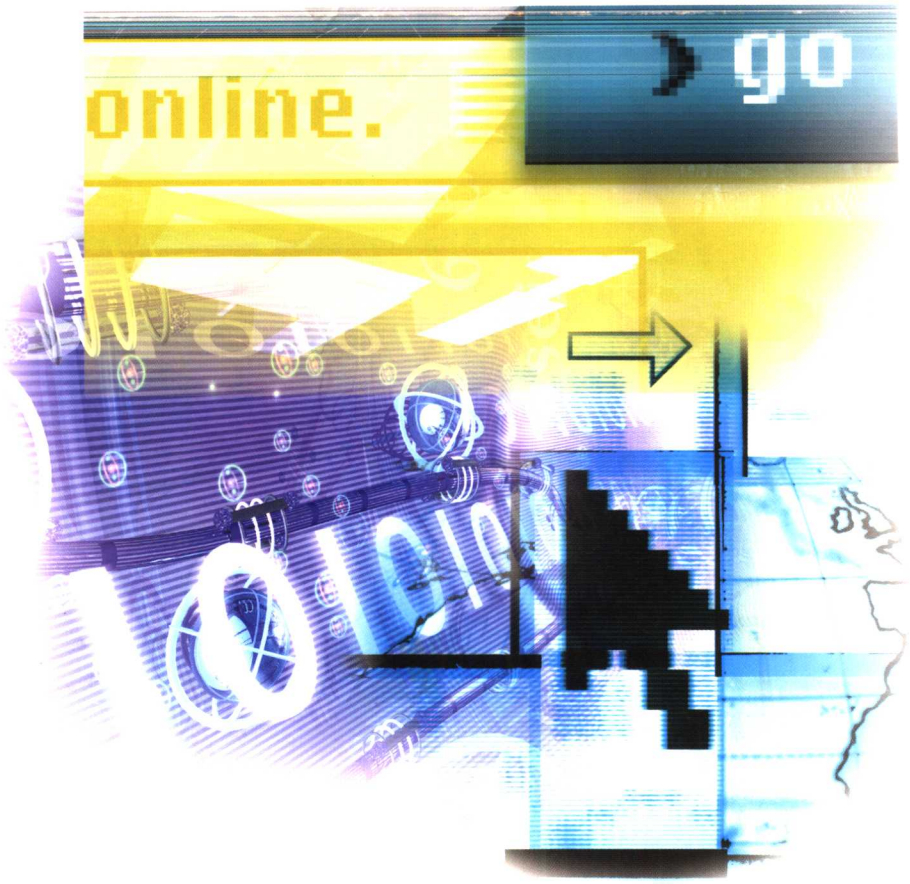
白领就业指南：Java Web开发设计师之路

赵　辉　姚胤含　编著



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

白领就业指南：
Java Web开发设计师之路

赵　辉　姚胤含　编著

電子工業出版社
Publishing House of Electronics Industry
北京・BEIJING

内 容 简 介

随着网络技术的迅猛发展，各种网站开发语言不断出现。Java目前正是流行的Web开发语言，基于J2EE技术平台的Java Web开发模式已经被广泛使用。

本书指导读者最大程度地使用Java工具实现Web应用的开发，循序渐进的讲解方式，使读者快速掌握Java Web的开发方法，为读者实现IT就业的梦想。本书内容涉及到Java Web开发的各项技术层面，案例丰富，可以使读者迅速达到实际应用的水平，成为一名合格的Java Web开发人员和设计师。

本书适合广大Java Web技术的爱好者和专业网络应用开发技术人员阅读，同时也可作为大中专院校学生学习的辅助教材。

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

白领就业指南：Java Web开发设计师之路/赵辉，姚胤含编著.—北京：电子工业出版社，2006.9
ISBN 7-121-02605-8

Ⅰ.白… Ⅱ.①赵… ②姚… Ⅲ.Java语言—程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2006）第044335号

责任编辑：徐云鹏
特约编辑：卢国俊
印　　刷：北京天竺颖华印刷厂
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
　　北京市海淀区翠微东里甲2号　邮编：100036
经　　销：各地新华书店
开　　本：787×1092 1/16　印张：23.625　字数：590千字
印　　次：2006年9月第1次印刷
定　　价：35.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换，若书店售缺，请与本社发行部联系。联系电话：010-68279077。质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

