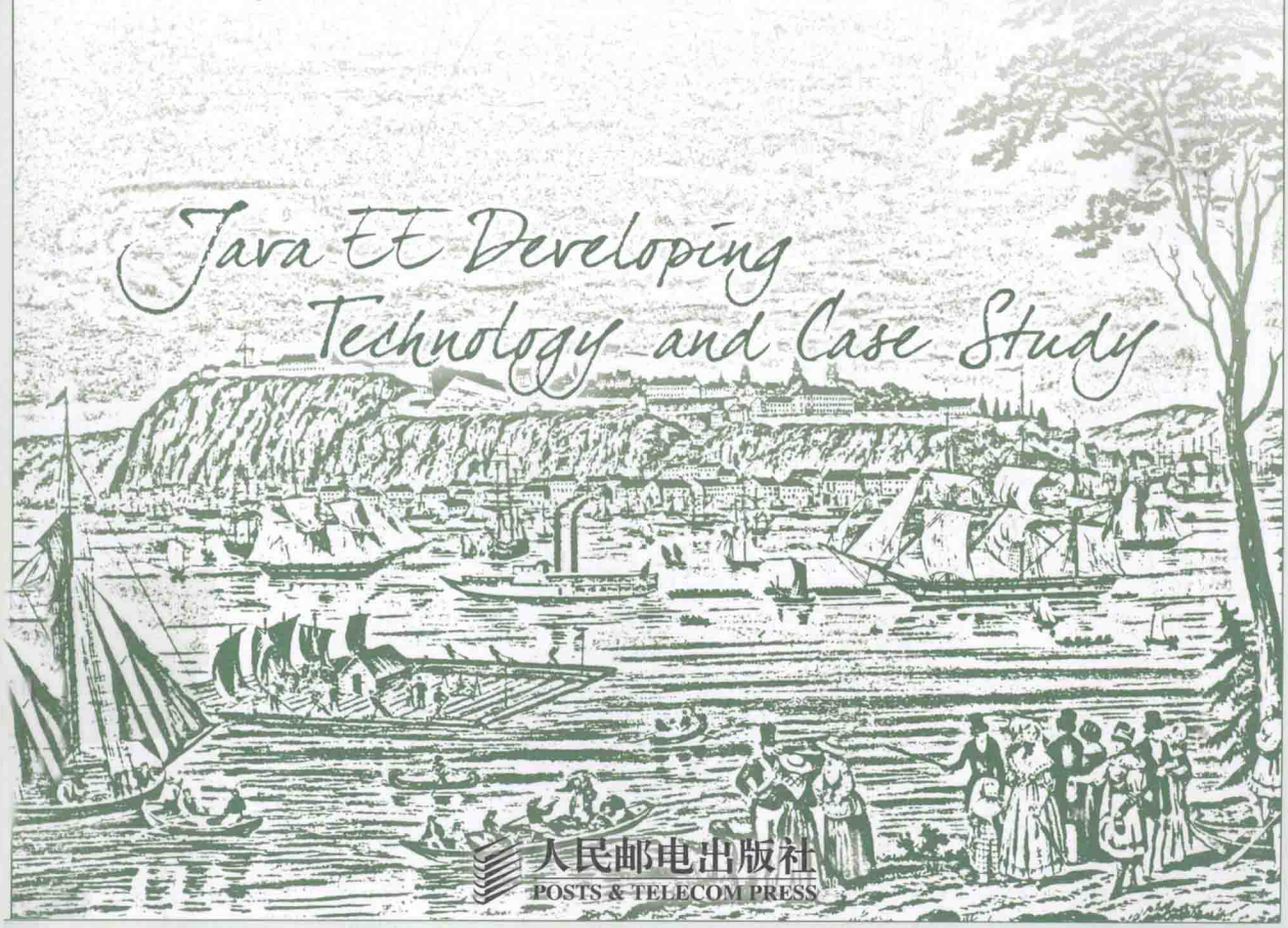
普通高等教育
软件工程 “十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

