第2章 Servlet 核心技术

本章学习 Servlet 核心编程技术，包括 HttpServlet 类、HttpServletRequest 请求对象、HttpServletResponse 响应对象、处理请求参数、请求转发、处理响应等。

2.1 知识点总结

（1）Servlet API 是 Java Web 开发的基础，它由 4 个包组成：javax.servlet、javax.servlet.http、javax.servlet.annotation 和 javax.servlet.descriptor。

（2）Servlet 接口是核心接口，每个 Servlet 必须直接或间接实现该接口，该接口定义了 init()、service()和 destroy()生命周期方法以及 getServletInfo()与 getServletConfig()。

（3）GenericServlet 抽象类实现了 Servlet 接口和 ServletConfig 接口。

（4）ServletConfig 在 Servlet 初始化时，容器将调用 init(ServletConfig)，并为其传递一个 ServletConfig 对象，该对象称为 Servlet 配置对象，使用该对象可以获得 Servlet 初始化参数、Servlet 名称、ServletContext 对象等。

（5）HttpServlet 类扩展了 GenericServlet 类，在 HttpServlet 中针对不同的 HTTP 请求方法定义了不同的处理方法，如处理 GET 请求的 doGet()，该方法有两个参数：一个是请求对象，一个是响应对象。

（6）HttpServletRequest 接口对象是请求对象，使用它可以检索客户请求信息，如使用 getParameter()可以获取请求参数，getMethod()可以获取请求的 HTTP 方法（如 GET 或 POST），getRequestURI()返回请求 URI 等。

（7）HttpServletResponse 接口对象是响应对象，通过它可向客户端发送响应消息，如 getWriter()返回 PrintWriter 对象，它可以向客户发送文本数据；setContentType()设置响应的内容类型；setHeader()设置响应头；sendRedirect()响应重定向等。

（8）在客户端发生下面的事件，浏览器就向 Web 服务器发送一个 HTTP 请求。① 用户在浏览器的地址栏中输入 URL 并按 Enter 键。② 用户点击了 HTML 页面中的超链接。③ 用户在 HTML 页面中填写一个表单并提交。

（9）要实现请求转发，调用请求对象的 getRequestDispatcher()得到 RequestDispatcher 对象，然后调用它的 forward()方法将请求转发其他资源（Servlet 或 JSP）。

（10）请求对象是一个作用域对象，通过它的 setAttribute()将一个对象作为属性存储到请求对象中，然后可以在请求作用域的其他资源中使用 getAttribute()检索出属性。

（11）每个 Web 应用程序在它的根目录中都必须有一个 WEB-INF 目录，其中 classes 目录存放类文件，lib 目录存放库文件，该目录下还应该有一个 web.xml 文件，称为部署描

述文件。

（12）在 Servlet 3.0 的 javax.servlet.annotation 包中定义了若干注解，使用@WebServlet 注解可以定义 Servlet。

下面一行是为 helloServlet 添加的注解。

```
@WebServlet(name="helloServlet",urlPatterns={"/hello-servlet"})
```

这里，name 元素指定 Servlet 名称，urlPatterns 元素指定 URL。该注解还可包含其他元素。注解在应用程序启动时被 Web 容器处理，容器根据具体的元素配置将相应的类部署为 Servlet。

（13）Web 容器在启动时会加载每个 Web 应用程序，并为每个 Web 应用程序创建一个唯一的 ServletContext 实例对象，该对象一般称为 Servlet 上下文对象。在 Servlet 中可以直接调用 getServletContext()得到 ServletContext 引用。ServletContext 对象也是一个作用域对象，它是 4 个作用域中最大的作用域对象，在其上也可以使用 setAttribute()存储属性，在其他资源中使用 getAttribute()返回属性值。

2.2 实 训 任 务

【实训目标】

学会通过 Servlet 处理表单数据；通过 Servlet 处理业务逻辑，实现请求转发等；掌握 ServletContext 获得资源的方法。

任务 1　学习获取表单请求参数

开发一个简单的考试系统，在 HTML 页面中建立一个表单，通过 POST 方法传递参数。题目类型包括单选题、多选题和填空题，要求程序给出考试成绩。

（1）在 Eclipse 中，新建一个名为 servlet-demo 的动态 Web 项目。在项目的 WebContent 目录中创建名为 question.html 的页面用于显示测试题目，代码如下，运行结果如图 2-1 所示。

