方法如下:

(1) 在桌面上选择“我的电脑”,单击鼠标右键,选择“高级”→“环境变量”按钮。

(2) 在“系统变量”区域单击“新建”按钮,在变量名中输入“path”,变量值中输入 JDK 的安装路径,比如作者的机器上输入“E:\JavaTool\jdk1.5.0_09\bin”,如图 1-7 所示。

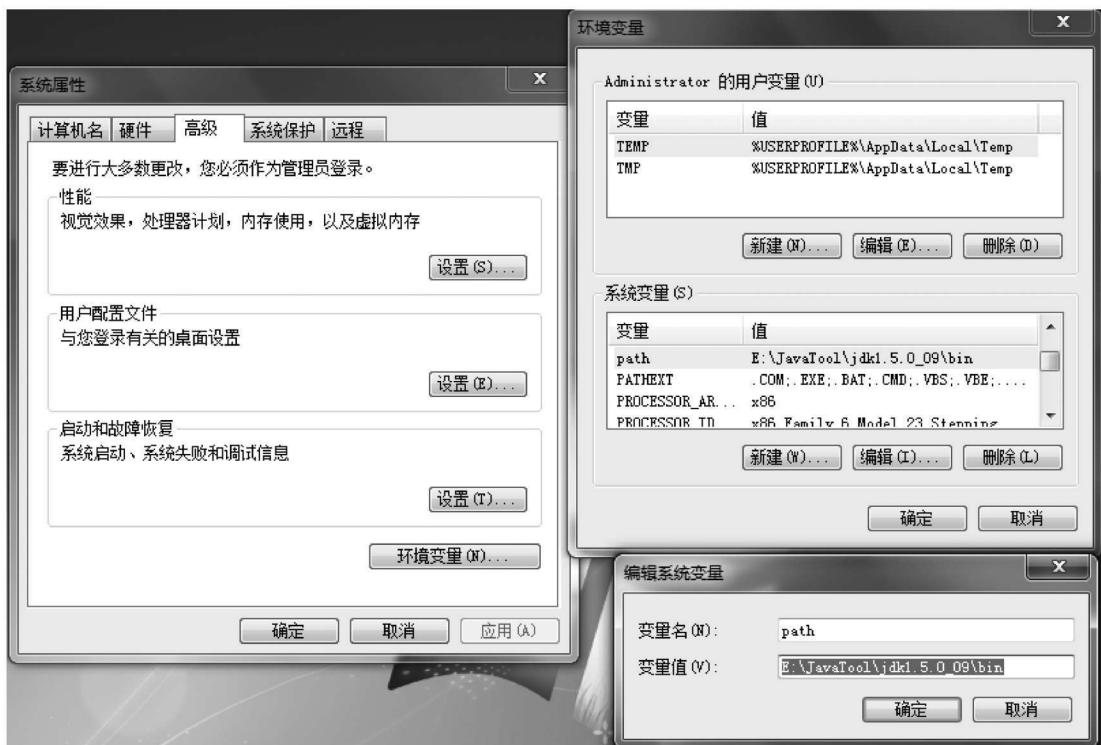


图 1-7 环境变量设置

用类似的方法,新建第 2 个环境变量“classpath”,变量值中输入 JDK 的安装路径\bin,比如作者的机器上输入“E:\JavaTool\jdk1.5.0_09\bin;.”。然后单击“确定”按钮。配置完成后,要重新启动计算机,环境变量才能生效。

2) JDK 的使用

JDK 工具包中直接包含了 Java 的编译器和解释器,无论通过什么工具编写保存的 Java 源程序(.java 文件),最终都要通过 JDK 的编译器 Javac 进行编译,编译成功后生成二进制的字节码文件(.class),最终通过 JDK 的解释器 Java 来执行字节码文件,可看到程序的运行结果;对于小程序,最后要通过小程序查看器命令来运行。

直接使用 JDK 命令时要打开 DOS 命令窗口,在命令窗口中执行 JDK 命令。



JDK 编译命令的格式如下：

```
javac ×××.java //javac 命令,编译×××.java 源文件
```

JDK 运行命令的格式如下：

```
java ××× //java 命令,运行×××.class 文件时略去.class 后缀,直接写类名即可。
```

JDK 小程序查看（运行网页文件）命令的格式如下：

```
appletViewer ×××.html //运行网页文件,可以看到小程序在网页中的运行结果。
```

1.4.2 Tomcat 的安装与启动

Tomcat 是 Apache 组织（Apache Software Foundation）推出的开放源代码的 Web 应用服务器，属于轻量级应用服务器，是一个世界上广泛使用的、支持 jsp 和 servlets 的 Web 服务器。它在 Java 上运行时能够很好地运行并支持 Web 应用部署。最新的 Servlet 和 JSP 规范总是能在 Tomcat 中得到体现。Tomcat 技术先进、性能稳定，是开发和调试 JSP 程序的首选。当在一台机器上配置好 Tomcat 服务器时，可利用它响应 HTML 页面的访问请求，为 HTML 页面服务，Tomcat 实际上运行 JSP 页面和 Servlet。另外，Tomcat 与 IIS 等 Web 服务器一样，具有处理 HTML 页面的功能。