Java EE 开发技术与案例教程

刘彦君 金飞虎 ◎ 主编
张仁伟 胡平 汪国武 杨沿航 ◎ 副主编

Java EE Developing Technology and Case Study



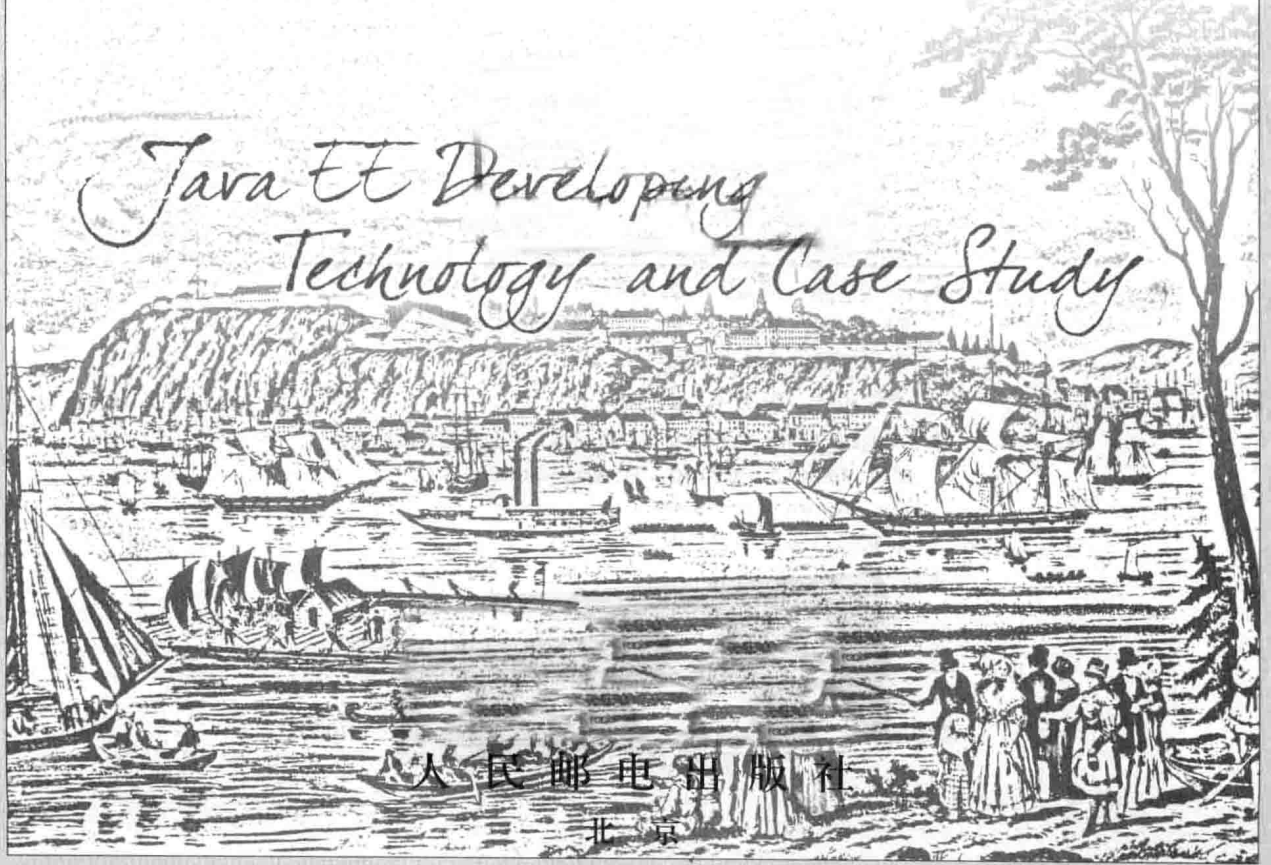
人民邮电出版社
POSTS & TELECOM PRESS

普通高等教育
软件工程 “十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

Java EE开发技术与案例教程

刘彦君 金飞虎 ◎ 主编
张仁伟 胡平 汪国武 杨沿航 ◎ 副主编



人民邮电出版社
北京

图书在版编目（CIP）数据

JavaEE开发技术与案例教程 / 刘彦君，金飞虎主编
-- 北京 : 人民邮电出版社，2014.2
普通高等教育软件工程“十二五”规划教材
ISBN 978-7-115-33741-2

Ⅰ. ①J… Ⅱ. ①刘… ②金… Ⅲ. ①JAVA语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2014)第013086号

内 容 提 要

全书共分为11章，主要内容包括：第1章介绍了Java EE的基本概念，第2章介绍了JDBC数据库编程的基础知识和应用方法，第3章介绍了Java Servlet，第4章介绍了JSP，第5章介绍了XML，第6章介绍了Struts2，第7章介绍了Hibernate，第8章介绍了Spring，第9章介绍了EJB，第10章介绍了SSH整合开发案例，第11章介绍了基于Java EE的测试。

本书可作为高等学校计算机专业、软件工程专业教材及从事相关开发领域程序设计人员自学及参考用书。

◆ 主　　编　刘彦君　金飞虎
副 主 编　张仁伟　胡　平　汪国武　杨沿航
责任编辑　许金霞
责任印制　彭志环　杨林杰

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京隆昌伟业印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：20　　2014年2月第1版
字数：528千字　　2014年2月北京第1次印刷

