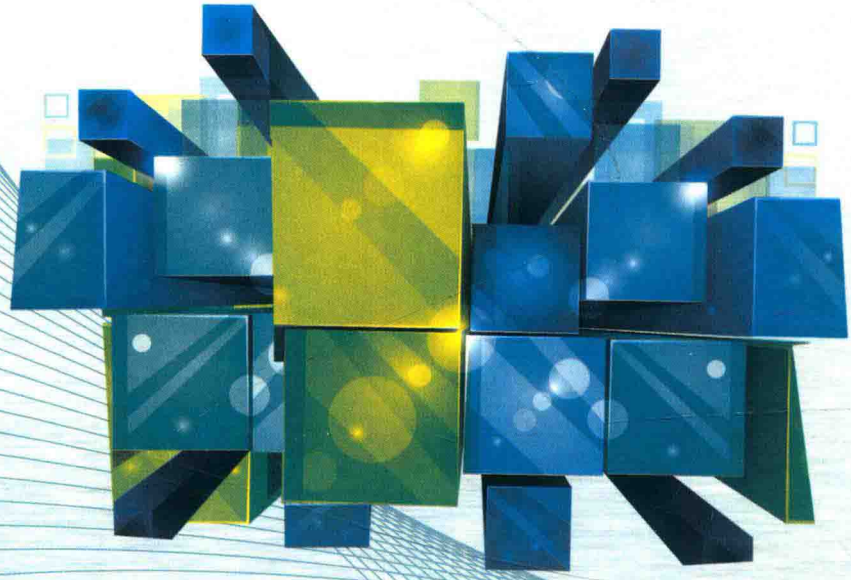
高等学校电子信息类“十三五”规划教材
与信息安全工程技术人才培养系列教材

Java EE 程序设计教程

Java EE Programming Tutorials

主　编　陈　丁
副主编　杨雪梅　吴雨芯



西安电子科技大学出版社
http://www.xduph.com

高等学校电子信息类“十三五”规划教材

应用型网络与信息安全工程技术人才培养系列教材

Java EE程序设计教程

主　编　陈　丁

副主编　杨雪梅　吴雨芯

参　编　赵　军　何林波　陈珊如　林春蔷

西安电子科技大学出版社

内 容 简 介

本书主要讲述基于 Spring、Hibernate、FreeMarker 三大主流开源框架进行 Java EE 应用开发的相关技术。书中，由 Java EE 产生的背景和发展过程入手，逐章节展开，从 Java EE 开发运行环境的搭建到最基础的 JSP+Servlet 开发，再到通过组装三大开源框架进行 Web 应用开发，由浅入深，循序渐进，非常适合初学者学习。

本书作者有多年的 Java EE 系列课程教学经验，作者正是结合自己的 Java EE 教学和 Web 应用系统开发经验编写了本书，比较详细地介绍了 Java EE 平台的基础构架和相关技术。全书共分为 10 章，内容包括：Java EE 概述、Servlet 开发、JSP 程序开发、Ajax 和 JSON、Hibernate 基础、Hibernate 高级编程、Spring 框架基础、Spring MVC 应用开发、FreeMarker 模板引擎和博客系统的设计与实现。其中最后一章为一个完整的案例，可帮助读者掌握 Java EE 开发全流程。本书内容丰富、注重实用，在理论知识点介绍完毕后一般都给出了使用的示范代码，部分代码有一定的实际设计意义。另外，每章后面附有习题，引导读者进行有关知识点的回顾和进一步的学习。

本书可作为高等院校计算机类、信息类、工程类、电子商务类和管理类各专业本专科生的教材，也可作为普通程序开发人员的自学教材或参考书。

图书在版编目(CIP)数据

Java EE 程序设计教程 / 陈丁主编. —西安：西安电子科技大学出版社，2018.2

ISBN 978-7-5606-4819-4

Ⅰ. ① J… Ⅱ. ① 陈… Ⅲ. ① JAVA 语言—程序设计 Ⅳ. ① TP312.8

中国版本图书馆 CIP 数据核字(2018)第 000721 号

策　　划　李惠萍

责任编辑　王　斌　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　陕西华沐印刷科技有限责任公司

版　　次　2018 年 2 月第 1 版　　2018 年 2 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　　印　张　21.875