前　言

当前，Java 2 Enterprise Edition（J2EE）技术平台已经被广泛使用，在国内的软件开发公司，大多都将基于J2EE平台的Java Web开发技术应用到其项目中，为其各种类型的客户提供Web应用。可以说，掌握了Java Web开发技术，白领们在求职路上就会一帆风顺。

J2EE是一种利用Java 2平台来简化企业解决方案的开发、部署，并管理相关的复杂问题的体系结构。J2EE技术的基础就是核心Java平台或Java 2平台的标准版，J2EE不仅巩固了标准版中的许多优点，例如“编写一次、随处运行”的特性、方便存取数据库的JDBC API、CORBA技术，以及能够在Internet应用中保护数据的安全模式等，同时还提供了对EJB（Enterprise JavaBeans）、Java Servlets API、JSP（Java Server Pages）以及XML技术的全面支持。其最终目的就是成为一个能够使应用开发者大幅缩短开发时间和提高开发效率的体系结构。

在实际的应用中，使用J2EE框架作为Web应用开发的结构，是Java作为Web应用开发工具的重要途径和方法。而J2EE体系中的核心技术成员：JDBC、CORBA、EJB、Servlet和JSP等，则成为Java进行Web应用开发的一套组合利器。

本书采用循序渐进的方式，详细介绍了Java Web实际开发过程中涉及到的技术知识和相关的开发方法和经验，使读者迅速掌握相应的内容，达到实战的水平。同时，本书中的案例均已通过调试，且注释明晰，方便读者将案例应用到实际的工作中去。

本书分为9章，按照内容分别介绍了Java Web技术所涉及到的各个层面，内容安排上力求由浅入深，指导读者快速掌握Java Web开发技术，具体涵盖了如下方面：

- Java基础知识
- JSP页面语言
- JDBC数据库操作
- JavaBean和JSP的应用开发
- Servlet技术
- EJB技术应用和开发
- Java Web中的分布式技术RMI和CORBA的开发
- Java Web对XML的处理
- Struts框架开发

本书中拥有大量的实例，使读者在实践中学用Java Web技术，并可以很快达到一个较高的应用层次。作者长期使用Java Web，对该Web开发语言的特点、功能及应用有较深入的理解和体会，因此本书力图能够兼顾各方人士的不同需求，尤其对于实现Java就业的求职者，更具有指导意义！

无论是Java Web的使用新手，还是已经有相当经验的Java开发高手，在本书中都可以找到自己所需要的内容。通过对本书的学习，希望读者可以迅速高效地掌握Java Web的开发方法和经验。

本书的编写过程得到了北京美迪亚电子信息有限公司的大力协助，在此表示衷心的感谢。本书由赵辉、姚胤含编写，赵忠诚、王秀荣、李艳君、杨丽敏、赵敏、李军琪、李逸凡、姚忠朋、杨丽明等同志参与了书稿的整理工作。

由于本书的编写时间仓促和限于编者水平，书中错误与不妥之处在所难免，恳请读者批评指正。

为方便读者阅读，本书配套资料请登录“华信教育资源网”（http://www.hxedu.com.cn），在“教学资源”频道的“综合资源下载”栏目下载。

此为试读，需要完整PDF请访问：www.ertongbook.com

目　录

第1章

了解Java在当前Web应用开发中的地位

1.1 Java的主要发展历程及其发展方向

1995年，美国Sun Microsystems公司正式向IT业界推出了Java语言，该语言具有安全、跨平台、面向对象、简单、适用于网络等显著特点，当时以Web为主要形式的因特网正在迅猛发展，Java语言的出现迅速引起所有程序员和软件公司的极大关注，程序员们纷纷尝试用Java语言编写网络应用程序，并利用网络把程序发布到世界各地进行运行。包括IBM、Oracle、微软、Netscape、Apple、SGI等大公司纷纷与Sun Microsystems公司签订合同，被授权使用Java平台技术。

Java语言的诞生对IT产业带来了一次变革，从某种意义上讲对人们的日常生活也产生了深远的影响。Java作为一种类C/C++，其平台无关性以及与因特网发展紧密结合的特点，预计未来必定成为因特网和计算机应用的主流。Java当之无愧地被纽约时报评为1995年的十大科技成果之一，Java将作为一项重大发明载入科技史册！微软总裁比尔·盖茨曾在观察了一段时间后，十分惭愧地说："Java是长时间以来最卓越的程序设计语言"，并确定微软整个软件开发的战略从PC单机时代向着以网络为中心的计算时代转移，而购买Java则是他的重大战略决策。当然，微软与Sun也曾为Java对峙法庭，微软必将直接或间接将Java技术融入到其产品体系中。Sun公司的总裁Scott McNealy认为Java为Internet和WWW开辟了一个崭新的时代。

