高等学校应用型本科创新人才培养计划指定教材
高等学校计算机类专业“十三五”课改规划教材

轻量级 Java EE 程序设计及实践

青岛英谷教育科技股份有限公司　编著

西安电子科技大学出版社
http://www.xduph.com

高等学校应用型本科创新人才培养计划指定教材

高等学校计算机类专业“十三五”课改规划教材

轻量级 Java EE 程序设计及实践

青岛英谷教育科技股份有限公司　编著

西安电子科技大学出版社

内 容 简 介

本书分为理论篇和实践篇，全面介绍了 Java EE 轻量级的三个开源框架：Struts2、Hibernate 和 Spring。其中，在 Struts2 部分主要讲解 MVC 设计思想、Struts2 的处理流程及配置、Struts2 常用控制器组件以及 Struts2 常用标签库的使用；在 Hibernate 部分主要讲解 O/R Mapping 的设计理念、Hibernate 对 O/R Mapping 的支持、Hibernate 的配置及多种关系映射的实现，以及 HQL 查询数据；在 Spring 部分主要讲解 IoC 的原理、Spring 对 Bean 的管理机制、Spring AOP 编程以及声明事务的配置和管理。

本书结构合理、重点突出、偏重应用，不仅在理论篇设有若干示例，而且在实践篇以一个完整在线购物系统贯穿全书的技术要点，进一步强化读者对 Struts2、Hibernate、Spring 框架的应用及整合技巧，全面提高动手能力。

本书适应面广，可作为本科计算机科学与技术、软件工程、网络工程、计算机软件、计算机信息管理、电子商务和经济管理等专业的程序设计课程的教材，也可作为科研、程序设计等人员的参考书籍。

图书在版编目(CIP)数据

轻量级 Java EE 程序设计及实践/青岛英谷教育科技股份有限公司编著. —西安：西安电子科技大学出版社，2015.8

高等学校计算机类专业“十三五”课改规划教材

ISBN 978-7-5606-3791-4

Ⅰ. ① 轻… Ⅱ. ① 青… Ⅲ. ① JAVA 语言—程序设计—高等学校—教材 Ⅳ. ① TP312

中国版本图书馆 CIP 数据核字(2015)第 191197 号

策　　划　毛红兵

责任编辑　马武装

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　　邮　　编　710071

网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com

经　　销　新华书店

印刷单位　北京京华虎彩印刷有限公司

版　　次　2015 年 8 月第 1 版　　2015 年 8 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印　张　27.5

字　　数　652 千字

印　　数　1～1000 册

定　　价　67.00 元

ISBN 978-7-5606-3791-4/TP

XDUP 4083001-1

如有印装问题可调换

❖❖❖ 前　　言 ❖❖❖

本科教育是我国高等教育的基础，而应用型本科教育是高等教育由精英教育向大众化教育转变的必然产物，是社会经济发展的要求，也是今后我国高等教育规模扩张的重点。应用型创新人才培养的重点在于训练学生将所学理论知识应用于解决实际问题，这主要依靠课程的优化设计以及教学内容和方法的革新。

另外，随着我国计算机技术的迅猛发展，社会对具备计算机基本能力的人才需求急剧增加，“全面贴近企业需求，无缝打造专业实用人才”是目前高校计算机类专业教育的革新方向。为了适应高等教育体制改革的新形势，积极探索适应 21 世纪人才培养的教学模式，我们组织编写了高等院校计算机专业系列课改教材。

该系列教材面向高校软件专业应用型本科人才的培养，强调产学研结合，内容经过了充分的调研和论证，并参照了多所高校一线专家的意见，具有系统性、实用性等特点，旨在使读者在系统掌握软件开发知识的同时，提高其综合应用能力和解决实际问题的能力。

该系列教材具有如下几个特色。

1. 以培养应用型人才为目标

本系列教材以培养应用型人才为目标，在原有体制教育的基础上对课程进行了改革，强化“应用型”技术的学习，使读者在经过系统、完整的学习后能够掌握如下技能：

- ✧ 掌握软件开发所需的理论和技术体系以及软件开发过程规范体系；
- ✧ 能够熟练地进行设计和编码工作，并具备良好的自学能力；
- ✧ 具备一定的项目经验，能够进行代码调试、文档编写、软件测试等；
- ✧ 达到软件企业的用人标准，做到学校与企业的需求无缝对接。

