

Java 开源项目

Spring+Hibernate+Struts

项目开发详解

覃庆炎 张伟 等编著
冼进 审



附
光
盘

各章实例完整源代码和配置文件



清华大学出版社

Java 开源项目

Spring+Hibernate+Struts

项目开发详解

覃庆炎 张伟 等编著
冼进 审



清华大学出版社
北 京

内 容 简 介

本书以 Java 开源项目为理念,以 Spring+Hibernate+Struts 开源项目为基础,由浅入深地讲解了轻量级 J2EE 系统构架技术的原理和开发过程。本书共分为 4 篇 14 章,分别以 Hibernate、Struts、Spring 开源项目为基础详细地介绍其开发原理和开发过程,基础实例篇讲解了新闻发布系统、在线拍卖系统和博客系统 3 个实用系统的开发过程,其他部分介绍了开源搜索引擎 Lucene、模板引擎 Freemarker、安全框架 Acegi、上传组件 Commons FileUpload 等内容,并介绍了科研资料信息管理系统的开发过程。

本书结构脉络清晰,代码实例丰富且针对性强,适合缺乏项目经验的 Java 初、中级读者学习使用,也可作为 IT 培训班的教材。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

Java 开源项目——Spring+Hibernate+Struts 项目开发详解 / 覃庆炎等编著. —北京:清华大学出版社, 2008.7

ISBN 978-7-302-17321-2

I. J… II. 覃… III. Java 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2008)第 049204 号

责任编辑:夏兆彦 王冰飞

责任校对:徐俊伟

责任印制:王秀菊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

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

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京市清华园胶印厂

装 订 者:三河市金元印装有限公司

经 销:全国新华书店

开 本:185×260 印 张:27 字 数:653 千字

附光盘 1 张

版 次:2008 年 7 月第 1 版

印 次:2008 年 7 月第 1 次印刷

印 数:1~4000

定 价:49.80 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770177 转 3103 产品编号:025059-01

FOREWORD

前言

本书向广大读者介绍全面、系统而深入的轻量级 J2EE 构架技术及其相关的应用知识，具有如下特点。

主流技术解析：本书以 Java 开源项目为理念，选择目前行业内前沿的轻量级 J2EE 架构技术为介绍对象，顺应了行业的技术发展趋势，将广大读者带到崭新的技术前沿，迎合了广大读者的学习需求。

代码实例分析：本书以理论概念为辅，强调培养读者的实际应用能力，在本书的各个章节中都通过代码实例来逐步阐述构架技术的核心，减轻读者的理论负担，让读者通过实例来掌握本书各章节的核心。需要特别指出的是，读者通过对本书实例的学习，学到的不仅仅是某一段代码的重用，而是一个通用开发框架的重用，本书在编写的思想上是一个巨大的创新。

结构主次分明：本书层次分明，由浅入深，共分为 4 篇，分别为基础应用篇、基础实例篇、高级应用篇、高级实例篇。以培养读者的基础能力为起点，让读者在学习了每一章之后都能掌握实际的构架能力，通过每一章节的积累，最终将读者培养成全面的熟练的构架人才。

讲解通俗易懂：本书的讲解符合广大读者的学习习惯，使用通俗的语言，摒弃高深的理论词汇。在各章节中均以丰富的图表说明为主，减少繁琐的文字叙述，让广大读者轻松、形象地学习。

本书体系完整，共分为 4 篇。

第 1 篇为基础应用篇，包括第 1 章～第 6 章。本篇主要介绍了 Hibernate、Struts 以及 Spring 等相关的知识点，并对每一个知识点都引入一个快速入门的例子，对于学习一门新的技术体系，本书认为例子往往要比纯粹的理论直接，对初学者的帮助更大。

第 2 篇为基础实例篇，包括第 7 章～第 9 章。在本篇中本书引入了新闻发布系统、在线拍卖系统和博客系统等开发实例，这 3 个例子都是先从系统构架的角度开始阐述如何构建 Web 应用程序的框架，接着用一个应用实例来详细地阐述如何在该框架上进行快速的开发。读者通过这样一个过程，不管学习的是什么例子，都能够从本书所提供的例子中学到构架的思想以及详细的构架过程，也就是说读者可以从框架层次上面对例子进行复用，而不是低级的代码级的复用。

第 3 篇为高级应用篇，包括第 10 章～第 13 章，本篇将会把读者提高到一个更高的水平。本篇主要介绍开源搜索引擎 Lucene、模板引擎 Freemarker、安全框架 Acegi、上传组件 Commons FileUpload 的集成。同样，在本篇中，为了使读者能够快速入门，采用了每讲解一个知识点就详细介绍一个快速入门例子。

