第 1 章　Java Web 应用开发简介

随着计算机软硬件技术的高速发展以及计算机网络的普及，软件应用已从以往的单机软件扩展到了基于网络的软件，并随之产生了基于 Internet 的 Web 应用程序，Java 作为业内重要的软件开发语言，也提供了 Web 应用的开发机制。

本章简要介绍了 Java Web 应用开发涉及的相关技术。希望读者通过本章的学习能够了解 Java Web 应用开发的整体技术组成，以及各部分技术在项目开发中的相关应用，同时对 Java Web 应用开发有一个全面的了解，以便后续相关内容的开展。

1.1　Java Web 编程简介

Java Web 是用 Java 技术来解决相关 Web 互联网领域的技术总和。Web 包括服务器和客户端两部分，Java 在客户端的应用原本就有 Java Applet，不过目前用得很少，而 Java 在服务器端的应用则非常丰富，比如 Servlet、JSP 和第三方框架等。Java 技术对 Web 领域的发展注入了强大的动力。

1.1.1　C/S 和 B/S

目前，应用软件系统的开发有两种基本的结构：C/S 结构和 B/S 结构。C/S 结构，即大家熟知的客户机和服务器结构。它是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到 Client 端和 Server 端来实现，降低了系统的通信开销。

目前相当多的应用软件系统都是 Client/Server 形式的两层结构，采用这种结构编写的软件需要分别在客户机和服务器上进行安装，该结构的使用在用户数据录入方面很有优势，也降低了系统的通信开销，但是也有一定的不足之处。例如，当版本更新时，所有的客户端软件都必须进行升级安装或者重新安装，这就给软件的使用者或系统管理员带来很大的不便。与此同时，由于不同的客户可能使用不同版本的客户端软件，在设计和升级服务器端软件的时候不得不考虑各个版本的兼容性。随着浏览器的功能越来越强大，在许多场合下，浏览器可以取代 C/S 模式中的客户端软件，这就使得浏览器作为统一客户端的想法成为可能，出现了另外一种结构——B/S 结构。

B/S 结构是 Web 兴起后的一种网络结构模式，即浏览器/服务器模式（Browser/Server），Web 浏览器是客户端最主要的应用软件。这种结构统一了客户端，将系统功能实现的核心部分集中到服务器上，简化了系统的开发、维护和使用。客户机上只要安装一个浏览器（如Internet Explorer或Netscape Navigator）就可以与服务器进行数据交互。

B/S 结构可以说是 C/S 结构的变体或者说是改进，B/S 结构围绕 Web 服务器进行开发。Web 服务器是安装了 Web 服务软件的计算机，它能接收客户端发出的 HTTP 请求。如在浏览器地址栏中输入 http：//www.baidu.tom 即会向 Web 服务器发出请求，Web 服务器处理请求后，产生 HTML 脚本发回客户端，而客户端就不需要像 C/S 那样专门编写一个程序，而是变为 IE

浏览器，接收服务器返回的 HTML，然后将该 HTML 显示出来，提供人机交互界面。

B/S 开发中的重点就是编写 Web 程序。目前大部分流行的 Web 程序都采用 HTML 加服务器端嵌入式脚本的方式组织。这样的 Web 程序本质上就是一个纯文本文件，可以随时对其进行修改而不需要重新编译。服务器在接收了一个 HTTP 地址后，就按照事先排定的规则（可能是服务器端的设置或是由配置文件说明），将请求映射到一个 Web 文件，然后由 Web 服务器来加载这个 Web 文件，并且解释执行。执行后得到 HTML 的结果（可以是其他的文档类型，如 doc、xls、jpg 等），将这个 HTML 结果返回客户端，由客户端的浏览器解释执行。

B/S 模式的优点在于：

（1）客户端基于统一的 Web 浏览器，减少了投资，解决了系统维护升级的问题。

（2）具有灵活性和可扩展性。系统可根据规模的不断扩大，在不影响用户日常工作的前提下，对 Web 服务器和数据库服务器等设备进行扩展。

（3）简易性。操作直观、简单，培训方便，对使用人员的计算机操作水平要求不高。

（4）实施成本低。充分利用现有的办公网络，避免了网络重复建设。

1.1.2 静态 Web 和动态 Web

一个典型的 Web 应用中通常包含静态页面和动态页面。这两种页面的请求处理方式有所不同。静态网页也称为普通网页，是相对于动态网页而言的，并不是指网页中的元素都是静止不动的，而是指在通过浏览器进行浏览时，Web 服务器中不再发生动态改变（没有表单处理程序或其他应用程序的执行），因此网页不是即时生成的。