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>简单测试</title></head>
<body>
<p>请回答下面的问题：</p>
<form action="simpletest.do" method="post">
<p> 1. Windows操作系统是哪个公司的产品？
    <input type="radio" name="q1" value="1"> Apple公司
    <input type="radio" name="q1" value="2"> IBM公司
    <input type="radio" name="q1" value="3"> Microsoft公司<br>
<p> 2. 编写Servlet程序应继承哪个类？
```

```
    <input type="text" name="q2" size="30"><br>
<p> 3. 下面的程序设计语言，哪些是面向对象的?
    <input type="checkbox" name="q3" value="1"> Java语言
    <input type="checkbox" name="q3" value="2"> C语言
    <input type="checkbox" name="q3" value="3"> C++语言<br>
<p> 4.下图是哪种编程语言的徽标?
    <input type="radio" name="q4" value="1">38
    <input type="radio" name="q4" value="2">40
    <input type="radio" name="q4" value="3">44<br>
    <img src="logo.png.jpg" width=100 /><br>
<p>交卷请单击:<input type="submit" value="交卷">
    重答请单击: <input type="reset" value="重答">
</form>
</body>
</html>
```

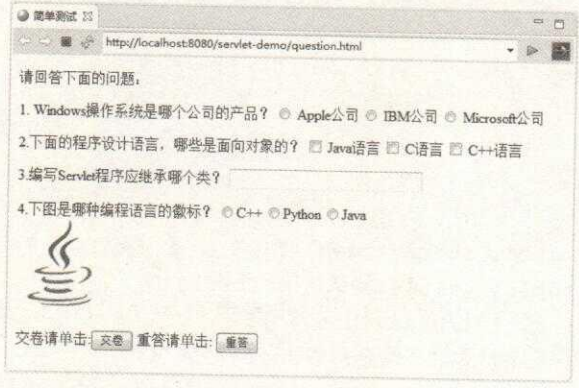

图 2-1　question.html 运行结果

（2）在 src 目录中创建 com.demo 包，然后在其中创建 ExamServlet.java 文件，文件代码如下：

```
package com.demo;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(name="examServlet",urlPatterns={"/exam-servlet"})
public class SimpleTestServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
```

```
                        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        String quest1 = request.getParameter("q1");
        String quest2 = request.getParameter("q2").trim();
        String[] quest3 = request.getParameterValues("q3");
        String quest4 = request.getParameter("q4").trim();
        int score = 0;
        if(quest1!=null && quest1.equals("3")){
          score = score+25;   //答对一题加25分
        }
        if(quest2!=null&& (quest2.equals("HttpServlet")||
          quest2.equals("javax.servlet.http.HttpServlet"))){
          score = score+25;
        }
        if(quest3!=null&&quest3.length==2&&quest3[0].equals("1")&&
          quest3[1].equals("3")){
          score = score+25;
        }
        if(quest2!=null&& quest2.equals("3")){
          score = score+25;
        }
        out.println("<html><head>");
        out.println("<title>测试结果</title>");
        out.println("</head><body>");
        out.println("你的成绩是："+score+"分");
        out.println("</body></html>");
    }
}
```

当在页面中选择了题目答案，单击“交卷”按钮时，将调用 ExamServlet，在该 Servlet 中首先检索请求参数，然后判断用户答案是否正确，最后给出分数。

（3）在 question.html 页面的编辑区右击，在弹出的快捷菜单中选择 Run As→Run on Server 即可执行该页面。在页面中选择题目答案，单击“交卷”按钮，将执行 ExamServlet，在浏览器中显示结果如图 2-2 所示。

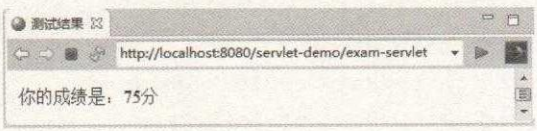

图 2-2　ExamServlet 运行结果

任务 2　学习请求转发与请求作用域

完成下面的综合应用，程序运行首先显示一个页面，输入学号和姓名后，单击“登录”

按钮，控制转发到 FirstServlet，在其中检索出学号和姓名信息，创建一个 Student 对象并将它存储到请求（request）作用域中，将控制转发到 SecondServlet，在其中从请求作用域检索出 Student 对象并显示学号和姓名，同时显示一个链接。

（1）在 servlet-demo 项目的 WebContent 目录中创建名为 input.jsp 的 JSP 页面，其中包括一个表单，表单中包含两个文本域，分别供用户输入学号和姓名，该页面也包含提交和重置按钮。