字　　数　517 千字

印　　数　1～3000 册

定　　价　45.00 元

ISBN　978-7-5606-4819-4 / TP

XDUP　5121001-1

中国电子教育学会高教分会推荐
高等学校电子信息类“十三五”规划教材
应用型网络与信息安全工程技术人才培养系列教材

编审专家委员会名单

前　言

自 1999 年 Sun 公司首次发布 Java 企业版(Java EE，以前是 J2EE)以来，经过十多年的发展，Java EE 已经演变为当前企业的主流计算平台。目前，Java EE 的产业和技术链已经渗入到各行各业的企业信息系统中。尤其是随着整个 Java 平台的开源，越来越多的开源实体参与到 Java EE 许多重要的技术规范制定工作中，第三方开源框架如雨后春笋般地涌现出来，如 Struts、Spring、Hibernate、Mybatis、JBoss Seam、Tapestry、FreeMarker、Thymeleaf、Play 等。虽然层出不穷的框架能够有效地解决 Java EE 应用开发中的很多问题，但却让许多初学者不知所措、望而却步。为此，本书选择业内公认的三大主流框架 FreeMarker、Spring、Hibernate，提取每个框架中的常用功能进行了介绍，这样做的目的在于帮助读者在有限的时间内，尽快掌握基于这三大框架的在 Java EE 企业级的应用开发技术。

相对于 Java EE 规范中的 JNDI、EJB、JTA、JMS 而言，“FreeMarker+Spring+Hibernate”三大框架组合是一种轻量级的解决方案。每个框架各司其职，在不同的业务层面发挥作用。FreeMarker 框架作为前端网页的模板引擎，负责将网页模板和数据进行组装后呈现给用户；Hibernate 框架是一个 ORM 框架，负责对数据库的所有操作；Spring 框架是整个 Web 应用的核心框架，它既包含 Spring MVC(以前是用 Struts)，同时又把 FreeMarker 和 Hibernate 无缝整合在一起。这三者高效地组合，极大地降低了 Java EE 应用开发的难度，并在保证系统稳定性和扩展性的同时，大大提高了开发人员的工作效率。

本书在编写过程中，提倡“Learning by Doing”的学习方式，在讲解理论基础的同时，配合由浅入深的示例程序，希望读者亲自动手多加练习。全书面向有 Java 语言基础的读者，尽量用简单易懂的语言来描述相关的知识点，全部示例程序已在 Eclipse 上调试通过。由于 Struts 2 框架暴露出了较大的安全性漏洞，现在很多公司的 Web 项目的控制层技术框架已由 Struts 2 迁移到 Spring MVC，所以本书没有对 Struts 框架进行讲解，有这部分需求的读者可以参考其他教材。

在本书编写过程中，作者参考了互联网上一些技术文档和相关资源，在此向这些资料的作者深表谢意。同时，还要特别感谢笔者爱人的大力支持，她不仅承担了许多家务，还承包了书中第 7～10 章的绘图工作，正是在她的帮助和鼓励下才有这本书的出版。另外，作者也感谢西安电子科技大学出版社的编辑们，尤其是李惠萍女士的关心和建议，正是他们的努力才让本书得以顺利出版。

本书第1、2、3章由杨雪梅老师编写，第4、5、6章由吴雨芯老师编写，第7、8、9、10章由陈丁、赵军、何林波、陈珊如、林春蔷老师编写。由于作者水平有限，书中难免存在不妥之处，请广大读者批评指正。作者的联系邮箱为：chending@cuit.edu.cn，我们将虚心接受广大读者的建议和意见。

陈 丁

2017年10月

目　录

第 1 章　Java EE 概述

1.1　Java EE 的产生与发展

Java EE(Java Enterprise Edition)是建立在 Java 平台上的企业级应用的软件架构，同时是一种设计思想、一套规范、一种标准平台。Java EE 提供了一组可移植的、健壮的、可伸缩的、可靠的、安全的、快速的、可用于开发和运行服务器端应用程序的应用程序编程接口(Application Programming Interface，API)。

