

JSP+Oracle数据库 组建动态网站 经典实例

赵 光 等编著

JSP+Oracle
JSP+Oracle JSP+Oracle
JSP+Oracle



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

JSP+Oracle 数据库 组建动态网站经典实例

赵 光 等编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书系统地介绍了 JSP+Oracle 开发动态网站的交互式技术, 主要包括 JSP 预备知识、JSP 编程基础以及在 JSP 中如何使用数据库, 最后通过 6 个典型实例系统地介绍了各类网站从系统分析、代码设计到方案实施的全过程。

本书不但由浅入深地介绍了 JSP 程序设计的原理、方法和技术, 还提供了大量的 JSP 应用开发实例, 给出了相应的实用技巧、操作步骤及优化思路。

本书着重于 JSP 技术的应用和可操作性, 不但融合了编者在 JSP 应用开发过程中遇到的各种常见问题和应用案例, 还针对目前读者的需求进行了设计和布局。本书涉及到许多不同类型的应用, 多角度地引导读者学习相关知识, 使读者能够真正掌握 JSP 结合 Oracle 开发动态网站交互式的技术, 并在实际工作中灵活运用。

本书适合广大网站设计人员学习和使用。

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

JSP+Oracle 数据库组建动态网站经典实例/赵光等编著. —北京: 电子工业出版社, 2005.12
ISBN 7-121-01997-3

I. J… II. 赵… III. ①JAVA 语言—主页制作—程序设计 ②关系数据库—数据库管理系统, Oracle—程序设计 IV. ①TP393.092 ②TP311.138

中国版本图书馆 CIP 数据核字 (2005) 第 140479 号

责任编辑: 沈艳波 特约编辑: 张 健

印 刷: 北京市天竺颖华印刷厂

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

经 销: 各地新华书店

开 本: 787×1092 1/16 印张: 23.25 字数: 592 千字

印 次: 2005 年 12 月第 1 次印刷

印 数: 5 000 册 定价: 35.00 元 (含光盘 1 张)

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 zllts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前 言

本书介绍了 JSP 结合 Oracle 开发动态网站的交互式技术,以及相关技巧。全书以 6 个实用的系统为例,详细地讲述了用 JSP 创建开发动态网站的详细过程,全书逐层深入,难度逐渐加深。各章都穿插许多 JSP 开发的基本知识和技巧,同时突破了只讲编程技术,不讲开发思路的桎梏。书中处处渗透着软件工程的思想,希望通过每个系统的开发,提供给读者一些软件设计的理念。

本书最大特色是:所有的程序均来自工程实践,注重知识的系统性、连贯性和规范性,同时程序全面遵守高质量 Java 编程的编码规范。

全书共分 9 章。第 1 章介绍 JSP 与动态网站;第 2 章介绍 JSP 中连接数据库的技术;第 3 章介绍 Oracle 数据库;第 4 章介绍论坛系统;第 5 章介绍人事管理系统;第 6 章介绍资源下载网站;第 7 章介绍网上订票系统;第 8 章介绍新闻发布系统;第 9 章介绍留言版系统。