2. 以新颖的教材架构来引导学习

本系列教材采用的教材架构打破了传统的以知识为标准编写教材的方法，采用理论篇与实践篇相结合的组织模式，引导读者在学习理论知识的同时，加强实践动手能力的训练。

- ✧ 理论篇：学习内容的选取遵循“二八原则”，即重点内容由企业中常用的 20%的技术组成。每章设有本章目标，明确每一章学习重点和难点，章节内容结合示例代码，引导读者循序渐进地理解和掌握这些知识和技能，培养读者的逻辑思维能力，掌握软件开发的必备知识和技巧。
- ✧ 实践篇：多点集于一线，以任务驱动，将完整的具体案例贯穿始终，力求使读者在动手实践的过程中加深对课程内容的理解，培养读者独立分析和解决实际问题的能力，并配备相关知识的拓展讲解和拓展练习，拓宽读者的知识面。

另外，本系列教材借鉴了软件开发中的“低耦合，高内聚”的设计理念，组织结构上遵循软件开发中的 MVC 理念，即在保证最小教学集的前提下可以根据自身的实际情况对整个课程体系进行横向或纵向裁剪。

3. 提供全面的教辅产品来辅助教学实施

为充分体现“实境耦合”的教学模式，方便教学实施，该系列教材配备可配套使用的项目实训教材和全套教辅产品。

- ✧ 实训教材：集多线于一面，以辅助教材的形式，提供适应当前课程（及先行课程）的综合项目，遵循软件开发过程，进行讲解、分析、设计、指导，注重工作过程的系统性，培养读者解决实际问题的能力，是实施“实境”教学的关键环节。
- ✧ 立体配套：为适应教学模式和教学方法的改革，本系列教材提供完备的教辅产品，主要包括教学指导、实验指导、电子课件、习题集、实践案例等内容，并配以相应的网络教学资源。教学实施方面，提供全方位的解决方案(课程体系解决方案、实训解决方案、教师培训解决方案和就业指导解决方案等)，以适应软件开发教学过程的特殊性。

本书由青岛英谷教育科技股份有限公司编写，参与本书编写工作的有王燕、宁维巍、朱仁成、宋国强、何莉娟、杨敬熹、田波、侯方超、刘江林、方惠、莫太民、邵作伟、王千等。本书在编写期间得到了各合作院校专家及一线教师的大力支持与协作，在此，衷心感谢每一位老师与同事为本书出版所付出的努力。

由于编者水平有限，书中难免有不足之处，欢迎大家批评指正！读者在阅读过程中发现问题，可以通过邮箱(yujin@tech-yj.com)发给我们，以期进一步完善。

本书编委会

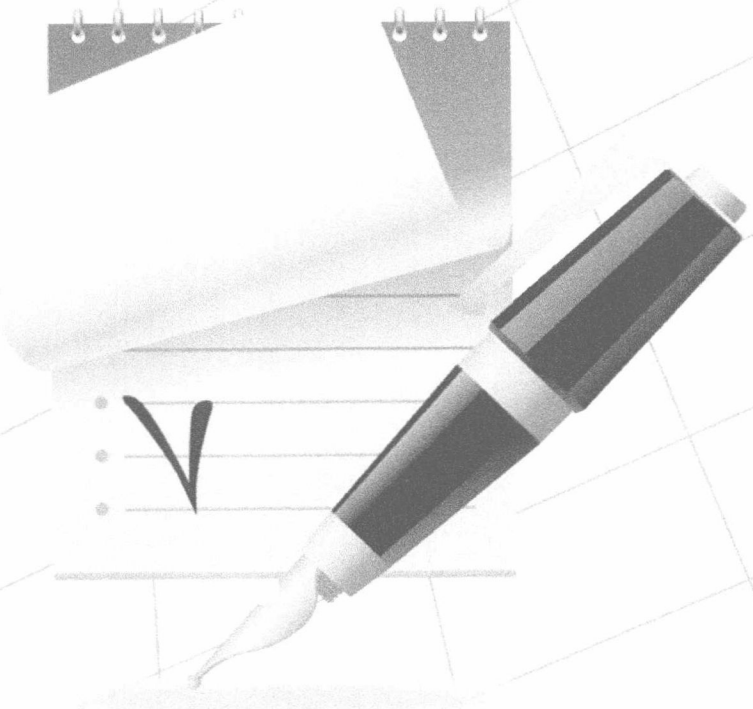
2015 年 4 月

❖❖❖ 目　　录 ❖❖❖

理　论　篇

实 践 篇

理论篇



第1章　Java EE 应用

本章目标

- 了解 Java EE 的开发模型
- 了解 Model1 的特点
- 了解 Model2 的特点
- 掌握 MVC 设计思想
- 熟悉多层架构模式
- 了解 Java EE 的常见架构技术

