第 1 章　JSP 概述

1.1　JSP 简介

早期的网站一般都是由静态网页制作的，静态网页指网页的内容是固定的，不会根据浏览者的不同需求而改变。静态网页一般使用 HTML（超文本标记语言）进行编写。静态网页一般是运行于客户端的程序、网页、插件、组件，显示内容属于静态的，是永远不变的。在信息技术日新月异的今天，静态网页已无法满足人们对信息丰富性和多样性的强烈需求。现在大部分的网页都是动态网页，动态网页是指在接到用户访问要求后动态生成的页面，页面内容会随着访问时间和访问者发生变化，动态网页一般是在服务器端运行的程序、网页等。目前，解决 Web 动态网站的开发技术很多，而 JSP 是目前应用最为广泛的一种。

JSP 的全称是 Java Server Page，是 Sun 公司于 1999 年推出的动态网页技术。利用这一技术可以建立安全、跨平台的 Web 应用程序。JSP 以 Java 编程语言作为脚本语言，JSP 的安全性和跨平台得益于 Java 语言，这是因为 Java 语言具有不依赖于平台、面向对象和安全等优良特性，已经成为网络程序设计的佼佼者。许多和 Java 有关的技术得到了广泛的应用和认可，JSP 技术就是其中之一。

目前，Web 动态网站开发技术有 ASP、PHP、JSP 等，ASP 是微软（Microsoft）公司开发的动态网页语言，只能运行于微软公司的服务器产品和 PWS 之上。在 UNIX 环境下可以通过 ChiliSoft 的插件来支持 ASP，但是 ASP 本身的功能有限，必须通过 ASP+COM 的组合来扩充，UNIX 下的 COM 实现起来非常困难。PHP3.0 可在 Windows、UNIX 和 Linux 的 Web 服务器上正常运行，还支持 IIS 和 Apache 等通用 Web 服务器，用户更换平台时，无须变换 PHP 代码即可使用。与 ASP、PHP 相比，JSP 是一种完全与平台无关的开发技术，完全克服了目前 ASP 和 PHP 的脚本级执行的缺点。它将极高的运行效率、较短的开发周期、超强的扩展能力、完全开放的技术标准和自由的开发方式等众多的完美特性集于一身。JSP 可以在 Servlet 和 JavaBean 的支持下，完成功能强大的动态网站程序的开发，使构造基于 Web 的应用程序更加容易和快捷。

JSP 应该是未来发展的趋势，已广泛地应用于电子商务、电子政务等各个行业的管理应用软件中，一些大的电子商务解决方案提供商都采用 JSP。比较著名的有 IBM 公司的 E-business，它的核心技术采用 JSP 的 Web Sphere。

1.2　JSP 工作原理

当客户端浏览器向服务器发出请求访问一个 JSP 页面时，服务器根据该请求加载相应的 JSP 页面，并对该页面进行编译，然后执行。

JSP 的具体处理过程如下所示。

（1）客户端通过 Web 浏览器向 JSP 服务器发送请求。

（2）JSP 服务器检查是否已经存在 JSP 页面对应的 Servlet 源代码，若存在则进行第（3）步，否则转至（4）。

（3）JSP 服务器检查 JSP 页面是否有更新修改，若存在更新修改则进行第（4）步，否则转（5）。

（4）JSP 服务器将 JSP 代码转译为 Servlet 源代码。

（5）JSP 服务器将 Servlet 源代码编译后执行。

（6）将产生的结果返回到客户端。

JSP 工作原理如图 1-1 所示。

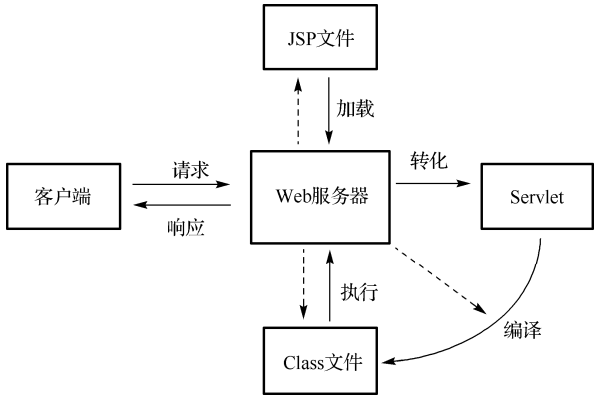

图 1-1　JSP 工作原理

从前面的介绍中，可以知道服务器在获取了客户端发送的请求后，依据请求调用相关的 JSP 处理页面。如果该页面是第一次执行，则需要把 JSP 页面中的代码转换为 Servlet 代码形式，转换完毕后，需要转换的 Java 文件即编译成了 class 文件，编译完成后，使用 JVM 执行编译过的文件，并将执行结果返回到客户端。如果该页面不是第一次执行，就会直接调用该页面的 class 文件执行。

可以看出，JSP 页面的第一次执行需要耗费一些时间，这些时间是耗费在 JSP 文件到 Servlet 文件转换并编译的过程中，而再次访问时会感觉快了很多。如果被请求的页面经过修改，服务器将会重新编译这个文件，然后执行。

由 JSP 的处理过程可以看到，JSP 和 Servlet 有很深的渊源关系，Servlet 是 JSP 技术的发展前身，它是 Java 技术对 CGI 编程的回应。Servlet 具有更高的效率，更容易使用，功能更强大，具有更好的可移植性、更节省投资等优点，因此通常用来做 MVC 模式中的控制器，但 Servlet 也有自身的不足之处，即所有响应代码都是通过 Servlet 程序生成的，如 HTML 标记。一个 Servlet 程序，其中大量的代码都用于生成这些 HTML 标记响应代码，只有少部分代码用于数据的处理和响应。并且，开发 Servlet 程序起点要求较高，Servlet 产生之后，没有像 PHP 和 ASP 那样，快速得到应用。因此，Sun 公司在结合了 Servlet 技术和 ASP 技术等特点后，又推出了 JSP 技术，JSP 技术完全继承了 Servlet 技术的优势，并具备了一些新的优势。