在经历了以大型机为代表的集中计算模式和以PC为代表的分散计算模式之后，因特网的出现使得计算模式进入了网络计算时代。网络计算模式的一个特点是计算机是异构的，即计算机的类型和操作系统是不一样的，例如，Sun工作站的硬件是SPARC体系，软件是UNIX中的Solaris操作系统，而PC的硬件是Intel体系，操作系统是Windows或者是Linux，因此，相应的编程语言基本上只是适用于单机系统，例如COBOL、FORTRAN、C、C++等；网络计算模式的另一个特点是代码可以通过网络在各种计算机上进行迁移，这就迫切需要一种跨平台的编程语言，使得用它编写的程序能够在网络中的各种计算机上正常运行，Java就是在这种需求下应运而生的。正是因为Java语言符合了因特网时代的发展要求，才获得了巨大的成功。

Sun公司绝没想到本想用于消费电子产品开发的编程语言却率先在网络中得到了广泛应用，但是也可以说是"东方不亮西方亮"，正是因为Java语言在设计目标上的正确性使得Java语言"是金子总会发光的"。C语言是面向过程的语言，也是使用率非常高的语言；而面向对象的思想引入到编程语言之后，C语言就被改造成为面向对象的C++语言，得到了广泛的应用。但是C++语言必须兼容C语言，因此，C++语言是面向过程和面向对象混合的语言。

Java语言产生于C++语言之后，是完全的面向对象的编程语言，充分吸取了C++语言的优点，采用了程序员所熟悉的C和C++语言的许多语法，同时又去掉了C语言中指针、内存申请和释放等影响程序健壮性的部分，可以说Java语言是站在C++语言这个“巨人的肩膀上”前进的。

Java语言的一个目标是跨平台，因此采用了解释执行而不是编译执行的运行环境，在执行过程中根据所在的不同的硬件平台把程序解释为当前的机器码，实现跨平台运行。而动态下载程序代码的机制完全是为了适应网络计算的特点，程序可以根据需要把代码实时地从服务器中下载过来执行，在此之前还没有任何一种语言能够支持这一点。

Java是一个小岛，盛产咖啡，而程序员往往喜欢喝咖啡，由此命名了Java语言。看来，目前Java这杯咖啡已经飘香在世界各地。

任何事物的产生既有必然的原因也有偶然的因素，Java语言的出现也验证了这一点。1991年，美国Sun Microsystems公司的某个研究小组为了能够在消费电子产品上开发应用程序，积极寻找合适的编程语言。消费电子产品种类繁多，包括PDA、机顶盒、手机等，即使是同一类消费电子产品所采用的处理芯片和操作系统也不相同，也存在着跨平台的问题。当时最流行的编程语言是C和C++语言，Sun公司的研究人员就考虑是否可以采用C++语言来编写消费电子产品的应用程序，但是研究表明，对于消费电子产品而言，C++语言过于复杂和庞大，并不适用，安全性也并不令人满意。于是，Bill Joy先生领导的研究小组就着手设计和开发出一种语言，称之为Oak。该语言采用了许多C语言的语法，提高了安全性，并且是面向对象的语言，但是Oak语言在商业上并未获得成功。时间转到了1995年，因特网在世界上蓬勃发展，Sun公司发现Oak语言所具有的跨平台、面向对象、安全性高等特点非常符合因特网的需要，于是改进了该语言的设计，要达到如下几个目标：

- 创建一种面向对象的程序设计语言，而不是面向过程的语言。
- 提供一个解释执行的程序运行环境，使程序代码独立于平台。
- 吸收C和C++的优点，使程序员容易掌握。
- 去掉C和C++中影响程序健壮性的部分，例如指针、内存申请和释放，使程序更安全。
- 实现多线程，使得程序能够同时执行多个任务。
- 提供动态下载程序代码的机制。
- 提供代码校验机制以保证安全性。

Java语言具有能独立于平台而运行、面向对象、可对动态画面进行设计与操作、坚固性等特点，又具有多线程、内置校验器用来防止病毒入侵等功能，所以用来在Internet上研制与开发软件时，特别受到用户的欢迎。