Sun 公司 1998 年 12 月发布了 JDK 1.2 版本，开始使用的名称为 Java 2 Platform，即 Java 2 平台，其平台包括标准版(Java 2 Standard Edition，J2SE)、企业版(Java 2 Enterprise Edition，J2EE)和微型版(Java 2 Micro Edition，J2ME)三个版本。J2SE 主要用于桌面应用软件的编程；J2ME 主要应用于嵌入式系统开发；J2EE 主要用于分布式网络程序的开发。2006 年 5 月，Sun 公司对 Java 的各种版本进行了更名，J2SE 更名为 Java SE，J2EE 更名为 Java EE，J2ME 更名为 Java ME。

1998 年 Sun 公司发布了 EJB 1.0 标准。EJB 为企业级应用中的数据封装、事务处理、交易控制等功能提供了良好的技术基础。至此，J2EE 平台的三大核心技术 JSP、Servlet 和 EJB 都已先后问世。1999 年，Sun 公司正式发布了 J2EE 的第一个版本。紧接着，遵循 J2EE 标准、为企业级应用提供支撑平台的各类应用服务软件相继涌现出来。IBM 的 WebSphere、BEA 的 WebLogic 都是这一领域里成功的商业软件平台。随着开源活动的兴起，JBoss 等开源的应用服务器软件也吸引了许多用户的注意力。2003 年，Sun 的 J2EE 版本已经升级到 1.4 版本，其中三个关键组件的版本也升级到了 JSP 2.0、Servlet 2.4 和 EJB 2.1。至此，J2EE 体系及相关的软件产品已经成为 Web 服务端开发的一个强有力的支撑环境。

但从 1999 年诞生的第一个 J2EE 版本一直到 J2EE 1.4 版本，J2EE 经常被人们所抱怨，即使实现一个简单的 J2EE 程序，都需要大量的配置文件，尽管有些配置文件并不是必需的。Sun 公司一直在试图改变此状况，但一直未能如愿。2002 年，J2EE 1.4 推出后，J2EE 的复杂程度达到顶点。尤其是 EJB 2.0，开发和调试的难度非常大。直到 2006 年 5 月，Sun 公司正式发布了 J2EE1.5 标准，并改名为 Java EE 5。Java EE 5 大大降低了开发难度。2009 年 12 月，Sun 公司正式发布了 Java EE 6 标准，同时 EJB 3.1 发布，进一步简化了使用，并改进了许多常见的开发模式。2013 年 6 月，甲骨文(Oracle)公司发布了 Java EE 7 标准，Java EE 7 扩展了 Java EE 6，加强了对 HTML 5 动态可伸缩应用程序的支持，提高了开发人员的生产力，更能满足苛刻的企业需求。

1.2　Java EE 应用模型

Java EE 使用多层的分布式应用模型，应用逻辑按功能划分为组件，各个应用组件根据

它们所在的层分布在不同的机器上。事实上，Sun 设计 Java EE 的初衷正是为了解决两层模式(Client/Server)的弊端。在传统模式中，客户端担当了过多的角色而显得臃肿，在这种模式中，第一次部署时比较容易，但难于升级或改进，可伸展性也不理想，而且经常基于某种专有的协议，通常是某种数据库协议，使得重用业务逻辑和界面逻辑非常困难。现在 Java EE 的多层企业级应用模型将两层化模型中的不同层面切分成许多层。一个多层化应用能够为不同的每种服务提供一个独立的层，以下是 Java EE 典型的四层结构，如图 1-1 所示。

- 客户层——运行在客户端机器上的客户层组件。
- Web 层——运行在 J2EE 服务器上的 Web 层组件。
- 业务层——运行在 J2EE 服务器上的业务层组件。
- EIS 层——运行在 EIS 服务器上的企业信息系统(Enterprise Information System，EIS)层软件。

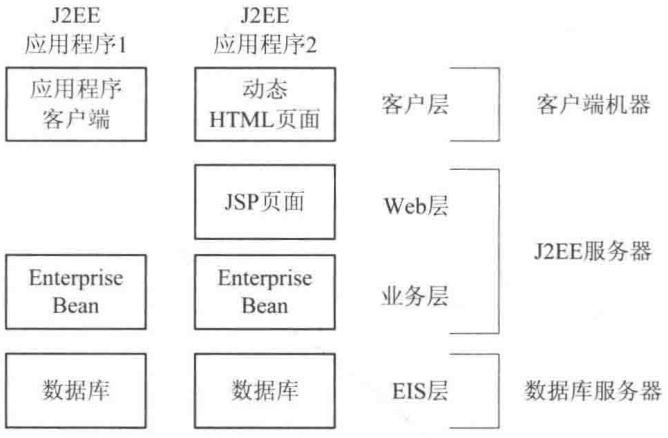


图 1-1　Java EE 的四层结构

1. Java EE 应用程序组件

Java EE 应用程序是由组件构成的。Java EE 组件是具有独立功能的软件单元，它们通过相关的类和文件组装成 Java EE 应用程序，并与其他组件交互。Java EE 说明书中定义了以下的 Java EE 组件：