1.3　JSP 程序体系结构

1.3.1　比较 C/S 结构与 B/S 结构

软件开发中，目前最常用的体系结构有两种，即 C/S 结构和 B/S 结构。下面介绍这两种结构，并进行比较。

1．C/S 结构

C/S 结构即 Client/Server（客户/服务器）结构，它通过将任务合理分配到 Client 端和 Server 端，降低了系统的通信开销，可以充分利用两端硬件环境的优势。C/S 结构的出现是为了解决费用和性能的矛盾，最简单的 C/S 体系结构的数据库应用由两部分组成，即客户应用程序和数据库服务器程序。二者可分别称为前台程序与后台程序。运行数据库服务器程序的机器称为应用服务器，一旦服务器程序被启动，就随时等待响应客户程序发来的请求；客户程序运行在用户自己的计算机上，对应于服务器计算机，可称为客户计算机。当需要对数据库中的数据进行操作时，客户程序就自动地寻找服务器程序，并向其发出请求，服务器程序根据预定的规则做出应答，返回结果。

C/S 结构的优点是能充分发挥客户端 PC 的处理能力，很多工作可以在客户端处理后再提交给服务器，对应的优点就是客户端响应速度快。但它也有自身的局限性，其缺点如下：

（1）只适用于局域网。而随着互联网的飞速发展，移动办公和分布式办公越来越普及，这需要我们的系统具有扩展性。这种方式远程访问需要专门的技术，同时要对系统进行专门的设计来处理分布式数据。

（2）客户端需要安装专用的客户端软件。首先涉及安装的工作量，其次任何一台计算机出问题，如病毒、硬件损坏，都需要进行安装或维护。特别是有很多分部或专卖店的情况，不是工作量的问题，而是路程的问题。另外，在系统软件升级时，每一台客户机需要重新安装，其维护和升级成本非常高。

（3）对客户端的操作系统一般也会有限制。可能适应于 Windows 98，但不能用于 Windows 2000 或 Windows XP。或者不适用于微软新的操作系统等，更不用说 Linux、UNIX 等。

虽然 C/S 结构有一定的缺点，但目前仍有大量的软件开发采用的是该结构，例如 QQ、MSN、PP Live、迅雷等、eMule 等。

2．B/S 结构

B/S 结构即 Browser/Server（浏览器/服务器）结构，是随着 Internet 技术的兴起，对 C/S 体系结构的一种变化或者改进的结构。在 B/S 体系结构下，用户界面完全通过 WWW 浏览器实现，一部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。B/S 结构利用不断成熟和普及的浏览器技术实现原来需要复杂专用软件才能实现的强大功能，并节约了开发成本，是一种全新的软件系统构造技术。

基于 B/S 体系结构的软件，系统安装、修改和维护都在服务器端解决。Web 应用程序的访问不需要安装客户端程序，可以通过任一款浏览器（例如 IE 或者 Firefox）来访问各类 Web 应用程序。当 Web 应用程序进行升级时，并不需要在客户端做任何更改。和 C/S 结构的应用程序相比，Web 应用程序可以在网络上更加广泛地进行传播和使用。一般的网站都是 B/S 结构的，例如 Google、Baidu。

虽然 B/S 有诸多优点，但也是存在一些缺点的，例如 B/S 结构程序在跨浏览器的使用上总是不能尽如人意。另外，因为 B/S 结构的程序的大量工作都是由服务器完成的，所以如何设计算法使得访问效率得到保证也是一个很大的问题。

本书所介绍的 JSP 程序的体系结构将要采用的结构就是 B/S 结构，B/S 结构的程序是非常注重程序架构的，常用的程序架构有三层架构和两层架构，具体内容在下面介绍。

1.3.2 三层架构

传统的 JSP 开发中通常将业务处理的代码与 JSP 代码混在一起，这样程序的可读性很差，不易于阅读，更不易于代码维护。如何解决这个弊端？通常采用分层模式的设计理念来解决这个问题。分层模式是最常见的一种架构模式。分层模式是很多架构模式的基础，通过分层模式将解决方案的组件分隔到不同的层中，实现在同一个层中组件之间保持内聚性，并保持层与层之间松耦合，如何进行分层呢？

一般的做法是在客户端与数据库之间加入一个“中间层”，从而形成三层架构。这里所说的三层架构，不是指物理上的三层，而是逻辑上的三层，即把这三个层放置到一台机器上。三层架构的三层指的是表示层、业务逻辑层、数据层，各层的作用如下。

（1）表示层：主要作用为数据显示或者和后台进行交互，因此表示层通常对应于 HTML 页面或 JSP 页面。

（2）业务逻辑层：主要是针对具体问题的操作，也可以理解成对数据层的操作，对数据进行业务逻辑处理。

（3）数据层：主要是对非原始数据（数据库或者文本文件等存放数据的形式）的操作层，而不是指原始数据。也就是说，是对数据库的操作，而不是数据，具体为业务逻辑层或表示层提供数据服务。

注意：这里所说的数据层并不是数据库，而是与数据库密切相关的操作代码。

表示层、业务逻辑层、数据访问层之间的访问关系如图 1-2 所示。

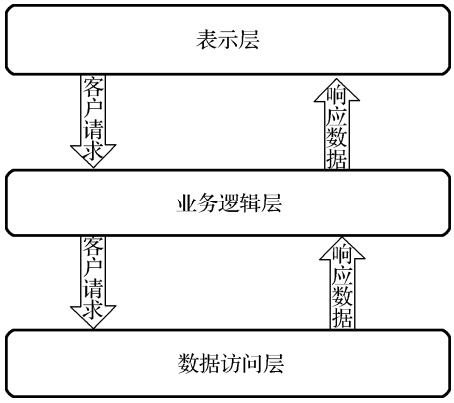

图 1-2 三层架构

表示层访问业务层，业务层为表示层的访问提供数据或相应的方法；业务层访问数据层，数据层为业务层的访问提供数据或方法。也可以说，表示层依赖业务层，业务层依赖数据层，层和层之间是单向的依赖关系，下层不知道上层的存在，仅完成自身的功能，而不关心结果如何使用，每一层仅知道其下层的存在，忽略其他层的存在，只关心结果的取得，而不关心结果的实现过程。