定价：48.00元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前　言

Java EE 技术经过多年的发展越来越趋于成熟完善、目前已成为最佳的企业应用解决方案之一。Java EE 是 Java 的高级应用，它与市场紧密衔接，有利于读者掌握前沿开发技术，理解和掌握新的软件开发思想，有助于培养学生的实际动手能力。为了适应形势的发展，许多高校都在开设 Java EE 课程。但是目前市场上关于 Java EE 的书籍多为技术参考书，这些书或为专论某种 Java EE 技术规范，或为集中介绍某些应用案例，它们往往内容偏多、偏难，且不能很好地做到理论基础和案例的有机结合，因此不适合作为教材。本书是基于作者多年从事 Jave EE 课程的教学实践，对教学内容取舍及重点内容确定的理解编写的。

本书的主要特点包括：以实用为目的，选择常用技术加以详尽阐述，对 JDBC 技术、JSP、XML、EJB 的理论知识和应用进行了深入的讨论。体现项目驱动和案例教学的思想，在每一章都运用大量应用程序例证，并以一个实用系统的开发案例作为学生综合运用所学的技术的一个实战项目。案例选择不片面追求大规模和高复杂性，而是力求典型，使案例与知识相辅相成，有机联系，形成整体，使知识讲解更直接具体，有针对性和目的性，有利于学生接受和掌握。理论知识讲解精练，在保证知识的系统性的同时，注重精选内容。从典型的项目出发，基于项目需要展开知识，围绕知识讲解案例，对知识的讲解不追求全面细致，而是强调针对性和目的性，突出重点，内容新颖。关于框架的内容，介绍最新的或较新的版本，反映技术发展的新趋势。

全书共分 11 章，各章的主要内容如下。

第 1 章概要介绍了 Java EE 的产生与发展沿革、Java EE 的新特性、Java EE 企业应用系统分层框架、Java EE 技术规范、敏捷轻型框架以及 Java EE 开发环境包括 Tomcat 和 MySQL 等的安装与使用方法。

第 2 章介绍了 JDBC 数据库编程的基础知识和应用方法。包括 JDBC 驱动程序、用 JDBC 访问数据库的方法步骤、部分数据库操作所必需的类和接口以及它们的属性和方法、事务的概念与事务处理。最后介绍了提高数据库存取效率的几种常用技术包括预处理语句的使用方法、存储过程的定义与调用方法和数据库连接池的概念与应用程序等。

第 3 章主要内容包括：Java Servlet 概述、Java Servlet 编程基础、Java Sevlet 生命周期、Java Servlet 常用的类、Java Servlet 应用举例。Java Servlet 是早期用于实现 Web 应用服务的一种技术。通过本章的学习，读者应该了解什么是 Java Servlet 以及 Java Servlet 常用类、接口的基本特点和用法，并掌握如何完成 Java Servlet 的创建和应用。

第 4 章介绍 JSP 的基本语法和编程知识，详细介绍了 JSP 的脚本元素、指令元素、动作元素和 9 大内置对象的语法及使用方法。另外还介绍了 EL 表达

式和 JSTL 的基础知识和具体应用方法。通过本章的学习，读者应该具备 JSP 的使用和开发所需要的知识和技能。

第 5 章主要内容包括：XML 简介、DOM 和 SAX、XPath。本章对 XML 在 Internent 以及基于 Internet 的应用中的作用、对 XML 的语法知识包括 DTD 和 Schema 并做了详细阐述。用程序实例对 Java 中访问 XML 文档的方法进行了对比分析。介绍了文档对象模型（DOM）和用于 XML 的简单 API（SAX）的基本原理和 API 进行了说明。最后介绍了文档查询的基本概念和技术实现。

第 6 章主要内容包括：Struts2 简介、Struts2 工作原理、Struts.xml 配置、拦截器的机制和应用、Struts2 类型转换以及 Struts2 的输入校验等内容。Struts 是目前使用最广泛的一种框架。Struts 建立在 Servlet，JSP，XML 等技术上，很好地实现了 MVC 设计模式，使得软件设计人员可以把精力放在复杂的业务逻辑上。使用 Struts 框架，开发人员可以快速开发易于重用的 Web 应用程序。本章通过大量的 Struts2 应用的简单例子说明其工作原理，达到易于理解和掌握的效果。

第 7 章主要内容包括：Hibernate3 基础知识、Hibernate 对象状态、Hibernate 事务、Hibernate 反向工程和 HQL。Hibernate 是一种对象关系映射解决方案，是使用 GNU 通用公共许可证发行的自由、开源的软件，是一种使用方便的框架，它为面向对象的领域和传统的关系型数据库的映射提供了比较好的解决方案。

第 8 章介绍了 Spring 的基本概念和特点、Spring 的框架结构和工作原理。具体阐述了 Spring 的控制反转（IoC）和依赖注入（DI）的基本概念和依赖注入的形式、IoC 的装载机制、面向方面编程（AOP）的实现原理、AOP 框架等内容。