(1) 应用程序客户端和 Applet 是客户层组件。

(2) Java Servlet 和 Java Server Page(JSP)是 Web 层组件。

(3) Enterprise Java Bean(EJB)是业务层组件。

2. 客户层组件

Java EE 应用程序可以是基于 Web 方式的，也可以是基于传统方式的。

3. Web 层组件

Java EE Web 层组件可以是 JSP 页面或 Servlet，按照 Java EE 规范，静态的 HTML 页面和 Applet 不算是 Web 层组件，如图 1-2 所示。

Web 层可能包含某些 JavaBean 对象，用来处理用户输入，并把输入发送给运行在业务层上的 Enterprise Bean 进行处理。

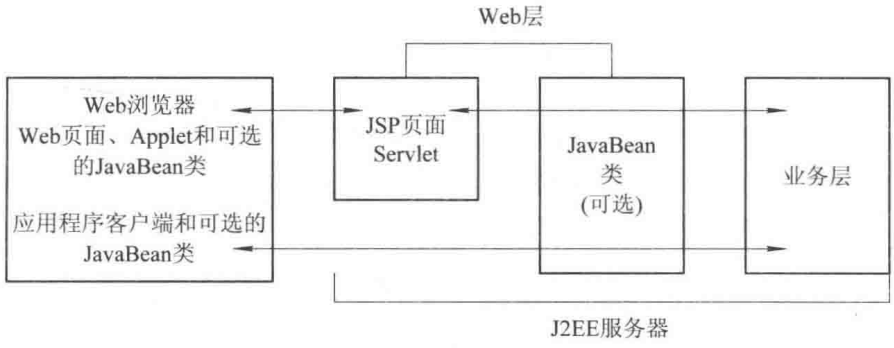


图 1-2　Web 层组件

4. 业务层组件

业务层组件代码用来满足银行、零售、金融等特殊商务领域的需要，由运行在业务层上的 Enterprise Bean 进行处理。图 1-3 表明了一个 Enterprise Bean 是如何从客户端程序接收数据，进行处理，并发送到 EIS 进行层储存的，这个过程也可以逆向进行。

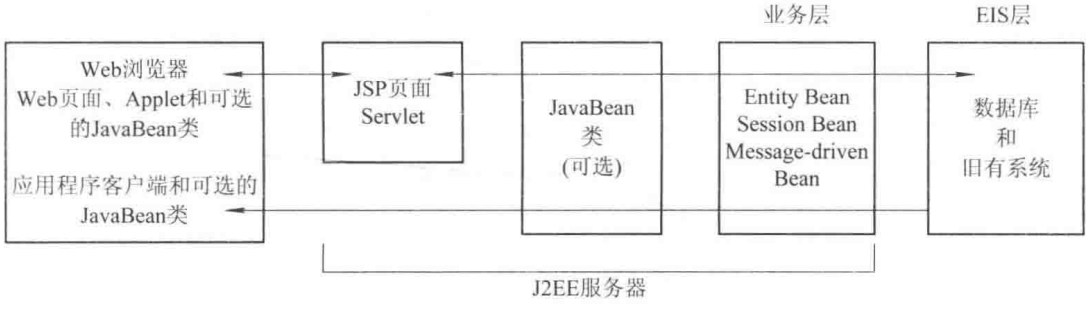


图 1-3　业务层组件

业务层有三种企业级的 Bean：会话(Session) Bean、实体(Entity) Bean 和消息驱动(Message-driven) Bean。会话 Bean 表示与客户端程序的临时交互，当客户端程序执行完后，会话 Bean 和相关数据就会消失；实体 Bean 表示数据库的表中一行永久的记录，当客户端程序中止或服务器关闭时，就会有潜在的服务保证实体 Bean 的数据得以保存；消息驱动 Bean 结合了会话 Bean 和 JMS 的消息监听器的特性，允许一个业务层组件异步接收 JMS 消息。