1.1 Java EE 概述

Java EE 经过多年发展，已经成为一个稳定、开源、安全的企业级开发平台。在传统的 Java EE 应用中，EJB(Enterprise Java Bean，企业级 JavaBean)是核心，但在轻量级 Java EE 应用中，EJB 不再是必需的，目前流行的轻量级 Java EE 应用框架有 Struts2、Spring、Hibernate、MyBatis 和 Spring MVC 等。

1.1.1 Java EE 应用分层模型

Java EE 应用大致可分为如下几层：

- ✧ 数据持久层：该层由一系列负责操纵 POJO(Plain Old Java Object，普通的、传统的 Java 对象)的类组成，这些类负责把数据进行持久化，一般是把数据保存到数据库中。
- ✧ 数据访问层：该层由一系列的 DAO(Data Access Object)组件组成，实现对数据库的增删改查等细粒度的操作。
- ✧ 业务逻辑层：该层由一系列的业务逻辑对象组成，实现系统所需要的业务逻辑方法。
- ✧ 控制层：该层由一系列控制器组成，用于拦截用户请求并调用业务逻辑对象的业务方法来处理请求，根据处理结果转发到不同的表示层。
- ✧ 表示层：该层由一系列的视图组件(例如 JSP 页面)组成，负责收集用户请求并显示处理结果。

Java EE 应用分层模型如图 1-1 所示。

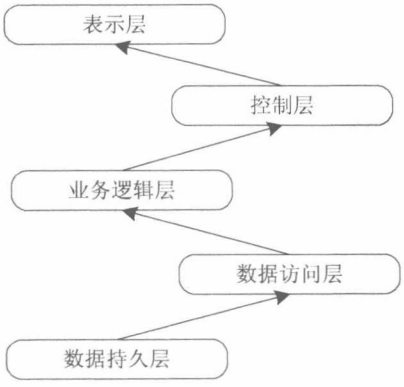


图 1-1 Java EE 应用分层模型

Java EE 各层组件之间以松散的方式耦合在一起，而非硬编码的方式，这种方式让应用具有更好的扩展性。图 1-2 显示了 Java EE 分层框架图，从上到下，上层组件的实现依赖于下层组件的功能；从下到上，下层组件支持上层组件的功能实现。

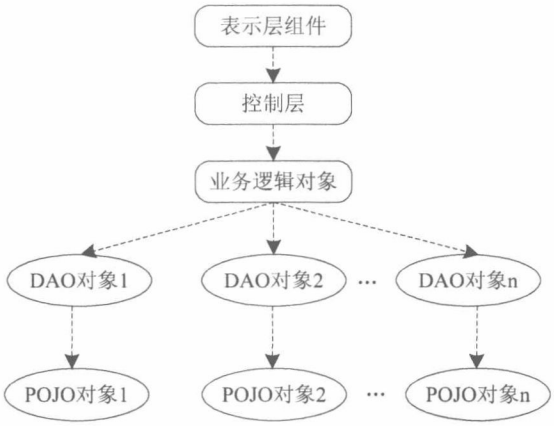


图 1-2　Java EE 分层框架图

1.1.2　Model1 与 Model2

Java 动态 Web 编程技术经历了 Model1 和 Model2 时代。

1．Model1

Model1 就是整个 Web 应用几乎全部由 JSP 页面组成，JSP 页面接收并处理客户端请求，用少量的 JavaBean 来处理数据库的连接、访问等操作。图 1-3 显示了 Model1 模式流程。