第 9 章主要内容包括：EJB 简介、会话 Bean、消息服务和消息驱动 Bean 和 EJB 生命周期。EJB（Enterprise Java Beans）是 Sun 公司提出的服务器端组件模型，是 Java 技术中服务器端软件构件的技术规范和平台支持。其最大的用处是部署分布式应用程序，类似微软的.com 技术。凭借 Java 跨平台的优势，用 EJB 技术部署的分布式系统可以不限于特定的平台。和其他 Java EE 技术一样，EJB 大大地增强了 Java 的功能，并推动了 Java 在企业级应用程序中的应用。

第 10 章主要介绍一个实用的应用系统开发案例，并给出全部实现源代码。具体内容包括：系统概述、SSH 工程的配置、DOMAIN 层、DAO 层、验证码、用户注册、用户登录等内容。

第 11 章主要内容包括：单元测试方法、基于 QTP 的功能测试方法、基于 Jmeter 的性能测试方法。测试是一种产品质量保证的重要手段。对软件的质量要求越来高，这必然引起了对测试工作的重视，一款好软件的问世，不但要求有强大的开发人员，还需要水平高超的测试人员。

本教材的第 1~2 章和第 9 章由刘彦君编写，第 3~4 章由张仁伟编写，第 5 章和第 11 章由杨沿航编写，第 6~7 章由金飞虎编写，第 8 章由汪国武编写，第 10 章由胡平编写，全书由刘彦君负责统稿和审稿。

由于编者水平有限，书中难免存在疏漏之处，恳请读者和同仁批评指正。

编　者

2013 年 11 月

目 录

第1章 Java EE概述

本章内容

- Java EE 简介
- Java EE 应用分层架构
- Java EE 技术规范
- 敏捷轻型框架
- Java EE 开发环境

Java EE（Java Enterprise Edition）是建立在 Java 平台上的企业级应用解决方案。Java EE 基于 Java SE（Java Standard Edition）平台，提供了一组用于开发和运行的可移植的、健壮的、可伸缩的、可靠的和安全的服务器端应用程序的应用程序编程接口（Application Programming Interface，API）。

1.1 Java EE 简介

1.1.1 什么是 Java EE

Java EE 是基于 Java 的解决方案，是 Java 平台的企业级应用，是一套技术架构。Java EE 的核心是一组技术规范与指南，它使开发人员能够开发具有可移植性、安全性和可复用的企业级应用。Java EE 的体系结构保证了开发人员更多地将注意力集中于架构设计和业务逻辑上。

Java EE 的全称是 Java 2 Platform Enterprise Edition，Sun 公司于 1998 年推出 JDK1.2 版的时候使用的名称是 Java 2 Platform，即 Java 2 平台。后来修改为 Java 2 Platform Software Developing Kit，即 J2SDK，包括标准版（Standard Edition，J2SE）、企业版（Enterprise Edition，J2EE）和微型版（Micro Edition，J2ME）。2006 年 5 月，这三个 Java 版本更改为现在的名称，J2SE、J2EE 和 J2ME 分别改称为 Java SE、Java EE 和 Java ME。Java EE 是由 Sun 公司领导大厂商共同制定的被业界广泛认可的工业标准，JCP（Java Community Process）等开放性组织对其发展也做出了非常大的贡献。

Java EE 技术具有 Java SE 技术的所有功能，同时还提供对 EJB、Servlet、JSP、XML 等技术的支持，它已经发展成为一个支持企业级开发的体系结构，简化了企业解决方案的开发、部署和管理等问题。目前，它已成为企业级开发的工业标准和首选平台。Java EE 是一个规范，各个平台开发商可按照 Java EE 规范开发 B/S 模式的 Java EE 应用服务器，克服了传统的 C/S 模式的弊端。需要说明的是：Java EE 不是 Java SE 的替代品，二者的关系为 Java SE 是 Java EE 的核心，它为 Java EE 提供了基本的语言框架，是 Java EE 所有组件的基础。Java 开发人员在学习了 Java SE 之

后，在 Java EE 中将会学习更多的组件和 API，并可利用所学的编程知识进行不同的应用的开发。

1.1.2 Java EE 的新特性

Java EE 5 的发布给业界带来很大的震撼，这源于它与 Java EE1.4 相比，增加了许多重要的新特性和做了很多改变。所有改变的目标是为开发人员提供功能更加强大的 API 库、使系统架构适合于快速的开发和部署的要求、提高软件性能、降低开发难度等。