第4篇为高级实例篇，包括第14章。在本篇中，也安排了一个代码实例，这个例子的不同寻常之处是加入了一些前面所介绍的高级特性，这个例子不仅仅是代码重用，重要的是这个例子所采用的框架本身就是一个成熟的 Web 应用的解决方案。

本书在编写过程中引用了成都理工大学李申申的硕士论文，在此表示感谢。同时，还引用了其他材料，在此向原作者表示深切的感谢。

本书由覃庆炎、张伟编著，洗进审校。同时，参与本书编写的人员还有冯强、曾德惠、许庆华、程亮、周聪、黄志平、胡松、邢永峰、邵军、边海龙、刘达因、赵婷、马鸿娟、侯桐、赵光明、李胜、李辉、侯杰、王红研、王磊、闫守红、康涌泉、李欢、蒋杼倩、王小东、张森、张正亮、宋利梅、何群芬、程瑶等，在此一并表示感谢。

为充分展现本书的编写特点，帮助读者深刻理解本书编写意图与内涵，进一步提高对本书教学的使用效率，我们建立了本书使用指导联络方式，E-mail 是 china_54@tom.com。

由于编写时间仓促，书中疏漏之处在所难免，欢迎广大读者和同行批评指正。

编 者

2008 年 2 月

CONTENTS

目录

第一篇 基础应用篇

第 1 章 轻量级 J2EE 构架技术	2
1.1 Java Web 应用开发基础知识	2
1.1.1 部署 Web 应用程序	2
1.1.2 组织 Web 应用程序源代码	5
1.1.3 Web 应用程序开发流程	7
1.2 轻量级开发技术	8
1.2.1 轻量级开发概念	8
1.2.2 轻量级容器	8
1.3 轻量级 J2EE 构架技术	9
1.3.1 业务逻辑层	9
1.3.2 数据持久层	10
1.3.3 视图表现层	11
1.4 基于 Java 开源项目构架轻量级 J2EE Web 应用	12
1.4.1 Java 开源 Web 框架 Struts	12
1.4.2 Java 开源持久层框架 Hibernate	13
1.4.3 Java 开源 J2EE 框架 Spring	14
1.4.4 Spring+Hibernate+Struts 整合构架轻量级 J2EE Web 应用	14
1.5 本章小结	16
第 2 章 搭建开发平台	17
2.1 配置开发环境	17
2.1.1 下载和安装 JDK	17
2.1.2 下载和安装开源应用服务器 Tomcat	19
2.1.3 下载和安装开源编译工具 Ant	21
2.1.4 下载和安装开源数据库 MySQL	21
2.1.5 配置环境变量	22
2.2 安装开发工具	23
2.2.1 下载和安装开源开发工具 Eclipse	24
2.2.2 下载和安装 Eclipse 语言包	25
2.3 Ant 下常用的 Web 项目构建脚本	26
2.3.1 Ant 下的 Web 应用服务器控制	27
2.3.2 Ant 下的 Web 项目构建与部署	28
2.3.3 Ant 下的 Web 项目管理与监控	29
2.4 开发一个简单的 Java Web 测试实例	30
2.4.1 创建 Java Web 工程 HelloWorld	31

2.4.2	编辑 HelloWorld 源代码	31
2.4.3	部署 HelloWorld	33
2.4.4	运行 HelloWorld	33
2.5	本章小结	34

IV

第3章 Spring+Hibernate+Struts

整合应用入门实例：注册

登录系统

3.1	概述	35
3.2	下载 Spring、Struts、Hibernate 以及 相应的 JAR 包	36
3.2.1	配置运行环境	36
3.2.2	配置开发环境	36
3.3	注册登录系统的目录结构	39
3.4	创建实体对象 mode	40
3.4.1	基础业务对象 BaseObject	40
3.4.2	创建实体对象	40
3.5	创建 DAO 类	43
3.5.1	创建 DAO 基类	44
3.5.2	创建实体对象 User 和 Role 相应的 DAO 文件	45
3.5.3	实现实体对象 User 和 Role 相应的 DAO 文件的接口	46
3.5.4	配置 applicationContext-hibernate .xml 文件	48
3.5.5	为 DAO 编写单元测试代码 DaoTest	49
3.6	创建 Managers	52
3.6.1	创建基础逻辑处理对象 Manager 接口	53
3.6.2	创建实体对象 User 和 Role 的 Managers 接口	54
3.6.3	实现实体对象 User 和 Role 的 Managers 接口	55
3.6.4	配置 applicationContext-service .xml 文件	56
3.6.5	为 Managers 编写单元测试代码 ManagerTest	56
3.7	新用户注册	58