浏览器“阅读”静态网页的执行较为简单，浏览器向网络中的 Web 服务器发出请求，该请求可以是用户在浏览器地址栏里输入一个 Web 地址，也可以是用户单击一个链接指向某个普通网页，浏览器会使用标准的 HTTP 协议（HyperText Transfer Protocol，超文本传送协议），把一个 HTTP 请求（HTTP Request）发送给 Web 站点的服务器。当 Web 站点接收到浏览器端发送来的一个 HTTP 请求后，服务器检查其上是否存在客户端所请求的文件，如果该文件存在，则用 HTTP 响应（HTTP Response）的形式把客户端所请求的文件送回给客户端的浏览器。当客户端浏览器接收到了 HTTP 响应后，开始解读 HTML 标签，然后将其转换，将结果显示出来。

动态网页是指网页中除了静态网页中的元素外，还包括一些应用程序。这些应用程序使浏览器与 Web 服务器之间发生交互，而且应用程序的执行有时需要应用程序服务器支持才能够完成。例如百度搜索页面，当在文本框中输入不同的搜索内容，单击“百度一下”按钮时，则会把要查找的内容作为参数发送给对应的 Web 应用程序（在此指表单处理），然后 Web 应用程序会根据发送来的参数进行处理（如读取数据库），生成一个 HTML 页面，把生成的页面送回给客户端的浏览器。

当客户端浏览器向网络中的 Web 服务器发出请求时，浏览器会将用户输入的信息一起发送到 Web 服务器。Web 服务器接收到请求信号后，将该网页发送至 Web 容器。Web 容器检查该网页，执行其中的应用程序，在执行应用程序的过程中可能会查询数据库，查询完数据库，应用程序服务器会将查询到的数据插入网页中，此时动态网页变为静态网页。Web 服务器将完成的静态网页传给浏览器，浏览器接到 Web 服务器送来的信号后，开始解读 HTML 标签并将其转换，有时还执行脚本程序，然后将结果显示出来。

1.1.3　Java Web 工作原理

互联网上的所有网站大部分都是使用 HTTP 构建的 Web 应用程序，一个基本的 Web 应用程序需要 Web 服务器和 Web 客户端浏览器。通常情况下，Web 服务器和 Web 客户端浏览器通过 HTTP 进行网络通信。其中，Web 服务器的作用是接收客户端请求，然后向客户端返回一些结果；浏览器的作用是允许用户请求服务器上的某个资源，并且向用户显示请求的结果，通常情况下浏览器从服务器得到的是一个 HTML 页面，HTML 页面可以告诉浏览器怎样向用户显示内容。

HTTP 是一套计算机在网络中通信的规则。在 TCP/IP 层次中，HTTP 属于应用层协议，位于 TCP/IP 的顶层。HTTP 是一种无状态的协议，是指在 Web 浏览器和 Web 服务器之间不需要建立持久的连接。整个过程就是当一个客户端向服务器端发送一个请求（Request），然后 Web 服务器返回一个响应（Response），之后连接就关闭了。HTTP 遵循请求/响应（Request/Response）模型，所有的通信交互都被构造在一套请求和响应模型中。浏览 Web 时，浏览器通过 HTTP 与 Web 服务器交换信息，Web 服务器向 Web 浏览器返回的内容都是与之相关的类型，这些信息类型的格式由 MIME（Multipurpose Internet Mail Extensions，多用途互联网邮件扩展类型）定义。

