第3章 Servlet基础

本章要点：

- 能知道 Servlet 的历史发展和特点。
- 能理解 Servlet 的生命周期和运行过程。
- 会编写 Servlet 代码。
- 会使用 Servlet 开发 Web 网站。
- 会部署 Servlet 到 Web 服务器 Tomcat。

3.1 Servlet 概述

在 Servlet 出现之前，Web 服务器通过 CGI 技术与客户端进行交互。具体实现时，需要使用 Perl、Shell Script 或 C 语言编写 CGI 程序，再部署到 Web 服务器，从而使 Web 服务器能处理来自客户端网页表单的输入信息并将处理后的信息反馈给客户端。但是，CGI 技术存在很多缺点，主要如下。

- CGI 程序开发比较困难。要求开发人员具备处理参数传递的能力，需要开发经验。
- CGI 程序不可移植。为某一特定平台编写的 CGI 程序只能运行于该特定平台中。
- CGI 程序内存和 CPU 开销大，而且在同一进程中不能服务多个客户。因为每个 CGI 程序存在于一个由客户端请求激活的 Web 服务器进程中，并且在请求完成后被卸载。

随着 Java 语言的广泛使用，Servlet 迅速成为动态网站的主要开发技术。实际上，Servlet 可以定义为一种基于 Java 技术的 Web 组件，用来扩展以请求和响应为模型的 Web 服务的能力。它是一种独立于平台和协议的服务器端的 Java 应用程序，与传统的从命令行启动的 Java 应用程序不同，Servlet 本身没有 main 方法，不是由用户或开发人员调用的，而是由另外一个应用程序——Servlet 容器调用和管理的。其中，Servlet 容器又称为引擎，承担着运行环境的作用，管理和维护所有 Servlet 的完整生命周期，并将所有的 Servlet 编译成字节码从而生成动态的内容以供 Web 请求调用。

本书使用 Tomcat 作为运行 Java Web 网站的服务器，也就是说，Tomcat 是 Servlet 容器，也是 Servlet 的运行环境。具体运行时，Tomcat 负责把客户端的请求传递给 Servlet，并把处

理结果返回给客户端。当多个客户端请求同一个 Servlet 时，Tomcat 将处理这些并发问题并负责为每个客户端启动一个线程，之后，这些线程的运行和销毁由 Tomcat 负责。

相比于 CGI 技术，Servlet 具有以下特点。

- 高效。在服务器上仅有一个 Java 虚拟机在运行，其优势在于当多个来自客户端的请求访问时，Servlet 为每个请求分配一个线程，而不是进程。
- 方便。Servlet 提供了大量的实用程序，如处理复杂的 HTML 表单数据、读取和设置 HTTP 头，以及处理 Cookie 和跟踪会话等。
- 跨平台。Servlet 用 Java 语言编写，可以在不同操作系统平台和不同应用服务器平台运行。
- 功能强大。在 Servlet 中，许多使用传统 CGI 程序很难完成的任务都可以轻松地完成。例如，Servlet 能够直接和 Web 服务器交互，而 CGI 程序一般不能实现。Servlet 还能够在各个程序之间共享数据，使得数据库连接池之类的功能很容易实现。
- 灵活性和可扩展性强。采用 Servlet 开发的 Web 应用具备 Java 的面向对象特性，因此应用灵活，可扩展性强。

3.2 Servlet 的生命周期与运行过程

Servlet 运行在 Web 服务器的 Servlet 容器里，容器根据 Servlet 的规范，执行 Servlet 对象的初始化、运行和卸载等动作。Servlet 在容器中从创建、初始化、执行到卸载的过程被称为 Servlet 的生命周期。

（1）当客户端第一次请求 Servlet 时，容器加载开发人员编写的 HttpServlet 类。在 Java Web 网站开发过程中，开发人员编写 Servlet 代码一般直接继承 javax.servlet.http 包中的 HttpServlet 类。

（2）Servlet 容器根据客户端请求创建 HttpServlet 对象实例，并把这些实例加入 HttpServlet 实例池中。

（3）Servlet 容器调用 HttpServlet 的初始化方法 init，完成初始化。

（4）Servlet 容器创建一个 HttpServletRequest 对象实例和一个 HttpServletResponse 对象实例。HttpServletRequest 对象实例封装了客户端的 HTTP 请求信息，而 HttpServletResponse 对象实例是一个新实例。

（5）Servlet 容器把 HttpServletRequest 对象实例和 HttpServletResponse 对象实例传递给 HttpServlet 的执行方法 service。service 方法可被多次请求调用，各调用过程运行在不同的线程中，互不干扰。

（6）HttpServlet 对象实例在执行 service 方法时，通常从 HttpServletRequest 对象读取 HTTP 请求数据，访问来自 HttpSession 或 Cookie 对象的状态信息，执行特定应用的处理并且用 HttpServletResponse 对象生成 HTTP 响应数据。

（7）当 Servlet 容器关闭时会自动调用 HttpServlet 对象实例的 destroy 方法，释放运行时占用的资源。

从图 3-1 中可以看出，当多个客户端请求同一个 Servlet 时，Servlet 容器为每个客户请

求启动一个线程。init 方法只在 Servlet 第一次被请求加载时调用一次，当该 Servlet 再次被客户端请求时，Servlet 容器将启动一个新的线程，在该线程中调用 service 方法响应客户端请求。在 Servlet 生命周期的各个阶段中，执行 service 方法时是真正处理业务的阶段，所以开发人员在编写 Servlet 相关代码时，主要在这个方法中编写代码实现业务功能。