Java EE 5 的新特性主要有以下几点。

1. 标注

标注（annotation）是 Java EE 5 引入的一个新特性。标注是一种元数据，按照其作用可以分为 3 类：编写文档、代码分析和编译检查。用于编写文档的标注是通过代码里的标注元数据生成文档（例如@Documented）用于定制 javadoc 不支持的文档属性，并在开发中使用。用于代码分析的标注（如@Deprecated）,指出这是个不建议使用的方法。而用于编译检查的标注是通过代码里的标注元数据使编译器能实现基本的编译检查，如@Override 标注能实现编译时的检查，在方法前加此标注的作用是声明该方法用于覆盖父类中的方法。如果该方法未覆盖父类方法，例如，该名字的方法不是父类方法或参数不符合覆盖父类方法的语法规定，则编译会报错。

引入标注可以实现多种功能的简化，例如：

（1）定义和使用 Web Service。

（2）开发 EJB 组件。

（3）映射 Java 类到 XML 文档。

（4）映射 Java 类到数据库。

（5）依赖注入。

（6）指定部署信息等。

有了标注，XML 部署描述符就不再是必需的了。在 Web 应用开发中直接在代码中使用标注就可以告知 Java EE 服务器如何部署及运行，而不必再编辑 WEB-INF/web.xml 文件了。

2. EJB 3

EJB 3 是 EJB 2 的升级，持久化变得更加简化，是轻量级的框架。它不再需要 EJB home 接口，不再需要实现 SessionBean 接口，JDNI API 也不再是必需的。EJB 部署描述符变成可选的功能。此外，EJB 3 中还引入了拦截器功能。拦截器是 AOP 在 EJB 中的实现，是可以对 Bean 的业务方法进行拦截的组件。拦截器可以用于无状态会话 Bean、有状态会话 Bean 和消息驱动 Bean。拦截器用来监听程序的一个或者多个方法，它对方法调用提供了控制功能。

拦截器用@Interceptors 标注或在配置文件中配置。@Interceptors 可以用在类级别上，或者用在方法上。如果用在类上，则说明整个类的所有方法都被拦截。如果用在方法上，说明仅拦截被标注的方法。

3. JPA

JPA（Java Persistence API），即数据持久化 API，它是一个轻量级的对象持久化模型，是 Java EE 的又一新特性。

Sun 公司推出 JPA 规范的目的在于简化现有 Java EE 和 Java SE 应用的对象的持久化工作，希望统一 ORM 技术。在 JPA 出现之前，各种 ORM 框架之间的 API 差异很大，使用了某种 ORM 框架的系统会受制于该 ORM 的标准。在 JPA 中，实体是 POJO 对象。实体对象不再是组件，也不是必须在 EJB 模块中。JPA 在充分吸收现有的 ORM 框架技术的基础上，提出了一个易于使用的

伸缩性强的 ORM 规范，通过标注或 XML 描述对象关系的映射，将 POJO 对象持久化到数据库中。

JPA 本质上是一种 ORM 规范，并未提供 ORM 实现，具体实现由其他的厂商提供。程序员若要使用 JPA，需要选择 JPA 的实现框架，Hibernate 3 即是这样一个实现了 JPA 的框架。

4. Web Service 支持

Java EE 5 所提供的 Web Service 支持更简单，内容更广泛。其中包括：

（1）Java API for XML-based Web Services（JAX-WS 2.0，JSR 224）。

（2）Java Architecture for XML Binding（JAXB 2.0，JSR 222）。

（3）Web Services Metadata（JSR 181）。

（4）SOAP with Attachments API for Java（SAAJ 1.3）。

什么是 Web Service？众所周知，如果所有人都是使用 Java 来开发应用程序，处理客户端和服务器的通信问题，那问题就简单了。而实际情况是，很多商用程序仍然在继续使用 C、C++、Visual Basic 和其他各种各样的语言编写。现在，除了少数简单程序外，很多应用程序都需要与运行在其他异构平台上的应用程序集成并进行数据交换。这样的任务通常由一些特殊的方法，如文件传输和分析、消息队列，还有某些专用的 API 来完成。现在，使用 Web Service，客户端和服务器就能自由地利用 HTTP 进行通信，而不管两个应用程序的平台和编程语言是什么。

Web Service 是建立可互操作的分布式应用程序的新平台，程序员可能曾经使用 COM 或 DCOM 建立过基于组件的分布式应用程序，或者曾经使用 CORBA、RMI 等技术实现远程调用。Web Service 平台也是这样的一套标准，而且它做得更好。它定义了应用程序如何在 Web 上实现互操作。Web Service 平台需要一套协议来实现分布式应用程序的创建。

JAX-WS 2.0 支持可扩展的传输协议和异步客户端，并且支持代表状态传输（Representational State Transfer，REST）应用。在服务器端，用户只需要通过 Java 语言定义远程调用所需要实现的接口 SEI（Service Endpoint Interface），并提供相关的实现，通过调用 JAX-WS 的服务发布接口就可以将其发布为 Web Service 接口。

