

■ 高等学校计算机教材 ■

Java EE

实用教程

■ 郑阿奇 主编 ■



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

高等学校计算机教材

Java EE 实用教程

郑阿奇 主编

图书在版编目(CIP)数据

Java EE 实用教程 / 郑阿奇主编. — 北京: 电子工业出版社, 2003.11
ISBN 978-7-121-00737-0
I. J... II. 郑... III. JAVA... IV. TP311.72
中国版本图书馆CIP数据核字(2003)第...号
北京市西城区... 100039
... 807...
... 2003年11月第1次印刷
... 4100册 定价: 42.00元

电子工业出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书主要包含实用教程、实验指导和综合应用实习三部分，实用教程部分系统介绍 Java EE 开发环境配置、Struts 2 应用、Hibernate 应用、Spring 应用及它们之间的整合，其中的习题主要用于强化基本概念。实验指导部分着重训练学生的动手能力，训练内容仍然是实用教程部分的实例。Struts 2、Hibernate、Spring 每部分讲解完成后都实现了同样功能的实例，但分别是应用 Struts 2、Struts 2+Hibernate、Struts 2+Hibernate+Spring 完成的，让读者可以清楚地对比它们之间的应用方式。综合应用实习部分介绍学生成绩管理系统。本书免费提供教学课件、程序源代码和没有冲突的 Jar 包。

本书可作为大学本科和大专相关课程教材和教学参考书，也可供从事 Java EE 应用系统开发的用户学习和参考。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

Java EE 实用教程 / 郑阿奇主编. —北京: 电子工业出版社, 2009.11

高等学校计算机教材

ISBN 978-7-121-09727-0

I. J… II. 郑… III. JAVA 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2009) 第 189125 号

责任编辑: 王昭松

印 刷: 北京季蜂印刷有限公司

装 订: 三河市万和装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编: 100036

开 本: 787×1092 1/16 印张: 30.00 字数: 807 千字

印 次: 2009 年 11 月第 1 次印刷

印 数: 4 000 册 定价: 42.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前 言

Java EE 技术经过多年的发展日趋成熟，已经成为最佳电子商务解决方案之一，关注的人也越来越多。所以，学习 Java EE 不仅是一种时尚，更是一种需要。为了适应形势发展的需要，我国许多高校开始开设 Java EE 课程。但目前讲解 Java EE 的书多为面向企业开发的中高级应用而编写的，不能适应学生学习的需要。经过一段时间的思考和研究，我们根据多年的教学经验及 Java 开发应用的实际，编写了本教材。

Java EE 技术包含很多内容，本书从目前市场最流行的几个 Java EE 框架入手，分别介绍 Struts 2、Hibernate、Spring 以及它们的相互整合应用。每种技术都从最基础的例子着手，一步一步引导读者学习和应用这些技术。为了让读者能够看得懂、学得会，教材初稿通过初步审读后，由仅有 Java 基础的人员来阅读本书，如发现讲解不清楚或做不通的问题，由编写人员继续修改和完善。教程所有示例都试做通过。

本书主要包含**实用教程、实验指导和综合应用实习三部分，涵盖了理论和实践教学的全过程**。实用教程部分系统介绍 Java EE 开发环境配置、Struts 2 应用、Hibernate 应用、Spring 应用及它们之间的整合，其中的习题主要用于强化基本概念。实验指导部分着重训练学生的动手能力，训练内容仍然是实用教程部分的实例。Struts 2 部分实现了一个图书管理系统的部分功能，在 Hibernate 部分又应用 Struts 2 与 Hibernate 的整合完成了在 Struts 2 部分实现的图书管理系统的部分功能，让读者能够深刻理解 Hibernate 在项目开发中的应用。最后在 Spring 部分又应用 SSH 三者之间的完美整合来实现前面所做的图书管理系统的部分功能，让读者体会到应用框架的妙处，从而深刻理解这 3 个框架的详细应用。综合应用实习部分从贴近学生的角度实现了学生成绩管理系统的开发。