1. Tomcat 的下载

本书采用的是 Tomcat 9.0 版本，读者可以到 Tomcat 官方网站进行下载，网址是：<https://tomcat.apache.org/download-90.cgi>，下载界面如图 1-8 所示，下载后的文件名称为 apache - tomcat - 9.0.10.exe。

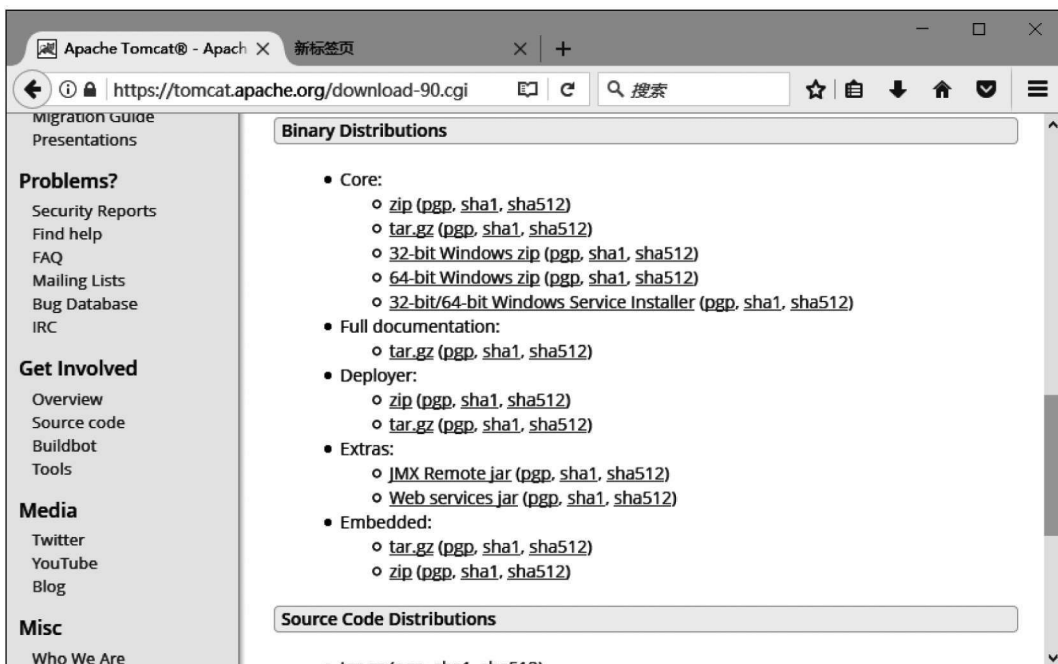


图 1-8 网站下载列表



2. Tomcat 的安装

Tomcat 下载完成后，就可以通过下载的文件来安装 Tomcat 服务器，安装步骤如下：

(1) 双击 apache - tomcat - 9.0.10.exe 文件，打开“安装向导”对话框，单击“Next”按钮，将打开“许可协议”对话框。

(2) 单击“I Agree”按钮，接受许可协议，将打开“Choose Components”对话框，在该对话框中可选择安装需要的组件，通常保留默认选项，如图 1-9 所示。

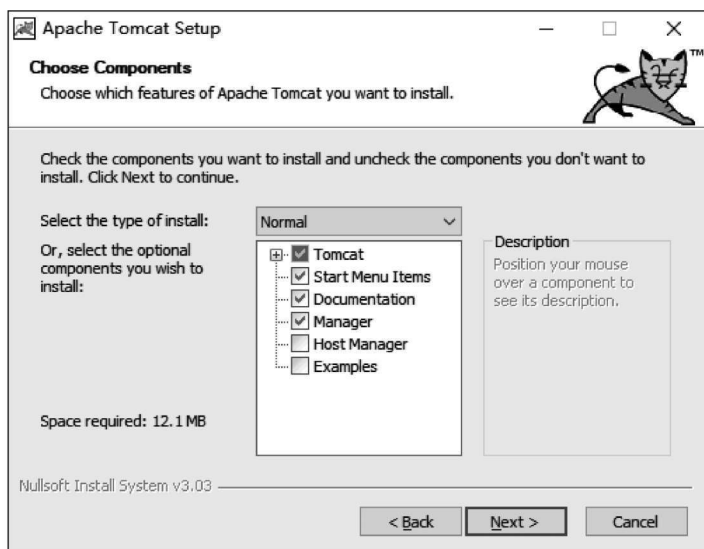


图 1-9 “Choose Components”对话框

(3) 单击图 1-9 中的“Next”按钮，将弹出“Configuration”对话框（见图 1-10），在该对话框中设置 Tomcat 服务器的端口号、用户名和密码，通常保留默认设置，即端口号为 8080，用户名和密码为空。