这种单向的依赖关系使得在同一个层中组件之间保持内聚性，并保持层与层之间的松耦合，从而实现软件工程要求的高内聚和低耦合的设计目标，提高程序的可复用性。

1.3.3　两层架构

三层架构虽然优秀，但理念相对复杂，不利于初学者掌握，因此在本书的 JSP 程序设计中将采用两层架构。所谓两层架构就是由同一程序来实现逻辑计算和数据处理，即把逻辑层与数据处理层合并为一层。这时，前台就是表示层页面，后台就是业务层和数据层的 java 代码。

两层架构如图 1-3 所示。

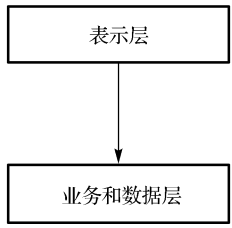

图 1-3　两层架构

1.4　搭建 JSP 的运行环境

1.4.1　JDK 的安装与配置

JSP 程序中的脚本语言是 Java 语言，Java 程序是运行在 Java 的虚拟机即 JVM 上的，所以 JSP 程序的开发首先需要搭建 Java 开发环境。搭建 Java 开发环境主要包括如下步骤：下载并安装 Java Develop Kit（JDK）；配置环境变量。下面分别进行详细介绍。

安装 JDK 及配置环境变量

在 Windows 下安装 JDK 与安装其他程序的步骤基本相同，请选择默认项直到单击“下一步”按钮，还需要进行系统环境变量的配置。

所谓环境变量，就是在操作系统中一个具有特定名字的对象，它包含了一个或多个应用程序将使用到的信息。如果安装完 JDK 之后，不配置 Java 的环境变量，那么在 DOS 命令行环境下就找不到 Java 的编译程序和 Java 的运行程序，也就不能在 DOS 环境下进行 Java 编译与运行程序了。与 JDK 或 JRE 的使用有关的是 path、classpath 两个环境变量。path 变量中存储的是 JDK 命令文件的路径，path 变量用来告诉操作系统到哪里去查找某个命令，只有设置好 path 变量，才能正常地编译和运行 Java 程序。classpath 则表示的是“类”路径，classpath 变量中存储的是 JDK 的类文件的路径，classpath 变量用来告诉 Java 执行环境，在哪些目录下可以找到执行 Java 程序所需要的类或包，在这些包中包含了常用的 Java 方法和常量。path 变量的值是 JDK 命令文件的路径，它的值应该设置成：“C:\Program Files (x86)\Java\jdk1.8.0_11\bin;”。classpath 变量的值是 JDK 类文件的路径，它的值应该设置成：“.;C:\Program Files (x86)\Java\jdk1.8.0_11\lib;”。注意，C:\Program Files (x86)是根路径，用户可以根据自己 JDK 的安装位置，调整 C:\Program Files (x86)的值。下面分别对这两个环境变量进行设置。

在环境变量设置窗口（如图 1-4 所示），单击“新建”按钮，添加一个名字是 path 的环境变量，变量的值是：“C:\Program Files (x86)\Java\jdk1.8.0_11\bin;”，如图 1-5 所示。

输入完成后，单击“确定”按钮，即可进行保存。path 变量就出现在系统变量列表中了，如图 1-6 所示。

图 1-4　环境变量设置窗口

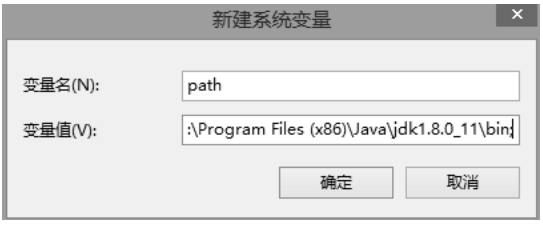

图 1-5　新建系统变量

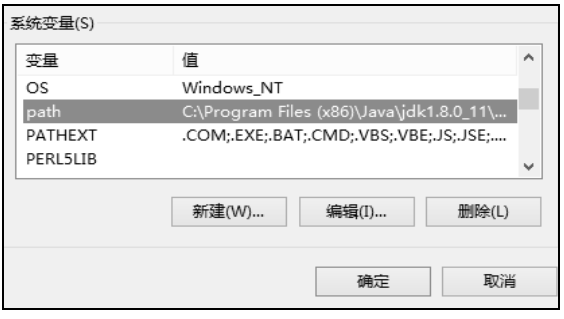

图 1-6　系统变量列表

注意：因为安装某些其他软件也需要配置 path 变量，可以选中 path 变量，单击“编辑”按钮对该变量进行编辑。如果因为其他软件 path 变量问题使得 JDK 运行异常，可以将 Java 的 path 变量的值放在其他软件 path 变量的值的前面，最后以分号结束，这样就能解决这个问题。

然后配置 classpath 环境变量，单击“新建”按钮，添加一个名字是 classpath 的环境变量，变量的值是：“.;C:\Program Files (x86)\Java\jdk1.8.0_11\lib;”，如图 1-7 所示。

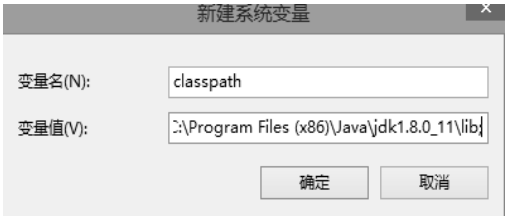

图 1-7　新建系统变量

输入完成后，单击“确定”按钮，即可进行保存。至此，环境变量配置完成，可以编写 Java 程序来测试环境变量的配置是否正确。

1.4.2　Tomcat 的安装、运行与目录结构

JSP 程序的运行必须依赖于服务器。目前能够运行 JSP 的服务器软件有很多，例如 Tomcat、JBoss、Resin、WebLogic 等。每个服务器都有自己的特点，其应用方面也不相同。本书将选择 Tomcat 服务器，Tomcat 是 Apache 公司的产品，目前版本已经升级到 9.x。Tomcat 服务器在中、小型的 JSP 网站上应用比较广泛，并且是完全开源免费的。