注释：MIME 类型就是设定某种扩展名的文件用某一种应用程序来打开的方式类型，当该扩展名文件被访问的时候，浏览器会自动使用指定应用程序来打开。

HTTP 构建的 Web 应用程序的工作步骤如下：

（1）客户端和 Web 服务器建立连接。

（2）客户端发送 HTTP 请求。

（3）服务器端接收客户端的 HTTP 请求，生成 HTTP 响应回发。

（4）服务器端关闭连接，客户端解析响应，恢复页面。

1.2　Java Web 开发技术简介

应用 Java 技术开发 Web 应用系统需要掌握一系列相关技术，主要包括 HTML、XML、JavaScript、JDBC、Servlet、JSP 等方面。

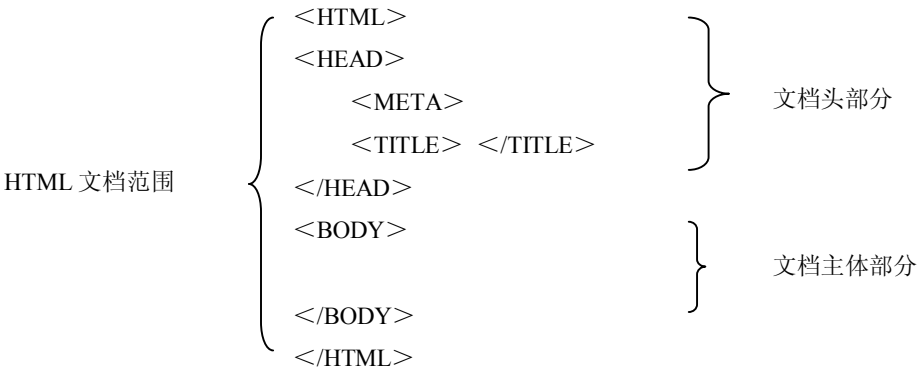
1.2.1　HTML

超文本标记语言 HTML 是当前网页设计领域最基础的应用语言，使用 HTML 语言所编写超文本文件（或称 HTML 文档）成为万维网上最普遍的网页形式之一。HTML 语言来源于著名的标准通用标记语言 SGML（Standard Generalized Markup Language），由万维网之父 Tim Berners-Lee 于 1990 年创建一个基于超文本的分布式应用系统时提出。作为 SGML 语言的子集，HTML 语言摒弃了 SGML 语言过于复杂、不利于信息传递和解析的不足，选用最基本的元素——标记（Tags）进行超文本描述，达到了简化、易懂的目的。

HTML 文档主要是由将要显示在网页上的文档内容和一系列标记所组成。当用户浏览 HTML 文档时，浏览器就会把这些标记解释成它应有的含义，并按照一定的格式，将这些被标识的文档内容显示在浏览器窗口中。在 HTML 文档中有些标记必须以“＜标记＞”开始，

而以“＜/标记＞”结束，这些标记称之为“成对标记”；有些标记并不需要确定作用域，称之为“非成对标记”。

一个标准的 HTML 文档具有如下结构：



＜HTML＞标记是成对标记。一个完整的 HTML 文档是以＜HTML＞标记开始，以＜/HTML＞标记结束的，用来告知浏览器该成对标记之间的内容是使用 HTML 格式编写的，浏览器会使用 HTML 规范来解释和显示其中的内容。

＜HEAD＞标记是成对标记。＜HEAD＞和＜/HEAD＞标记之间的内容是 HTML 文档的头部分，用来规定该文档的标题（出现在 Web 浏览器窗口的标题栏中）和文档的一些属性。在＜HEAD＞标记之间可引用＜META＞、＜TITLE＞等标记。

＜META＞标记是非成对标记，它位于＜HEAD＞标记之间，用以记录当前页面的一些重要信息，例如网页所依据的字符集、开发者、开发语言版本、网页关键字等。

＜TITLE＞标记是成对标记，用以规定 HTML 文档的标题。位于该成对标记之间的内容将显示在 Web 浏览器窗口的标题栏中。

＜BODY＞标记是成对标记，该成对标记之间的内容将显示在 Web 浏览器窗口的用户区域内，它是 HTML 文档的主体部分。

1.2.2 XML

XML（eXtensible Markup Language，可扩展标记语言）（标准通用标记语言的子集），是一种简单的数据存储语言，使用一系列简单的标记描述数据，而这些标记可以用方便的方式建立，虽然可扩展标记语言占用的空间比二进制数据要占用更多的空间，但可扩展标记语言极其简单，易于掌握和使用。

可扩展标记语言（XML）与 Access、Oracle 和 SQL Server 等数据库不同，数据库提供了更强有力的数据存储和分析能力，例如：数据索引、排序、查找、相关一致性等，XML 的宗旨是传输数据，而与其同属标准通用标记语言的 HTML 主要用于显示数据。XML 的最大优点是极其简单，而正是这一优点使得 XML 与众不同。