本书主要由赵光编写,参加编写的人员还有王波波、姜艳波、顾正大、艾丽香、赵辉、辛征、李志、李长林、张玉平。

本书附带的光盘包含每个实例的源代码、典型的技术难点、相应案例的常见问题解答和亮点展示。

编著者

2005 年 11 月

目 录

第 1 章 JSP 与动态网站	(1)
1.1 JSP 概述	(1)
1.1.1 JSP 与 ASP、PHP 3/4、ISAPI、CGI 的比较	(1)
1.1.2 使用 JSP 技术开发电子商务	(7)
1.1.3 JSP 开发实例	(9)
1.2 JSP 基本知识	(11)
1.2.1 JSP 页面的基本组成	(11)
1.2.2 JSP 脚本元素 (Scripting Elements)	(12)
1.2.3 JSP 基本语法	(12)
1.2.4 JSP 指令	(17)
第 2 章 JSP 中连接数据库的技术	(19)
2.1 JDBC 概述	(19)
2.1.1 JDBC 的使用步骤	(19)
2.1.2 JDBC API	(19)
2.1.3 JDBC API 使用场合	(21)
2.1.4 利用 JDBC 连接数据库	(22)
2.2 实例应用	(23)
2.2.1 JSP 建立和关闭数据库连接程序	(23)
2.2.2 JSP 读取 MySQL 数据库程序	(26)
2.3 SQL 语句运用	(28)
2.3.1 创建数据库	(28)
2.3.2 创建数据表	(28)
2.3.3 查询语句 SELECT	(31)
2.3.4 JDBC 中执行查询语句的步骤	(33)
2.3.5 一个比较通用的查询数据库的程序实例	(34)
2.4 参数式操作数据库	(37)
2.4.1 插入数据	(38)
2.4.2 修改记录数据	(40)
2.4.3 删除记录	(40)
2.4.4 建立索引	(41)
2.4.5 JSP 中参数操作	(41)
2.4.6 向数据库中添加数据的程序实例	(42)
2.4.7 既可添加记录又可删除记录的程序实例	(46)
第 3 章 Oracle 数据库	(52)
3.1 系统概述	(52)

3.1.1	系统硬件的选择与使用	(52)
3.1.2	Oracle 数据库安全性策略	(53)
3.2	应用数据库的设计与开发	(57)
3.2.1	数据和关系	(57)
3.2.2	设计和开发	(59)
3.3	Oracle 数据库的建模与设计	(60)
3.3.1	逻辑建模	(60)
3.3.2	物理设计	(61)
3.3.3	关于数据库参数的设计	(61)
3.3.4	与开发软件的接口问题	(61)
3.3.5	实例	(62)
3.4	Oracle 密码文件的创建、使用和维护	(63)
3.4.1	密码文件的创建	(63)
3.4.2	设置初始化参数	(63)
3.4.3	密码文件中增加、删除用户	(64)
3.4.4	使用密码文件登录	(64)
3.4.5	密码文件的维护	(65)
3.5	Oracle sqlplus 命令	(66)
3.5.1	Oracle 的启动和关闭	(66)
3.5.2	有效利用数据字典	(67)
3.5.3	查看数据库的 SQL	(69)
3.5.4	Oracle 用户连接的管理	(71)
3.5.5	SQL*Plus 使用	(71)
3.5.6	Oracle 逻辑备份的 SH 文件	(72)
3.6	自动备份 Oracle 数据库	(73)
3.6.1	导出数据库	(73)
3.6.2	磁带备份	(74)
3.6.3	异地备份	(74)
3.6.4	启动备份进程	(75)
3.7	Oracle 数据库恢复技术	(76)
3.7.1	恢复的方法	(76)
3.7.2	Oracle 表空间恢复方案	(76)
第 4 章	论坛系统	(80)
4.1	系统设计	(80)
4.1.1	系统结构	(80)
4.1.2	设计思路	(82)
4.2	数据库设计	(82)
4.2.1	Oracle 9i 数据库的安装与配置	(82)
4.2.2	Oracle 9i 数据库的创建	(87)
4.2.3	数据库需求分析	(92)

4.2.4	数据库逻辑设计	(92)
4.3	论坛系统的实现	(95)
4.3.1	界面头的设计	(95)
4.3.2	界面尾的设计	(96)
4.3.3	主界面的设计	(96)
4.3.4	用户注册的设计	(105)
4.3.5	用户登录的设计	(110)
4.3.6	管理员公告	(111)
4.3.7	新用户注册	(111)
4.3.8	排行榜	(112)
4.3.9	搜索	(115)
4.3.10	管理员登录页面	(117)
4.3.11	管理员修改页面	(119)
4.3.12	管理员删除页面	(123)
4.3.13	查看论坛信息	(125)
4.3.14	论坛详细信息	(129)
4.3.15	用户详细信息	(134)
第 5 章	人事管理系统	(137)
5.1	系统概述	(137)
5.1.1	应用背景	(137)
5.1.2	系统特点	(137)
5.1.3	系统设计思想	(138)
5.2	系统功能模块和结构设计	(138)
5.3	系统数据字典	(138)
5.4	配置 WebLogic	(141)
5.5	数据库系统的实现	(146)
5.5.1	数据库和表	(146)
5.5.2	建立 Oracle 数据库	(146)
5.5.3	数据库逻辑结构设计	(151)
5.6	人事管理系统模块设计及代码	(153)
5.6.1	工作管理模块	(153)
5.6.2	人员管理模块	(162)
5.6.3	用户管理模块	(185)
第 6 章	资源下载网站	(195)
6.1	建立 Oracle 数据库	(195)
6.1.1	创建用户	(195)
6.1.2	创建表	(196)
6.2	系统功能分析	(199)
6.3	用户模块	(199)
6.3.1	index.jsp	(199)