3.7.1	创建 UserForm、RoleForm 以及 AddressForm	58
3.7.2	创建 SignupAction 实现用户 注册	60
3.7.3	为 SignupAction 编写单元测试 代码 SignupActionTest	62
3.8	用户登录	63
3.8.1	创建 UserAction 实现用户登录	63
3.8.2	为 UserAction 编写单元测试 代码 UserActionTest	69
3.9	部署运行注册登录系统	71
3.9.1	部署 registration.war	71
3.9.2	配置数据库	71
3.9.3	运行注册登录系统	72
3.10	本章小结	74

第4章 数据持久层技术 Hibernate

4.1	Hibernate 框架概述	75
4.1.1	概述	75
4.1.2	核心接口	76
4.2	Hibernate 开发环境配置	78
4.2.1	下载 Hibernate	78
4.2.2	下载和安装 Hibernate 开发插件	78
4.2.3	创建由 Spring 控制的数据源	79
4.2.4	创建 SessionFactory	79
4.2.5	配置 Hibernate 的基本属性	80
4.3	Hibernate 对象持久化	82
4.3.1	POJO 概述	82
4.3.2	构建数据持久层的基类	84
4.3.3	实现 equals() 和 hashCode() 方法	84
4.4	Hibernate 数据映射	84
4.4.1	基本的对象关系映射	84
4.4.2	一对多映射	87
4.4.3	多对一映射	87
4.5	数据库操作的实现	88

4.5.1	编写 CRUD (创建、检索、 更改、删除) 操作的 DAO (数据 进入层) 接口	88
4.5.2	编写 applicationContext-hibernate. xml 文件	91
4.6	本章小结	92
第 5 章 表现层技术 Struts 93		
5.1	Struts 框架概述	93
5.1.1	概述	93
5.1.2	Struts 的体系结构与工作原理	94
5.2	Struts 框架入门	95
5.2.1	在 Eclipse 中建立 J2EE Web 应用程序目录结构	96
5.2.2	编写视图	98
5.2.3	编写 ActionForm Bean	98
5.2.4	编写 Action 类	99
5.2.5	配置 struts-config.xml 文件	100
5.2.6	调试运行 strutsSample	101
5.3	ActionForm 和 DynaActionForm	103
5.3.1	创建 ActionForm	103
5.3.2	使用 DynaActionForm	105
5.3.3	注册登录系统中创建 UserForm	106
5.4	Struts Validator 验证框架	108
5.4.1	Validator 框架入门	108
5.4.2	Validator 验证规则	110
5.4.3	为 DynaActionForm 创建 验证规则	112
5.5	Actions	113
5.5.1	创建 Action	113
5.5.2	在注册登录系统中创建 SignupAction	114
5.6	Struts 常用标签库	117
5.6.1	JSP 的局限性	117
5.6.2	Struts 标签库	117
5.6.3	HTML 标签库	118
5.6.4	Bean 标签库	119
5.6.5	logic 标签库	122

5.7	在注册登录系统中集成 Struts	124
5.7.1	配置 web.xml	124
5.7.2	配置 struts-config.xml 文件	126
5.8	本章小结	127
第 6 章 业务逻辑层技术 Spring 128		
6.1	Spring 概述	128
6.1.1	Spring 的发展历史	128
6.1.2	Spring 的模块结构	129
6.1.3	Spring 使 J2EE 开发变得 更容易	130
6.2	BeanFactory 的工作原理	130
6.2.1	控制反转	130
6.2.2	BeanFactory	131
6.2.3	Bean 的生命周期	133
6.2.4	ApplicationContext	136
6.3	Spring 集成 Hibernate	139
6.3.1	概述	139
6.3.2	管理 Hibernate 资源	139
6.3.3	在 ApplicationContext 中的 Bean 声明	140
6.3.4	通过 HibernateTemplate 和 HibernateDaoSupport 访问 Hibernate	141
6.3.5	Spring 中配置 Hibernate	142
6.4	Spring 集成 Struts	143
6.4.1	注册 Spring plug-in	143
6.4.2	实现 Spring-aware- Strutsactions	143
6.4.3	代理 Action	144
6.5	AOP	144
6.5.1	AOP 的概念	144
6.5.2	创建 Advice	146
6.5.3	定义 PointCuts	148
6.5.4	使用 ProxyFactoryBean 创建 AOP 代理	149
6.5.5	使用 Autoproxying 创建 AOP 代理	152
6.6	声明式事务处理	155