XML的前身是标准通用标记语言，同HTML一样，可扩展标记语言是标准通用标记语言的一个子集，它是描述网络上的数据内容和结构的标准。尽管如此，XML 不像 HTML，HTML 仅仅提供了在页面上显示信息的通用方法（没有上下文相关和动态功能），XML 则对数据赋予上下文相关功能，它继承了标准通用标记语言的大部分功能，却使用了不太复杂的技术。

为了使得标准通用标记语言显得友好，XML 重新定义了标准通用标记语言的一些内部值和参数，去掉了大量的很少用到的功能，这些繁杂的功能使得标准通用标记语言在设计网站时显得复杂。XML 保留了标准通用标记语言的结构化功能，这样就使得网站设计者可以定义自己的文档类型，XML 同时也推出一种新型文档类型，使得开发者也可以不必定义文档类型。

因为 XML 是 W3C 制定的，XML 的标准化工作由 W3C 的 XML 工作组负责，该小组成员由来自各个地方和行业的专家组成，他们通过 E-mail 交流对 XML 标准的意见，并提出自己的看法。因为 XML 是个公共格式，不专属于任何一家公司，所以不必担心 XML 技术会成为少数公司的盈利工具，XML 不是一个依附于特定浏览器的语言。

1.2.3　JavaScript

JavaScript 是 NetScape 公司为 Navigator 浏览器开发的，起初命名 LiveScript。在 NetScape 发展 LiveScript 的同时，Sun 的公司也正在发展 Java 语言，为了辅助 Java 的网页程式方面的设计，两家公司合作开发 LiveScript 语言，并更名为 JavaScript。JavaScript 是写在 HTML 文件中的一种脚本语言，能实现网页内容的交互显示。当用户在客户端显示该网页时，浏览器就会执行 JavaScript 程序，用户通过交互式的操作来变换网页的内容，以实现 HTML 语言所不能实现的效果。

JavaScript 在 Navigator 成功应用之后不久，Microsoft 公司也推出了用于 IE 浏览器的、类似 JavaScript 的程序语言，并将其命名为 JScript。1999 年，欧洲计算机制造商协会（ECMA）在 JavaScript 1.5 版基础上制定了“ECMAScript 程序语言规范书”（ECMA-262 标准），该标准被国际标准化组织（ISO）采纳，作为各个浏览器使用的脚本程序的统一标准。ECMA-262 标准公布后，JavaScript 和 JScript 之后开发的新功能都要遵循该标准，但 JavaScript 与 JScript 两个名称仍被两大公司分别使用。JavaScript 与 JScript 绝大部分是相同的，但是也有区别。

JavaScript 是一种解释性的语言，其源代码不经过编译而在运行时直接被“翻译”，也称为“脚本式”语言。正是由于 JavaScript 的代码是解释执行，所以不同浏览器对 JavaScript 的支持是不一样的。

1.2.4　JDBC

JDBC（Java DataBase Connectivity，Java 数据库连接）是一种用于执行 SQL 语句的 Java API，可以为多种关系数据库提供统一访问，它由一组用Java 语言编写的类和接口组成。JDBC 提供了一种基准，据此可以构建更高级的工具和接口，使数据库开发人员能够编写数据库应用程序，同时，JDBC 也是个商标名。

有了 JDBC，向各种关系数据发送 SQL 语句就是一件很容易的事。换言之，有了 JDBC API，就不必为访问Sybase 数据库专门写一个程序，为访问Oracle 数据库又专门写一个程序，或为访问 Informix 数据库又编写另一个程序等，程序员只需用 JDBC API 写一个程序就够了，它可向相应数据库发送 SQL 调用。同时，将 Java 语言和 JDBC 结合起来可以使程序员不必为不同的平台编写不同的应用程序，只须写一遍程序就可以让它在任何平台上运行，这也是 Java 语言“编写一次，处处运行”的优势。

Java 数据库连接体系结构是用于 Java 应用程序连接数据库的标准方法。JDBC 对 Java 程

序员而言是 API，对实现与数据库连接的服务提供商而言是接口模型。作为 API，JDBC 为程序开发提供标准的接口，并为数据库厂商及第三方中间件厂商实现与数据库的连接提供了标准方法。JDBC 使用已有的 SQL 标准并支持与其他数据库连接标准，如 ODBC 之间的桥接。JDBC 实现了所有这些面向标准的目标，并且具有简单、严格类型定义且高性能实现的接口。