1．下载 Tomcat

获取 Tomcat 非常容易，可以直接在网上搜索或者从 Tomcat 官方网站获取。访问“http://tomcat.apache.org/”，下载 Tomcat 软件“apache-tomcat-7.0.27.exe”，下载完毕后，就可以使用 Tomcat 服务器了。

2．安装 Tomcat

单击下载的可执行程序，会弹出一个如图 1-8 所示的窗口，在该窗口中单击 Next 按钮，会弹出如图 1-9 所示的窗口。

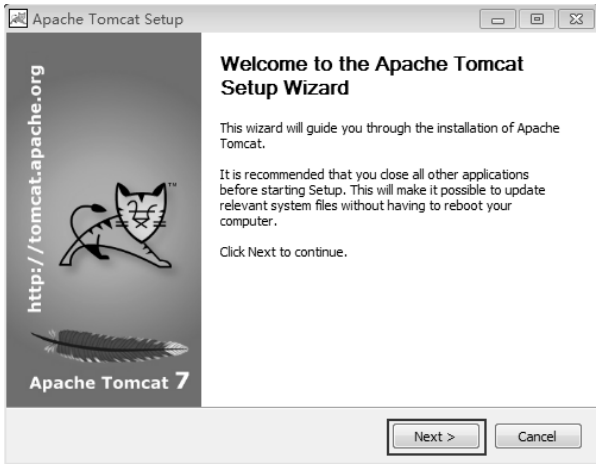

图 1-8　Tomcat 安装启动窗口

在如图 1-9 所示的窗口中单击 I Agree 按钮，进入安装选项窗口，如图 1-10 所示。在该窗口中需要对相关的插件进行选择，在这里把所有的插件全部选中，即选择 Full 选项，选择好后单击 Next 按钮，会显示如图 1-11 所示的窗口。

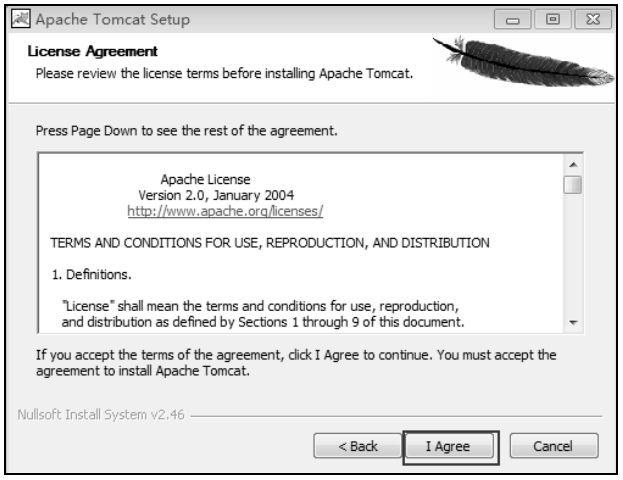

图 1-9　Tomcat 安装显示窗口

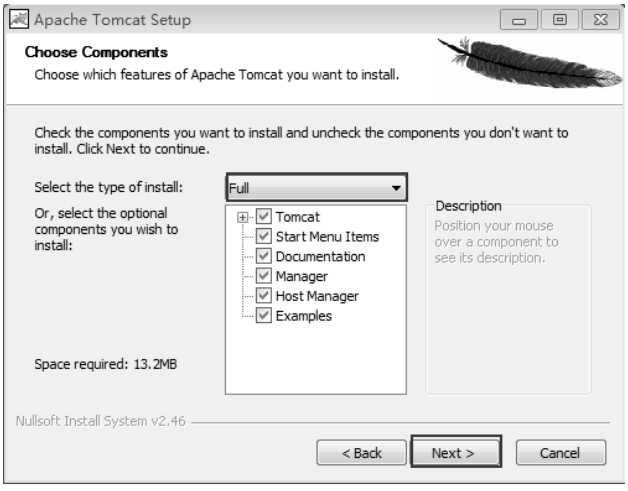

图 1-10　安装选项窗口

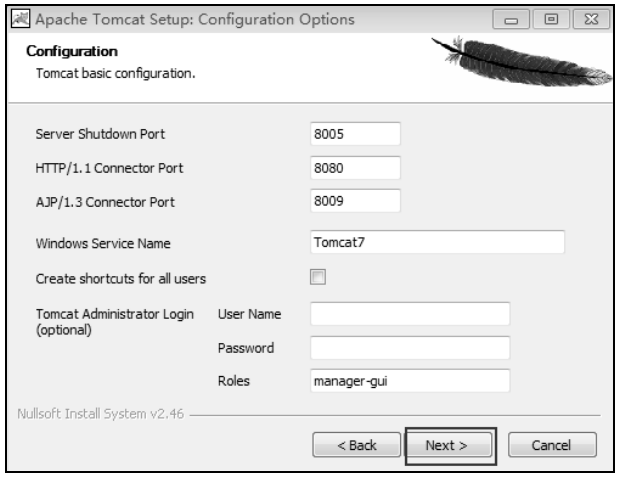

图 1-11　进行端口的配置

在如图 1-11 所示的窗口中，主要进行端口的配置，即配置所编写的 JSP 程序在哪个端口运行，这里 Tomcat 默认的是操作系统的 8080 端口。单击 Next 按钮，会进入下一个窗口（如图 1-12 所示）。