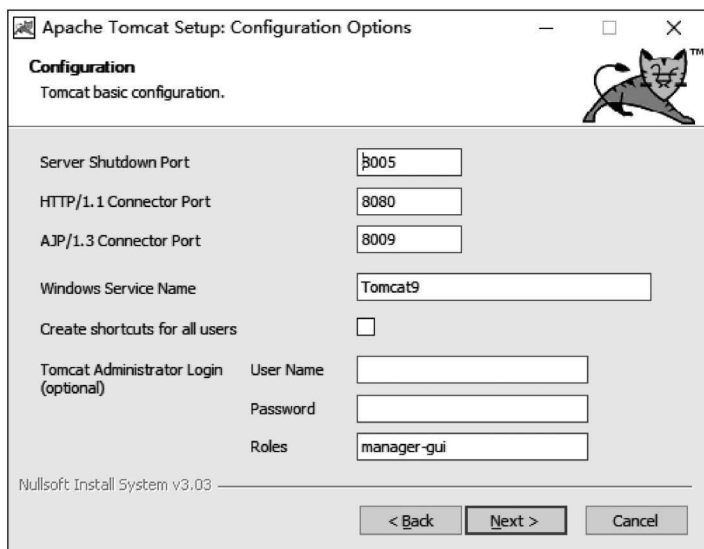


图 1-10 “Configuration”对话框



(4) 单击图 1-10 中的“Next”按钮，将弹出“Java Virtual Machine”对话框，如图 1-11 所示。在该对话框中可以选择 JRE 的路径。

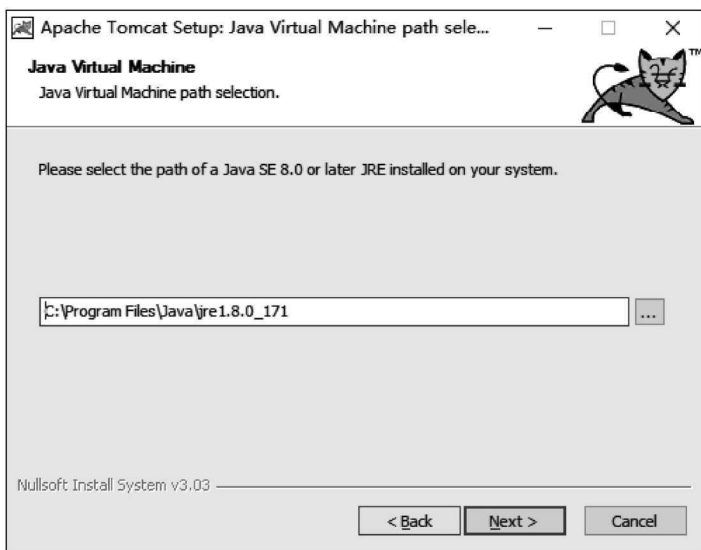


图 1-11 “Java Virtual Machine”对话框

(5) 单击图 1-11 中的“Next”按钮，将弹出“Choose Install Location”对话框，如图 1-12 所示。在该对话框中可以更改 Tomcat 的安装路径。

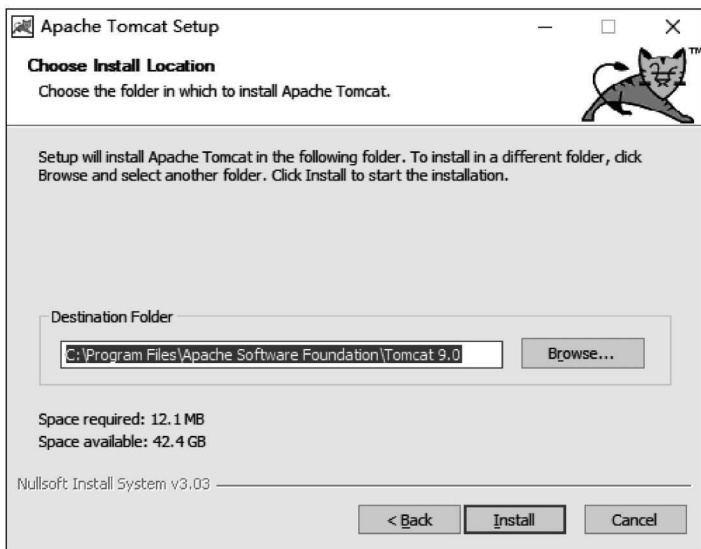


图 1-12 “Choose Install Location”对话框

(6) 单击图 1-12 中的“Next”按钮，开始安装 Tomcat。

3. Tomcat 的启动

安装完成后，启动并访问 Tomcat 服务器，具体如下：

(1) 单击“开始”→“程序”→“Monitor Tomcat”菜单项，在任务栏中将出现  图