

第 1 章 Java EE 框架概述

1.1 Java Web 程序体系结构

1.1.1 比较 C/S 结构与 B/S 结构

在 Java Web 软件开发中，目前最常用体系结构有两种，即 C/S 结构和 B/S 结构。下面对这两种结构进行介绍和比较。

1. C/S 结构

C/S 结构即 Client/Server（客户/服务器）结构，它通过将任务合理地分配到客户端和服务端，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 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，一旦数据库服务器程序被启动，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，数据库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，这需要系统具有扩展性。这种方式的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，如病毒感染、硬件损坏，都需要进行安装或维护。特别是在有很多分部或专卖店的情况下，不是工作量的问题，而是路程的问题。另外，系统软件升级时，每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。可能适用于 Windows 98，但不能用于 Windows 2000 或 Windows XP。或者不适用于微软的新操作系统等，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该结构，如 QQ、MSN、PP Live、迅雷、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 结构的一种变化或改进的结构。在 B/S 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由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 结构的软件，系统安装、修改和维护全在服务器端解决。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（如 IE 或 Firefox）来访问各类 Web 应用程序。当对 Web 应用程序进行升级时，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传播和使用。一般的网站都采用 B/S 结构，如 Google、Baidu。

虽然 B/S 有诸多优点，但也存在一些缺点，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 结构的大量程序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 Java Web 程序的体系结构采用的就是 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，在下一节介绍三层架构。

1.1.2 三层架构

在传统的 Java Web 软件开发中，通常将业务处理的代码与 Java Web 代码混在一起，导致程序的可读性很差，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，实现在同一个层中的组件之间保持内聚性，并保持层与层之间的松耦合。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（如图 1-1 所示）的三层指的是表示层、业务层、数据持久层，各层的作用如下。

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

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

（3）数据持久层：主要指对非原始数据（数据库或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，而不是对数据的操作，为业务层或表示层提供数据服务。

注意：这里所说的数据持久层并不是数据库，而是与数据库密切相关的操作代码。

可从图 1-1 中看出表示层、业务层、数据持久层之间的访问关系。

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；业务层访问数据持久层，数据持久层为业务层的访问提供数据或方法。也可以这样说，表示层依赖业务层，业务层依赖数据持久层，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，而不关心结果如何被使用，每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的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。

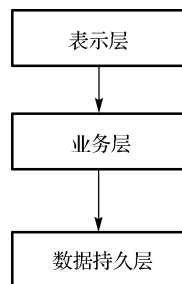


图 1-1 三层架构

1.2 Hibernate、Spring MVC、Spring 框架概述

在介绍 Hibernate、Spring MVC、Spring 这三个框架之前，提高必须先明确为什么要学习框架及什么是框架。使用框架会给程序员带来很多好处，包括提高代码重用性和一致性，获得对变化的适应性等，尤其是能够减轻程序员的开发强度，使程序员更加注重业务的开发，有利于提高软件开发速度及质量。如果一个项目的开发不使用框架，那么开发该项目所需工作量会随着项目复杂性的提高以几何级数递增，而对于使用框架的项目而言，开发所需工作量会随着项目复杂性的提高以代数级数递增。也就是说，在开发团队人数一样的情况下，如果一个没有使用框架的项目所需的周期为 6~9 个月，那么该项目使用框架则只需要 3~5 个月。

框架 (Framework) 的定义如下：框架是应用系统的骨架，将软件开发中反复出现的任务标准化，以可重用的形式提供使用；大多数框架提供了可执行的具体程序代码，支持迅速地开发出可执行的应用程序；但也有些框架仅提供了抽象的设计架构，可以帮助程序员开发出健壮的设计模型；一个设计成功的框架能够大大缩短应用系统开发的周期，在预制框架上加入定制的构件可以大量减少编码量，并容易进行测试。因此，框架实际上是某种软件系统的半成品，是一组协同工作组件的集合。框架具有以下特点：

- (1) 一般是成熟、健壮的。
- (2) 可以使程序员专注于软件系统的业务逻辑设计，提高软件开发效率，缩短开发周期。
- (3) 使程序在开发阶段有利于分工，并且更易于维护和扩展。
- (4) 不断升级的软件，使程序员可以直接享用软件升级带来的优势。
- (5) 具有良好的结构和可扩展性。