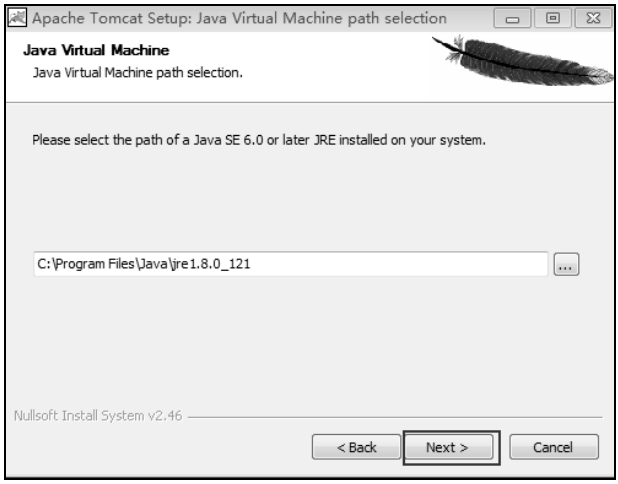

图 1-12　选择 Java 虚拟机窗口

在如图 1-12 所示的窗口中，要选择 Tomcat 服务器在运行时使用哪个开发工具包编译和解释执行 JSP 文件。JSP 文件实质上是一个 Java 文件，是由 Java 中的 Servlet 包产生的。在这里要选择的是 jre1.8.0 文件夹。

单击 Next 按钮，进入如图 1-13 所示的界面。

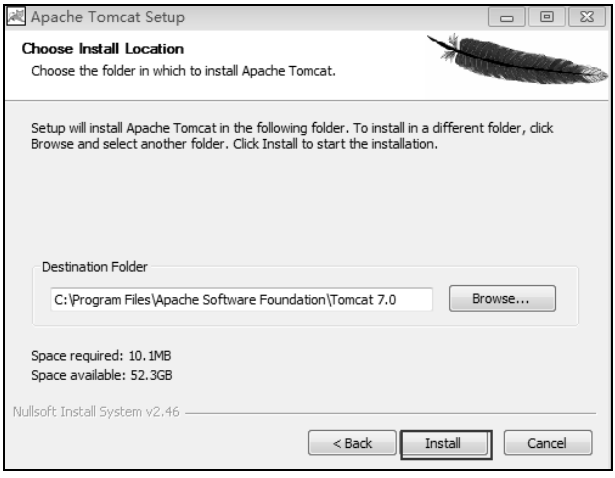

图 1-13　选择安装位置

在该界面中选择安装路径，选择好后，单击 Install 按钮，程序会自动完成安装。安装完成后，会弹出一个如图 1-14 所示的界面。

在如图 1-14 所示的界面中选择要运行的软件，如可以直接运行该 Tomcat 服务器，或打开 Tomcat 的使用说明书。在这里选择两个都运行，Tomcat 服务器运行后，会在右下角的状态栏中出现一个图标，绿色表示正常启动，可以使用，红色表示不可以使用。到此为止，Tomcat

已经安装完成了，检验是否安装成功，打开 IE 浏览器，在地址栏中输入“http://localhost:8080/”，单击“转到”按钮，会弹出一个如图 1-15 所示的窗口，这时就表明服务器已经正确安装了。

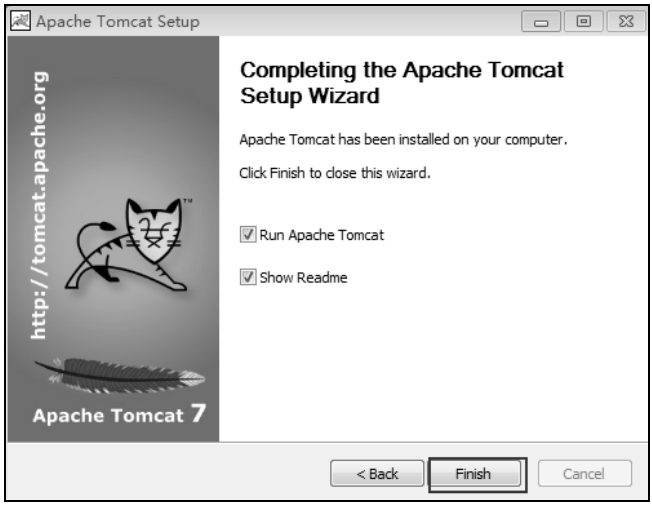

图 1-14　成功界面

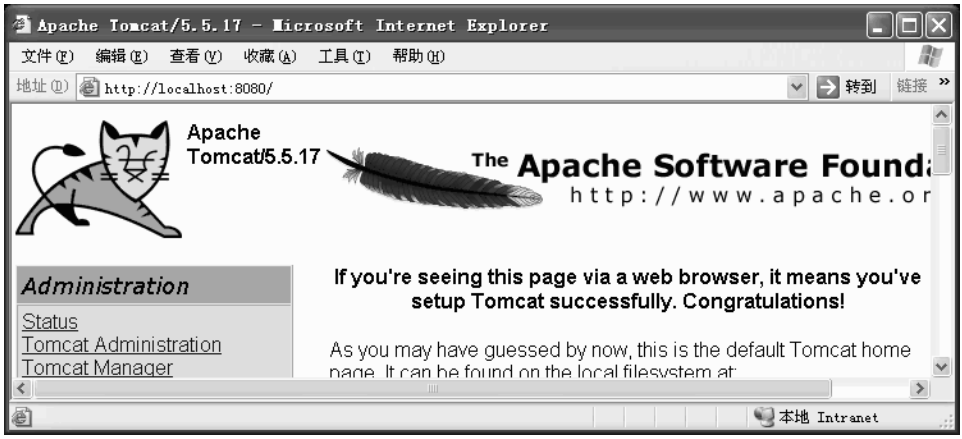

图 1-15　Tomcat 服务器主页运行窗口

Tomcat 安装完成后，其目录结构的详细介绍如下。

（1）bin 存放在 Windows 平台以及 Linux 平台上启动和关闭 Tomat 的脚本文件。

（2）conf 存放 Tomat 服务器的各种配置文件，其中最重要的文件是 Server.xml。

（3）server 包含 3 个子目录：classes、lib 和 webapps。

（4）server/lib 存放 Tomat 服务器所需的 Jar 文件。

（5）server/webapps 存放 Tomat 自带的两个 Web 应用：admin 应用和 manager 应用。

（6）common/lib 存放 Tomat 服务器以及所有 Web 应用都可以访问的 Jar 应用。

（7）share/lib 存放所有 Web 应用都可以访问的 Jar 文件。

（8）logs 存放 Tomat 的日志文件。

（9）webapps 在发布 Web 应用时，默认情况下把 Web 应用文件放于此目录下。

（10）work Tomcat 把由 JSP 生成的 Servlet 放于此目录下。

1.4.3　开发工具的选择

目前比较流行的开发 JSP 程序的工具为 MyEclipse 和 JavaEE 版的 Eclipse，下面分别进行介绍。

1．MyEclipse

MyEclipse 企业级工作平台（MyEclipse Enterprise Workbench，简称 MyEclipse）是对 Eclipse IDE 的扩展，利用它可以在数据库和 J2EE 的开发、发布，以及应用程序服务器的整合方面极大地提高工作效率。它是功能丰富的 J2EE 集成开发环境，包括了完备的编码、调试、测试和发布功能，完整支持 HTML, Struts, JSF, CSS, Javascript, SQL, Hibernate。