第二篇 基础实例篇

6.6.1 事务处理的概念	155
6.6.2 Spring 中的程式事务处理	155
6.6.3 声明式事务处理	156
6.7 本章小结	156

第 7 章 新闻发布系统

7.1 系统功能概述	158
7.2 系统需求分析	158
7.2.1 系统功能模块划分	158
7.2.2 数据库逻辑结构设计	159
7.3 系统框架搭建	160
7.3.1 模型层的实现	161
7.3.2 数据持久层的实现	161
7.3.3 业务逻辑层的实现	161
7.3.4 视图表现层的实现	162
7.4 用户管理模块	166
7.4.1 数据操作实现	166
7.4.2 业务逻辑实现	169
7.4.3 表现层的实现	171
7.5 新闻信息管理模块	178
7.5.1 数据操作实现	178
7.5.2 业务逻辑实现	181
7.5.3 表现层的实现	183
7.5.4 配置 struts-config.xml	187
7.6 权限管理模块	188
7.6.1 数据操作实现	188
7.6.2 业务逻辑实现	195
7.6.3 表现层的实现	197
7.7 本章小结	205

第 8 章 在线拍卖系统

8.1 系统功能概述	206
8.2 系统需求分析	207
8.2.1 系统功能模块划分	207
8.2.2 数据库逻辑结构设计	207
8.3 系统框架搭建	208
8.3.1 建立工程目录结构	209

8.3.2 数据持久层的实现	210
8.3.3 业务逻辑层的实现	212
8.3.4 视图表现层的实现	212
8.4 注册登录模块	214
8.4.1 数据操作实现	214
8.4.2 业务逻辑实现	218
8.4.3 表现层的实现	220
8.5 拍卖商品模块	226
8.5.1 数据操作实现	227
8.5.2 业务逻辑实现	232
8.5.3 表现层的实现	235
8.6 实名认证模块	240
8.6.1 数据操作实现	240
8.6.2 业务逻辑实现	245
8.6.3 表现层的实现	246
8.7 本章小结	250

第 9 章 博客系统

9.1 系统功能概述	251
9.2 系统需求分析	251
9.2.1 系统功能模块划分	252
9.2.2 数据库逻辑结构设计	252
9.3 系统框架搭建	254
9.3.1 建立工程目录结构	254
9.3.2 数据持久层的实现	255
9.3.3 业务逻辑层的实现	257
9.3.4 视图表现层的实现	259
9.3.5 Action 基类	259
9.3.6 页面视图	260
9.4 注册登录模块	261
9.4.1 数据操作实现	261
9.4.2 业务逻辑实现	265
9.4.3 表现层的实现	266
9.4.4 部署运行	273
9.5 发布/管理日志模块	274
9.5.1 数据操作实现	274
9.5.2 业务逻辑实现	278
9.5.3 表现层的实现	280

9.5.4 部署运行	284
9.6 留言板模块	285
9.6.1 数据操作实现	285
9.6.2 业务逻辑实现	288
9.6.3 表现层的实现	290
9.6.4 部署运行	292
9.7 本章小结	293

第三篇 高级应用篇

第 10 章 开源搜索引擎 Lucene

10.1 Lucene 简介	296
10.1.1 概述	296
10.1.2 Lucene 全文检索的实现机制	298
10.1.3 Lucene 框架剖析	299
10.1.4 建立索引	300
10.1.5 搜索文档	303
10.2 下载并测试 Lucene	304
10.2.1 下载 Lucene	304
10.2.2 配置 Lucene	304
10.2.3 测试 Lucene	304
10.3 在 Web 应用程序中集成 Lucene	305
10.3.1 实例需求分析	305
10.3.2 建立 Dynamic Web Project	306
10.3.3 设计 Web 工程体系结构	306
10.3.4 实现用户接口	307
10.3.5 实现控制器	308
10.3.6 实现搜索	308
10.3.7 实现索引	310
10.4 部署并测试 lucenesample	311
10.5 本章小结	313

第 11 章 Spring 构架中模板引擎

集成

11.1 Freemarker 模板引擎简介	314
11.2 Freemarker 脚本语言简介	316
11.2.1 模板	316

11.2.2 数据模型	318
11.3 搭建 Freemarker 开发环境	321
11.3.1 下载 Freemarker	321
11.3.2 下载和安装 Freemarker 开发插件	322
11.4 Freemarker 开发实例	322
11.4.1 实例需求分析	322
11.4.2 建立工程——Freemarker-Sample	323
11.4.3 创建 Freemarker 宏模板	323
11.4.4 客户信息录入页面	324
11.4.5 客户信息添加成功页面	328
11.4.6 客户信息列表页面	329
11.5 本章小结	330