本书为任课老师免费提供教学课件、系统所有源代码和没有冲突的 Jar 包。需要者可从电子工业出版社华信教育资源网 (www.hxedu.com.cn) 免费注册下载。

本书可作为大学本科和大专相关课程教材和教学参考书，也可供从事 Java EE 应用系统开发的用户学习和参考。

本书由郑阿奇（南京师范大学）主编，许多同志帮助完成了基础工作，在此一并表示感谢！

参加本套丛书编写的还有梁敬东、顾韵华、王洪元、杨长春、王一莉、曹弋、徐文胜、丁有和、刘启芬、姜乃松、殷红先、张为民、丁有和、彭作民、高茜、陈冬霞、朱毅华、时跃华、赵青松、刘毅、郑进、周怡君、赵阳、周旭琴、陈金辉、李含光、黄群、徐斌、毛凤伟、王志瑞等。刘彦博也参加了本书的编写工作，在此一并表示感谢！

由于 Struts 2、Spring、Hibernate 集成时会有可能发生 Jar 包冲突，建议读者在开发程序时，上网下载我们提供的没有冲突的版本，替换原来的 Jar 包。

由于我们的水平有限，疏漏和错误在所难免，敬请广大师生、读者批评指正。

意见和建议可反馈至作者邮箱：easybooks@163.com。

目 录

第 1 部分 实用教程

第 1 章	Java EE 概述	1
1.1	Java EE 应用概述	1
1.1.1	Struts 2 框架	1
1.1.2	Hibernate 框架	2
1.1.3	Spring 框架	2
1.2	Java EE 开发环境的构建	2
1.2.1	JDK 和 Tomcat 的安装与配置	2
1.2.2	MyEclipse 的安装与配置	6
习题 1	7
第 2 章	Struts 2 基础	8
2.1	Struts 2 框架的发展	8
2.1.1	MVC 模式的应用	8
2.1.2	Struts 框架应用	9
2.1.3	WebWork 框架应用	9
2.1.4	Struts 2 框架的诞生	10
2.2	Struts 2 工作流程结构	10
2.3	Struts 2 项目开发	11
2.4	Struts 2 项目的运行流程	17
2.5	程序文件详解	18
2.5.1	Action 类应用详解	18
2.5.2	<action>配置详解	24
2.5.3	<result>配置详解	27
2.5.4	<package>配置详解	30
2.5.5	struts.xml 文件	32
2.5.6	struts.properties 文件	34
2.5.7	web.xml 文件	37
习题 2	38
第 3 章	Struts 2 的标签库	39
3.1	Struts 2 的 OGNL	39
3.2	Struts 2 的标签库	41
3.2.1	数据标签	41
3.2.2	控制标签	51

3.2.3	表单标签	59
3.2.4	非表单标签	67
习题 3	69
第 4 章	Struts 2 的类型转换及输入校验	70
4.1	Struts 2 类型转换	70
4.1.1	Struts 2 的内置类型转换器	70
4.1.2	类型转换	70
4.2	自定义类型转换器	73
4.3	对数组类型及集合类型的转换	79
4.3.1	数组类型的使用	79
4.3.2	集合类型的使用	82
4.4	Struts 2 的输入校验	83
4.4.1	使用 execute()方法校验	85
4.4.2	重写 validate()方法进行校验	86
4.4.3	使用 Struts 2 校验框架	87
4.4.4	客户端校验与服务器端校验	90
习题 4	91
第 5 章	Struts 2 的拦截器	92
5.1	拦截器概述	92
5.2	拦截器的配置	94
5.3	自定义拦截器类	99
5.3.1	实例说明自定义拦截器的应用	99
5.3.2	应用拦截器处理重复提交	101
习题 5	103
第 6 章	Struts 2 的其他应用	104
6.1	Struts 2 的文件上传	104
6.1.1	单文件上传	104
6.1.2	多文件上传	107
6.2	Struts 2 的文件下载	109
6.3	Struts 2 的国际化应用	111
6.3.1	应用国际化的准备工作	111
6.3.2	国际化资源的访问	115
6.3.3	国际化应用实例	116
习题 6	118
第 7 章	Struts 2 综合应用实例	119
7.1	建立项目	119
7.2	运行主界面的实现	120