Java语言的有点主要表现在：简单、面向对象、自动内存管理、分布式计算、稳定、安全、解释执行、结构中立、平滑移植、多线程以及异常处理等方面。

1. 简单

由于Java的结构类似于C和C++，所以，一般熟悉C与C++语言的编程人员稍加学习就可以掌握Java的编程技术。Java所具有的自动内存管理机制也大大简化了Java程序设计开发。

2. 面向对象

简单地说，面向对象设计是一种以数据（对象）及其接口为中心的程序设计技术，面

向对象的设计可以说是定义程序模块如何“即插即用”的机制。Java的面向对象机制实际上可以看做是C++面向对象机制的延伸。Java提供了简单的类机制和动态的构架模型，对象中封装了它的状态变量和方法（函数、过程），实现了模块化和信息隐藏；而类则提供了一类对象的原型，通过继承和重载机制，子类可以使用或重新定义父类或超类所提供的过程，从而实现代码的复用。

3. 自动内存管理

Java的自动无用内存回收集（Auto Garbage Collectino）实现了内存的自动管理，因此简化了Java程序开发的工作，早期的GC（Garbage Collectino）对系统资源抢占太多而影响整个系统的运行，Java 2对GC进行的改良使Java的效率有了很大提高。GC的工作机制是周期性地自动回收无用存储单元。Java的自动内存回收机制在简化程序开发的同时，提高了程序的稳定性和可靠性。

4. 分布式计算

Java为程序开发提供了java.net包，该包提供了一组使程序开发者可以轻易实现基于TCP/IP的分布式应用系统。此外，Java还提供了专门针对因特网应用的类库，如URL、Java mail等。

5. 稳定性

人们最常见的应用程序错误就是“非法访问xxx内存”，其实质是程序指针使用出错。Java拥有一种指针（Pointer）模型，能够排除发生内存被覆盖和毁损数据的可能性。Java不采用指针算术法，而是提供真正的数组（Array），运行程序下标检查；另外，它也不会发生对象类型转换将一个任意数转换成指针的情形。Java的自动内存管理在减少编程工作的同时，大大减少了运行态错误。

6. 安全性

Java的设计目的是提供一个用于网络/分布式的计算环境。因此，Java强调安全性，如确保无病毒、小应用程序运行安全控制等。Java的验证技术是以公钥（Public-Key）加密算法为基础的，而且从环境变量、类加载器、文件系统、网络资源和命名空间等方面实施安全策略。

7. 解释执行

Java解释器（Interpreter）可以直接在任何已移植的解释器的机器上解释、执行Java字节代码，不需重新编译。当然，其版本向上兼容，因此，如果是高版本环境下编译的Java字节码到低版本环境下运行，也许会有部分问题。

8. 跨异构环境

Java是网络空间的“世界语”，编译后的Java字节码是一种“结构中立性”（Architecture Neutral）的目标文件格式，可以在所有提供Java虚拟机（JVM）的多种不同主机、不同处理器上运行。

9. 平滑移植

“Write once,run every where!”这也许是Java最诱人的特点。用Java开发而成的系统其移植工作几乎为零，一般情况下只需对配置文件、批处理文件做相应修改即可实现平滑移植。

此为试读，需要完整PDF请访问：www.ertongbook.com

10. 多线程

Java的多线程机制使程序可以并行运行。同步机制保证了对共享数据的正确操作。多线程使程序设计者可以用不同的线程分别实现各种不同的行为，而不需要采用全局的事件循环机制，因此，使用Java语言可以非常轻松地实现网络上的实时交互行为。

11. 异常处理

C语言程序员大都有使用goto语句来做条件跳转，Java编程中不支持goto语句。Java采用异常模型使程序的主流逻辑变得更加清晰明了，并且能够简化错误处理工作。

1.2 Java的工作原理

1.2.1 Java虚拟机

Java虚拟机是软件模拟的计算机，可以在任何处理器上（无论是在计算机中还是在其他电子设备中）安全并且兼容地执行保存在.class文件中的字节码。Java虚拟机的机器码保存在.class文件中，有时也可以称之为字节码文件。Java程序的跨平台主要是指字节码文件可以在任何具有Java虚拟机的计算机或者电子设备上运行，Java虚拟机中的Java解释器负责将字节码文件解释成为特定的机器码进行运行。Java源程序需要通过编译器编译成为.class文件（字节码文件）。

但是，Java虚拟机的建立需要针对不同的软硬件平台做专门的实现，既要考虑处理器的型号，也要考虑操作系统的种类。目前在UNIX、Linux、Windows和部分实时操作系统上都有Java虚拟机的实现。