```
<%@ page contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<html>
<head>
<title>登录页面</title>
</head>
<body>
<form action="first-servlet" method="post">
    学号<input type="text" name="sno" size="15" /><br>
    姓名<input type="text" name="sname" size="15"/><br>
    <input type="submit" value="登录" />
    <input type="reset" value="取消" />
</form>
</body>
</html>
```

（2）在 servlet-demo 项目中新建 com.demo 包，在该包中定义一个名为 Student 的类，其中包括学号 sno 和姓名 name 两个 private 的成员变量，定义一个带两个参数的构造方法，为属性定义访问和修改 sno 和 name 的方法。

```
package com.demo;
public class Student {
    private String sno;
    private String name;
    public Student(String sno, String name) {
        this.sno = sno;
        this.name = name;
    }
    public String getSno() {
        return sno;
    }
    public void setSno(String sno) {
        this.sno = sno;
        }
    public String getName() {
        return name;
    }
    public void setName(String name) {
```

```
        this.name = name;
    }
}
```


（3）在 com.demo 包中创建名为 FirstServlet 的 Servlet，要求当用户在 input.jsp 中输入信息后单击“登录”按钮，请求 FirstServlet 对其处理。在 FirstServlet 中使用表单传递的参数（学号和姓名）创建一个 Student 对象并将其作为属性存储在请求对象中，然后通过请求对象的 getRequestDispatcher()获得 RequestDispatcher 对象，将请求转发到 SecondServlet。

```
package com.demo;
import java.io.IOException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
@WebServlet("/first-servlet")
public class FirstServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
              throws ServletException, IOException {
    String sno = request.getParameter("sno");
    String name = request.getParameter("sname");
    Student student = new Student(sno,name);
    request.setAttribute("student", student);
    RequestDispatcher rd =
         request.getRequestDispatcher("/second-servlet");
    rd.forward(request, response);
    }
}
```

（4）在 com.demo 包中创建名为 SecondServlet 的 Servlet，在它的 doPost()方法中从请求作用域中取出存储的 Student 对象，然后用输出流对象 out 输出该学生的学号和姓名。输出中还包含一个超链接，单击该链接可以返回到 input.jsp 页面。

```
package com.demo;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
@WebServlet("/second-servlet")
public class SecondServlet extends HttpServlet {
```

```
protected void doPost(HttpServletRequest request,
          HttpServletResponse response)
        throws ServletException, IOException {
    Student student= (Student)request.getAttribute("student");
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    out.println("学号: "+student.getSno()+"<br>");
    out.println("姓名: "+new String(
          student.getName().getBytes("iso-8859-1"),"UTF-8")+"<br>");
    out.println("<a href='input.jsp'>返回输入页面</a>");
    }
}
```

（5）访问 input.jsp 页面，输入学号和姓名，如图 2-3 所示。单击“登录”按钮，请求 FirstServlet，然后控制又转发到 SecondServlet，显示结果如图 2-4 所示。

图 2-3　input.jsp 页面显示结果

图 2-4　SecondServlet 显示结果

2.3　思考与练习答案

1．下面哪个方法不是 Servlet 生命周期方法？（　　）

A．public void destroy()

B．public void service()

C．public ServletConfig getServletConfig()

D．public void init()

【答】 C。Servlet 生命周期方法包括 init()、service()和 destroy()。

2．要使向服务器发送的数据不在浏览器的地址栏中显示，应该使用什么方法？（　　）

A．POST　　B．GET　　C．PUT　　D．HEAD

【答】 A。POST 方法发送的数据不在浏览器地址栏显示，而 GET 方法发送的数据将附加在请求 URL 后面并显示在浏览器地址栏。

3．考虑下面的 HTML 页面代码：

```
<a href="/HelloServlet">请求</a>
```

当用户在显示的超链接上单击时将调用 HelloServlet 的哪个方法？（　　）

A．doPost()　　B．doGet()　　C．doForm()　　D．doHref()

【答】 B。单击超链接向服务器发送 GET 请求。