5. 企业信息系统层

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(如企业资源计划(ERP))、大型机事务处理、数据库系统和其他的遗留信息系统。例如，Java EE 应用组件可能为了数据库连接需要访问企业信息系统。

1.3　Java EE 常用技术

1. JDBC

JDBC(Java Database Connectivity，Java 数据库连接)是一种用于执行 SQL 语句的 Java

API。它可以为多种关系数据库提供统一访问的途径，其由一组用 Java 语言编写的类和接口组成。

2. JNDI

JNDI(Java Naming and Directory Interface，Java 命名和目录接口)是一组在 Java 应用中访问命名和目录服务的 API，它为开发人员提供了查找和访问各种命名和目录服务的通用、统一的方式。借助于 JNDI 提供的接口，JNDI 能够通过名字定位用户、机器、网络、对象服务等。

3. EJB

EJB(Enterprise Java Bean)是 Java EE 的服务器端组件模型，用于部署分布式应用程序，即把已编写好的程序打包放在服务器上执行，客户端则以 C/S 形式对服务器上的程序进行调用。凭借 Java 跨平台的优势，用 EJB 技术部署的分布式系统可不限于特定的平台。

4. RMI

RMI(Remote Method Invocation，远程方法调用)定义了调用远程对象中的方法的标准接口，并通过使用连续序列方式在客户端和服务器端传递数据。RMI 是一种被 EJB 使用的更底层的协议。

5. JSP

JSP(Java Server Page，Java 服务器页面)是由 Sun 公司倡导，并由许多公司参与建立的一种动态网页技术标准。类似 ASP 技术，JSP 是在传统的网页 HTML 文件(*.htm 和 *.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。用 JSP 开发的 Web 应用是跨平台的，既能在 Linux 下运行，也能在其他操作系统上运行，相比传统的 ASP 脚本的解释方式，JSP 运行速度更快。

6. Servlet

Servlet(Server Applet)是用 Java 编写的服务器端程序，其主要功能在于交互式地浏览和修改数据，生成动态 Web 内容。Servlet 运行于支持 Java 的应用服务器中，当被请求时开始执行，常用来扩展基于 HTTP 协议的 Web 服务器。

7. JMS

JMS(Java Messaging Service)是用于和面向消息的中间件相互通信的应用程序接口(API)。它既支持点对点的消息模型，又支持发布/订阅(Publish/Subscribe)的消息模型，还提供对经认可的消息传递、事务型消息的传递、一致性消息和具有持久性的订阅者等消息类型的支持。

8. JTA

JTA(Java Transaction Architecture，Java 事务架构)定义了面向分布式事务服务的标准 API，应用系统由此可存取各种事务监控，如事务范围的界定、事务的提交和回滚。

9. JTS

JTS(Java Transaction Service)是 CORBA OTS(CORBA Object Transaction Service，CORBA 对象事务服务)事务监控的基本实现，它具体规定了事务管理器的实现方式。JTS 事务管理器是在高层支持 JTA 规范，并且在较低层实现 OMG OTS specification 的 Java 映

像。JTS 事务管理器为应用服务器、资源管理器、独立的应用及通信资源管理器提供了事务服务。

10. Java IDL/CORBA

在 Java IDL 的支持下，开发人员可将 Java 和 CORBA 集成在一起，这样既可创建 Java 对象并使之可在 CORBA ORB 中展开，还可创建 Java 类并作为和其他 ORB 一起展开的 CORBA 对象的客户。

11. Java Mail and JAF(JavaBean Activation Framework)

Java Mail 是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，不仅支持 SMTP 服务器，也支持 IMAP 服务器。Java Mail 利用 JAF 来处理 MIME 编码的邮件附件。MIME 的字节流可被转换成 Java 对象或转换自 Java 对象，故大多数应用不需要直接使用 JAF。

12. JSF

JSF(Java Server Face，Java 构建框架)是一种用于构建 Web 应用程序的 Java 框架。它是 Java EE 表示层的技术，其主旨是为了使 Java 开发人员能够快速地开发基于 Java 的 Web 应用程序。较之于其他 Java 表示层技术，其最大优势是采用的组件模型和事务驱动确保了应用程序具有更高的可维护性。