在明确框架的重要性及框架的定义后，下面将分别对 Hibernate、Spring MVC、Spring 这三个框架进行简单介绍。

1.2.1 Hibernate 简介

目前，大多数应用软件的开发都是基于数据库操作的，当应用程序直接访问数据库时，软件开发人员要编写大量烦琐的数据库操作代码，不但容易出错，而且还影响项目的开发效率。

幸运的是，Hibernate 框架对 JDBC 的代码进行了封装，能够使开发人员从繁重的操作数据库的编码工作中解放出来。在实现数据持久层开发时，通常要选择 Hibernate 框架进行开发，这样程序员可以把精力放在数据表示和业务逻辑处理的代码编写上，提高了项目的开发效率、可维护性和可移植性。

Hibernate 是一个开放源码的对象-关系映射 (ORM) 框架，它对 JDBC 进行了轻量级封装，开发人员可以使用面向对象的编程思想来进行数据持久层开发，操作数据库。还可以使用 Hibernate 提供的 HQL (Hibernate Query Language) 直接从数据库中获得 Java 对象。

2001 年年末，Hibernate 发布了第一个正式版本。该版本发布后受到开发人员的一致好评。之后，Hibernate 势不可挡，不断推出新的版本，成长速度惊人。直到 2003 年 6 月，Hibernate 2 发布了，由于该版本对各主流数据库提供的完美支持及完善的开发文档，使其一跃成为最流行的数据持久层开发工具。

2003 年 9 月，Hibernate 开发团队进入 JBoss 公司，开始全职开发 Hibernate，从这个时候

开始，Hibernate 得到了突飞猛进的普及和发展。

2004 年，整个 Java 社区开始从实体 Bean 向 Hibernate 转移，特别是在 Rod Johnson 的著作 *Expert One-on-One J2EE Development without EJB* 出版后，由于这本书以扎实的理论、充分的论据和翔实的论述否定了 EJB（Enterprise Java Bean，企业 Java Bean），提出了轻量级敏捷开发理念之后，以 Hibernate 和 Spring 为代表的轻量级开源框架开始成为 Java 世界的主流和事实标准。在 2004 年由 Sun 公司领导的 J2EE 5.0 标准制定中的持久化框架标准正式以 Hibernate 为蓝本。

2005 年 3 月，Hibernate 3 的发布成为 Hibernate 发展史上的一个里程碑，使 Hibernate 进入一个新的发展阶段。Hibernate 无疑已经占据了数据持久层设计领域的主导地位。

2006 年，J2EE 5.0 标准正式发布以后，持久化框架标准 Java Persistent API（简称 JPA）基本上是参考 Hibernate 实现的，而 Hibernate 从 3.2 版本开始已经完全兼容 JPA 标准。

2012 年 11 月，Hibernate 4.1.8 发布。

2018 年 11 月，Hibernate 5.4 发布。

1.2.2 Spring MVC 简介

在实现表示层时，通常要选择 MVC 框架，目前常用的 MVC 框架有 Spring MVC 和 Struts2。Spring MVC 属于 SpringFrameWork 的后续产品，已经融合在 Spring Web Flow 里。自 Spring 2.5 发布后，由于支持注解配置，Spring MVC 框架的易用性有了大幅度的提高。虽然 Struts2 也是非常优秀的 MVC 构架，但由于 Struts2 采用了值栈、OGNL 表达式、Struts2 标签库等，因而导致其应用性能下降，而 Spring MVC 以其使用灵活、简单，学习成本低，代码书写方便及良好的扩展性等优势深受软件开发者的青睐。

1.2.3 Spring 简介

业务层是最容易被忽视的一层，一些初级程序员经常纠结于“为什么要单独开辟业务层”，这些开发者更习惯于将业务层的内容书写在表示层或数据持久层中。这种编程习惯是不好的，因为它会造成程序代码的强耦合，这样的代码难以维护。而 Spring 框架的引入能很好地解决业务层的这些问题，因此在实现业务层开发时，通常要使用 Spring 框架。

Spring 是 Java 平台上的一个开源应用框架，它有着深厚的历史根基。Spring 起源于由 Rod Johnson 在 2002 年所著的 *Expert One-on-One: J2EE Design and Development* 一书中的基础性代码。在该书中，Rod Johnson 阐述了大量 Spring 框架的设计思想，并对 J2EE 平台进行了深层次的思考，指出了 EJB 存在的结构臃肿的问题。他认为：采用一种轻量级、基于 JavaBean 的框架就可以满足大多数程序开发的需要。

2003 年，Rod Johnson 公开了所描述框架的源代码，这个框架逐渐演变成我们所熟知的 Spring 框架。在 2004 年 3 月发布的 1.0 版本是 Spring 的第一个具有里程碑意义的版本。这个版本发布之后，Spring 框架在 Java 社区中变得异常流行。现在，Spring 已经获得广泛的欢迎，并被许多公司认为是具有战略意义的重要框架。Spring 框架是基于 Java 平台的，它为应用程序的开发提供了全面的基础设施支持。Spring 专注于基础设施，这使开发者能更好地致力于应用开发而不必关心底层的架构。