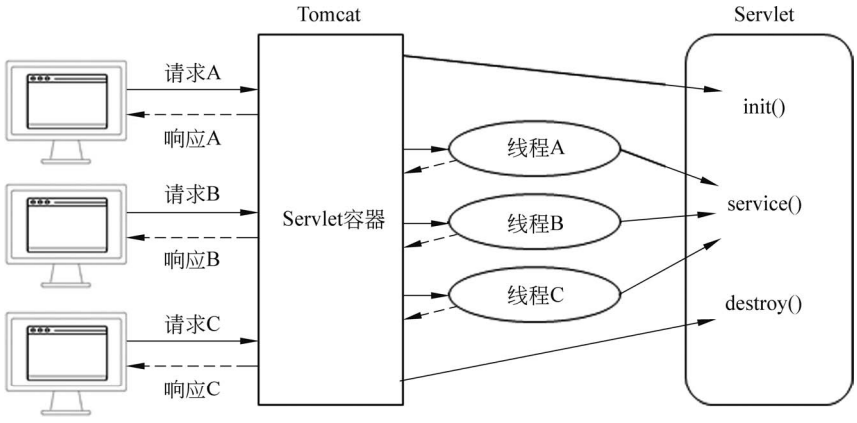

图 3-1　Servlet 的生命周期与运行过程

3.3 开发第一个 Servlet

开发编写 Servlet 类的方式是继承 javax.servlet.http 包中的 HttpServlet 类，应在 service 方法中编写代码响应请求。但是 HttpServlet 类中的 service 方法已经实现，部分源代码如下。

源程序：HttpServlet 类中 service 方法的部分代码

```
protected void service(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException
   {
      String method = req.getMethod();
      if (method.equals(METHOD_GET)) {    //若为 Get 请求
         long lastModified = getLastModified(req);
         if (lastModified == -1) {
            doGet(req, resp);
         } else {
            long ifModifiedSince = req.getDateHeader(HEADER_IFMODSINCE);
            if (ifModifiedSince < lastModified) {
               maybeSetLastModified(resp, lastModified);
               doGet(req, resp);
            } else {
               resp.setStatus(HttpServletResponse.SC_NOT_MODIFIED);
            }
         }

      } else if (method.equals(METHOD_POST)) {   //若为 Post 请求
```

```
            doPost(req, resp);
        } else { //若为其他请求
            …
        }
    }
```

从上述源代码中可知，service 方法根据请求的类型 Get、Post 等，调用相应的方法 doGet、doPost 等，所以开发人员实际编写 Servlet 类时，只要实现 doGet、doPost 等方法即可。


实例 3-1

实例 3-1　第一个 Servlet 程序

本实例利用 Servlet 的 doGet 方法实现在页面上输出节能环保的提示信息。

源程序：FirstServlet.java

```
package com.example.ch03;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;
import java.io.IOException;
import java.io.PrintWriter;
@WebServlet("/FirstServlet")
public class FirstServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request,
                         HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        response.setCharacterEncoding("UTF-8");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>节能环保 tips：空调设置温度：" +
                    "夏季不得低于 26℃；冬季不得高于 20℃。</h3>");
        out.println("</body></html>");
    }
    @Override
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
            throws ServletException, IOException {
        doGet(request, response);
    }
}
```

操作步骤：

（1）在 Book 项目中创建 ch03 模块。右击 Book 项目名称，在弹出的快捷菜单中选择 New→Module 命令，在弹出的对话框中选择模块类型为 Java Enterprise，设置模块名称为 ch03、位置为 D:\ideaProj\Book\ch03，选择模块结构为 Web application，选择应用服务器为 Tomcat 9.0.29，然后单击 Next 按钮，在弹出的对话框中单击 Finish 按钮，ch03 模块建立完成。

（2）右击 com.example. ch03 包，在弹出的快捷菜单中选择 New→Servlet 命令，然后输

入名称 FirstServlet，按 Enter 键确认建立文件，项目文件结构如图 3-2 所示。

图 3-2　项目文件的结构图

（3）在 FirstServlet.java 文件中输入源代码。

（4）单击运行工具条，选择 Edit Configurations，如图 3-3 所示。

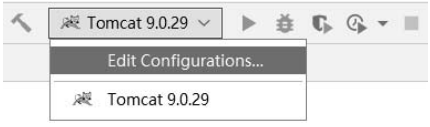

图 3-3　选择配置 Tomcat

（5）在 Run/Debug Configurations 对话框中选择 Deployment 选项卡，单击“+”按钮，选择 Artifact 选项，在下拉列表中选择 ch03:war exploded，单击 OK 按钮，最后修改 Application context 为“/ch03”。单击 OK 按钮，完成 ch03 模块网站在 Tomcat 中的部署。配置界面如图 3-4 所示。