6.3.2	find.jsp	(208)
6.3.3	result.jsp	(211)
6.3.4	class.jsp	(222)
6.3.5	source.jsp	(230)
6.4	管理模块	(239)
6.4.1	login.jsp	(239)
6.4.2	ministrate.jsp	(242)
6.4.3	sourceadd.jsp	(243)
6.4.4	sourceedit.jsp	(250)
第 7 章	网上订票系统	(255)
7.1	系统功能概述	(255)
7.1.1	实现的功能	(255)
7.1.2	系统安装与配置	(256)
7.2	数据库的实现	(264)
7.2.1	建立 Oracle 数据库	(264)
7.2.2	数据库的逻辑设计	(265)
7.2.3	建立 Oracle 数据表	(266)
7.3	订票系统模块设计及代码	(269)
7.3.1	用户管理模块	(274)
7.3.2	订票管理模块	(285)
第 8 章	新闻发布系统	(300)
8.1	系统功能设计	(300)
8.1.1	系统设计规划	(300)
8.1.2	系统设计思想	(301)
8.1.3	设计目标	(301)
8.2	创建数据库	(302)
8.2.1	新建数据库	(302)
8.2.2	创建用户和表	(306)
8.3	系统详细设计	(308)
8.3.1	Bean 的编写	(308)
8.3.2	head.jsp	(310)
8.3.3	foot.jsp	(313)
8.3.4	index.jsp	(314)
8.3.5	admin.jsp	(317)
8.3.6	check.jsp	(319)
8.3.7	manage.jsp	(320)
8.3.8	dealing.jsp	(322)
8.3.9	logout.jsp	(324)
8.3.10	change-pwd.jsp	(325)

第 9 章 留言版系统	(328)
9.1 系统概述	(328)
9.1.1 目标设计	(328)
9.1.2 设计思路	(328)
9.1.3 系统功能分析与设计	(328)
9.2 数据库设计	(329)
9.2.1 数据库的建立	(329)
9.2.2 数据库的导入	(329)
9.2.3 数据库需求分析	(333)
9.2.4 数据库逻辑结构设计	(333)
9.2.5 数据库的连接	(333)
9.3 系统模块的实现	(334)
9.3.1 用户注册	(335)
9.3.2 用户登录	(337)
9.3.3 主页面	(338)
9.3.4 留言信息	(347)
9.3.5 回复留言	(353)
9.3.6 删除模块	(358)

第 1 章 JSP 与动态网站

利用 JSP 和其他有关技术, 一个用户可以将一个 HTTP 请求发送给一个可执行的应用程序而不是一个静态的 HTML 文件, 服务器将会立即运行这个限定的程序, 对用户的输入做出反应, 将处理结果返回客户端, 或者对数据的记录进行更新。通过这个模型, 就可以在服务器和客户端之间有效地进行交互。

1.1 JSP 概述

JSP (Java Server Pages) 是由 Sun Microsystem 公司倡导, 许多公司参与, 共同建立的一种动态网页技术标准, JSP 技术是在 Servlet 技术基础上发展起来的。它正在飞速发展, 现在已经成为 Java 服务器编程的重要组成部分。