MyEclipse 有两种发行方式。第一种是以集成的方式发行，该发行版本集成了 Eclipse 和 JRE。在安装的过程中不需要网络连接。在下载完集成版的 MyEclipse 后，直接运行安装程序，并按着提示一步步地安装即可。

第二种是 pulse 发行方式。该发行版本的安装程序非常小（6MB 左右），也是一个可执行的安装程序。在运行该安装程序后，会自动从 MyEclipse 的官方网站下载当前版本的 MyEclipse。也就是说，该发行版本的安装文件虽小，但在安装时需要稳定的网络连接。

MyEclipse 虽然功能强大，但是它是收费的，因此在使用前需要输入正确的注册码，其界面如图 1-16 所示。

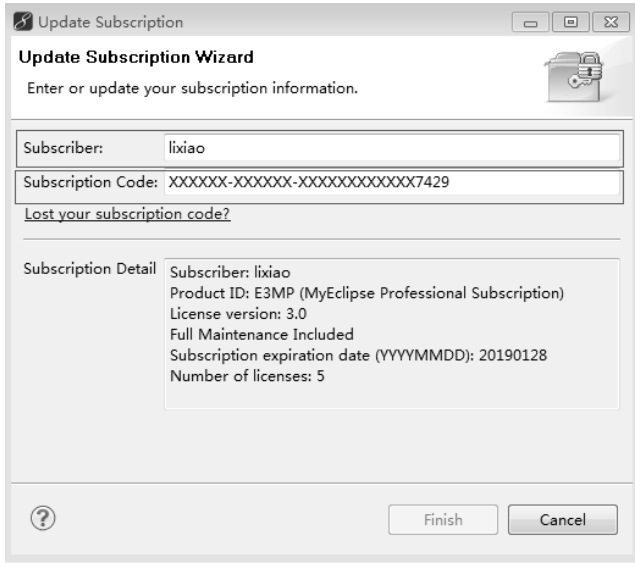

图 1-16　注册码输入界面

在本书的学习中，我们将使用 JavaEE 版的 Eclipse，下面对 Eclipse 进行介绍。

2．Eclipse

Eclipse 是一种可扩展的开放源代码 IDE。2001 年 11 月，IBM 公司捐出价值 4000 万美元的源代码组建了 Eclipse 联盟，并由该联盟负责这种工具的后续开发。集成开发环境（IDE）经常将其应用范围限定在“开发、构建和调试”的周期之中。为了帮助集成开发环境（IDE）克服目前的局限性，业界厂商合作创建了 Eclipse 平台。Eclipse 允许在同一 IDE 中集成来自不

同供应商的工具，并实现了工具之间的互操作性，从而显著改变了项目工作流程，使开发者可以专注在实际的嵌入式目标上。

利用 Eclipse 可以将高级设计（也许是采用 UML）与低级开发工具（如应用调试器等）结合在一起。如果这些互相补充的独立工具采用 Eclipse 扩展点彼此连接，那么当用调试器逐一检查应用时，UML 对话框可以突出显示我们正在关注的器件。事实上，由于 Eclipse 并不了解开发语言，所以无论是 Java 语言调试器、C/C++调试器，还是汇编调试器都是有效的，并可以在相同的框架内同时瞄准不同的进程或节点。

Eclipse 自创建至今，已经有很多版本，如表 1-1 所示。

表 1-1　Eclipse 版本与发布日期

代号	版本	发布日期
IO	Eclipse 3.1	2005 年 6 月 27 日
Callisto	Eclipse 3.2	2006 年 6 月 26 日
Europa	Eclipse 3.3	2007 年 6 月 27 日
Ganymede	Eclipse 3.4	2008 年 6 月 25 日
Galileo	Eclipse 3.5	2009 年 6 月 24 日
Helios	Eclipse 3.6	2010 年 6 月 23 日
Indigo	Eclipse 3.7	2011 年 6 月 22 日
Juno	Eclipse 3.8/4.2	2012 年 6 月 27 日
Kepler	Eclipse 4.3	2013 年 6 月 26 日
Luna	Eclipse 4.4	2014 年 6 月 25 日
Mars	Eclipse 4.5	2015 年 6 月 25 日
Neon	Eclipse 4.6	2016 年 6 月 25 日

从 Eclipse 3.7 Indigo 版的 Eclipse 起都是集成了 JavaEE 开发插件，可以进行 Java Web 程序的开发。本书的程序将采用 Indigo 版的 Eclipse 进行开发，在下面的内容中将利用 Eclipse 开发第一个 JSP 程序。