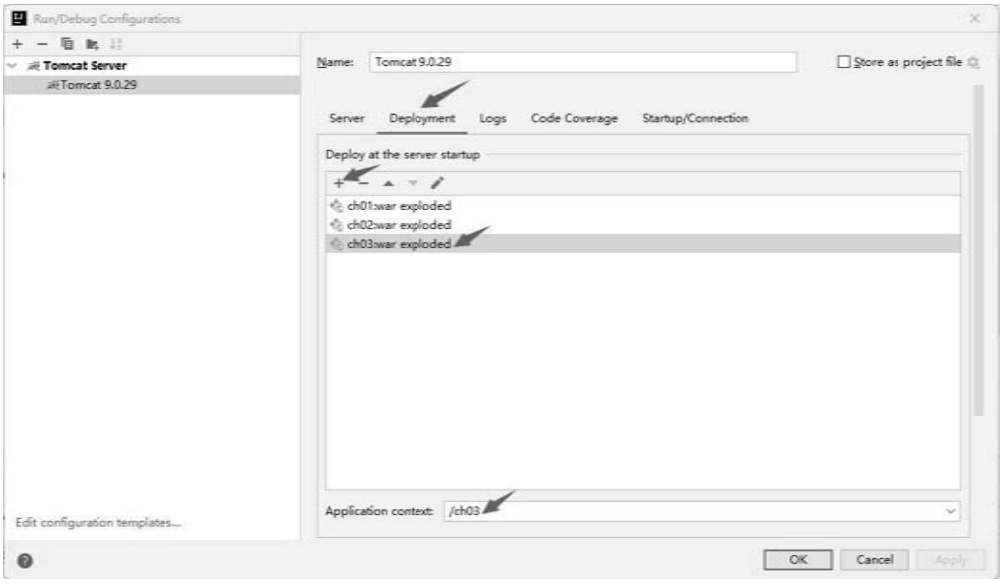

图 3-4　部署 ch03 模块网站

（6）单击运行工具栏中的“运行”按钮 ▶，启动 Tomcat。Tomcat 在启动时，IDEA 界面中将输出 Tomcat 启动和部署当前项目的相关信息，如图 3-5 所示。其中①表示 Tomcat 耗时 32ms 完成启动；②表示当前项目已经成功部署到 Tomcat 中。

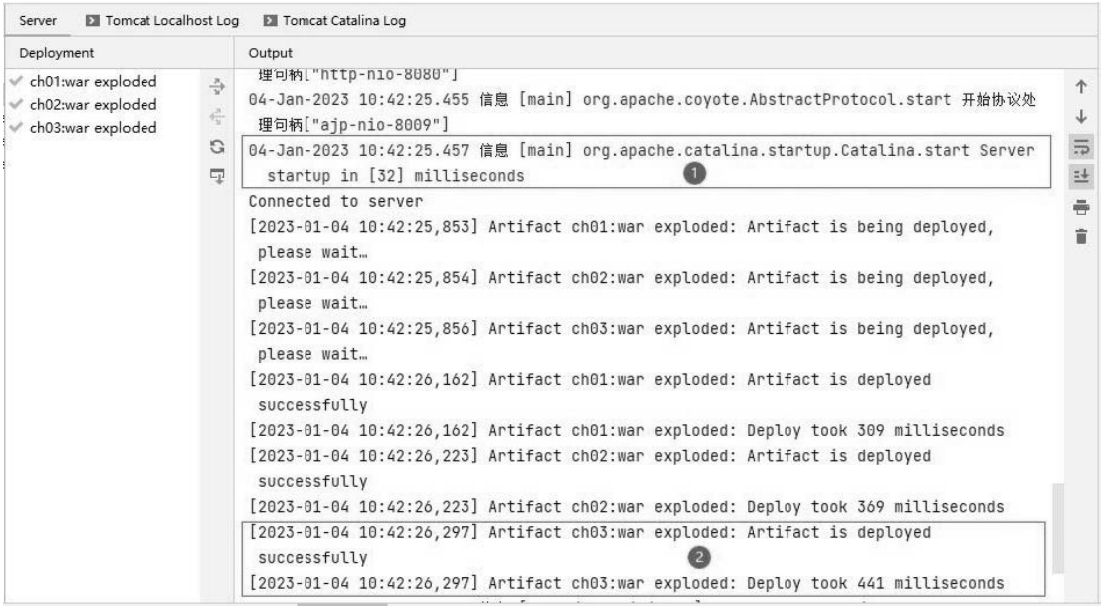

图 3-5　Tomcat 启动时的提示信息

（7）在浏览器地址栏中输入网址 http://localhost:8080/ch03/FirstServlet，查看运行效果，如图 3-6 所示。

图 3-6　FirstServlet.java 的运行效果

程序说明：

- “response.setContentType("text/html");”表示响应内容格式为 HTML 文本。
- “response.setCharacterEncoding("UTF-8");”表示响应内容语言编码字符集为 UTF-8。
- 在 Servlet 直接输出 HTML 页面信息，必须执行上述两行代码。
- “操作步骤”中的（4）（5）两步，仅需要在 ch03 模块网站第一次运行时设置。

3.4 Servlet 的部署方法

Servlet 容器是 Servlet 的运行环境，管理和维护 Servlet 的完整生命周期，所以 Servlet 必须要部署到 Servlet 容器中，才能运行起来处理客户端请求。Servlet 的部署方法有两种：