在以前, 开发网络数据库应用程序主要由专业程序员来做, 因为如同 ISAPI、CGI 类型的开发技术, 使用 C/C++、Perl、Delphi、Visual Basic 作为开发语言, 开始比较难, 程序代码难写难懂, 调试修改也十分困难, 如果不是熟练的程序员, 并且没有无比的耐心, 那么将很难开发出一个完美的 Web 应用程序。后来, 代替 CGI、ISAPI 的技术纷纷出台, 如 Microsoft 的 ASP (Active Server Pages)、PHP (Personal HomePage Tools)、Allaire 的 CFML、Sun 的 Java/Servlet/JSP 等。JSP/Servlet 技术就是它们当中的佼佼者, 在 Web 开发领域独占鳌头。

1.1.1 JSP 与 ASP、PHP 3/4、ISAPI、CGI 的比较

开发 Internet 应用程序的方法大概有以下几种方案: ASP、PHP、JSP、ISAPI、CGI, 这些开发技术各有其优缺点。这里我们选择的是 JSP+Servlet。

为了给读者一个形象的概念, 下面先举一个简单的例子来说明用 JSP 开发 Web 程序是如何的简单。

在计算机语言中, 第一个入门的程序往往就是著名的“Hello World!”程序, 即输出一行“Hello World!”的文字。

下面分别用 JSP、ASP、PHP、CGI、ISAPI 五种技术输出“Hello World!”这一 Web 页面, 从而比较一下用这五种技术开发 Web 应用程序的难易程度。

例如: helloworld.jsp

```
<%@ page language=="java" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> Hello World! </TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
<%
```

```

out.print("Hello World!");
%>
</BODY>
</HTML>

```

运行效果如图 1-1 所示。

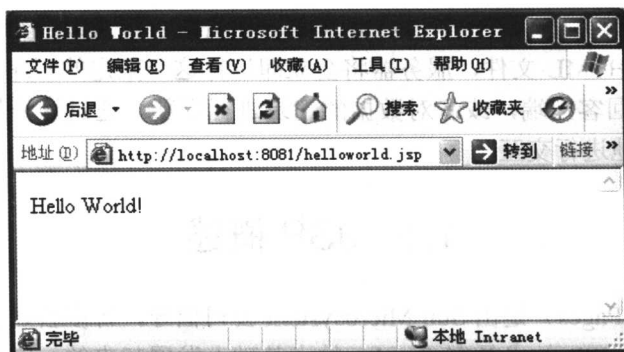


图 1-1 helloworld.jsp 程序的运行效果

例如: hello.asp

```

<%@ language="vbscript" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> Hello World </TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
<%
response.write "Hello World!"
%>
</BODY>
</HTML>

```

运行效果如图 1-2 所示。

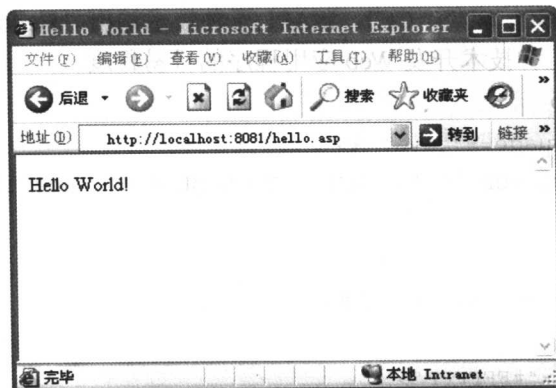


图 1-2 hello.asp 程序的运行效果

例如: **Hello.cgi**

```
#!/usr/bin/perl
print "Hello World!\n";
```

例如: **Hello.php3**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> Hello World! </TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
    <?php
    echo "Hello World!";
    ?>
</BODY>
</HTML>
```

例如: **helloworld.dll**

下面是我们使用 Microsoft Visual C++ 6.0 开发的 ISAPI 程序 **Helloworld.dll**, 该程序完整的源代码有十多个文件, 现在只列出最重要的程序文件 **Helloworld.cpp** 和它的头文件 **Helloworld.h**, 由于篇幅关系, 其他文件在此不再列出。