13. Web Service

Web Service 是一种通过 WWW 的 HTTP 进行交互和交流的方式，可使运行在不同的平台和框架的软件应用程序之间进行互操作。Web Service 可以以松耦合的方式完成复杂的操作，具有强大的互操作能力和可扩展能力。

14. Web Socket

Web Socket 协议是 HTML 5 一种新的协议。它实现了浏览器与服务器全双工通信(Full-duplex)。HTML 5 定义了 Web Socket 协议，能更好地节省服务器资源和带宽，并达到实时通信的要求。在 Java EE 7 中实现了 Web Socket 协议。

15. AJAX

AJAX(Asynchronous Javascript and XML，异步 JavaScript 和 XML)是一种创建快速动态网页的开发技术，通过在后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。即在不重新加载整个网页的情况下，对网页的某部分进行更新。

16. JAX-RS

JAX-RS(Java API for RESTful Web Service)是一个 Java 编程语言的应用程序接口，支持按照表述性状态转移(REST)架构风格创建 Web 服务。它是 JAVA EE 6 引入的一个新技术。JAX-RS 使用了 Java SE 5 引入的 Java 标注来简化 Web 服务的客户端和服务端的开发和部署。它提供了一些标注，将一个资源类和一个 POJO Java 类封装为 Web 资源。

17. JSR

JSR(Java Specification Requests，Java 规范提案)指向 JCP(Java Community Process)，提出新增一个标准化技术规范的正式请求。任何人都可提交 JSR，以向 Java 平台增添新的 API 和服务。JSR 已成为 Java 界的一个重要标准，JSR 107 成为了 Java EE 7 的一部分。

18. XML

XML 是一种可用来定义其他标记语言的语言，它被用来在不同的商务过程中共享数据。XML 的发展和 Java 是相互独立的，但它和 Java 具有相同的目标，即平台独立性。通过将 Java 和 XML 组合，可得到一个完美的具有平台独立性的解决方案。

1.4 Java EE 7 新特性

Java EE 7 扩展了 Java EE 6，该版本的新特性主要集中在提高开发人员的生产力、加强对 HTML 5 动态可伸缩应用程序的支持和进一步满足苛刻的企业需求这三个方面。Java EE 7 使开发人员可以写更少的样板代码，通过丰富的组件来提供一个完整、全面、集成的堆栈以支持和构建最新的 Web 应用程序和框架，同时提供更具扩展性、丰富性和简易的功能。企业可以从 Java EE 7 的便捷式批处理、改进的扩展性等新功能中获益。下面通过对该版本中新增组件 WebSocket 1.0、JSON Processing 1.0、JAX-RS 2.0、JSF 2.2 和 JMS 2.0 的介绍对以上三个特性进行详细的剖析。

1. 减少冗余代码

Java EE 7 一直致力于减少在核心业务逻辑代码运行前必须执行的样板代码。减少样板代码的三大核心区域是默认资源、JMS 2.0 和 JAX-RS 客户端 API。默认资源是一个新的功能，要求平台提供商预配置一个默认的数据源和一个默认的 JMS 连接工厂。这可以让开发人员直接使用默认的资源而无需进行额外的定义。JMS 2.0 在可伸缩性和可移植性方面进行了重大的改进，减少了冗余代码。事实证明 JMS 2.0 是一个良好的规范，能够较好地满足企业的需求。

2. 更多带注释的 POJO

通过注释 Java EE 使开发人员更专注于 Java 对象的编程而无需关注繁琐的配置。现在 CDI 在默认情况下不需要使用“Bean.xml”文件就可直接使用。开发人员可以不需要任何配置而是简单地使用@Inject 来注入任何 Java 对象。新的资源注释@JMSDestination Definition 和@MailSessionDefinition，使得开发人员在源代码中就可以指定元数据资源，简化了 DevOps 体验。

3. 更紧密集成的平台

Java EE 6 引入了 Managed Beans 1.0 作为迈向 EJB、JSF Managed Bean 和 CDI Bean 方向发展的第一步。Java EE 7 继承了这一点，例如，对 JSF Managed Bean 进行了改进，以便更好地支持 CDI Bean。Java EE 7 为平台引入了易用的 EJB 容器管理事务，使用基于 CDI 拦截器的解决方案来保证事务在 CDI Managed Bean 和其他 Java EE 组件中的使用，并把注释@Transactional 应用到任何 CDI Bean 或者任何支持事务的方法中。