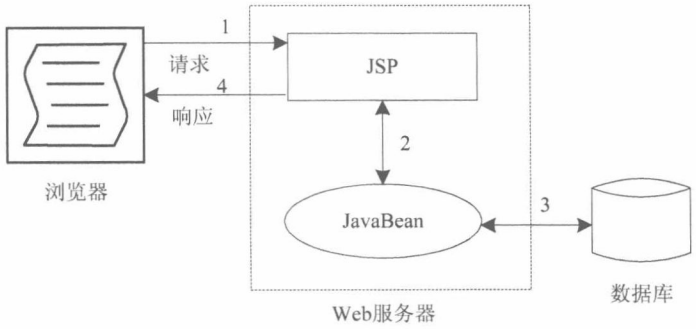


图 1-3　Model1 模式流程

Model1 模式的实现比较简单，适用于快速开发小型规模的项目，但此种模式具有局限性：JSP 页面身兼表示层和控制层两种角色，将控制逻辑和显示数据的代码混杂在一起，导致代码的可重用性非常低，不利于提高应用的可扩展性和可维护性。

2．Model2

Model2 是基于 MVC 架构的设计模式，在此模式中，Servlet 作为前端控制器，负责接收客户端发送的请求，Servlet 调用 JavaBean 完成实际的业务逻辑处理，处理结果显示到相应的 JSP 页面。如图 1-4 所示，显示了 Model2 模式流程。

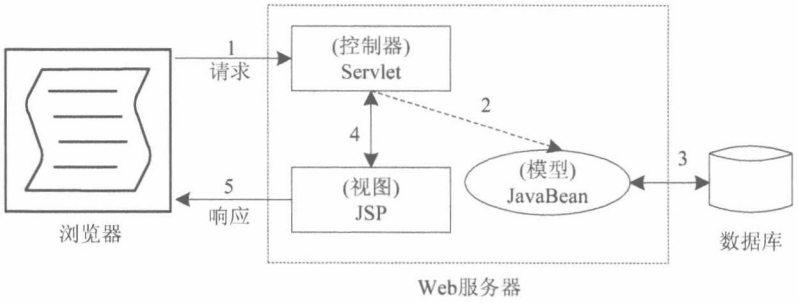


图 1-4　Model2 模式流程

1.1.3　MVC 思想及其优势

MVC 是一种架构模式，目的是将模型(业务逻辑)和视图(表示层)分离，使模型和视图可以独立修改，互不影响。大多数软件在设计架构时都采用此种模式。使用 MVC 模式有很多好处，当一个通过浏览器浏览的系统想要开发手机版本时，只需要重新开发视图，模型部分的业务逻辑可以重用。许多软件需要同时推出 B/S 和 C/S 版本，采用 MVC 模式，模型部分可完全重用，只需开发不同的视图即可。

MVC 思想将一个应用分成三个基本部分：M(Model，模型)、V(View，视图)和 C(Controller，控制器)。其中 M 表示处理业务逻辑的部分，V 表示显示数据和获取用户输入的部分，C 类似“中介”，保证 M 和 V 不会直接交互。如图 1-5 所示，显示 MVC 三部分之间的关系。

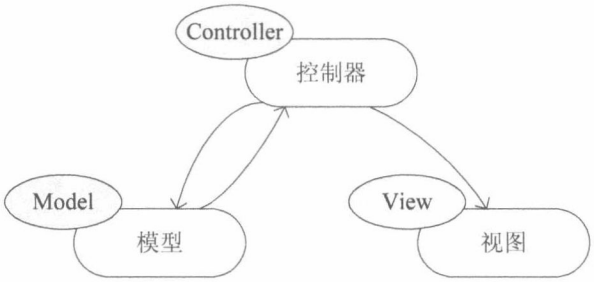


图 1-5　MVC 结构图

概括起来，MVC 具有如下特点：

(1) 数据的获取与显示分离。

(2) 控制器将不同的模型和视图组合在一起。

(3) 应用分为三部分：模型、视图和控制器，三部分之间松耦合并协同工作，从而提高应用的可扩展性和可维护性。

(4) 各层负责应用的不同功能，各司其职，每一层的组件具有相同的特征，便于通过工程化和工具化产生程序代码。

1.2　自定义 MVC 框架

Java EE 领域的 MVC 框架有很多，本书首先会设计一个简单的自定义 MVC 框架，以帮助读者体会 MVC 框架的含义和实现过程，以及在 MVC 设计模式中了解控制器(Controller)的作用。通过此框架的实现还可以帮助读者理解 Struts2 的原理。