Spring 框架本身并未强制使用任何特别的编程模式。从设计上看，Spring 框架给予了 Java 程序员许多自由度，但同时业界存在的一些常见问题也提供了规范的文档和易于使用的方

法。Spring 框架的核心功能适用于任何 Java 应用。在基于 Java 企业平台上的大量 Web 应用中，积极的拓展和改进已经形成。而 Spring 的用途也不仅限于服务器端的开发，从简单性、可测试性和松耦合的角度来说，任何 Java 应用都可以从 Spring 中获得好处。

1.3 Java Web 开发环境搭建

1.3.1 开发工具选择

1. 代码编写工具：Eclipse

Eclipse 是一种可扩展的开放源代码 IDE（Integrated Development Environment，集成开发环境）。2001 年 11 月，IBM 公司捐出价值为 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。IDE 经常将其应用范围限定在“开发、构建和调试”的周期中。为了帮助 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指向不同的进程或节点。

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

Eclipse 自 4.6 Neon 版起都集成了 Java EE 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。

2. 服务器软件：Tomcat 7.0.27

Tomcat 是一个免费的开源的 Servlet 容器，它是 Apache 基金会的 Jakarta 项目中的一个核

心项目，由 Apache、Sun、其他公司及个人共同开发而成。由于 Sun 公司的参与和支持，最新的 Servlet 和 Java Web 规范总能在 Tomcat 中得到体现。

目前，Tomcat 的最新版为 Tomcat 9，本书使用的是 Tomcat 7.0.27。Tomcat 提供各种平台的版本下载，可以从 <http://jakarta.apache.org> 上下载其源代码版或二进制版。由于 Java 的跨平台特性，基于 Java 的 Tomcat 也具有跨平台性。

3. 数据库：Oracle

数据库是信息管理系统应用程序开发的核心，应用程序的开发往往都是围绕数据库展开的。根据数据的存储规模数据库可以分为大型数据库、中型数据库和小型数据库。大型数据库有 SyBase、DB2、Oracle 等，中型数据库有 SQL Server、MySQL 等、小型数据库有 Access 等。在上面所列的这些数据库中，目前在 Oracle 公司旗下的有 Oracle 和 MySQL，本书的案例将采用 Oracle 数据库进行开发。

Oracle 数据库之所以备受用户喜爱是因为它具有以下突出的特点。

(1) 支持大数据库、多用户的高性能的事务处理。Oracle 支持大型数据库；支持大量用户，同时在同一数据上执行各种数据应用，并使数据争用性最小，保证数据一致性。系统维护具有高的性能，Oracle 每天可连续 24 小时工作，正常的系统操作（后备或个别计算机系统故障）不会中断数据库的使用；可控制数据库数据的可用性，可在数据库级或在子数据库级上控制。

(2) Oracle 遵守数据存取语言、操作系统、用户接口和网络通信协议的工业标准。因此，它是一个开放系统，保护了用户的投资。美国标准化和技术研究所（NIST）对 Oracle 7 Server 进行检验，发现它 100%地与 ANSI/ISO SQL89 标准的二级相兼容。

(3) 实施安全性控制和完整性控制。Oracle 为限制各监控数据存取提供系统、可靠的安全性。Oracle 实施数据完整性，为可接受的数据指定标准。

(4) 支持分布式数据库和分布处理。Oracle 为了充分利用计算机系统和网络，允许将处理分为数据库服务器和客户应用程序，所有共享的数据管理由数据库管理系统的计算机处理，而运行数据库应用的工作站集中于解释和显示数据。通过网络连接的计算机环境，Oracle 将存放在多台计算机上的数据组合成一个逻辑数据库，供全部网络用户存取。分布式系统像集中式数据库一样具有透明性和数据一致性。

(5) 具有可移植性、可兼容性和可连接性。由于 Oracle 软件可在许多不同的操作系统上运行，所以在 Oracle 上开发的应用可移植到任何操作系统，只需做微小修改或不用修改。Oracle 软件同工业标准相兼容，包括许多工业标准的操作系统，开发的应用系统可在任何操作系统上运行。可连接性是指 Oracle 允许不同类型的计算机和操作系统通过网络共享信息。

1.3.2 开发环境搭建

1. 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，是指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的信息。如果在安装 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。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 变量表示的是“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 1-2 所示）中，单击“新建”按钮，添加一个名为 path 的系统变量，变量的值是：“C:\Program Files (x86)\Java\jdk1.8.0_11\bin;”，如图 1-3 所示。