1.2.2 无用内存自动回收机制

在程序的执行过程中，部分内存在使用过后就处于废弃状态，如果不及时进行无用内存的回收，就会导致内存泄漏，进而导致系统崩溃。在C++语言中是由程序员进行内存回收的，程序员需要在编写程序的时候把不再使用的对象内存释放掉；但是这种人为的管理内存释放的方法却往往由于程序员的疏忽而致使内存无法回收，同时也增加了程序员的工作量。而在Java运行环境中，始终存在着一个系统级的线程，专门跟踪内存的使用情况，定期检测出不再使用的内存，并进行自动回收，避免了内存的泄露，也减轻了程序员的工作量。

1.2.3 代码安全性检查机制

安全和方便总是相对矛盾的。Java编程语言的出现使得客户端机器可以方便地从网络上下载Java程序到本机上运行，但是如何保证该Java程序不携带病毒或者不怀有其他险恶目的呢？如果Java语言不能保证执行的安全性，那么它就不可能存活到今天。虽然有时候少数程序员会抱怨说Applet连文件系统也不能访问，但是正是各种安全措施的实行才确保了Java语言的生存。

字节码的执行需要经过三个步骤，首先，由类装载器（class loader）负责把类文件（.class文件）加载到Java虚拟机中，在此过程需要检验该类文件是否符合类文件规范；其次，字节码校验器（bytecode verifier）检查该类文件的代码中是否存在着某些非法操作，例如Applet

程序中写本机文件系统的操作；如果字节码校验器检验通过，由Java解释器负责把该类文件解释成为机器码进行执行。Java虚拟机采用的是“沙箱”运行模式，即把Java程序的代码和数据都限制在一定内存空间里执行，不允许程序访问该内存空间外的内存，如果是Applet程序，还不允许访问客户端机器的文件系统。

1.3　Java Web应用开发的组合利器

Java 2平台有三个版本：它们是适用于小型设备和智能卡的Java 2平台Micro版（Java 2 Platform Micro Edition，J2ME）、适用于桌面系统的Java 2平台标准版（Java 2 Platform Standard Edition，J2SE）、适用于创建服务器应用程序和服务的Java 2平台企业版（Java 2 Platform Enterprise Edition，J2EE）。其中最重要的就是J2EE平台。

J2EE 是一种利用Java 2平台来简化企业解决方案的开发、部署，并管理相关的复杂问题的体系结构。J2EE 技术的基础就是核心Java平台或Java 2平台的标准版，J2EE不仅巩固了标准版中的许多优点，例如“编写一次、随处运行”的特性、方便存取数据库的JDBC API、CORBA技术以及能够在Internet应用中保护数据的安全模式等，同时还提供了对EJB（Enterprise JavaBean）、Java Servlets API、JSP（Java Server Pages）以及XML技术的全面支持。其最终目的就是成为一个能够使企业开发者大幅缩短投放市场时间的体系结构。

在实际的应用中，使用J2EE框架作为进行Web应用开发的结构，是Java作为Web应用开发工具的重要途径和方法。而J2EE体系中的核心技术成员：JDBC、CORBA、EJB、Servlet和JSP等，则成为Java进行Web应用开发的一套组合利器。

1.3.1　J2EE分布式技术平台

J2EE 平台规范是一个由Sun公司定义的用于简化分布式企业级应用开发与部署的基于组件的模式。它提供了一个多层次的分布式应用模型和一系列开发技术规范。多层次分布式应用模型是根据功能把应用逻辑分成多个层次，每个层次支持相应的服务器和组件，组件在分布式服务器的组件容器中运行（如Servlet组件在Servlet容器上运行，EJB组件在EJB容器上运行），容器间通过相关的协议进行通信，实现组件间的相互调用。

J2EE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J2EE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，第一次部署的时候比较容易，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。它使得重用业务逻辑和界面逻辑非常困难。现在，J2EE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服务提供一个独立的层，如图1-1所示，是J2EE规范的4个层次及相应的组件。

这4层分别是运行在客户端机器上的客户端层（Client-Side Presentation）、运行在Web服务器上的表现层（Server-Side Presentation）、运行在EJB服务器上的业务层（Server-Side Business Logic）和运行在EIS服务器上企业信息系统层（Enterprise Information System）。其中，表现层和业务层共同组成了三层J2EE应用的中间层，其他两层是客户端层和存储层或企业信