一种是通过 web.xml 部署；另一种是通过注解部署。

■ 3.4.1　通过 web.xml 部署 Servlet

在 IDEA 开发工具中创建 Web Application 项目时，会自动创建 webapp 和 WEB-INF 文件夹。webapp 是本项目 Web 站点的根节点，用于存放 html、jsp、css、js、jpg 等类型资源文件，WEB-INF 中包含 Web 站点的配置文件 web.xml。

实例 3-2　通过 web.xml 部署 Servlet

本实例利用 web.xml 文件设置 Servlet 的配置信息。


实例 3-2

源程序：**DeployServlet.java**

```
package com.example.ch03;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.IOException;
import java.io.PrintWriter;
public class DeployServlet extends HttpServlet {
   @Override
   protected void doGet(HttpServletRequest request,
                        HttpServletResponse response)
         throws ServletException, IOException {
      response.setContentType("text/html");
      response.setCharacterEncoding("utf-8");
      PrintWriter out = response.getWriter();
      out.println("<html><body>");
      out.println("<h3>节能环保 tips：节能减排靠大家;低碳生活一同享。</h3>");
      out.println("</body></html>");
   }
   @Override
   protected void doPost(HttpServletRequest request,
                         HttpServletResponse response)
         throws ServletException, IOException {
      doGet(request, response);
   }
}
```

源程序：**web.xml 文件代码**

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
             http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
        version="4.0">
   <servlet>
      <servlet-name> DeployServlet</servlet-name>
      <servlet-class>com.example.ch03.DeployServlet</servlet-class>
```

```
        <init-param>
            <param-name>count</param-name>
            <param-value>100</param-value>
        </init-param>
        <load-on-startup>5</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>DeployServlet</servlet-name>
        <url-pattern>/DeployServlet</url-pattern>
    </servlet-mapping>
</web-app>
```

操作步骤：

（1）右击 com.example.ch03 包，在弹出的快捷菜单中选择 New→Servlet 命令，然后输入名称 DeployServlet，按 Enter 键确认建立文件。

（2）在 DeployServlet.java 文件中输入源代码，务必删除@WebServlet 所在行的代码。

（3）在 webapp\WEB-INF\web.xml 文件中输入源代码。

（4）单击运行工具栏中的“运行”按钮 ▶，启动 Tomcat。

（5）在浏览器地址栏中输入网址 http://localhost:8080/ch03/DeployServlet，查看运行效果，如图 3-7 所示。

图 3-7　实例 3-2 的运行效果

程序说明：

- 删除@WebServlet所在行的代码，是因为开发工具创建的Servlet文件默认包含注解，而本实例是通过 web.xml 部署 Servlet，所以需要删除注解。
- <web-app>标记。XML 文件中必须有一个根标记，web.xml 的根标记是<web-app>。
- <servlet>标记。可以配置多个不同名称的<servlet>标记。<servlet>标记中有四个子标记：①<scrvlet-name>子标记的内容是 Tomcat 创建 Servlet 对象的名字，不能重名；②<servlet-class>子标记的内容是指定 Tomcat 用什么类来创建 Servlet 对象；③<load-on-startup>子标记的内容是正整数或者 0 时，表示 Tomcat 在启动时就加载并初始化这个 Servlet，值越小就越先被加载，如果不指定或者为负整数则在客户端第一次请求时再加载；④<init-param>子标记内容是参数初始值，在 Servlet 的 init 方法内通过 getInitParameter 方法读取。
- <servlet-mapping>标记。需要与<servlet>标记成对使用。<servlet-mapping>中有两个子标记：①<servlet-name>子标记必须与对应<servlet>标记中的<servlet-name>内容相同；②<url-pattern>子标记用来指定客户端请求 Servlet 模式，即访问 Servlet 的 URL。例如，若<url-pattern>子标记的内容是“/SecondServlet”，那么客户端浏览器请求访问时在地址栏中输入的 URL 内容应为 http://localhost:8080/ch03/SecondServlet。

■ 3.4.2　通过注解方式部署 Servlet

Servlet 3.0 版本包含注解（Annotation）新特性，提供了更加便利的部署方式，简化了开发流程。本书中采用的开发环境为 JDK8 与 Tomcat9，支持 Servlet 4.0 版本，所以在 IDEA 开发工具中新建的 Servlet 默认使用注解方式部署 Servlet。本书后续的实例都采用注解方式部署 Servlet。

@WebServlet 注解将继承于 javax.servlet.http.HttpServlet 的类标注为可以处理用户请求的 Servlet。Tomcat 根据注解的具体属性配置将相应的类部署为 Servlet。该注解具有表 3-1 给出的常用属性。表中的属性均为可选属性，但是 value 属性或者 urlPatterns 属性是必需的，且二者不能共存，如果同时指定则忽略 value 属性。

表 3-1　注解@WebServlet 的属性

属　性　名	类　　型	描　　述
name	String	Servlet 的名称，等价于<servlet-name>。没有显式指定则默认值为全类名（包含包名和类名）
urlPatterns	String[]	访问 Servlet 的 URL，指定一组 Servlet 的 URL。等价于<url-patterm>标签
value	String[]	访问 Servlet 的 URL，等价于 urlPatterns 属性
loadOnStartup	int	Servlet 的加载优先级，等价于<load-on-startup>标签
initParams	String	Servlet 的初始化参数，指定一组 Servlet 初始化参数，等价于<init-param>标签

实例 3-3　通过注解方式部署 Servlet


实例 3-3

本实例利用@WebServlet 注解在 Tomcat 中部署 Servlet。

源程序：AnnotationServlet.java 的部分代码