图 1-2 环境变量设置窗口

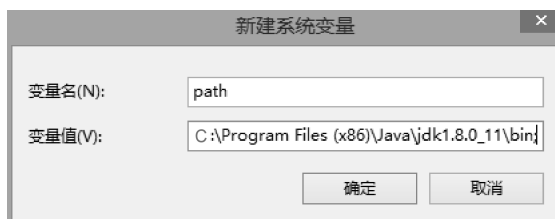


图 1-3 新建 path 变量

输入完成后，单击“确定”按钮，即可保存。path 变量就出现在系统变量列表中了，如图 1-4 所示。

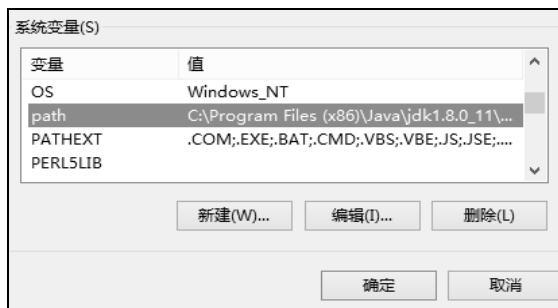


图 1-4 系统变量列表

注意：因为安装某些其他软件也需要配置 path 变量，所以可以选中 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，则可以将 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 变量，单击“新建”按钮，添加一个名为 classpath 的变量，该变量的值是：“.;C:\Program Files (x86)\Java\jdk1.8.0_11\lib;”，如图 1-5 所示。

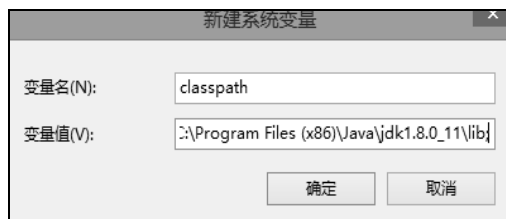


图 1-5 新建 classpath 变量

输入完成后，单击“确定”按钮，即可保存。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。

2. Tomcat 的安装、运行与目录结构

JSP 程序的运行必须依赖服务器。本书选择 Tomcat 作为程序运行的服务器，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 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，弹出如图 1-6 所示的窗口，在该窗口中单击 Next 按钮，弹出如图 1-7 所示的窗口。

在如图 1-7 所示的窗口中单击 I Agree 按钮，进入安装选项窗口，如图 1-8 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 1-9 所示的窗口。