1.5　第一个 JSP 应用

1.5.1　创建 JSP 页面

选择 File→New→Dynamic Web Project，如图 1-17 所示。

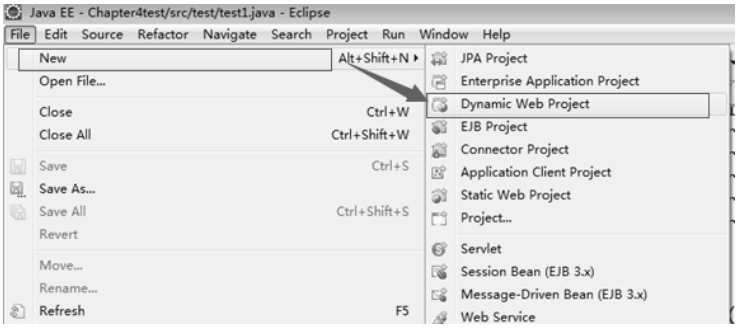

图 1-17　选择动态 Web 工程

单击 Dynamic Web Project 后将会出现如图 1-18 所示的界面，在该界面中输入工程名称 Chapter1_1，单击 Finish 按钮即可。

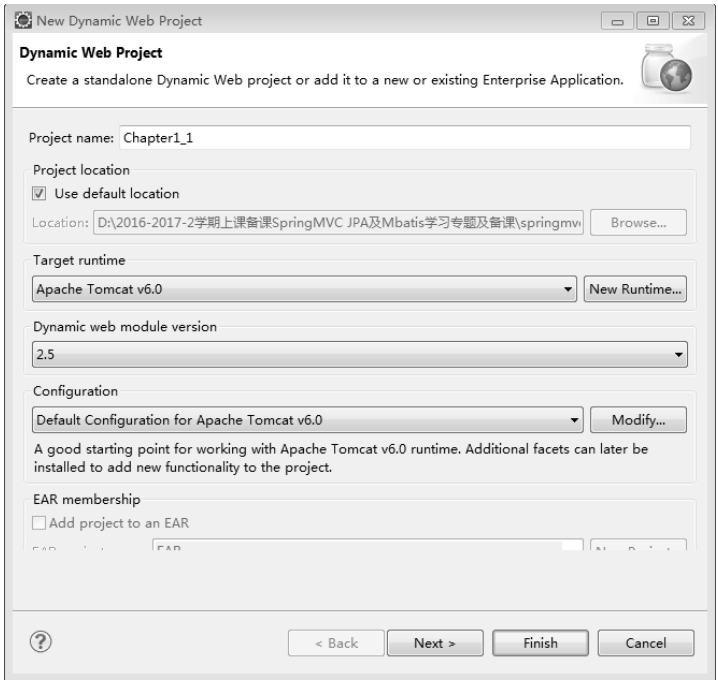

图 1-18　创建动态 Web 工程

建好的动态 Web 工程目录结构如图 1-19 所示。

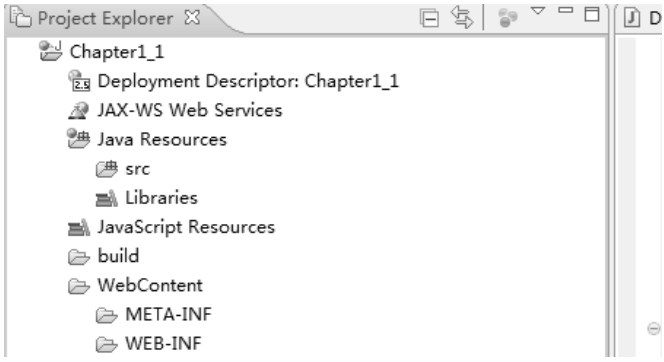

图 1-19　动态 Web 工程目录结构

在动态 Web 工程目录结构中，JavaBean 需要建在 Java Resources 的 src 目录下，Web 元素需要建立在 WebContent 目录下，WebContent 下是要发布到服务器上的内容。其中，META-INF 文件夹下存放的是工程自身相关的一些信息，通常由开发工具自动生成。而文件 web.xml：完成 Servlet 在 Web 容器的注册。web.xml 是 Web 应用程序的部署描述文件，是用来给 Web 服务器解析并获取 Web 应用程序相关描述的。不按照 Sun 公司的规范做应用 Web 程序的结构，则 Web 容器找不到相关资源，比如，xml 文件写错了，启动 Tomcat 的时候会报错。凡是客户端能访问的资源（*.html,*.jpg），则必须与 WEB-INF 在同一目录下。也就是说，

放在 WebContent 根目录下的资源，从客户端是可以通过 URL 地址直接访问的，而不能放在 WEB-INF 下面，因为凡是 WEB-INF 里的文件都不能被客户端直接访问（比如隐藏的信息）。WEB-INF 目录下的资源对用户来说是不可见的，而对 Web 服务器来说则没有这样的限制。

在 WebContent 目录上右击，选择 New→JSP File，创建 JSP 的第一个 jsp 文件，如图 1-20 所示。

图 1-20　创建 helloworld.jsp 文件

单击 Finish 按钮，生成 JSP 文件的程序如下：

```
<%@ page language="java"  pageEncoding="ISO-8859-1"%>
<html>
<head>
<title>Insert title here</title>
</head>
<body>
helloworld!
</body>
</html>
```