在客户端，用户可以通过 JAX-WS 的 API 创建一个代理（用本地对象来替代远程的服务）来实现对于远程服务器端的调用。

JAXB 2.0 全面支持 XML Schema，能够绑定 Java 类到 XML Schema 中，支持更小的内存占用、更快的编组和更灵活的反编组，支持局部绑定 XML 文档到 JAXB 对象。

总之，Java EE 5 所提供的 Web Service 支持，使得在这个平台上开发的基于 Web 的应用更为简单而高效、更加健壮而灵活易用，且能够涵盖更广泛的应用范围。

5. 依赖注入

所谓依赖注入（dependency injection）是指当某个角色（可能是一个 Java 实例，调用者）需要另外一个角色（另外一个 Java 类的实例，被调用者）的协助时，在传统的程序设计过程中，通常是由调用者来创建被调用者的实例。在一些轻型框架如 Spring 中，创建被调用者的任务不再由调用者完成，而是由 Spring 容器完成，然后以某种方式注入给调用者，称为依赖注入，也称为控制反转。

通过依赖注入降低了代码的耦合度，使得资源访问变得更加容易。

下面的例子说明依赖注入的基本功能。

例如，类 A 中用到一个类 B 的实例，而用依赖注入的方式是在 applicationContext.xml 文件里面写入下面内容。

```
<bean id="id1" class="A"><property name="B" ref="id2"></bean>
<bean id="id2" class="B"></bean>
```

则在类 A 里原来需要定义 B 的实例的地方就没必要再定义了。

6. 泛型

泛型（generics）是程序设计语言的一种特性，支持泛型的程序设计语言允许程序员在编写代码时定义一些可变部分，那些部分在使用前必须做出指明。各种程序设计语言及其编译器、运行环境对泛型的支持均不一样。泛型将类型参数化以达到代码复用、提高软件开发工作的效率。泛型主要是引入了类型参数这个概念。

使用泛型的好处是：它允许程序员将一个实际的数据类型的确定延迟至创建泛型的实例的时候。泛型为开发者提供了一种高性能的编程方式，能够提高代码的重用性，并允许开发者编写非常优秀的解决方案。

Java EE 5 通过引入泛型，使得集合元素类型参数化，避免了运行时出现类型转换错误，因此不必要加入显式强制类型转换的操作了。

下面的例子对此做了说明。

不使用泛型时：

```
ArrayList  list = new ArrayList();
list.add(0,new Integer(42));
int total = ((Integer)list.get(0)).inValue();
```

使用了泛型后：

```
ArrayList <Integer> list = new ArrayList<Integer>();
list.add(0,new Integer(42));
int total = list.get(0).inValue();
```

读者可以体会这两个例子之间的微妙差异，进而理解用泛型的好处。

7. 枚举

枚举类型是 Java EE 5 开始引入的类型，本质上枚举类型就是一个命名变量的列表。枚举类型通过关键字 enum 来声明。下面是一个枚举的例子。

```
public enum Week{
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday,
    Sunday
}
```

对命名常量可以通过类似对象成员的方法或者通过方法 values()、valueOf()、ordinal()、name() 等方法进行存取操作。

下面的 for 循环将输出枚举的所有命名常量。

```
for(Week w:Week.values())
    System.out.println(w);
```

8. 增强的 for 循环

Java EE 5 中的增强的 for 循环简化了数组和集合的遍历操作，其语法更简单，可以防止下标越界的问题出现，而且还可以避免由于强制类型转换导致的错误。下面是一个使用增强 for 循环

对数组元素进行遍历的例子。

```
int a[] = {1,2,3,4,5,6};
for(int num:a)
   System.out.println(num);
```

9. 可变参数

Java EE 5 之前的版本中，方法的参数个数是固定的。Java EE 5 允许创建具有可变参数的方法，这使得某些操作变得更方便了。下面的程序例子可以说明这一点。

```
public class VarArgument{
public static void main(String args[]){
    System.out.println(add(2,3));
    System.out.println(add(2,3,4,5));
}
public static int add(int…args){
    int sum=0;
    for(int i=0;i<args.length;i++)
    {
        sum+=args[i];
    }
return sum;
    }
}
```

10. 静态导入

在 Java EE 5 之前的版本中，程序中使用静态成员要在其前面加类名引导。Java EE 5 引入静态导入意味着不必再写类名，而是直接通过静态成员的名字来访问它们。例如：

```
//静态导入
import static java.lang.System.*;
import static java.lang.Math.*;
…
    //调用静态成员
    out.println(sqrt(6));
    //不再是 Math.sqrt(6)
```