第 12 章 Spring 集成 Acegi

安全框架

12.1 Acegi 安全框架概述 ^[8]	331
12.2 Acegi 安全框架体系结构 ^[8]	332
12.2.1 Acegi 安全框架组件	332
12.2.2 控制反转	333
12.2.3 XML 配置文件	333
12.3 安全过滤器 ^[8]	333
12.3.1 Session Integration Filter	333
12.3.2 Authentication Processing Filter	334
12.3.3 Logout Processing Filter	335
12.3.4 Exception Translation Filter	335
12.3.5 Interceptor Filter	336
12.4 在 Web 应用程序中集成 Acegi 安全框架	337
12.4.1 建立一个新的工程	337
12.4.2 设计 Web 工程目录结构	338
12.4.3 配置 Acegi 安全框架	338
12.5 配置 acegisecurity-config.xml 文件	339
12.6 部署并运行 acegisample	345
12.7 本章小结	346

第 13 章 Spring 构架中上传组件

集成.....347

13.1 Commons FileUpload 组件简介.....347

13.2 Commons FileUpload 开发流程.....348

13.3 Commons FileUpload 开发环境搭建.....350

13.3.1 下载 Commons FileUpload.....350

13.3.2 配置 Commons FileUpload.....351

13.4 Commons FileUpload 开发实例.....351

13.4.1 实例需求分析.....351

13.4.2 建立工程——FileUpload

Sample352

13.4.3 编写前台页面.....352

13.4.4 编写 Servlet 来负责文件的

上传354

13.4.5 部署和测试.....356

13.5 本章小结.....357

第四篇 高级实例篇

第 14 章 科研资料信息管理系统.....360

14.1 系统功能概述.....360

14.2 系统需求分析.....361

14.2.1 系统功能模块划分.....361

14.2.2 数据库逻辑结构设计^[12].....36114.3 系统各模块效果图^[12].....363

14.4 系统基础框架搭建.....365

14.4.1 数据持久层的实现^[12].....36514.4.2 业务逻辑层的实现^[12].....37014.4.3 视图表现层的实现^[12].....378

14.5 其他高级框架集成.....382

14.5.1 集成 Commons FileUpload

组件.....382

14.5.2 集成 Acegi 安全框架.....382

14.5.3 Freemarker 模板引擎.....384

14.5.4 集成 Struts-Menu 菜单

框架^[12].....38514.5.5 集成 ValueList 标签库^[12].....386

14.5.6 集成 Log4j 日志管理组件.....388

14.6 用户信息管理模块.....389

14.6.1 数据操作实现^[12].....38914.6.2 业务逻辑实现^[12].....39514.6.3 表现层的实现^[12].....398

14.7 用户组信息管理模块.....401

14.7.1 数据操作实现^[12].....40114.7.2 业务逻辑实现^[12].....40514.7.3 表现层的实现^[12].....407

14.8 系统权限管理模块.....408

14.8.1 数据操作实现^[12].....40814.8.2 业务逻辑实现^[12].....41114.8.3 表现层的实现^[12].....413

14.9 文件信息管理模块.....414

14.9.1 数据操作实现^[12].....41414.9.2 业务逻辑实现^[12].....41714.9.3 表现层的实现^[12].....418

14.10 本章小结.....419

参考文献.....420

第一篇

基础应用篇

在 Java 应用开发中，除了使用 Java 语言外，还需要使用一些第三方库，如 Spring、Hibernate 以及 Struts 等。这些库为开发者提供了许多现成的解决方案，使得开发工作更加高效。本篇将介绍这些库的基本用法，帮助读者快速上手。

1.1 Java Web 应用开发基础知识

本章主要介绍 Java Web 应用开发的基础知识，包括 Web 应用的结构、开发流程以及常用的开发工具。通过本章的学习，读者可以了解 Java Web 应用的基本概念，为后续的学习打下基础。

1.1.1 部署 Web 应用程序

在部署 Web 应用之前，需要先了解 Web 应用的结构。Web 应用通常由 HTML、CSS、JavaScript 以及 JSP 等文件组成。这些文件需要打包成一个 WAR 文件，然后部署到 Web 容器（如 Tomcat）中。部署过程中需要注意一些细节，如文件权限、端口配置等。

部署 Web 应用时，首先需要将 WAR 文件复制到 Web 容器的 webapps 目录下。然后，启动 Web 容器，应用会自动部署并运行。如果遇到问题，可以通过查看日志来排查原因。部署完成后，可以通过浏览器访问应用，验证其是否正常运行。

第1章

轻量级 J2EE 构架技术

知识点:

- Java Web 应用开发基础知识
- 轻量级开发技术
- 轻量级 J2EE 构架技术
- 基于 Java 开源项目框架轻量级 J2EE Web 应用

本章导读:

在 Java 应用领域中,有许多优秀、成熟的开源软件,比如 Struts、Hibernate 以及 Spring,利用它们同样可以打造优越、灵巧的 J2EE 应用框架。本章首先讨论 Java Web 应用开发的基础知识,接着简单的介绍基于开源项目 Spring、Hibernate 以及 Struts 整合构建轻量级 J2EE Web 应用。