```
@WebServlet( name="AnnotationServlet",
             value="/AnnotationServlet",
             initParams = {@WebInitParam(name="count", value = "100")},
             loadOnStartup =5)
public class AnnotationServlet extends HttpServlet {
   @Override
   protected void doGet(HttpServletRequest request,
     HttpServletResponse response) throws ServletException, IOException {
       response.setContentType("text/html");
       response.setCharacterEncoding("utf-8");
       PrintWriter out = response.getWriter();
       out.println("<html><body>");
       out.println("<h3>节约粮食 tips：向舌尖上的浪费说不，向光盘的您看齐。</h3>");
       out.println("</body></html>");
   }
}
```

操作步骤：

（1）右击 com.example.ch03 包，在弹出的快捷菜单中选择 New→Servlet 命令，然后输

入名称 AnnotationServlet，按 Enter 键确认建立文件。

（2）在 AnnotationServlet.java 文件中输入源代码。

（3）单击运行工具栏中的“运行”按钮 ▶，启动 Tomcat。

（4）在浏览器中输入网址 http://localhost:8080/ch03/AnnotationServlet，查看运行效果，如图 3-8 所示。

图 3-8　实例 3-3 的运行效果

程序说明：

- 本实例中，若@WebServlet 只有一个属性 value="/AnnotationServlet"，那么 value 和=符号也可以省略，即最简模式为@WebServlet("/AnnotationServlet")。
- 本书后续实例中的 Servlet 注解都采用最简模式。

3.5 请求 Servlet 的三种方式

请求 Servlet 的常用方式有三种，分别为通过超链接方式请求、通过 JSP 页面或者 HTML 页面的表单方式请求，以及通过 JavaScript 脚本的 Ajax 方式请求。

■ 3.5.1 超链接请求 Servlet


实例 3-4

实例 3-4　超链接请求 Servlet

本实例利用超链接标签中的 href 属性请求 Servlet。

源程序：young.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>超链接请求 Servlet</title>
</head>
<body>
    <a href="YoungServlet">学习古诗《教子诗》</a>
</body>
</html>
```

源程序：YoungServlet.java 的部分代码

```
@WebServlet("/YoungServlet")
public class YoungServlet extends HttpServlet {
```

```
    @Override
    protected void doGet(HttpServletRequest request,
       HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        response.setCharacterEncoding("utf-8");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>教子诗</h3>");
        out.println("<h4>【宋】余良弼</h4>");
        out.println("<p>白发无凭吾老矣，青春不再汝知乎。</p>");
        out.println("<p>年将弱冠非童子，学不成名岂丈夫。</p>");
        out.println("<p>幸有明窗并净几，何劳凿壁与编蒲。</p>");
        out.println("<p>功成欲自殊头角，记取韩公训阿符。</p>");
        out.println("</body></html>");
    }
}
```

操作步骤：

（1）右击 com.example.ch03 包，在弹出的快捷菜单中选择 New→Servlet 命令，然后输入名称 YoungServlet，按 Enter 键确认建立文件。

（2）右击 webapp 文件夹，在弹出的快捷菜单中选择 New→HTML 命令，然后输入名称 young，按 Enter 键确认建立文件。

（3）分别在 young.html 和 YoungServlet.java 文件中输入源代码。

（4）单击运行工具栏中的“运行”按钮 ▶，启动 Tomcat。

（5）在浏览器中输入网址 http://localhost:8080/ch03/young.html，查看浏览效果，如图 3-9 所示。

图 3-9　young.html 的运行效果

（6）单击页面上的超链接，请求访问 YoungServlet，运行效果如图 3-10 所示。

图 3-10　YoungServlet.java 的运行效果

程序说明：

- 超链接属性 href 的值应与@WebServlet 注解中 value 属性值一致，不包含“/”字符。
- 超链接被单击激活时，浏览器以 Get 方式发送请求。
- 在浏览器地址栏中输入请求 Servlet 的 URL，与超链接请求效果一致。

■ 3.5.2 表单请求 Servlet

通过 JSP 页面或者 HTML 页面的表单请求 Servlet，两者的方法是一致的。


实例 3-5

实例 3-5　表单请求 Servlet

本实例利用 HTML 页面提交表单方式请求 Servlet，实现登记无偿献血者信息的功能。

源程序：**donate.html** 的部分代码

```
<body>
    <h3>无偿献血登记</h3>
    <form action="DonateServlet" method="post">
        <label for="name">姓名</label>
        <input type="text" name="name" id="name"><br>
        <label >血型</label>
        <input type="radio" name="bloodtype" id="bloodtypeA" value="A">
        <label for="bloodtypeA">A 型</label>
        <input type="radio" name="bloodtype" id="bloodtypeB" value="B">
        <label for="bloodtypeB">B 型</label>
        <input type="radio" name="bloodtype" id="bloodtypeO" value="O">
        <label for="bloodtypeO">O 型</label> <br>
        <input type="submit" value="提交">
    </form>
</body>
```