7.3	实现“登录”功能	127
7.4	实现“借书”功能	130
7.4.1	查询已借图书	132
7.4.2	“借书”功能	144
7.5	实现“图书管理”功能	150
7.5.1	“图书追加”功能	151
7.5.2	“图书删除”功能	159
7.5.3	“图书查询”功能	161
7.5.4	“图书修改”功能	162
	习题 7	163
第 8 章	Hibernate 基础	164
8.1	ORM 简介	164
8.2	建立第一个 Hibernate 程序	164
8.2.1	建立项目	165
8.2.2	创建 POJO 类及其映射文件	166
8.2.3	创建 Hibernate 的核心配置文件	167
8.2.4	创建并运行测试类	168
8.3	Hibernate 工作流程	170
8.4	Hibernate 常用接口	171
8.4.1	Configuration 接口	171
8.4.2	SessionFactory 接口	172
8.4.3	Session 接口	172
8.4.4	Transaction 接口	172
8.4.5	Query 接口	173
	习题 8	173
第 9 章	Hibernate 的映射机制	174
9.1	主键映射	174
9.1.1	代理主键的映射	174
9.1.2	单个自然主键的映射	176
9.1.3	复合自然主键的映射	176
9.2	Hibernate 的映射类型	181
9.3	对象关系映射	182
9.3.1	继承关系映射	183
9.3.2	关联关系映射	192
9.4	动态类的使用	206
	习题 9	208
第 10 章	Hibernate 对持久化对象的操作	209
10.1	Session 的常用方法	209

10.1.1	Session 的 save()方法	209
10.1.2	Session 的 get()和 load()方法	209
10.1.3	Session 的 delete()方法	210
10.1.4	Session 的 update()方法	210
10.1.5	Session 的 saveOrUpdate()方法	211
10.2	HQL 查询	211
10.2.1	基本查询	212
10.2.2	条件查询	212
10.2.3	分页查询	214
10.2.4	连接查询	214
10.2.5	子查询	216
10.2.6	SQL 查询	216
10.3	Hibernate 的批量操作	220
10.3.1	批量插入	220
10.3.2	批量更新	222
10.3.3	批量删除	223
10.4	持久对象的生命周期	223
习题 10	225
第 11 章	Hibernate 的高级特性	226
11.1	Hibernate 的事务管理	226
11.1.1	事务的概念	226
11.1.2	Hibernate 的事务	226
11.2	Hibernate 的并发	229
11.2.1	并发产生的问题	229
11.2.2	解决方案	231
11.3	Hibernate 的拦截器	232
11.3.1	Interceptor 接口	232
11.3.2	应用举例	233
习题 11	237
第 12 章	Hibernate 与 Struts 2 的整合应用实例	238
12.1	Hibernate 的反向工程	238
12.2	编写 DAO 接口及实现类	247
12.3	功能实现	255
12.3.1	“登录”功能的实现	255
12.3.2	“查询已借图书”功能的实现	257
12.3.3	“借书”功能的实现	260
12.3.4	“图书管理”功能的实现	263
习题 12	268