JDBC 扩展了 Java 的功能。例如，用 Java 和 JDBC API 可以发布含有 Applet 的网页，而该 Applet 使用的信息可能来自远程数据库。企业也可以用 JDBC 通过 Intranet 将所有职员连到一个或多个内部数据库中（即使这些职员所用的计算机有 Windows、Macintosh 和 UNIX 等各种不同的操作系统）。随着越来越多的程序员开始使用 Java 编程语言，对从 Java 中便捷地访问数据库的要求也在日益增加。

MIS（Management Information System，管理信息系统）管理员们都喜欢 Java 和 JDBC 的结合，因为它使信息传播变得容易和经济。企业可继续使用它们安装好的数据库，并能便捷地存取信息，即使这些信息是储存在不同数据库管理系统上。新程序的开发期很短。安装和版本控制将大为简化。程序员可只编写一遍应用程序或只更新一次，然后将它放到服务器上，随后任何人就都可以得到最新版本的应用程序。对于商务上的销售信息服务，Java 和 JDBC 可为外部客户提供获取信息更新的更好方法。

1.2.5 Servlet

Servlet（Java Servlet）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。通常来讲，Servlet 是指任何实现了 Servlet 接口的类。Servlet 运行于支持 Java 的应用服务器中。从实现上讲，Servlet 可以响应任何类型的请求，但绝大多数情况下 Servlet 只用来扩展基于 HTTP 协议的 Web 服务器。最早支持 Servlet 标准的是 JavaSoft 的 Java Web Server。此后，一些其他基于 Java 的 Web 服务器开始支持标准的 Servlet。

Servlet 的主要功能在于交互式地浏览和修改数据，生成动态 Web 内容。这个过程通常包括以下 4 个步骤：

（1）客户端发送请求至服务器端。

（2）服务器将请求信息发送至 Servlet。

（3）Servlet 生成响应内容并将其传给服务器。响应内容动态生成，通常取决于客户端的请求。

（4）服务器将响应返回给客户端。

Servlet 看起来像是平常的 Java 程序。Servlet 导入特定的属于 Java Servlet API 的包。因为是对象字节码，可动态地从网络加载，可以说 Servlet 对 Server 就如同 Apple 对 Client 一样。但是，由于 Servlet 运行于 Server 中，它们并不需要一个图形用户界面。

一个 Servlet 就是 Java 编程语言中的一个类，它被用来扩展服务器的性能，服务器上驻留着可以通过“请求—响应”编程模型来访问的应用程序。虽然 Servlet 可以对任何类型的请求产生响应，但通常只用来扩展 Web 服务器的应用程序。

1.2.6 JSP

JSP（Java Server Pages，Java 服务器页面）是一个简化的Servlet设计，它是由Sun Microsystems 公司倡导、许多公司参与一起建立的一种动态网页技术标准。JSP 技术有点类似 ASP 技术，它是

在传统的网页HTML（标准通用标记语言的子集）文件（.htm,.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。

JSP 实现了 HTML 语法中的 Java 扩展（以<%，%>形式）。JSP 与 Servlet 一样，是在服务器端执行的，通常返回给客户端的就是一个 HTML 文本，因此客户端只要有浏览器就能浏览。

JSP 技术使用 Java 编程语言编写类 XML 的 Tags 和 Scriptlets，来封装产生动态网页的处理逻辑。网页还能通过 Tags 和 Scriptlets 访问存在于服务端的资源的应用逻辑。JSP 将网页逻辑与网页设计的显示分离，支持可重用的基于组件的设计，使基于 Web 的应用程序的开发变得迅速和容易。JSP 是一种动态页面技术，它的主要目的是将表示逻辑从Servlet中分离出来。

Java Servlet 是 JSP 的技术基础，而且大型的 Web 应用程序的开发需要 Java Servlet 和 JSP 配合才能完成。JSP 具备了面向因特网的所有特点，简单易用、完全地面向对象，平台无关性且安全可靠。

1.3　Java Web 开发环境的搭建

项目开发需要合适的开发环境，Java Web 涉及 XML、Servlet、JSP、JDBC 等相关技术，需要诸多软件相互配合，共同搭建开发环境。由于 Servlet、JDBC 等内容将在后面陆续展开，这里先讲解基本开发环境的搭建，随着内容的进一步深入，开发环境的搭建将越来越完善。