源程序：**DonateServlet.java** 的部分代码

```
@WebServlet("/DonateServlet")
public class DonateServlet extends HttpServlet {
    @Override
    protected void doPost(HttpServletRequest request,
       HttpServletResponse response) throws ServletException, IOException {
        request.setCharacterEncoding("UTF-8");
        String name = request.getParameter("name");
        String bloodtype = request.getParameter("bloodtype");
        response.setContentType("text/html");
        response.setCharacterEncoding("UTF-8");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>无偿献血是无私奉献、救死扶伤的崇高行为。</h3>");
        out.println("<p>感谢您！"+bloodtype+"型血的"+name+"朋友。</p>");
        out.println("</body></html>");
```

```
    }
}
```

操作步骤：

（1）右击 com.example.ch03 包，在弹出的快捷菜单中选择 New→Servlet 命令，然后输入名称 DonateServlet，按 Enter 键确认建立文件。

（2）右击 webapp 文件夹，在弹出的快捷菜单中选择 New→HTML 命令，然后输入名称 donate，按 Enter 键确认建立文件。

（3）分别在 donate.html 和 DonateServlet.java 文件中输入源代码。

（4）单击运行工具栏中的“运行”按钮 ▶，启动 Tomcat。

（5）在浏览器中输入网址 http://localhost:8080/ch03/donate.html，查看运行效果，如图 3-11 所示。

图 3-11　donate.html 的运行效果

（6）在页面上输入姓名，选择血型，单击“提交”按钮，运行效果如图 3-12 所示。

图 3-12　DonateServlet.java 的运行效果

程序说明：

- 表单属性 action 的值应与@WebServlet 注解中 value 属性值一致，不包含“/”字符。
- 表单属性 method 的值应与 Servlet 中的服务方法对应，即 get 对应 doGet 方法，post 对应 doPost 方法。
- “request.getParameter("name");”为读取表单中的姓名信息。
- “request.getParameter("bloodtype");”为读取表单中的血型信息，4.7.1 节将详细介绍该方法。
- 通常情况下，Servlet 的 doGet 与 doPost 方法中的处理请求代码相同，所以可以在 doGet 中编写处理请求的详细代码，而在 doPost 中调用 doGet，从而优化代码量。

■ 3.5.3 Ajax 方法请求 Servlet


实例 3-6

实例 3-6 Ajax 方法请求 Servlet

本实例利用 JavaScript 脚本中的 Ajax 方法请求 Servlet，实现了在公益志愿者注册时校验用户名是否重复的功能。

源程序：register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>公益志愿者用户注册</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
<form action="RegisterServlet" method="post">
    <fieldset>
        <legend>公益志愿者用户注册</legend>
        <label for="userName">用户名称：</label>
        <input type="text" name="userName" id="userName">
        <span id="msg"></span><br>
        <label for="userEmail">邮箱地址：</label>
        <input type="email" name="userEmail" id="userEmail" ><br>
        <label for="realName">真实姓名：</label>
        <input type="text" name="realName" id="realName"><br>
        <label >服务类别:</label>
        <input type="checkbox" name="serviceType"
               id="serviceType1" value="社区建设">
        <label for="serviceType1">社区建设</label>
        <input type="checkbox" name="serviceType"
               id="serviceType2" value="环境保护">
        <label for="serviceType2">环境保护</label>
        <input type="checkbox" name="serviceType"
               id="serviceType3" value="大型赛会">
        <label for="serviceType3">大型赛会</label>
        <input type="checkbox" name="serviceType"
               id="serviceType4" value="应急救助">
        <label for="serviceType4">应急救助</label><br>
        <input type="submit" value="提交">
    </fieldset>
</form>
<script>
    $(document).ready(function (){
        //用户名称输入文本框失去焦点事件代码
        $("#userName").blur(function (){
```

```
            let inputNameString = $(this).val();
            $.ajax({
                type: "post",
                url: "CheckNameServlet",
                data: {"userName":inputNameString},
                success: function (returnData){
                    if(returnData == "isUsed"){
                        $("#msg").text("用户名被占用，请修改");
                        $("#userName").select();
                    }
                    else{
                        $("#msg").text("用户名可用");
                    }
                }
            });
        })
    })
</script>
</body>
</html>
</html>
```

源程序：CheckNameServlet.java 的部分代码