图 1-6 Tomcat 安装启动窗口

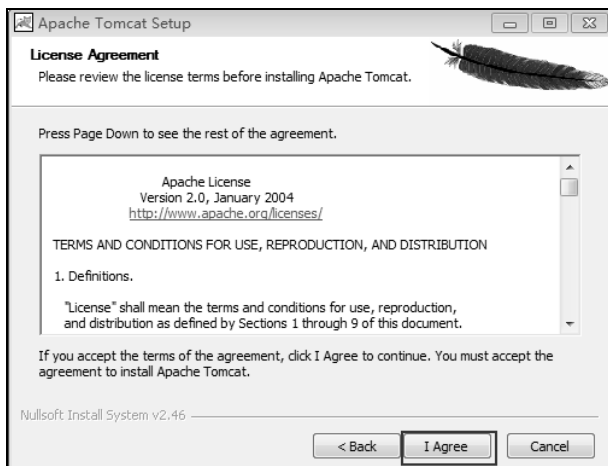


图 1-7 Tomcat 安装显示窗口

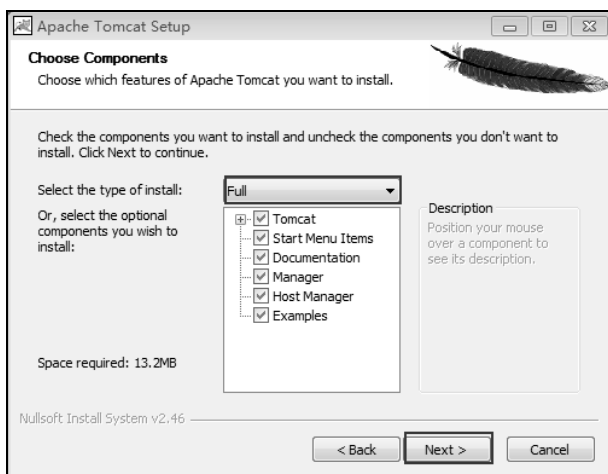


图 1-8 安装选项窗口

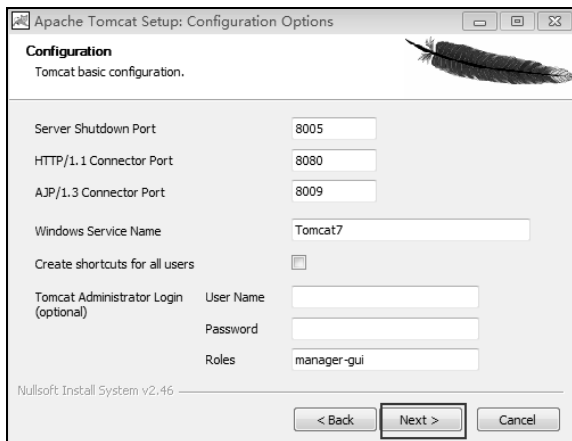


图 1-9 端口配置窗口

在如图 1-9 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，会进入如图 1-10 所示的窗口。

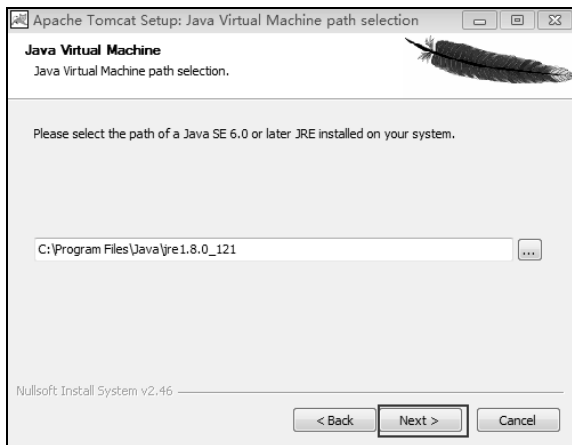



图 1-10 选择 Java 虚拟机窗口

在如图 1-10 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。在这里选择的是 jre1.8.0 文件夹。

单击 Next 按钮，进入如图 1-11 所示的界面。

在该界面中选择安装路径后，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 1-12 所示的界面。

在如图 1-12 所示的界面中选择要运行的软件，例如可以直接运行该 Tomcat 服务器，或打开 Tomcat 的使用说明书。在这里将“Run Apache Tomcat”和“Show Readme”两个选项都选中，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 安装完成，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 1-13 所示的窗口，这时就表明服务器已经正确安装了。

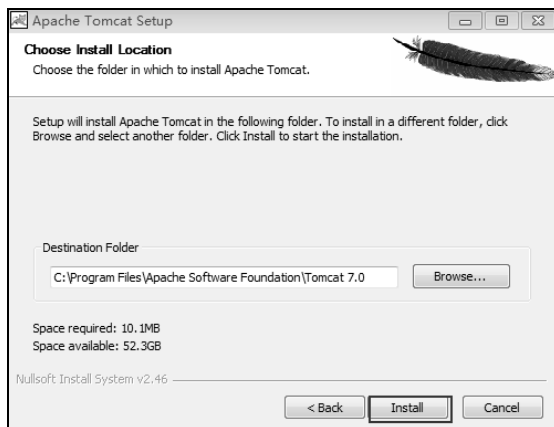


图 1-11 选择安装位置界面

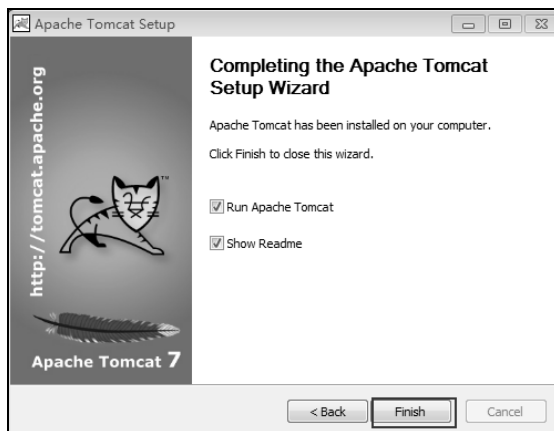


图 1-12 成功界面



图 1-13 Tomcat 服务器主页运行窗口

习 题 1

1. 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
2. 在下列选项中，不是 JSP 运行必需的要素是（ ）。

- A. 操作系统
 - B. Java JDK
 - C. 支持 JSP 的 Web 服务器
 - D. 数据库
3. 简述 B/S 模式和 C/S 模式。
 4. 简述三层架构及其特点。
 5. 简述 Java Web 程序开发环境的搭建过程。
 6. 简述你对框架概念的认识。
 7. 简述 Hibernate 框架的发展历程。
 8. Spring 框架的作用是什么？