第 13 章 Spring 基础	269
13.1 Spring 简介	269
13.2 项目实例开发	270
习题 13	273
第 14 章 Spring 的核心机制及容器	274
14.1 Spring 的核心机制	274
14.1.1 依赖注入的概念	274
14.1.2 依赖注入的两种方式	277
14.1.3 两种注入方式的比较	280
14.2 Spring 容器	280
14.2.1 Bean 的定义	280
14.2.2 Bean 的基本属性	280
14.2.3 Bean 的生命周期	283
14.2.4 Bean 的管理	288
14.2.5 Bean 的依赖关系	290
14.2.6 Bean 的自动装配	292
14.3 Bean 中对集合的注入	295
14.3.1 对 list 的注入	295
14.3.2 对 Set 的注入	297
14.3.3 对 Map 的注入	298
14.4 两种后处理器	299
14.4.1 Bean 后处理器	299
14.4.2 容器后处理器	301
习题 14	303
第 15 章 Spring 的 AOP 及事务支持	304
15.1 Spring 的 AOP	304
15.1.1 代理机制	304
15.1.2 AOP 的术语与概念	308
15.1.3 Spring 1.x 的 AOP 支持	310
15.1.4 Spring 2.x 的 AOP 支持	320
15.2 Spring 的事务支持	329
15.2.1 使用 TransactionProxyFactoryBean 生成事务代理	331
15.2.2 利用继承简化配置	332
15.2.3 用 BeanNameAutoProxyCreator 自动创建事务代理	333
15.2.4 用 DefaultAdvisorAutoProxyCreator 自动创建事务代理	335
习题 15	337
第 16 章 Spring 的其他功能	338
16.1 Spring 的 Web 框架	338

16.1.1	第一个 Spring 的 MVC 应用	338
16.1.2	Spring MVC 的控制器	343
16.1.3	Spring MVC 的分发器	348
16.1.4	数据绑定	349
16.1.5	Validator 应用	354
16.2	定时器的应用	357
16.2.1	使用程序启动 Timer 定时器	357
16.2.2	使用 Web 监听方式	358
16.2.3	Spring 定制定时器	359
习题 16	360
第 17 章	Struts 2、Hibernate、Spring 的整合应用实例	361
17.1	Spring 与 Struts 2 的整合	361
17.2	Spring 与 Hibernate 整合	365
17.3	Struts 2、Hibernate 与 Spring 的整合	374
17.3.1	项目创建	375
17.3.2	添加项目开发所需的 JARs	376
17.3.3	实现 Hibernate 持久层	377
17.3.4	编写 DAO 接口及实现类	378
17.3.5	编写业务逻辑接口及实现类	382
17.3.6	“登录”功能的实现	388
17.3.7	“查询已借图书”功能的实现	390
17.3.8	“借书”功能的实现	393
17.3.9	“图书管理”功能的实现	395
习题 17	401

第 2 部分 实验指导

实验 1	Struts 2 基础应用	402
实验目的	402
实验内容	402
思考与练习	402
实验 2	Struts 2 综合应用	403
实验目的	403
实验内容	403
思考与练习	404
实验 3	Hibernate 基础应用	404
实验目的	404
实验内容	404

思考与练习·····	404
实验 4 Hibernate 与 Struts 2 整合应用·····	405
实验目的·····	405
实验内容·····	405
思考与练习·····	405
实验 5 Spring 基础应用·····	405
实验目的·····	405
实验内容·····	405
思考与练习·····	405
实验 6 Spring AOP 应用·····	406
实验目的·····	406
实验内容·····	406
思考与练习·····	406
实验 7 Spring 与 Struts 2 整合应用·····	406
实验目的·····	406
实验内容·····	406
思考与练习·····	406
实验 8 Spring 与 Hibernate 整合应用·····	406
实验目的·····	406
实验内容·····	407
思考与练习·····	407
实验 9 Struts 2、Hibernate、Spring 的整合应用·····	407
实验目的·····	407
实验内容·····	407
思考与练习·····	407

第 3 部分 综合应用实习

实习 1 学生成绩管理系统·····	408
P1.1 项目创建及功能描述·····	408
P1.2 层次划分·····	409
P1.3 添加 SSH 开发 JARs·····	410
P1.4 实现 Hibernate 持久层·····	410
P1.5 实现 DAO·····	418
P1.6 实现业务逻辑层·····	425
P1.7 实现 Web 层·····	433
P1.7.1 web.xml 及 struts.properties 配置文件·····	433
P1.7.2 分页实现·····	434
P1.7.3 系统主界面·····	434

P1.7.4	“登录”功能实现	437
P1.7.5	“学生信息管理”功能实现	442
P1.7.6	“学生成绩信息管理”功能实现	456
附录 A	SQL Server 2005 图书管理系统数据库	465
A.1	登录表	465
A.2	读者信息表	465
A.3	图书信息表	465
A.4	借阅信息表	466