1.2　Java EE 应用分层架构

Java EE 使用多层的分布式应用模型，按功能划分为不同组件，根据组件所在的层分布在不同的机器中。

1.2.1　分层模式概述

分层模式是常见的架构模式。分层描述的是这样一种架构设计过程：最低抽象级别称为第 1 层，从最低级的抽象逐步向上进行抽象，直至达到功能的最高级别。

分层模式的特点如下。

- 伸缩性：伸缩性是指应用程序能支持更多用户的能力。应用的层数少，可以增加资源（如 CPU、内存等）的机会就少。反之，则可以把每层分布在不同的机器上。

- 可维护性：指的是发生需求变化时，只需修改软件的局部，不必改动其他部分的代码。
- 可扩展性：可扩展性是指在现有系统增加新功能的能力。在分层的结构中，可扩展性较好，这是由于可以在每个层中插入功能扩展点，而不改变原有的整体框架。
- 可重用性：可重用性指的是同一程序代码可以满足多种需求的能力。例如，业务逻辑层可以被多种表示层共享，即业务逻辑层的代码被重用了。
- 可管理性：指管理系统的难易程度。

1.2.2 Java EE 的结构

Java EE 使用多层分布式的应用模型，该模型通过以下 4 层来实现。

（1）客户层：运行在客户计算机上的组件。

（2）Web 层：运行在 Java EE 服务器上的组件。

（3）业务层：同样是运行在 Java EE 服务器上的组件。

（4）企业信息系统层（EIS）：是指运行在 EIS 服务器上的软件系统。

有时我们把客户层和 Web 层视为一个层，这样就可以将以上结构按 3 层来划分，如图 1-1 所示。

在这个分层体系中，客户层组件可以是基于 Web 方式的，也可以是基于传统方式的。Web 层组件可以是 JSP 页面或者 Servlet。

对于业务逻辑层组件，其代码是处理（如银行、零售等）具体行业或领域的业务需要，由运行在业务层上的 Enterprise Bean 进行处理。

企业信息系统层处理企业信息系统软件，包括企业基础建设系统（例如企业资源计划）、大型机事务处理、数据库系统和其他遗留系统。

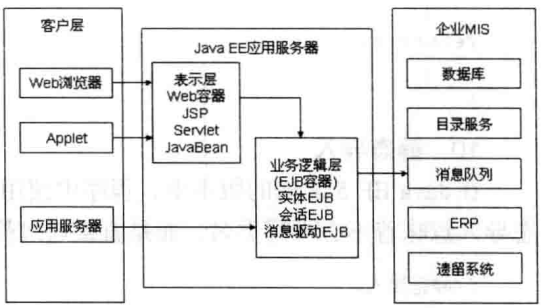


图 1-1　三层 Java EE 体系结构

1.3 Java EE 技术规范

Java EE 作为一个分布式企业应用开发平台，通过一系列的企业应用开发技术来实现。其技术框架可分为 3 部分：组件技术、服务技术和通信技术。其中，组件是构成 Java EE 应用的基本单元，组件包括：客户端组件、Web 组件和 EJB 组件。服务技术是指方便编程的各种基础服务技术，如命名服务、事务处理、安全服务、数据库连接。而通信技术则是提供客户和服务器之间，以及服务器上不同组件之间的通信机制等，相关支持技术包括 RMI、消息技术等。下面对 Java EE 中的常用技术规范进行简要的介绍。

1. JDBC（Java Database Connectivity）

JDBC API 为访问不同的数据库提供了一种统一的机制，像 ODBC 一样，JDBC 使操纵数据库的细节对开发者透明。另外，JDBC 对数据库的访问也具有平台无关性。

2. JNDI（Java Name and Directory Interface）

名字和目录服务，为应用提供一致的模型来访问企业级资源，如 DNS 和 LDAP、本地文件系统或应用服务器中的对象。

3. EJB（Enterprise Jav Bean）

企业 Java 组件，提供一个框架来描述分布式商务逻辑，开发具有可伸缩性和复杂的企业级应用。EJB 规范定义了组件何时、如何与它们的容器进行交互。容器负责提供公用的服务，如目录、事务管理、安全性等。需要说明的是，EJB 并不是实现 Java EE 企业应用的唯一渠道，它的意义在于：它是专为分布式大型企业应用而设计，用它编写的程序具有良好的可扩展性和安全性。

4. RMI（Remote Method Invoke）

远程方法调用，顾名思义，它用于调用远程对象的方法。在客户端和服务器端传递数据使用了序列化方式。

5. Java IDL/CORBA（Java Interface Definition Language/Common Object Request Broker Architecture）

Java 接口定义语言/公用对象请求代理结构。为 Java 平台添加了公用对象请求代理体系结构（Common Object Request Broker Architecture，CORBA）功能，从而可提供基于标准的互操作性和连接性。Java IDL 使分布式、支持 Web 的 Java 应用程序可利用 Object Management Group 定义的行业标准对象管理组接口定义语言（Object Management Group Interface Definition Language，OMG IDL）及 Internet 对象请求代理间协议（Internet Inter-ORB Protocol，IIOP）来透明地调用远程网络服务。运行时组件包括一个全兼容的 Java ORB，用于通过 IIOP 通信进行分布式计算。