1.3.1　JDK 的安装与配置

JDK 8.0 是 Java 虚拟机的最新版本，加入了很多新的特性。学习 Java，就必须有 JDK，可谓“工欲善其事，必先利其器”。JDK 8.0 有很多版本，分别对应 Windows 操作系统包括的 32 位版本或 64 位版本，以 JDK 8.0 的第 45 个更新包为例：32 位版本安装包对应 jdk-8u45-windows-i586.exe，64 位版本安装包对应 jdk-8u45-windows-x64.exe，要根据自己的机器分别下载相应的 JDK 8.0 版本。

首先，登录官网 http://www.oracle.com/technetwork/java/javase/downloads/index.html 下载 32 位版本安装包 jdk-8u45-windows-i586.exe，双击开始安装，弹出欢迎界面。如图 1.1 所示。

图 1.1　JDK 安装欢迎界面

单击“下一步”按钮，进入定制安装界面。如图 1.2 所示。

图 1.2　JDK 定制安装界面

在定制安装界面可以更改安装路径，默认情况下 JDK 安装在 C:\Program File\Java\jdk1.8.0_45\目录下，选择默认安装路径，单击“下一步”按钮，进入安装进度显示界面。如图 1.3 所示。

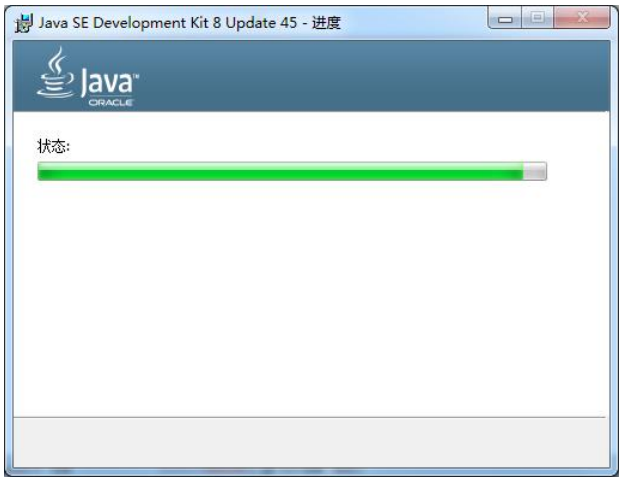


图 1.3　JDK 安装进度显示界面

当 JDK 安装将要完成时，会弹出 JRE 的安装提示界面，如图 1.4 所示。这里有必要说明一下 JDK、JRE 和 JVM 三者的区别。JDK（Java Development ToolKit，Java 开发工具包）是整个 Java 的核心，包括了 Java 运行环境（Java Runtime Environment）、Java 工具（javac、java、jdb 等）和 Java 基础的类库。JRE（Java Runtime Environment，Java 运行时环境）也就是我们说的 Java 平台，所有的 Java 程序都要在 JRE 下才能运行，它包括 JVM 和 Java 核心类库和支持文件，与 JDK 相比，它不包含开发工具（编译器、调试器）和其他工具。JVM（Java Virtual Machine，Java 虚拟机）是 JRE 的一部分，它是一个虚构出来的计算机，是通过在实际的计算机上仿真模拟各种计算机功能来实现的。JVM 有自己完善的硬件架构，如处理器、堆栈、寄

存器等，还具有相应的指令系统。JVM 的主要工作是解释自己的指令集（即字节码），并映射到本地 CPU 的指令集或 OS 的系统调用。Java 语言是跨平台运行的，其实就是不同的操作系统，使用不同的 JVM 映射规则，让其与操作系统无关，完成了跨平台性。JVM 对上层的 Java 源文件是不关心的，它关心的只是由源文件生成的类文件。类文件的组成包括 JVM 指令集、符号表以及一些补助信息。

图 1.4　JRE 安装提示界面

默认情况下 JRE 安装在 C:\Program File\Java\jre1.8.0_45\目录下，单击“下一步”按钮，进入 JRE 安装进度显示界面。如图 1.5 所示。