第 1 部分 实用教程

第 1 章 Java EE 概述

1996 年, Sun Microsystems 公司推出了一种新的纯面向对象的编程语言, 命名为 Java。根据不同的应用领域将 Java 语言划分为三大平台:

- **Java Platform Micro Edition** 简称 Java ME, 即 Java 平台微型版。主要用于开发掌上电脑、手机等移动设备使用的嵌入式系统。
- **Java Platform Standard Edition** 简称 Java SE, 即 Java 平台标准版。主要用于开发一般台式机应用程序。
- **Java Platform Enterprise Edition** 简称 Java EE, 即 Java 平台企业版。主要用于快速设计、开发、部署和管理企业级的软件系统。

本书将着重介绍 Java EE 的实际应用。

1.1 Java EE 应用概述

Java EE 是目前开发 Web 应用(特别是企业级 Web 应用)最流行的平台之一, 其实 Java Web 应用的核心技术就是 JSP、Servlet 与 JavaBean。但遗憾的是一直没有一套有效的开发规范来约束这些 JSP 程序员, 导致不同程序员写出不同风格的 JSP 程序。项目的规模越大, 这种弊端就越明显, 项目的可维护性就越差, 从而大大增加了项目管理的难度。这时, 一个基于 Web 框架开发的 Web 应用的年代已经来临了。继 Struts 的成功推行之后, 大批优秀的框架也紧随其后, 进入了整个市场。程序员根据不同框架的组合, 可以方便地开发出优秀的、利于维护和扩展的程序。本书将分别介绍目前流行的组合框架 Struts 2、Hibernate 3 和 Spring 2, 从基础到深入, 让读者能够在没有任何这方面基础的前提下由浅入深地了解每一个框架, 然后再介绍它们之间的整合应用, 达到开发效果。

1.1.1 Struts 2 框架

早期的 Web 应用全部是静态的 HTML 页面, 用于将一些个人信息呈现给浏览者。随着开发技术的发展, 有不少新技术问世。Model 1 模式就是其中之一, 所谓 Model 1 就是 JSP 大行其道的时代, 整个 Web 应用几乎全部都是由 JSP 页面组成的。将控制逻辑和显示逻辑混合在一起, 导致代码的重用性非常低, 而且还不利于维护与扩展。开发人员看出这种开发模式不是长久之计, 便提出了 MVC 的思想, MVC 即 Model (模型)、View (视图)、Controller (控制器)。视图层负责页面的显示工作, 而控制层负责处理及跳转工作, 模型层负责数据的存取, 这样它们的耦合性就大大降低了, 从而提高了应用的可扩展性及维护性。虽然如此, 程序员还是有很多工作要做, 而且代码的书写也没有一定的规范性, 不同的程序员可以写出不同的代

码，这种项目对于扩展及维护是非常不方便的。

MVC 模式的提出虽然改变了程序的设计思路，但代码的规范性还是很差，而 Struts 框架则具有组件的模块化、灵活性和重用性的优点，同时也简化了基于 MVC 的 Web 应用程序的开发。从应用的角度来说，Struts 有三大块，分别是 Struts 核心类、Struts 配置文件及 Struts 标签库。

由此可见，Struts 本身就实现了 MVC 模式，就 Struts 的发展来说，由于 Struts 1 框架的局限性，一个基于 Struts 1 及 WebWork 框架提炼出来的 Struts 2 框架已经逐步成为程序员们关注的焦点。Struts 2 框架有着 Struts 1 的市场及 WebWork 的精湛技术，将越来越受欢迎。故本书将重点讲述 Struts 2。

1.1.2 Hibernate 框架

传统的 Java 应用都是采用 JDBC 来访问数据库，它是一种基于 SQL 的操作方式，但对目前的 Java EE 信息化系统而言，通常采用面向对象分析和面向对象设计的过程。系统从需求分析到系统设计都是按面向对象方式进行，但是到详细的数据访问设计阶段，又回到了传统的 JDBC 访问数据库的老路上来，这是非常令人沮丧的。