息系统层。一般情况下，许多开放商把Web服务器和EJB服务器产品结合在一起发布，称为应用服务器或J2EE服务器。

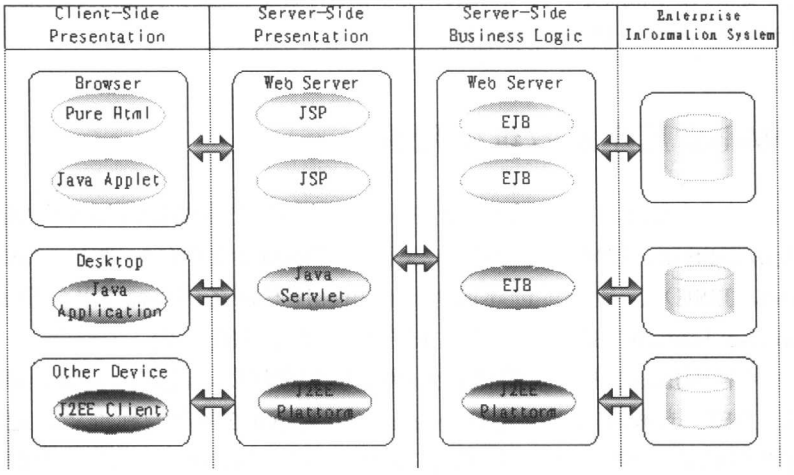


图1-1　J2EE规范的层次组件

J2EE是主流的技术体系，J2EE已成为一个工业标准。围绕着J2EE有众多的厂家和产品，其中不乏优秀的软件产品。合理集成以J2EE为标准的软件产品构建资金平台系统，可以得到较好的稳定性、高可靠性和扩展性。

1.3.2　JSP技术

JSP（Java Server Pages）是由Sun Microsystems公司倡导、许多公司参与一起建动态网页技术标准，其在动态网页的建设中有强大而特别的功能。

与ASP（Active Server Pages）、PHP一样，JSP在动态网页的建设中具有强大而特别的功能。SUN公司在Java的基础下开发出的JSP具有动态页面与静态页面分离，能够脱离硬件平台的束缚，以及编译后运行等优点而大大提高了其执行效率，逐渐成为因特网上的主流开发工具。

JSP规范是Web服务器、应用服务器、交易系统以及开发工具供应商间广泛合作的结果。Sun公司开发出这个规范来整合和平衡已经存在的规范，对Java编程环境（例如，Java Servlet和JavaBean）进行支持的技术和工具。其结果是产生了一种新的、开发基于Web应用程序的方法，给予使用基于组件应用逻辑的页面设计者以强大的功能支持。

JSP是一种服务器端嵌入Java代码的脚本语言，是开发Web动态同站重要而快速、有效的工具。在保证最大可操作性的前提下，它提供了比一般脚本语言（ASP、PHP）和CGI更快的执行速度。由于JSP是基于Java语言的，所以“一次编写，可随处运行”，即它的与平台无关的特性使其可以无缝地运行在UNIX/Linux和Windows平台上。在美国，EJB+Servlet+JSP几乎成为电子商开发标准。而在我国，JSP刚刚兴起，许多电子商务网站也开始使用JSP技术了。相信随着网络技术及电子商务的发展，JSP技术会很快在我国流行和普及的。

作为Java进行Web应用开发的页面语言，JSP具有如下的特点：

（1）将内容的生成和显示进行分离。

使用JSP技术，Web页面开发人员可以使用HTML或者XML（Extensih Marked Language）标识来设计和格式化最终页面。使用JSP标识或者脚本来生成页面上的动态内容，生成内容的逻辑被封装在标识和JavaBean组件中，并且捆绑在脚本中，所有的脚本在服务器端运行。如果核心逻辑被封装在标识和Bean中，那么其他人，如Web人员和页面设计者，能够编辑和使用JSP页面，而不影响内容的生成。

在服务器端，JSP引擎解释JSP标识和脚本，生成所请求的内容（例如，通过访问JavaBean组件，使用JDBC技术访问数据库，或者包含文件），并且将结果以HTML或XML的形式发送回浏览器。这有助于作者保护自己的代码，而又保证任何基于浏览器的完全可用性。

（2）生成可重用的组件。

绝大多数 JSP页面依赖于可重用的、跨平台的组件（JavaBean或者Enterprise JavaBean组件）来执行应用程序所要求的更为复杂的处理。开发人员能够共享和交换执行普通组件的操作，或者使得这些组件为更多的使用者所使用。基于组件的方法加速了总体开发过程，并且使得各种组织在他们现有的技能和优化结果的开发努力中得到平衡。