```
// HELLOWORLD.CPP - Implementation file for your Internet Server
// helloworld Extension
#include "stdafx.h"
#include "helloworld.h"
/////////////////////////////////////////////////////////////////
// The one and only CWinApp object
// NOTE: You may remove this object if you alter your project to no
// longer use MFC in a DLL.
CWinApp theApp;
/////////////////////////////////////////////////////////////////
// command-parsing map
BEGIN_PARSE_MAP(CHelloworldExtension, CHttpServer)
    // TODO: insert your ON_PARSE_COMMAND() and
    // ON_PARSE_COMMAND_PARAMS() here to hook up your commands.
    // For example:
    ON_PARSE_COMMAND(Default, CHelloworldExtension, ITS_EMPTY)
    DEFAULT_PARSE_COMMAND(Default, CHelloworldExtension)
END_PARSE_MAP(CHelloworldExtension)
/////////////////////////////////////////////////////////////////
// The one and only CHelloworldExtension object
CHelloworldExtension theExtension;
/////////////////////////////////////////////////////////////////
```

```

// CHelloworldExtension implementation
CHelloworldExtension::CHelloworldExtension()
{
}

CHelloworldExtension::~CHelloworldExtension()
{
}

BOOL CHelloworldExtension::GetExtensionVersion(HSE_VERSION_INFO* pVer)
{
    // Call default implementation for initialization
    CHttpServer::GetExtensionVersion(pVer);
    // Load description string
    TCHAR sz[HSE_MAX_EXT_DLL_NAME_LEN+1];
    ISAPIVERIFY(::LoadString(AfxGetResourceHandle(),
        IDS_SERVER, sz, HSE_MAX_EXT_DLL_NAME_LEN));
    _tcscpy(pVer->lpszExtensionDesc, sz);
    return TRUE;
}

BOOL CHelloworldExtension::TerminateExtension(DWORD dwFlags)
{
    // extension is being terminated
    //TODO: Clean up any per-instance resources
    return TRUE;
}

////////////////////////////////////
// CHelloworldExtension command handlers
void CHelloworldExtension::Default(CHttpServerContext* pCtxt)
{
    StartContent(pCtxt);
    WriteTitle(pCtxt);
    // *pCtxt << _T("This default message was produced by the Internet");
    // *pCtxt << _T(" Server DLL Wizard. Edit your CHelloworldExtension::
    Default());
    // *pCtxt << _T(" implementation to change it.\r\n");
    *pCtxt << _T(" Hello World!\n")
    EndContent(pCtxt);
}

// Do not edit the following lines, which are needed by ClassWizard.
#if 0
BEGIN_MESSAGE_MAP(CHelloworldExtension, CHttpServer)
    //{{AFX_MSG_MAP(CHelloworldExtension)
    //}}AFX_MSG_MAP

```

```

END_MESSAGE_MAP()

#endif    // 0

////////////////////////////////////

// If your extension will not use MFC, you'll need this code to make
// sure the extension objects can find the resource handle for the
// module. If you convert your extension to not be dependent on MFC,
// remove the comments around the following AfxGetResourceHandle()
// and DllMain() functions, as well as the g_hInstance global.
/****

static HINSTANCE g_hInstance;
HINSTANCE AFXISAPI AfxGetResourceHandle()
{
    return g_hInstance;
}

BOOL WINAPI DllMain(HINSTANCE hInst, ULONG ulReason,
                    LPVOID lpReserved)
{
    if (ulReason == DLL_PROCESS_ATTACH)
    {
        g_hInstance = hInst;
    }
    return TRUE;
}

****/

#if !defined(AFX_HELLOWORLD_H__435B5A45_8A89_11D4_8192_704264C1BD81__
INCLUDED_)
#define AFX_HELLOWORLD_H__435B5A45_8A89_11D4_8192_704264C1BD81__INCLUDED_
// HELLOWORLD.H - Header file for your Internet Server
// helloworld Extension
#include "resource.h"
class CHelloworldExtension : public CHttpServer
{
public:
    CHelloworldExtension();
    ~CHelloworldExtension();
// Overrides
// ClassWizard generated virtual function overrides
// NOTE - the ClassWizard will add and remove member functions here.
// DO NOT EDIT what you see in these blocks of generated code !
//{{AFX_VIRTUAL(CHelloworldExtension)
public:
    virtual BOOL GetExtensionVersion(HSE_VERSION_INFO* pVer);

```