人们的智慧是无穷的，遇到问题总会想办法解决它，而不是与之妥协或绕道而行。Hibernate 的问世解决了这个问题，Hibernate 是一个面向 Java 环境的对象/关系映射工具，它用来把对象模型表示的对象映射到基于 SQL 的关系数据模型中去，这样就不用再为怎样用面向对象的方法进行数据的持久化而大伤脑筋了。

1.1.3 Spring 框架

Spring 框架是 Rod Johnson 开发的，2003 年发布了 Spring 框架的第一个版本。Spring 是一个从实际开发中抽取出来的框架，因此它完成了大量开发中的通用步骤，从而大大提高了企业应用的开发效率。

Spring 为企业应用的开发提供了一个轻量级的解决方案。其中依赖注入、基于 AOP 的声明式事务管理、多种持久层的整合与优秀的 Web MVC 框架等最为人们关注。Spring 可以贯穿程序的各个层之间，但它并不是想取代那些已有的框架，而是以高度的开放性和它们紧密的整合。这也是 Spring 被广泛应用的原因之一。

上面简单介绍了这三种框架，读者只需有个印象就可以了，后面会分别对这三个流行框架进行一一讲解。

1.2 Java EE 开发环境的构建

工欲善其事，必先利其器。在开发 Java Web 项目之前，必须先准备好相应的开发环境及工具，包括 JDK、Tomcat 及 MyEclipse 开发工具。

1.2.1 JDK 和 Tomcat 的安装与配置

1. JDK 的安装与配置

安装 Java 开发包（Java 2 Software Development Kit，简称 JDK）是 Java 软件开发的前提，

其目前最高版本为 JDK 1.6, 但 JDK 1.5 仍是目前最常用的版本, 所以本书中所有范例都是基于 JDK 1.5 开发的。软件可以在 Sun 的官方网站上下载, 其安装过程这里就不再赘述了, 本书安装的目录是 “D:\Program Files\Java\jdk1.5”。

通过设置系统环境变量, 告知 Windows 操作系统 JDK 1.5 版本的安装位置。下面具体介绍设置系统环境变量的方法。

(1) 设置系统变量 JAVA_HOME。右击桌面 “我的电脑” 图标, 依次选择 “属性” → “高级” → “环境变量”, 弹出 “环境变量” 对话框, 如图 1.1 所示。在系统变量中单击 “新建” 按钮, 弹出 “新建系统变量” 对话框, 在 “变量名” 文本框中输入 “JAVA_HOME”, “变量值” 文本框中输入 JDK 的安装路径 “D:\Program Files\Java\jdk1.5”, 如图 1.2 所示, 单击 “确定” 按钮完成配置。

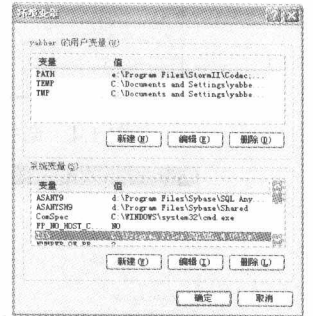


图 1.1 “环境变量” 对话框

(2) 设置系统变量 Path。选择 “属性” → “高级” → “环境变量”, 在 “系统变量” 中找到变量为 Path 的变量, 单击 “编辑”, 在前面输入 JDK 到 bin 的目录 “D:\Program Files\Java\jdk1.5\bin;”, 如图 1.3 所示, 单击 “确定” 按钮完成配置。

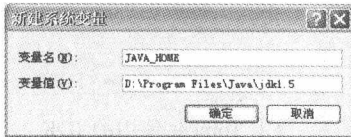


图 1.2 “新建系统变量” 对话框

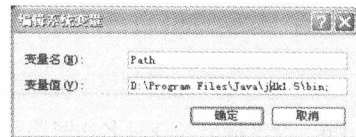


图 1.3 “编辑系统变量” 对话框

(3) 设置 CLASSPATH。与 (1) 同样的操作, 不同的是变量名为 “CLASSPATH”, 变量值为 “.;D:\Program Files\Java\jdk1.5\lib\dt.jar; D:\Program Files\Java\jdk1.5\lib\tools.jar”。