Bean Validation 在 Java EE 7 中使用广泛，并用可以应用于方法级别的验证，包括内置和自定义的约束。约束可被应用于方法的参数以及返回值。约束也可以使用灵活渲染和违反约束的字符串格式的 Java EE 的表达语言。

JAX-RS 2.0 也沿用了 Bean Validation。注释约束可以应用到公共构造函数的参数、方法参数、字段和 Bean 的属性。此外，它们还可以修饰资源类、实体参数和资源的方法。例

如，约束可以通过@ POST 和@ PUT 应用于 JAX-RS 方法参数来验证表单提交的数据。

4. 通过精简现有技术简化 Java EE

Java EE 7 中增加了许多新的特性，有些老的特性和功能已经被更简单的特性所替代或直接弃用。Java EE 6 为过时技术的弃用和功能的修剪引入了一个正式的流程，以下的 API 在 Java EE 7 中已成可选，但在 Java EE 8 中将会被移除：

① Java EE Management (JSR-77)。该 API 原本是用于为应用服务器创建监控管理的 API，可各大供应商对此 API 热情并不高。

② Java EE Application Deployment (JSR-88)。JSR-88 是当初用于 J2EE 应用程序在应用服务器上进行配置和部署的标准 API，可是该 API 始终没有得到众多供应商的支持。

③ JAX-RPC。该模块是早期通过 RPC 调用和 SOAP Web Service 进行交互的编程模型。Web Service 成熟后从 RPC 模型中分离了出来，被更加健壮和具备更多特性的 JAX-WS API 所替代。

④ EJB 2.x Entity Bean CMP。笨重、过度复杂的 EJB2.*的 Entity Bean 模型已经被 Java EE 5 的基于 POJO 的流行轻量级 JPA 持久层模型所代替。

5. 对 HTML 5 动态可伸缩应用程序的支持

HTML 5 是包括 HTML、JavaScript 和 CSS3 在内的一套技术组合，它加快了开发人员创建高度互动的应用程序的步伐。开发出的应用程序都可以高度互动的方式提供实时的数据，如聊天应用程序、比赛实况报道等，并且这些应用程序只需要编写一次，就可以应用在桌面、移动客户端等不同设备上，具有非常好的跨平台性。这些高度动态的应用程序，允许用户可以在任何地点任何时间进行访问，从而对服务器端向客户端传送数据的规模提出了更高的要求。Java EE 7 在更新现有技术如 JAX-RS 2.0、Java Server Face 2.2 和 Servlet 3.1 NIO 的基础上，又借助新的应用技术 WebSocket 和 JSON 处理为支持动态应用程序 HTML 5 奠定了坚实的基础。

6. 低延迟数据交换——Java API for WebSocket 1.0

越来越多的 Web 应用程序依赖于从中央服务器及时获取并更新数据。基于 HTTP 的 WebSocket 为解决低延迟和双向通信提供了一种解决方案。在 WebSocket API 的最基层是一个带注释的 Java 对象(POJO)，诸如客户端连接、接收消息和客户端断开这样的回调函数都可以用注释来指定。WebSocket API 的最基层支持发送和接收简单文本与二进制信息。API 的简单性使得开发人员可以快速入门。

当然，功能丰富的应用拥有更复杂的需求，因此需要支持对最基础的网络协议进行控制和自定义，WebSocket API 正好能够满足以上需求。另外，WebSocket 可利用现有 Web 容器的安全特性，开发人员只需付出较少的代价就可以建立良好的保密通信。

7. 简化应用数据分析和处理——Java API for JSON Processing 1.0

JSON 作为一个轻量级的数据交换格式被应用在许多流行的 Web 服务中，用来调用和返回数据。许多流行的在线服务都是使用基于 JSON 的 RESTful 服务。在 Java EE 7 之前，Java 应用程序使用了不同的类库去生成和解析 RESTful 服务中的 JSON 对象。然而，现在这个功能已被标准化。

通过 Java API 中的 JSON Processing 1.0，JSON 处理过程标准化为一个单一的 API，应