（3）采用标识简化页面开发。

Web页面开发人员不会都是熟悉脚本语言的编程人员。JSP技术封装了许多功能，这些功能是在XML标识中生成动态内容所需要的。标准的JSP标识能够访问和实例化JavaBean组件、设置或者检索组件属性、下载Applet，以及执行用其他方法更难于编码和耗时的功能。

通过开发定制化标识库，JSP技术是可以扩展的。今后，第三方开发人员和其他人员可以为常用功能创建自己的标识库。这使得Web页面开发人员能够使用熟悉的工具和如同标识一样地执行特定功能的构件来工作。

（4）内置脚本语言为Java。

由于JSP页面的内置脚本语言是基于Java编程语言的，而且所有的JSP页面都被编译成为Java Servlet，因此，JSP页面就具有Java技术的所有好处，包括健壮的存储管理和安全性。

（5）拥有Java的特点。

作为Java平台的一部分，JSP拥有Java编程语言“一次编写，可随处运行”的特点。随着越来越多的供应商将JSP支持添加到他们的产品中，开发人员可以使用自己所选择的服务器和工具，更改工具或服务器并不影响当前的应用。

1.3.3 Servlet技术

Servlet是Java 2.0中增加的一个全新功能。Servlet可以被认为是服务器端的Applet。Servlet被Web服务器执行，同Applet被Web浏览器执行是很类似的。

Java Servlet是运行在请求和面向请求服务器上的模块，比如一个支持Java的Web服务器，以及类似这样的场合。例如，一个Servlet可以从一个HTML订单表中获取数据，然后通过相应的算法来更新公司相应的订单数据库。

也就是说：Servlet能够像CGI脚本一样扩展Web服务器功能，但是Servlet占用很少密集资源，有很多用CGI脚本编制的一些站点由于访问量剧增，性能迅速下降，这是CGI脚本一个缺点。同时由于Servlet是用Java编写的，因此是跨平台的。

Servlet API（Applicatlon Programming Interface）是用来写Servlet程序的，编写Servlet时已经不会像编写CGI脚本那样，关心一个Servlet怎样被装载、Servlet运行的服务器环境是什么，或者用来传输数据的协议是什么等。这样，Servlet就可以融合在不同的Web服务器中了。

Servlet可以相当有效地替代CGI脚本：可以方便地产生容易编写而且运行快的动态文本；可以很方便地调试寻找出程序问题。

1.3.4 EJB技术

EJB的全称是Enterprise Java Bean，是Java中的商业应用组件技术。EJB结构中的角色EJB组件结构是基于组件的分布式计算结构，是分布式应用系统中的组件。

一个完整的基于EJB的分布式计算结构由6个角色组成，这6个角色可以由不同的开发商提供，每个角色所做的工作必须遵循Sun 公司提供的EJB 规范，以保证彼此之间的兼容性。这6个角色分别是EJB组件开发者（Enterprise Bean Provider）、应用组合者（Application Assembler）、部署者（Deployer）、EJB服务器提供者（EJB Server Provider）、EJB容器提供者（EJB Container Provider）、系统管理员（System Administrator）。

EJB分布式应用程序是基于对象组件模型的，低层的事务服务用了API技术。EJB技术简化了用Java语言编写的企业应用系统的开发、配置。EJB技术定义了一组可重用的组件：Enterprise Bean。可以利用这些组件像搭积木一样地建立的分布式应用程序。当把代码写好之后，这些组件就被组合到特定的文件中去。每个文件有一个或多个Enterprise Bean，在加上一些配置参数。最后，这些Enterprise Bean被配置到一个装了EJB容器的平台上。客户能够通过这些Bean的home接口，定位到某个Bean，并产生这个Bean的一个实例。这样，客户就能够调用Bean的应用方法和远程接口。

EJB服务器作为容器和低层平台的桥梁，管理着EJB容器和函数。它向EJB容器提供了访问系统服务的能力。例如：数据库的管理和事务的管理，或者对于其他的Enterprise的应用服务器。所有的EJB实例都运行在EJB容器中。容器提供了系统级的服务，控制了EJB的生命周期。EJB中的有一些易于使用的管理工具，如Security——配置描述器（The Deployment descriptor）定义了客户能够访问的不同的应用函数。容器通过只允许授权的客户访问这些函数来达到这个效果。Remote Connectivity——容器为远程链接管理低层的通信issues，而且对Enterprise Beas的开发者和客户都隐藏了通信细节。