6. JSP（Java Server Pages）

JSP 页面由 HTML 代码和嵌入其中的 Java 代码组成。服务器被客户端请求以后，对这些 Java 代码进行处理，然后将生成的 HTML 页面返回给客户端的浏览器。

7. Java Servlet

Servlet 是运行在服务器端的 Java 程序，它扩展了 Web 服务器的功能。作为一种服务器端的应用，当被请求时开始执行。Servlet 提供的功能和 JSP 一致，只是二者的构成不同。JSP 通常是 HTML 代码中嵌入 Java 代码，而 Servlet 全部由 Java 写成并且生成 HTML。

8. XML（eXtensible Markup Language）

扩展的标记语言，用来定义其他标记语言的语言。作为数据交换和数据共享的语言，适用于很多的应用领域。

9. JMS（Java Message Service）

Java 消息服务，是 Java 平台上用于建立面向消息中间件（MOM）的技术规范，它便于消息系统中的 Java 应用程序进行消息交换，并且通过提供标准的产生、发送、接收消息的接口，简化企业应用的开发。

许多厂商目前都支持 JMS，包括 IBM 的 MQSeries、BEA 的 Weblogic JMS service 等。使用 JMS 能够通过消息收发服务（有时称为消息中介程序或路由器）从一个 JMS 客户机向另一个 JMS 客户机发送消息。消息是 JMS 中的一种类型对象，由两部分组成：报头和消息主体。报头由路由信息以及有关该消息的元数据组成。消息主体则携带着应用程序的数据或有效负载。

10. JTA（Java Transaction Architecture）

Java 事务体系结构，定义了一组标准的 API，用于访问各种事务监控。

11. JTS（Java Transaction Service）

Java 事务服务，是 CORBA OTS（Object Transaction Service）事务监控的基本实现。

12. Java Mail

用于存取邮件服务器的 API，它提供了一套邮件服务器的抽象类。它不仅支持 SMTP 服务器，

也支持 IMAP 服务器。

13. JAF (JavaBeans Activation Framework)

JavaMail 利用 JAF 来处理 MIME 编码的邮件附件。MIME 的字节流可以被转换成 Java 对象，或者相反。

1.4 敏捷轻型框架

框架，即 framework。其实就是某种应用的半成品，就是一组组件，供你选用，完成你自己的系统。这些组件是把不同的应用中有共性的任务抽取出来加以实现，做成程序供人使用。简单地说，就是使用别人搭好的舞台，你来做表演。而且，框架一般是成熟的，不断升级的软件。 框架的概念最早起源于 Smalltalk 环境，其中最著名的框架是 Smalltalk 80 的用户界面框架 MVC (Model-View-Controller)。

框架可分为重型框架和轻型框架。一般称 EJB 这样的框架为重型框架，因其软件架构较复杂，启动加载时间较长，系统相对昂贵，需启动应用服务器加载 EJB 组件。而轻型框架则不需要昂贵的设备，软件费用较低，且系统搭建容易，服务器启动快捷，适合于中小型企业或项目。目前，使用轻型框架开发项目非常普遍，常用的轻型框架包括 Hibernate、Struts、Spring、WebWork、Tapestry、JSF 等。

1.4.1 轻型框架简介

1. 使用轻型框架的好处

软件技术发展至今，面临各类复杂的应用系统的开发。软件系统开发任务涉及的知识更综合、内容更丰富、问题更繁多。如何能使程序开发效率高、工作效果好，这是轻型框架设计的目的所在。框架可以完成开发中的一些基础性工作，开发人员可以集中精力完成系统的业务逻辑设计。总体而言，使用轻型框架的好处有以下几方面。

（1）减少重复开发工作量、缩短开发周期、降低开发成本。

（2）使程序设计更为规范、程序运行更稳定。

（3）软件开发更能适应需求变化，且运行维护费用也较低。

2. 目前流行的框架组合

开发人员可以根据自己对框架的熟悉程度，在充分了解不同框架的性能的基础上，根据系统功能和性能要求，可以自由地选择不同框架来搭配使用。下面是一些常见的框架组合。

（1）JSP+Servlet+JavaBean+JDBC

（2）Struts+MySQL+JDBC

（3）Hibernate+JDBC+JSP

（4）Struts+Hibernate

（5）Hibernate+Spring

（6）Spring+Struts+JDBC

（7）Struts+Hibernate+Spring

（8）Struts+EJB

（9）JSF+Hibernate

（10）Tapestry+Hibernate+Spring