【示例 1.1】 使用符合 MVC 设计模式的自定义框架实现加法计算器，要求所有的请求都发送给控制器，控制器根据请求路径找到相应的 Action(表示针对用户请求的一种处理操作)进行处理，Action 调用模型执行业务操作并获取数据，最后将结果返回给视图。

使用自定义 MVC 框架开发加法器程序的结构图如图 1-6 所示。

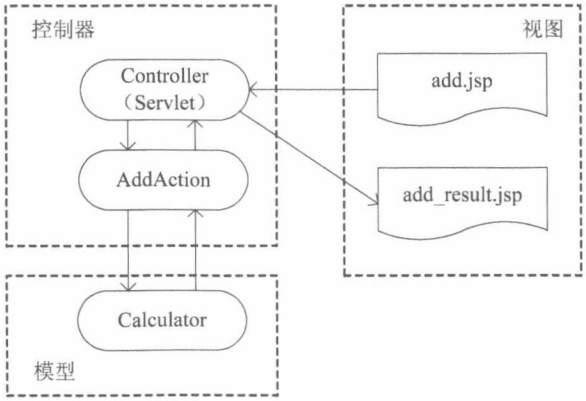


图 1-6　加法器的 MVC 结构图

此加法器程序需要用户在 add.jsp 页面中输入两个数，单击“加”按钮进行计算，计算的结果在 add_result.jsp 页面显示，如图 1-7 所示。

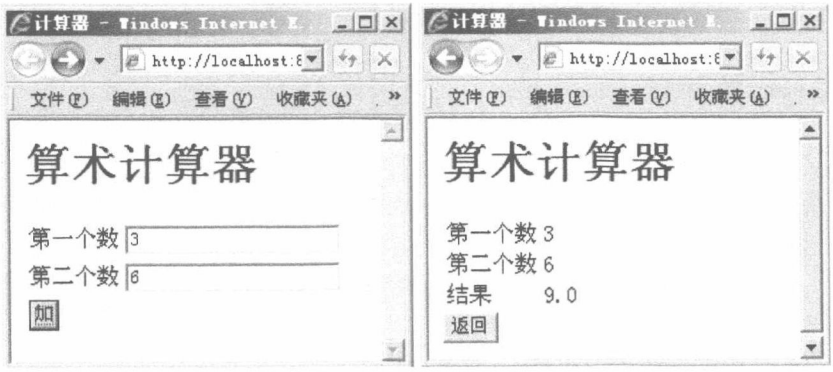


图 1-7　加法器运行效果

1.2.1　实现控制器

自定义的 MVC 框架的核心是控制器的实现：定义 Action 接口，实现 Controller 类。

首先在 com.dh.ch01.framework 包中创建 Action 接口，代码如下：

```
public interface Action {
    //定义该接口的实现类必须实现的 execute 方法
    String execute(HttpServletRequest request,HttpServletResponse response);
}
```

上述 Action 接口中定义了一个 execute()方法，该方法有请求对象 request 和响应对象 response 两个参数；该方法返回一个字符串类型的值，表示执行完操作后要转发到的页面。Action 接口对各种动作的执行方法进行统一，便于在控制器中进行调用和访问。

此处插入 AddAction.java 的部分代码，否则执行代码会出错。

然后在 com.dh.ch01.framework 包中创建一个名为 Controller 的 Servlet，代码如下：

```
/**
 * 自定义 MVC 框架：基于 Servlet 实现的控制器
 */
public class Controller extends HttpServlet {
    //声明由控制器 Controller 维护的 Action 映射，其中保存所有的 Action 实例
    private HashMap actionMap;
    /**
     * Servlet 初始化方法
     */
    @SuppressWarnings("unchecked")
    public void init() throws ServletException {
        // 初始化 actionMap
        actionMap = new HashMap();
        // 将 AddAction 对象放入到 actionMap 中
        actionMap.put("add", new AddAction());
    }
    /**
     * 根据 path 判断由哪个 action 执行操作
     */
    private Action determinActionByPath(String path) {
        //如：  从 http://localhost:8080/ch01/add.action 中得到 add
        String actionName =
                path.substring(path.lastIndexOf('/') + 1, path.length() - 7);
        // 获得该请求对应的 action 对象
        Action ret = (Action)actionMap.get(actionName);
        return ret;
    }
    /**
     * 处理页面以 get 方式提交的请求
     */
```