到此, JDK 的安装与配置就完成了, 读者可以自己测试是否配置成功。单击 “开始” → “运行”, 输入 “cmd”, 进入 DOS 界面。在命令行输入 “java -version”, 如果配置成功就会出现 Java 的版本信息。

2. Tomcat 的安装

Tomcat 是一个免费的开源的 Servlet 容器, 它是 Apache 基金会在 Jakarta 项目中的一个核心项目。本书采用 Tomcat 6.0 版本作为 Web 服务器。Tomcat 6.0 可以在官方网站 <http://tomcat.apache.org> 下载。

Tomcat 的具体安装过程如下所述。

(1) 运行 “apache-tomcat-6.0.14.exe” 安装程序, 开始安装。

(2) 选择安装内容及安装路径 (本书的安装路径是 “E:\Program Files\Apache Software Foundation\Tomcat 6.0”), 如图 1.4 与图 1.5 所示。可以把安装内容的 “Example” 选中, 作为开发的参考例子。

(3) 设定连接端口 (HTTP/1.1 Connector Port)、登录名 (User Name) 和密码 (Password), 这里, 采用系统默认值。连接端口为 “8080”, 登录名为 “admin”, 密码为空, 如图 1.6 所示。

(4) 设定 Tomcat 使用的 JDK 路径 “D:\Program Files\Java\jdk1.5”, 如图 1.7 所示。

(5) 安装完成并启动 Tomcat, 如图 1.8 所示。

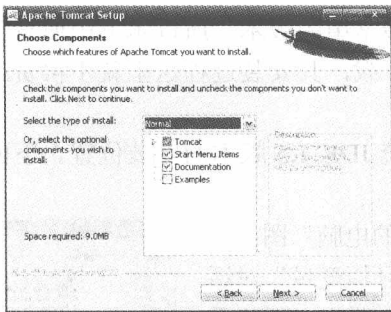


图 1.4 选择安装内容

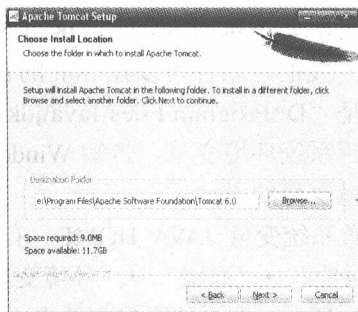


图 1.5 选择安装路径

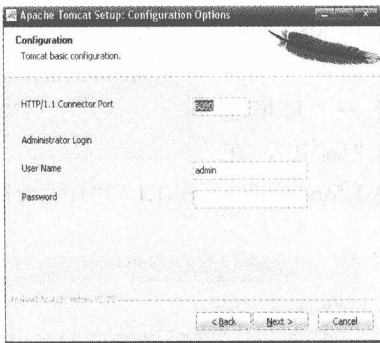


图 1.6 设定端口、登录名和密码

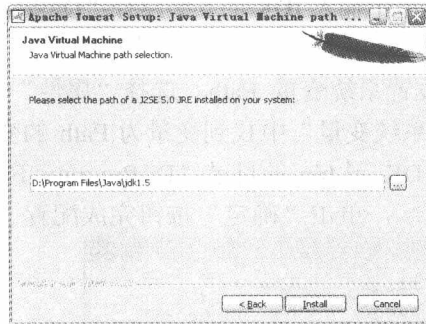


图 1.7 设定 Tomcat 使用的 JDK

(6) 测试是否安装成功。打开浏览器，输入“http://localhost:8080”，如果出现如图 1.9 所示界面，表明安装成功。



图 1.8 完成安装并启动 Tomcat

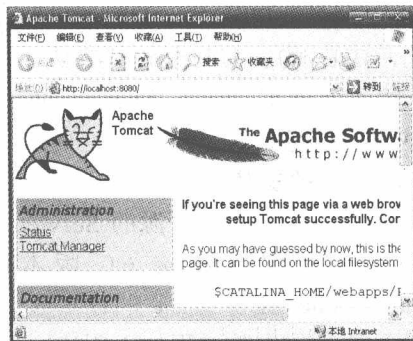


图 1.9 测试成功

关于 Tomcat 的配置和 JDK 差不多，主要就是设置“TOMCAT_HOME”环境变量。右键单击“我的电脑”→“属性”→“高级”→“环境变量”，打开环境变量对话框，在用户变量中“新建”→“环境变量”，在“变量名”中输入“TOMCAT_HOME”，“变量值”中输入 Tomcat 的安装路径，本书为“E:\Program Files\Apache Software Foundation\Tomcat 6.0”。

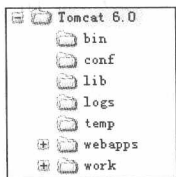


图 1.10 Tomcat 目录结构

Tomcat 安装和配置到这里就完成了，下面简要介绍其目录结构（见图 1.10）及每个文件夹的作用。