图 1.5　JRE 安装进度显示界面

JRE 安装完成后，JDK 也随即安装完成，系统显示 JDK 安装完成界面。如图 1.6 所示。

图 1.6　JDK 安装完成界面

JDK 安装完成后，打开 CMD，输入 java 命令，弹出如图 1.7 所示的提示信息，说明 JDK 安装成功。

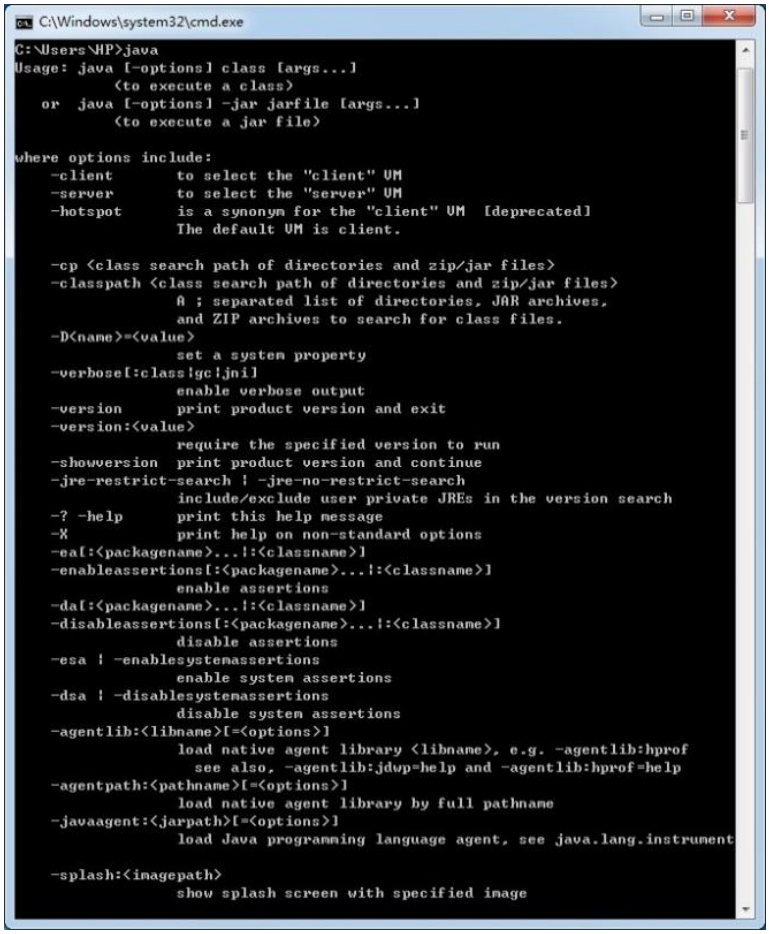


图 1.7　JDK 安装成功界面

1.3.2　Eclipse J2EE 的安装与配置

Eclipse 是一个开放源代码的、基于Java的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse 附带了一个标准的插件集，包括 Java开发工具（Java Development Kit，JDK）。Eclipse 有诸多版本，考虑到本书以讲解 Web 开发为主，所以选择 Eclipse IDE for Java EE Developers 这个版本，以方便我们建立 Web 项目。

从 Eclipse 官网http://www.eclipse.org/downloads/下载 eclipse-jee-luna-SR2-win32.zip（32 位版本）压缩包，如果操作系统是 64 位版本可下载 eclipse-jee-luna-SR2-win32-x86_64.zip 并直接解压至 C 盘，双击 eclipse.exe 即可启动 Eclipse。如图 1.8 所示。

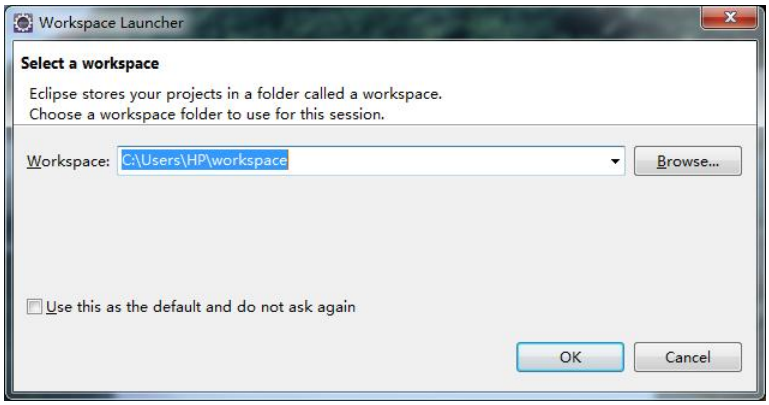


图 1.8　Eclipse 路径选择界面

在每次启动 Eclipse 时会出现路径选择界面，选择工程文件存放目录，建议大家在 Eclipse 的根目录下创建一个名为 workspace 的目录来存放工程文件，选择默认路径也可以，单击 OK 按钮进入 Eclipse 欢迎界面。如图 1.9 所示。