1.1 Java Web 应用开发基础知识

本节首先讨论 Java Web 应用开发的相关基础知识,以著名开源应用服务器 Tomcat 为 Web 应用程序的 Web 容器来讨论如何组织源代码,如何部署 Web 应用以及简单地介绍开发 Web 应用程序的一般流程。

1.1.1 部署 Web 应用程序

在讨论如何组织 Web 应用程序源代码目录结构之前,有必要先来了解一下 Web 应用程序在运行时的目录结构。对于 Web 应用程序在运行时的目录结构,在 Servlet API Specification 2.2 版本之前,在不同的服务器平台之间并没有统一的规范,但是,在 2.2 版本(或更高)规定,不同的服务器平台必须采用同一个标准的 Web 应用程序结构,下面将会进行详细的讨论。

一个 Web 应用程序通常被定义为有着标准层次结构的目录和文件,部署 Web 应用程序有两种方式,一种方式是采用不打包(Unpacked)的方式,每一个目录和文件以文件系统的形式存在;另一种方式则是采用打包(Packed)方式,通常是一个 WAR 文件。通常情况下,在程序开发的过程中采用前者会比较方便,而在程序发布时一般是采用后者。

1. 标准目录结构

为了方便地以标准的形式发布“可执行的”Web 应用程序，无论是采用不打包的方式，还是采用打包的方式，都必须采用如下标准的目录结构。

- *.html, *.jsp, 等 HTML 和 JSP 页面和其他相关文件（如 JavaScript 文件、层式样式文件、图片等）必须对客户端浏览器透明。在较大型的 Web 应用中，用户可以将这些文件按照其层次结构再放到不同的子目录中，但对于较小型的应用开发，通常简单地将这些文件放在同一个目录下进行管理和维护。
- /WEB-INF/web.xml Web 应用部署描述符。它是一个 XML 文件采用 XML 文件格式编写，用来描述 Servlets 以及构建系统中的其他组件，描述初始化参数以及希望服务器容器提供的容器管理安全约束。
- /WEB-INF/classes/ 这个文件夹存放的是系统应用所需要的 Java 类文件以及与之相关的资源文件，包括非 JAR 的 Servlet 类文件和非 Servlet 类文件。如果用户的类文件是以 Java 包的形式进行组织的，那么必须在 /WEB-INF/classes/ 下建立相关的层次目录结构。例如，一个 Java 类是按照 com.mycompany.mypackage.MyServlet 进行组织的，那么该类文件需要按照 /WEB-INF/classes/com/mycompany/mypackage/MyServlet.class 的目录体系进行存放。
- /WEB-INF/lib/ 这个目录存放的是将 Java 类文件以及相关资源文件打包在一起的 JAR 文件，如第三方类库或者 JDBC 驱动。

下面开发一个很简单的 Web 程序来讲解 Web 应用程序的程序部署目录结构。

(1) 目录结构

本例子是一个最简单的 Web 应用程序部署结构，主要包含一个 /WEB-INF/ 文件夹，该文件夹根目录下包含很重要的文件，Web 应用程序部署描述符 Web.xml 以及两个子文件夹 lib 以及 classes。lib 文件主要保存的是第三方的 JAR 包，classes 文件夹主要存放用户自己定义的一些类。这个小例子的项目名称为 example1，该项目的目录结构如图 1-1、图 1-2 所示。