```

//}}AFX_VIRTUAL
virtual BOOL TerminateExtension(DWORD dwFlags);
// TODO: Add handlers for your commands here.
// For example:
void Default(CHttpServerContext* pCtxt);
DECLARE_PARSE_MAP()
//{{AFX_MSG(CHelloworldExtension)
//}}AFX_MSG
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
before the previous line.
#endif

// !defined(AFX_HELLOWORLD_H_435B5A45_8A89_11D4_8192_704264C1BD81__INCLUDED)
Hello.dll 运行效果如图 1-3 所示。

```



图 1-3 Hello.dll 程序的运行效果

通过以上的实例比较，我们可以看出，用 JSP 开发 Web 程序是相当简单的一种方法，对于大型项目更是如此，ISAPI 则是最复杂的方法。

为了简明起见，下面将 JSP、ASP、PHP、ISAPI、CGI (Perl) 的性能列表进行比较，如表 1-1 所示。

表 1-1 JSP、ASP、PHP、ISAPI 及 CGI 性能比较

项 目	JSP	ASP	PHP	ISAPI	CGI(Perl)
易学性	容易	很容易	很容易	难	较难
运行速度	快	较快	较快	很快	慢
运行开销	小	较大	较大	小	较大
平台	均可	Windows	均可	Windows	均可
扩展性	很好	好	不好	不好	不好
安全性	好	不好	好	一般	好
分布式处理	支持	支持	不支持	支持	不支持

续表

项 目	JSP	ASP	PHP	ISAPI	CGI(Perl)
函数支持	广	不广	广	广	不广
数据库支持	多	多	多	多	多
应用程度	较广泛	较广泛	较广泛	少	少
厂商支持程度	广泛	少	较广泛	少	广泛
对 XML 的支持	支持	不支持	支持	不支持	不支持
对组件的支持	支持	支持	不支持	支持	不支持

由表 1-1 中我们不难看出 JSP 的优越性与独到之处。

1.1.2 使用 JSP 技术开发电子商务

近年来,随着 Internet 及全球信息网应用的快速发展,越来越多的人想使用其无国界、无时间、无地域限制的便利环境来经营拓展商务。因此电子商务(Electronic Commerce/EC)越来越流行,越来越多的企业也想使用该技术与方法进行商业上的交易以减低成本。

当然,还有更多的公司想使用电子商务的技术来架设一个虚拟的店面进行营业交易,企业与消费者通过网络完成交易,不但能使企业降低成本,还可以让消费者在一个舒适的地点享受逛街与购物的乐趣,因此电子商务所带来的商机与便利的前景将会是无比广阔的。随着电子商务风潮的扩大,将会有更多的公司连上网络进行各项业务,而不只是将公司或产品介绍网页放在网站上供人浏览而已。

电子商务并没有一致性的定义,一般而言,只要交易双方均以电脑通过网络来进行交易都可以称之为电子商务。电子商务的具体操作过程如下:消费者通过电脑网络进入电子商店,浏览商店目录并购买需要的商品,此动作称之为在线购物(所购买的商品可以为数字或实体商品),最后由电子工具支付货款。