1.5.2　运行 JSP 程序

JSP 程序的运行需要依赖 Tomcat 服务器，因此运行 JSP 程序需要首先配置 Tomcat 服务器，配置方法如下。

在 Servers 窗口中，单击 New Server Wizard，出现如图 1-21 所示的界面。

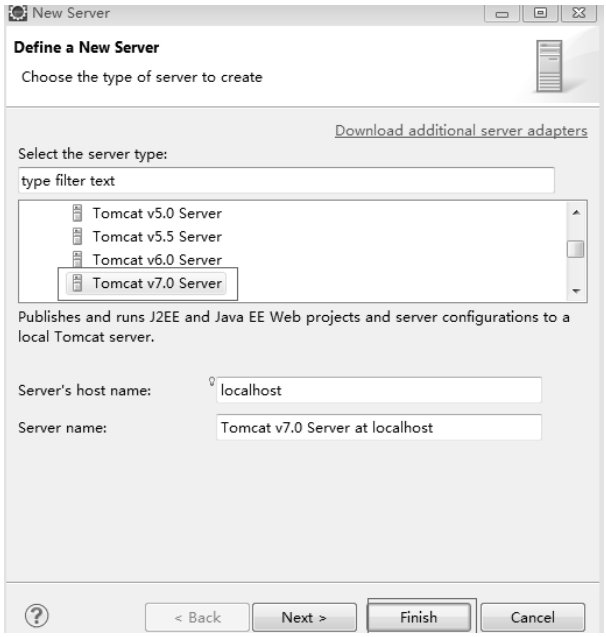

图 1-21　配置 Tomcat 服务器步骤 1

单击 Finish 按钮后出现如图 1-22 所示的界面，在该界面中单击 Browse 按钮后选择 Tomcat 的安装路径。

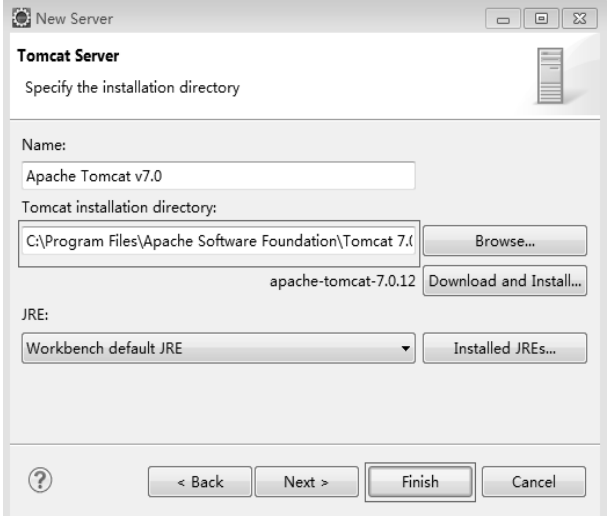

图 1-22　配置 Tomcat 服务器步骤 2

单击 Finish 按钮即可完成 Tomcat 服务器的配置。

配置完成后，在 Servers 窗口中显示如图 1-23 所示的内容。

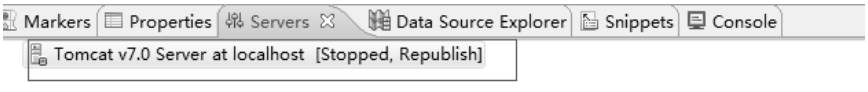

图 1-23　在 Servers 窗口中显示的内容

在 Servers 窗口中右击 Tomcat v7.0 Server at localhost [Stopped, Republish]，选择 Add and Remove 进入工程部署窗口（如图 1-24 所示）。

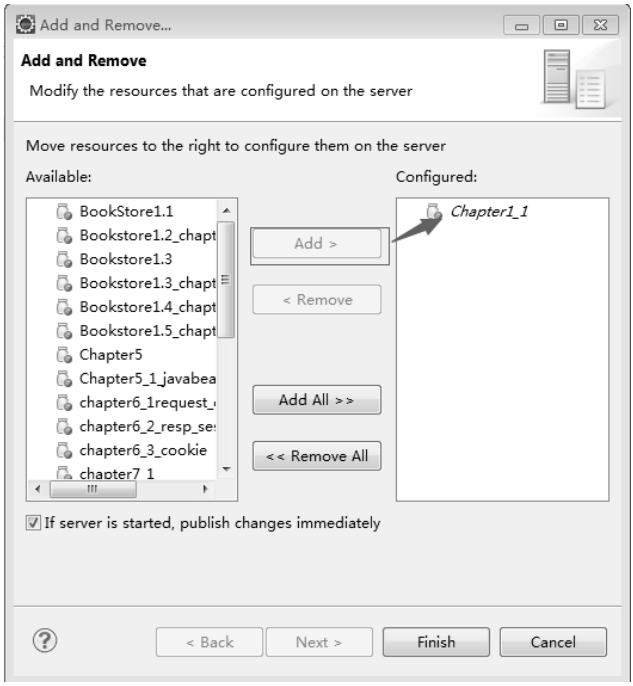

图 1-24　工程部署窗口

将左侧的待运行工程 Chapter1_1 通过单击 Add 按钮添加到右侧，单击 Finish 按钮即可完成部署。

在 Chapter1_1 工程下，选择待运行 JSP 文件 helloworld.jsp，右击该文件，选择 Run as→Run on Server，以 Run on Server 方式运行该文件。

若出现如图 1-25 所示的界面，则表示环境搭建成功。

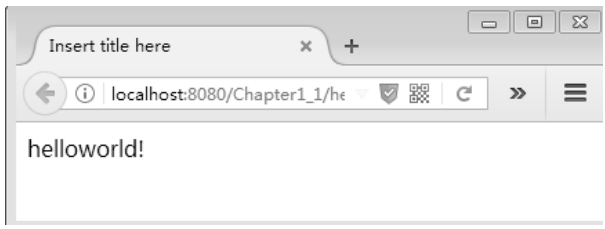

图 1-25　成功界面

习　题　1

1．如果想在 Tomcat 服务器启动时将 jar 包加载到内存，且该 jar 包可以被 Tomcat 服务器上所有的应用使用，应该将该 jar 包复制到 Tomcat 的（　　）目录下。

A．common　　B．server　　C．lib　　D．server\lib

2．Tomcat 服务器的默认端口为（　　）。

A．8888　　B．8001　　C．8080　　D．80

3．创建 JSP 应用程序时，配置文件 web.xml 应该在程序下的（　　）目录中。

A．admin　　B．servlet　　C．WebRoot　　D．WEB-INF

4．不是 JSP 运行必需的是（　　）。

A．操作系统　　B．Java JDK

C．支持 JSP 的 Web 服务器　　D．数据库

5．关于部署到 Tomcat 服务器的 Java Web 应用程序，正确的选项有（　　）。

A．Java Web 应用程序总是打包成 War 形式部署到 Tomcat 服务器中

B．Java Web 应用程序应该部署到 Tomcat 服务器的 server 子目录中

C．每个 Java Web 应用程序都有一个 web.xml 文件

D．Java Web 应用程序的根目录下不能存放任何文件，所有 html、gif 等文件必须存放到某一子目录中

6．简述 JSP 程序的运行过程。

7．简述 B/S 模式和 C/S 模式。

8．简述三层架构及其特点。

9．简述 Tomcat 的目录结构。

10．开发第一个 JSP 程序，测试 JSP 开发环境的搭建。