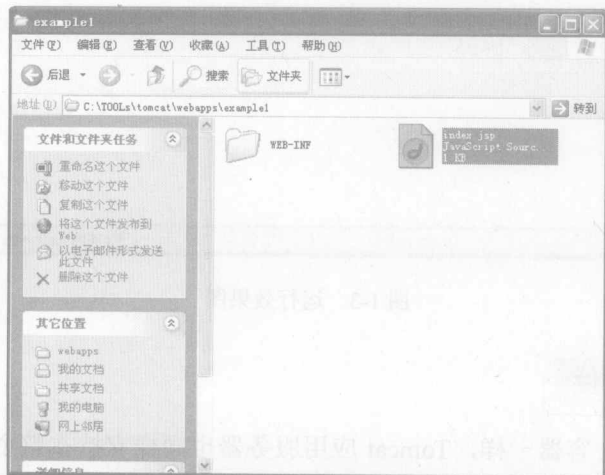


图 1-1 目录结构

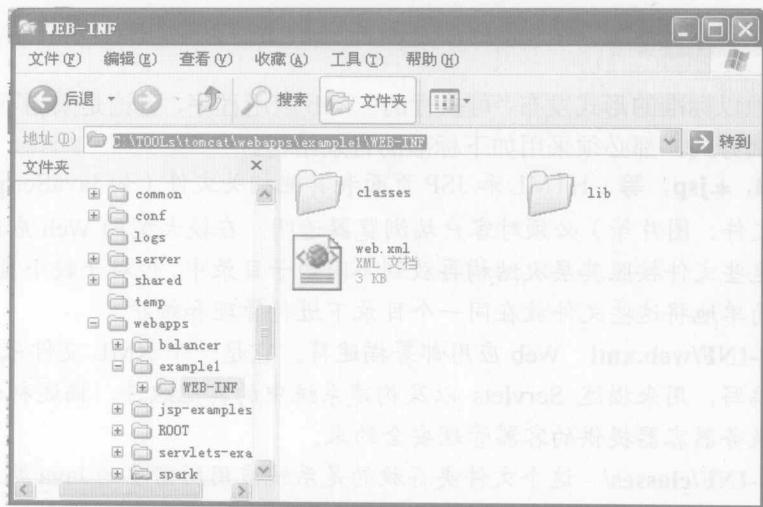


图 1-2 WEB-INF 目录

(2) 部署、运行

Web 应用程序部署相当简单，只需要将 example1 文件直接复制到 tomcat 安装目录下的 webapps 文件夹下，启动 Tomcat，然后在浏览器中输入“<http://localhost:8080/example1/>”。example1 运行效果图如图 1-3 所示。

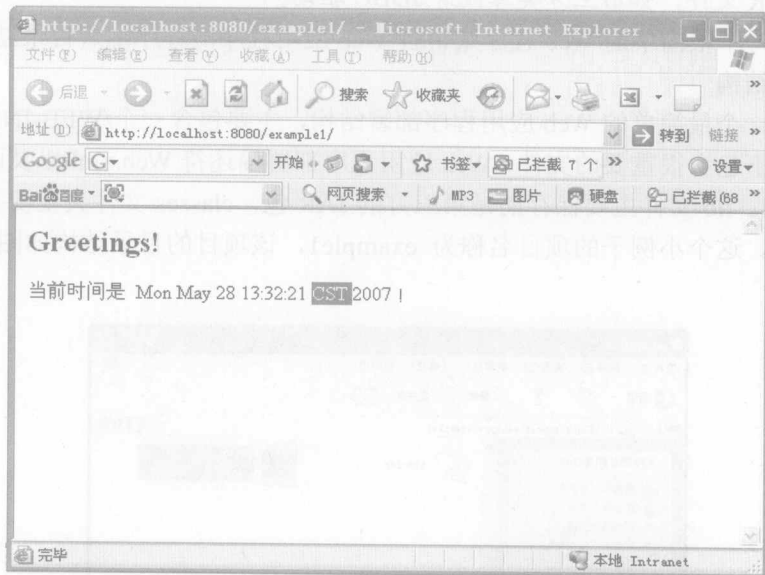


图 1-3 运行效果图

2. 共享库文件

与大多数 Servlet 容器一样，Tomcat 应用服务器也同样支持在整个容器中共享 JAR 库文件（或者未经打包的 classes），这些库文件对所有部署在该服务器的 Web 应用程序都是

可见的而不需要在每一个 Web 应用程序中都包含这些文件。下面介绍在 Tomcat 中如何在整个容器中实现库文件（或者公用文件）的共享。

- ❑ **\$CATALINA_HOME/common/lib** 存放在这里的 JAR 文件对部署在容器中的所有 Web 应用程序以及 Tomcat 内部的代码都是可见的。因此，这个文件夹是存放 JDBC 驱动器的一个好地方，因为一般的 Web 应用程序以及 Tomcat 内部代码都会使用到 JDBC 驱动。
- ❑ **\$CATALINA_HOME/shared/lib** 存放在这里的 JAR 文件对部署在容器中的所有 Web 应用程序都是可见的，但是对 Tomcat 系统代码是不可见的。因此，这是一个在不同 Web 应用程序之间共享库文件的好地方。

3. Web 应用程序部署描述符

在上面已经提到，/WEB-INF/web.xml 文件包含了 Web 应用程序部署描述。从文件的后缀名可以看出，该文件是一个 XML 文件，是 Web 应用程序与应用服务器“交流”的桥梁。

4. 在 Tomcat 应用服务器部署 Web 应用程序

一个 Web 应用程序要被执行，首先必须部署到一个 Servlet 容器里。下面介绍将 Web 应用程序部署到 Tomcat 5.0 的常用方法。

- ❑ 直接将没有打包的程序根目录复制到 \$CATALINA_HOME/webapps/ 目录下。Tomcat 会自动地分配一个上下文路径给该 Web 应用程序。值得注意的是，部署之后，必须重新启动 Tomcat 服务器。
- ❑ 直接将打包的 WAR 文件复制到 \$CATALINA_HOME/webapps/ 目录下，当 Tomcat 服务器重新启动以后，该 WAR 文件将会自动地解压成为非打包的形式。一般来讲，当发布 Web 应用程序的时候，通常采用的就是这种形式。