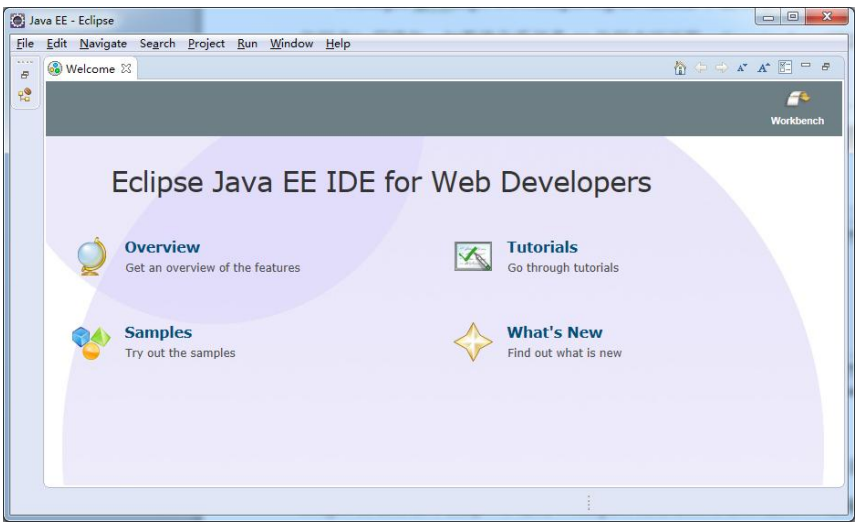


图 1.9　Eclipse 欢迎界面

关闭 Welcome 界面进入工作界面，如图 1.10 所示。

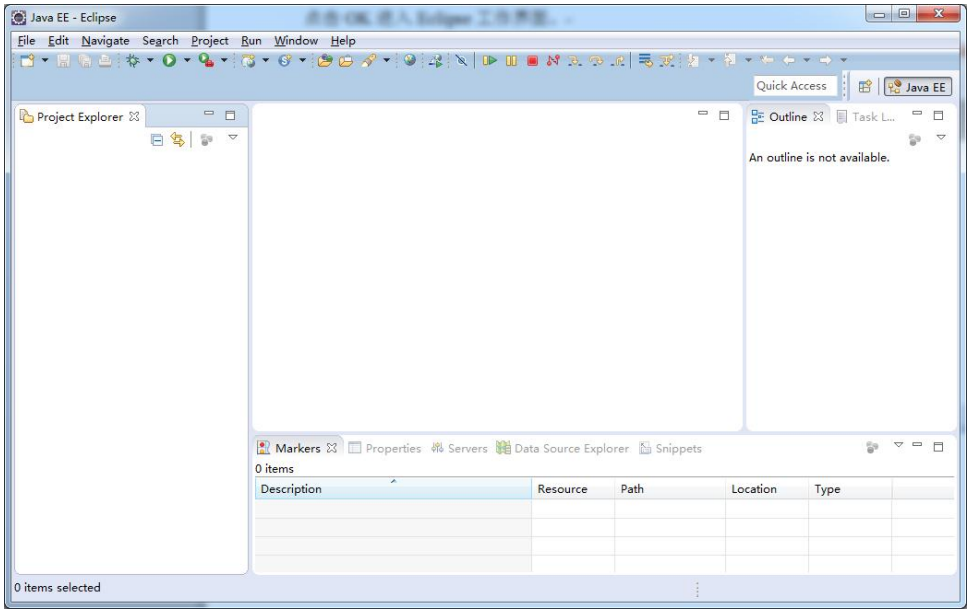


图 1.10　Eclipse 工作界面

1.4　小结

本章概括介绍了 Java Web 应用开发的相关概念、相关技术和相关工具，以及 Java Web 应用开发的基本开发环境。通过本章的学习，读者将对 Java Web 应用开发的相关问题有一个大致的了解。