EJB的开发者在编写应用方法的时候，就像是在调用本地的平台一样的。客户也不清楚他们调用的方法可能是在远程被处理的。Life Cyclemanagment——客户简单地创建一个Enterprise Bean的实例，并通常取消一个实例。而容器管理着Enterprise Bean的实例，使Enterprise Bean实现最大的效能和内存利用率。容器能够这样来激活和使Enterprise Bean失效，保持众多客户共享的实例池。

Transaction management——配置描述器定义了Enterprise Bean的事务处理的需求。容器管理着那些管理分布式事务处理的复杂事务。这些事务可能要在不同的平台之间更新数据库。容器使这些事务之间互相独立，互不干扰。保证所有的更新数据库都是成功发生的，否则就回滚到事务处理之前的状态。EJB组件是基于分布式事务处理的企业级应用程序的组件。所有的EJB都有如下的特点：EJB包含了处理企业数据的应用逻辑。定义了EJB的客户界面。

这样的界面不受容器和服务器的影响。于是，当一个EJB被集合到一个应用程序中去时，不用更改代码和重新编译。EJB能够被定制各种系统级的服务，例如安全和事务处理的特性，都不是属于EJB类的。而是由配置和组装应用程序的工具来实现。有两种类型的EJB：Session Bean和Entity Bean。Session Bean是一种作为单用户执行的对象。作为对远程的任务请求的相应，容器产生一个Session Bean的实例。一个Session Bean有一个用户，从某种程度上来说，一个Session Bean对于服务器来说就代表了它的那个用户。Session Bean也能用于事务，它能够更新共享的数据，但它不直接描绘这些共享的数据。Session Bean的生命周期是相对较短的。典型的是，只有当用户保持会话的时候，Session Bean才是活着的。一旦用户退出了，Session Bean就不再与用户相联系了。Session Bean被看成是瞬时的，因为如果容器崩溃了，那么用户必须重新建立一个新的Session对象来继续会话。

Session Bean典型地声明了与用户的互操作或者会话。也就是说，Session Bean在客户会话期间，通过方法的调用，掌握用户的信息。一个具有状态的Session Bean称为有状态的Session Bean。当用户终止与Session Bean互操作的时候，则会话就终止了，而且，Bean也不再拥有状态值。Session Bean也可能是一个无状态的Session Bean。无状态的Session Bean并不掌握它的客户的信息或者状态。用户能够调用Bean的方法来完成一些操作。但是，Bean只是在方法调用的时候才知道用户的参数变量。当方法调用完成以后，Bean并不继续保持这些参数变量。这样，所有的无状态的Session Bean的实例都是相同的，除非它正在方法调用期间。这样，无状态的Session Bean就能够支持多个用户，容器能够声明一个无状态的Session Bean。

Entity Bean对数据库中的数据提供了一种对象的视图。例如：一个Entity Bean能够模拟数据库表中一行相关的数据。多个客户端能够共享访问同一个Entity Bean，多个客户端也能够同时访问同一个Entity Bean。Entity Bean通过事务的上下文来访问或更新下层的数据。这样，数据的完整性就能够被保证。Entity Bean能存活相对叫长的时间，并且状态是持续的。只要数据库中的数据存在，Entity Bean就一直存活，而不是按照应用程序或者服务进程来说的。即使EJB容器崩溃了，Entity Bean也是存活的。Entity Bean生命周期能够被容器或者Bean自己管理。

Entity Bean是由主键（Primary Key）标识的。通常，主键与标识数据库中的一块数据（例如一个表中的一行）的主键是相同的。主键是客户端能够定位特定的数据块。

1.3.5　RMI技术

RMI应用程序通常包括两个独立的程序：服务器程序和客户机程序。典型的服务器应用程序将创建多个远程对象，使这些远程对象能够被引用，然后等待客户机调用这些远程对象的方法。而典型的客户机程序则从服务器中得到一个或多个远程对象的引用，然后调用远程对象的方法。RMI为服务器和客户机进行通信和信息传递提供了一种机制。

RMI 实现实际上由三个抽象层建立：

- Stubs/Skeletons Layer（存根/主架层）这一层会截获客户发出的对接口引用的方法调用，并且把这些调用重定向到一个远程对象。需要记住的是，这里的Stubs（存根信息）是针对客户方的，而Skeletons（主架信息）则位于服务器方。

此为试读，需要完整PDF请访问：www.ertongbook.com