```
@WebServlet("/CheckNameServlet")
public class CheckNameServlet extends HttpServlet {
    @Override
    protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException{
        request.setCharacterEncoding("UTF-8");
        String userName = request.getParameter("userName");
        response.setContentType("text/html");
        response.setCharacterEncoding("UTF-8");
        PrintWriter out = response.getWriter();
        //简化实例，假设用户名 userA 已被占用,当用户提交的用户名为 userA 时
        //Servlet 响应输出 isUsed，否则响应输出 notUsed
        if (userName.equals("userA")){
            //响应输出使用 print 方法
            //因为 println 方法会带上换行符号，不便于前端 JavaScript 代码进行字符串比较
            out.print("isUsed");
        }else{
            out.print("notUsed");
        }
    }
}
```

操作步骤：

（1）右击 com.example.ch03 包，在弹出的快捷菜单中选择 New→Servlet 命令，然后输

入名称 CheckNameServlet，按 Enter 键确认建立文件。

（2）右击 webapp 文件夹，在弹出的快捷菜单中选择 New→HTML 命令，然后输入名称 register，按 Enter 键确认建立文件。

（3）在 register.html 和 CheckNameServlet.java 文件中输入源代码。

（4）单击运行工具栏中的“运行”按钮 ▶，启动 Tomcat。

（5）在浏览器中输入网址 http://localhost:8080/ch03/register.html，查看运行效果，如图 3-13 所示。

图 3-13　register.html 的运行效果

（6）在“用户名称”文本框中输入 userA，单击“邮箱地址”文本框，此时触发“用户名称”文本框的失去焦点（blur）事件，执行 Ajax 请求，校验“用户名称”是否重复，运行效果如图 3-14 所示。

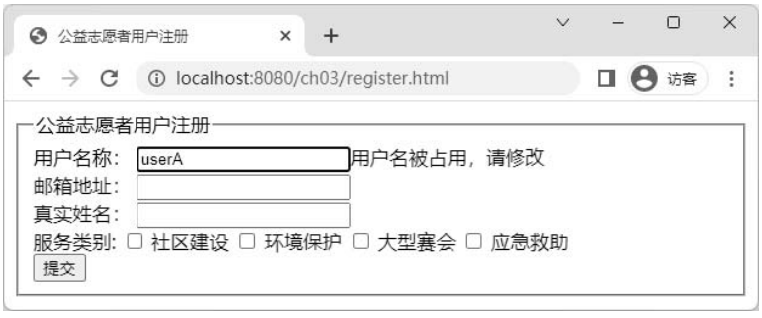

图 3-14　CheckNameServlet.java 的运行效果

程序说明：

- “<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>”表示在 HTML 文件中引入了 jQuery 库文件。因为是通过网络 URL 引入库文件的，所以在运行本实例时需要联网。
- 为了简化演示代码，假定用户名 userA 已经占用，其他用户名可用。
- “out.print("isUsed");”表示 Servlet 向客户端输出字符串，告知客户端该用户名已经被其他用户占用。这里一般情况下不使用 printIn 方法，因为该方法输出的字符信息结尾处带换行符，不便于 JavaScript 进行字符串比较。
- 为了简化演示代码，CheckNameServlet 响应输出设置为"text/html"格式，但在实际项目中，前后端之间的数据响应通常使用"application/json"格式。

3.6 小结

本章首先介绍了 Servlet 技术的发展历史，重点说明了 Servlet 的生命周期和运行过程。其次介绍了编写 Servlet 的方式以及如何将 Servlet 部署到 Web 服务器中。最后通过三个实例展示了 Serlvet 处理客户端请求的具体方法。

3.7 习题

1．填空题

（1）在 Servlet 出现之前，Web 服务器通过_____技术与客户端进行交互。

（2）部署 Servlet 的配置有两种方式：一种是通过_____部署；另一种是通过_____部署。

（3）在 IDEA 开发工具中创建 Web Application 项目时，会自动创建 webapp 和 WEB-INF 文件夹。其中_____是本项目 Web 站点根节点。

（4）使用注解方式部署 Servlet 时，若@WebServlet 只有一个属性 value="/Servlet1"，那么 value 和=符号也可以省略，即最简模式为_____。

（5）请求 Servlet 的常用方式有 3 种，分别为通过_____方式请求，通过页面的_____方式请求，以及通过 JavaScript 脚本的_____方式请求。

2．选择题

（1）开发编写 Servlet 类的方式是继承 javax.servlet.http 包中的（　　）。

A．Servlet 类　　B．HttpServlet 类

C．HttpGetServlet 类　　D．HttpPostServlet 类

（2）关于 Servlet 的说法不正确的是（　　）。

A．当多个来自客户端的请求访问时，Servlet 为每个请求分配一个进程

B．当 Servlet 容器关闭时会自动调用 destroy 方法，释放运行时占用的资源

C．Servlet 中的 service 方法可被多次请求调用，各调用过程运行在不同的线程中

D．Servlet 中的 init 方法仅执行一次

（3）关于@WebServlet 的注解说法正确的是（　　）。

A．Servlet 4.0 以上版本包含注解（Annotation）新特性

B．name 属性是必须显示指定的属性

C．urlPatterns 属性用于指定一组 Servlet 的 URL

D．value 属性或者 urlPatterns 属性是必需的，且二者可以共存