电子商店是建立在网络世界中的虚拟商店,与传统购物方式不同的是消费者不必出门,而是在家中通过电脑来连线选购。商家也不用将所有货物陈列在店面中(可以减少店面的昂贵租金),而是将商品的图片介绍与影像以多媒体的方式通过网络呈现在消费者的电脑屏幕中。我们以大家熟悉的 Amazon 公司为例,该公司是一间位于西雅图的书店(没有店面),该公司在 Internet 上卖各式各样的书籍(现在还卖其他物品,例如:CD、礼物等商品),当消费者下了订单之后,Amazon 公司会将该图书送到消费者手中,因此不需要库存地和店面。

电子商务的优点不仅在此,它还可以解除一些业务上的时空限制,例如:电子商店(网络服务器)可以全天候服务客户,而且搬迁容易,更能随时随地让客户通过网络订购商品。电子商务除了支持电子化环境中商品或服务的销售与购买活动之外,更强调商业流程中整体效率与效益的提升。例如:更好的服务品质,更高的客户满意度,较低的建制成本以及更有效率的交换过程等。

电子商务应用的涵盖范围相当广泛,可以分为以下三大类别:

- ① 企业对客户(Business-to-Customer/B to C);
- ② 企业组织内部(Intra-Business or Intra-Organization);
- ③ 企业对企业(Business-to-Business/B to B)。

电子商务应用系统必须具备以下特征:

- 基于标准;
- 以服务器为中心;
- 可伸缩;
- 能利用已有的核心系统;
- 可快速部署和易用;
- 易于管理。

我们一般是用基于 Java 2.0 平台的 JSP 技术构建安全高效的电子商务应用系统。为了适应未来商业应用和分布式处理的需要, Sun Microsystem 公司在 Java 1.1 的基础上推出了 Java 2.0 (Java Development Kit 1.2) 开发工具包, 提供了 Java 企业应用编程接口 (Java Enterprise API), 如图 1-4 所示。

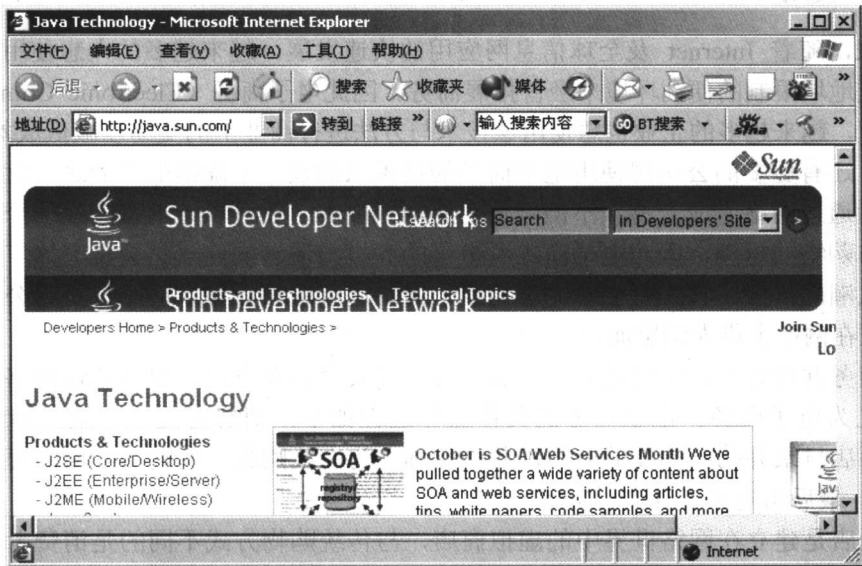


图 1-4 Sun 公司 Java 开发部门的网站

为企业计算以及电子商务应用系统提供了相关的技术和丰富的类库。

① JDBC (Java Database Connectivity) 提供了连接各种数据库的统一接口。

② EJB (Enterprise JavaBeans) 使得开发者方便地创建、部署和管理跨平台的基于组件的企业应用。

③ Java RMI (Java Remote Method Invocation) 用来开发分布式 Java 应用程序。一个 Java 对象的方法能被远程 Java 虚拟机调用。这样, 远程方法激活可以发生在对等的两端, 也可以发生在客户端和服务端之间, 条件是双方的应用程序都是用 Java 编写的。

④ Java IDL (Java Interface Definition Language) 提供与 CORBA (Common Object Request Broker Architecture) 的无缝操作, 这使得 Java 能够集成异构的商务信息资源。

⑤ JNDI (Java Naming and Directory Interface) 提供从 Java 平台到系统的无缝连接, 这个接口屏蔽了企业网络所使用的各种命名和目录服务。

⑥ JMAPI (Java Management API) 为异构网络系统、网络和服务管理的开发提供一整套丰富的对象和方法。

⑦ JMS (Java Message Service) 提供企业消息服务, 如可靠的消息队列、发布和订阅通