1.1.2 组织 Web 应用程序源代码

前面已经讨论过 Web 应用程序运行时的目录结构，下面讨论如何组织 Web 应用程序源代码，也就是如何安排 Web 应用程序的目录结构。最后讨论一下 Ant 配置文件 build.xml。

1. Web 应用程序目录结构

虽然在实际的开发过程中，用户可以随意地定义 Web 应用程序的文件组织结构。不过，通常采用下面所介绍的这种文件组织结构形式，这种代码组织结构在实际应用中被证明非常实用可行，即 Ant 配置文件 build.xml 配置文件（configuration file）所要求的形式。

- ❑ **docs/** 该目录存放应用程序的文档资料，在这个目录下的文件格式可以采用开发团队使用的任何格式。
- ❑ **src/** 该目录用于存放 Java 源代码文件，通常，在这个目录下存放应用程序的 Servlets、Beans 以及其他的 Java classes。如果源代码以包的形式存在，那么这个包在这个目录下面则表现为一个具有层次结构的目录。

- ❑ **web/** 组成网站的静态文件(HTML 页面、JSP 页面、JavaScript 文件、CSS stylesheet 文件以及图像文件等), 这些文件都可以被客户端所访问。
- ❑ **web/WEB-INF/** 应用程序所需的配置文件, 包括 Web 应用程序描述符(web.xml), 自定义标签库描述符以及其他需要包括在 Web 应用程序里面的资源文件。尽管这个目录是应用程序根目录下的子目录, 但是 Servlet 规范不允许客户直接访问这个目录里的内容(或它包含的任何文件)。所以, 这个地方常常用作存储敏感的配置信息(比如数据库连接用的用户名及密码)。

在程序开发过程中, 会临时产生另外两个目录。

- ❑ **build/** 执行默认 ant 指令后, 会在这个目录中产生应用程序的 WAR 文件。可以把此目录复制到 \$CATALINA_HOME/webapps 目录上去, 完成应用程序的部署。
- ❑ **dist/** 当执行 ant dist 任务后, 就会在这个目录中产生 Web 应用程序的二进制发布文件, 包括许可证信息、文档以及 README 文件。

注意

在实际的开发过程中, 如果团队使用了版本控制系统如 CVS, 则不要把这两个目录添加到源代码控制系统里, 因为这两个目录会在程序调试过程中不断地被删除并重新产生。此外, 不要在这两个目录里编辑源代码, 因为下一次执行 build 命令时, 所有更改会遗失。

2. Ant 配置文件 build.xml

为了方便用户快速入门, 提供了基本的 build.xml 文件的基本描述, 用户可以根据自己的项目需要自定义 build.xml 文件。通常, build.xml 文件提供以下基本任务。

- ❑ **清除 (Clean)** 这个任务用于清除已经编译之后产生的 build 和 dist 目录, 将该目录进行重新编译构建生成。
- ❑ **编译 (Compile)** 这个目标是用来编译在上次编译后又被更改过的源代码文件。所产生的类文件 (Class Files) 将被放置在 build 目录下的子目录 WEB-INF/classes 里, 与 Web 应用目录体系相一致。因为这个命令在程序开发过程中经常被执行, 所以通常设置为“默认”任务, 使用简单的 ant 命令便可以执行。
- ❑ **All** 这个任务是一个用来先执行清除任务, 然后再执行编译任务的快捷方式 (short cut)。因此, 它将重新编译所有的项目文件, 这样可以确保不会在不知不觉中对项目文件进行其他不正确的修改。
- ❑ **Javadoc** 这个任务用来为 Web 应用中的 Java 类生成对应的 Javadoc API 文档资料。
- ❑ **Dist** 这个任务用于为用户的应用开发创建一个分类目录体系, 包括任何所需要的文档资料、Java 类的 Javadoc 和 Web 应用档案 (WAR) 文件, 该 WAR 文件将被传递给需要安装 Web 应用程序的系统管理员。因为这个任务也依赖于 deploy 任务, 在项目部署时 Web 应用档案还应包括其他必需的外部文件。

为了方便与 Tomcat 5.0 进行交互式开发和代码测试, 定义了下面几个额外的任务 (Targets)。

- ❑ **安装 (Install)** 告诉正在运行的 Tomcat 5.0 使开发的程序可被立即用于执行和测