3．简答题

（1）简述 Servlet 的生命周期与运行过程。

（2）简述 web.xml 与注解部署 Servlet 的区别。

（3）简述开发编写以及测试运行 Servlet 的基本流程。

4．上机操作题

（1）建立并测试本章的所有实例。

（2）Servlet 应用练习——古诗学习。

习题背景：党的十八大以来，习近平总书记在多个场合谈到中国传统文化，表达了自己对传统文化、传统思想价值体系的认同与尊崇。2014 年 5 月 4 日他与北京大学学子座谈，也多次提到核心价值观和文化自信。习近平总书记在国内外不同场合的活动与讲话中，展现了中国政府与人民的精神志气，提振了中华民族的文化自信。

我们有博大精深的优秀传统文化。它能“增强做中国人的骨气和底气”，是我们最深厚的文化软实力，是我们文化发展的母体，积淀着中华民族最深沉的精神追求。诸如“自强不息”的奋斗精神，“精忠报国”的爱国情怀，“天下兴亡，匹夫有责”的担当意识，“舍生取义”的牺牲精神，“革故鼎新”的创新思想，“扶危济困”的公德意识，“国而忘家，公而忘私”的价值理念等，一直是中华民族奋发进取的精神动力。其中古诗是我们优秀传统文化中重要的一部分，学习优秀古诗是我们中国人的必修课之一。

习题要求：在 HTML 页面上设计一个表单，表单标题为“古诗学习欣赏”，表单内容为一个输入古诗名称的文本框，表单以 Post 方式提交到 Servlet；Servlet 中读取古诗名称，如果古诗名称是“劝学”，输出“劝学”古诗的信息，否则输出“抱歉暂未收录”。

命名规范：HTML 页面命名为 index31.html，Servlet 命名为 PoemServlet。

习题指导：①参考 3.5.1 节的内容，新建 HTML 文件和 Servlet 文件。②在 Servlet 中读取古诗名称，进行字符串比较时不能使用“= =”，应该使用 equals 方法。

参考资料：《劝学》唐-孟郊 击石乃有火，不击元无烟。人学始知道，不学非自然。万事须己运，他得非我贤。青春须早为，岂能长少年。《劝学》唐-颜真卿 三更灯火五更鸡，正是男儿读书时。黑发不知勤学早，白首方悔读书迟。

（3）Servlet 应用练习——北京冬奥会奖牌统计。

习题背景：2022 年北京-张家口冬季奥运会，第 24 届冬季奥林匹克运动会，2022 年 2 月 4 日至 2022 年 2 月 20 日在中华人民共和国北京市和张家口市联合举行。这是中国历史上第一次举办冬季奥运会，北京、张家口同为主办城市，也是中国继北京奥运会、南京青奥会后，中国第三次举办的奥运赛事。北京-张家口冬季奥运会设 7 个大项，102 个小项。北京将承办所有冰上项目，北京延庆和张家口将承办所有的雪上项目。

聚冰雪热爱展万象灵韵，迎八方宾客赏四季风华。一场精彩、非凡、卓越的冬奥会成为了中国送给世界的美好礼物。盛会闭幕，未来已来，“双奥之城”北京已用一份精彩的答卷点燃了舞动冰雪的激情，照亮了砥砺前行的征程。让我们再道一声“你好，双奥之城”，共赴美好前程。

习题要求：在 HTML 页面上设计一个表单，表单标题为“2022 年北京冬奥会-奖牌统计”，表单内容为 3 个文本框，分别输入中国队金、银、铜奖牌数量，表单以 Post 方式提交到 Servlet；Servlet 中计算奖牌总数，并输出 3 号标题“2022 年北京冬奥会上中国队获得？块奖牌！”，？号用计算结果代替，同时输出奖牌榜图片。

命名规范：HTML 页面命名为 index32.html，Servlet 命名为 OlympicServlet，奖牌榜图片命为 olympic.png。

习题指导：①参考 3.5.2 节的内容，新建 HTML 文件和 Servlet 文件，在 webapp 文件夹中新建 images 文件夹，然后把 olympic.png 复制到 images 文件夹中。②在 Servlet 中分别读取金、银、铜奖牌数量后，读取的值默认为字符串类型，需要转换为整数类型后再计算。③输出奖牌榜图片实际是输出<img src='images/olympic.png' alt='奖牌榜'>标记。④北京冬奥会奖牌榜如图 3-15 所示，请从网络中搜索北京冬奥会奖牌榜，并截图保存。

图 3-15　北京冬奥会奖牌榜

（4）Servlet 应用练习——志愿者登记。

习题背景：1993 年底，共青团中央决定实施中国青年志愿者行动。同年 12 月 19 日，2 万余名铁路青年率先打出了“青年志愿者”的旗帜，在京广铁路沿线开展了为旅客送温暖志愿服务。之后，40 余万名大中学生利用寒假在全国主要铁路沿线和车站开展志愿者新春热心行动，青年志愿者行动迅速在全国展开。青年志愿者行动不断发展，志愿服务的领域不断扩大，志愿者队伍日益壮大。据不完全统计，至 2000 年 6 月，全国累计已有 8000 多万人次的青年向社会提供了超过 40 亿小时的志愿服务。

为推动青年志愿服务事业的发展，团中央于 1994 年 12 月 5 日成立了中国青年志愿者协会，随后，各级青年志愿者协会也逐步建立起来。到 2000 年，已初步形成了由全国性协会、36 个省级协会和 2/3 以上的地（市）级协会及部分县级协会组成的志愿服务组织管理网络。

广大青年志愿者加入志愿者协会的第一步就是注册，下面通过练习题来实现简单的志愿者登记功能。

习题要求：在 HTML 页面上设计一个表单，表单标题为“中国青年志愿者注册”，表单内容为 3 个文本框，分别输入用户名称、邮箱地址、真实姓名，以及服务类别的复选项，包含社区建设、环境保护、大型赛会、应急救助等选项。表单以 Post 方式提交到 Servlet；Servlet 中读取并输出用户注册信息。

命名规范：HTML 页面命名为 index33.html，Servlet 命名为 RegServlet。

习题指导：①参考 3.5.3 节的内容新建 HTML 文件，参考 3.5.2 节的内容新建 Servlet。②在 Servlet 中读取服务类别时，使用 request. getParameterValues()方法，该方法返回类型为字符串数组，需要遍历数组输出。

扩展要求：使用 Ajax 技术实现用户名称重复校验功能，假定用户名 userA、userB 已占用，其他用户名可用。