bin: 存放 Tomcat 启动和关闭文件的文件夹。

conf: 存放一些 Tomcat 的配置文件。

lib: 存放一些 Tomcat 的核心类库，在应用中如果需要扩展

Tomcat 的功能，也可以将第三方类库赋值到该文件夹下。

logs: 该文件夹刚产生时是空的, 里面主要存放 Tomcat 每次运行后产生的日志。

temp: 该文件夹存放 Web 应用运行过程中生成的临时文件。

webapps: 该文件夹用于存放部署的 Web 应用, 只要将 Web 应用复制到该文件夹下, Tomcat 就会将该应用自动部署到容器中。

work: 该文件夹存放 Web 应用与运行时编译生成的 class 文件。JSP 文件在运行过程中会先被翻译成 Java 文件, 然后再被编译成 class 文件, 被翻译的 Java 文件也被存放在这里。

在前面测试 Tomcat 是否安装成功的时候, 在浏览器中输入了这样的网址“http://localhost:8080/”, 这里的“8080”是端口号, 在安装过程中应用的是默认的端口号, 如图 1.6 所示, 其实端口号是可以改动的, 在 conf 文件夹下找到 server.xml 文件, 找到这样的代码:

```
<Connector port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

其中, “port=8080”就是 Tomcat 的 Web 服务端口, 可以任意改变这个值, 保存后启动 Tomcat, 输入“http://localhost:自定义端口号”, 也会出现如图 1.9 所示的界面。

在图 1.9 所示的界面的左上角有“Tomcat Manager”链接, 这是进入 Manager 控制台的路径, 单击该链接, 会出现如图 1.11 所示的界面, 让用户输入用户名和密码。

在 webapps 文件夹下, 有 Manager 项目, 打开进入 WEB-INF 文件夹, 里面有个“web.xml”文件, 打开后在文件最后有下面的代码段:

```
<security-role>
  <description>
    The role that is required to log in to the Manager Application
  </description>
  <role-name>manager</role-name>
</security-role>
```

说明要想登录 Manager 控制台, 必须使用 Manager 角色。

打开 conf 文件夹下的 tomcat-users.xml 文件, 该文件内容如下:

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
</tomcat-users>
```

有的里面可能已经设置了登录名和密码, 有的没有设置, 不管有没有设置, 在这里用户都可以自己设定, 例如设定如下:

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
```

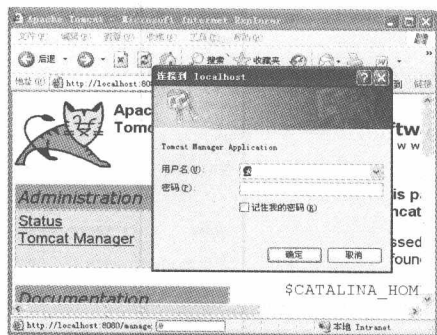


图 1.11 进入 Manager 控制台界面