



프로젝트로 배우는 자바 웹 프로그래밍

Servlet, JSP, JDBC

황희정 지음

Chapter 06. JSP 내장객체

목차

1. JSP 내장객체의 개요
2. request
3. response
4. out
5. session
6. 그 밖의 내장객체
7. JSP 내장객체와 속성 관리
8. [기본실습] JSP 내장객체 : 세션을 이용한 장바구니 기능
9. [응용실습] JSP 내장객체 : 트위터 구현

01. JSP 내장객체 개요

■ JSP 내장객체란?

- JSP 내장객체란 'JSP 내에서 선언하지 않고 사용할 수 있는 객체'라는 의미에서 붙여진 이름.
- 구조적으로는 JSP가 서블릿 형태로 자동 변환된 코드 내에 포함되어 있는 소속변수, 메소드 매개변수 등의 각종 참조 변수(객체)를 말함.

[표 6-1] JSP 내장객체

참조 변수 이름(내장객체)	자바 클래스	주요 역할
request	javax.servlet.http.HttpServletRequest	HTML 폼 요소의 선택 값 등 사용자 입력 정보를 읽으려고 사용한다.
response	javax.servlet.http.HttpServletResponse	사용자 요청에 대한 응답을 처리하려고 사용한다.
pageContext	javax.servlet.jsp.PageContext	현재 JSP 실행에 대한 context 정보를 참조하려고 사용한다.
session	javax.servlet.http.HttpSession	클라이언트의 세션 정보를 처리하려고 사용한다.
application	javax.servlet.ServletContext	웹 서버의 애플리케이션 처리와 관련된 정보를 참조하려고 사용한다.
out	javax.servlet.jsp.JspWriter	사용자에게 전달하기 위한 output 스트림을 처리하려고 사용한다.
config	javax.servlet.ServletConfig	현재 JSP의 초기화 환경을 처리하려고 사용한다.
page	java.lang.Object	현재 JSP의 클래스 정보를 보려고 사용한다.
exception	java.lang.Throwable	예외 처리를 하려고 사용한다.

01. JSP 내장객체 개요

1. JSP 내장객체의 구조적 특징

- 서블릿으로 변경된 JSP 코드는 모두 `_jspService()` 메소드에 위치함.
- 메소드 매개변수인 `request`, `response` 를 비롯한 `pageContext`, `session`, `application`, `page`, `config`, `out` 등 메소드 내에서 참조할 수 있는 참조변수들이 내장객체가 됨.

```
public void _jspService(HttpServletRequest request, HttpServletResponse response) throws java.io.IOException, ServletException {  
    JspFactory _jspxFactory = null;  
    javax.servlet.jsp.PageContext pageContext = null;  
    HttpSession session = null;  
    ServletContext application = null;  
    ServletConfig config = null;  
    JspWriter out = null;  
    Object page = this;  
    JspWriter _jspx_out = null;  
    try {  
        pageContext = _jspxFactory.getPageContext(this, request, response, null, true, 8192, true);  
        application = pageContext.getServletContext();  
        config = pageContext.getServletConfig();  
        session = pageContext.getSession();  
        out = pageContext.getOut();  
        ...  
        사용자 작성 JSP 코드가 오는 위치....  
    }  
}
```

01. JSP 내장객체 개요

2. 내장객체를 이용한 속성 관리 기법

- 내장객체가 단순히 특정한 기능을 제공하는 컨테이너 관리 객체라는 점 외에도 한 가지 특징이 있다.
바로 page, request, session, application 내장객체를 이용한 속성 관리 기법이다. 이들 내장객체는 각자 지정된 생명주기가 있으며 setAttribute(), getAttribute()라는 메서드를 통해 해당 생명주기 동안 자바 객체를 유지하는 기능을 제공한다.
- 속성 관리 기법은 '7절. JSP 내장객체와 속성 관리'에서 자세히 살펴볼 것이다.

02. request

■ request 내장객체

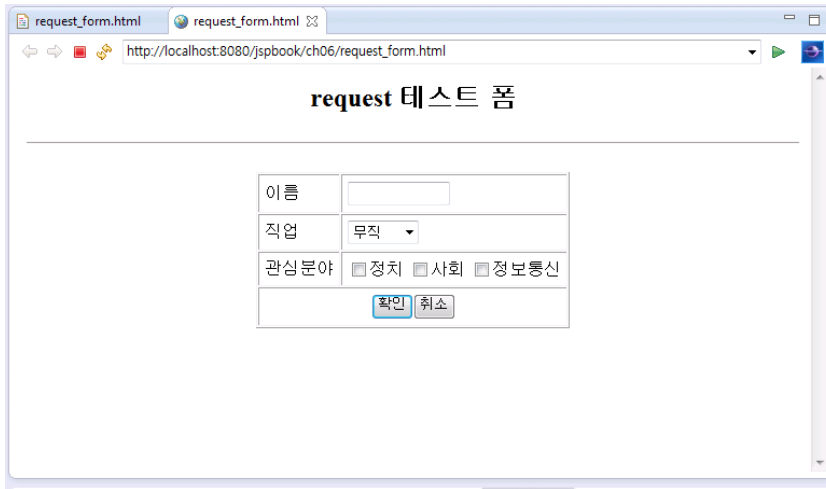
- request는 사용자 요청과 관련된 기능을 제공하는 내장객체로 javax.servlet.http.HttpServletRequest 클래스에 대한 참조 변수임.
- 주로 클라이언트에서 서버로 전달되는 정보를 처리하기 위해 사용한다.
- 대표적으로 HTML 폼을 통해 입력된 값을 JSP에서 가져올 때 사용함.

[표 6-2] request 주요 메소드

메서드	설명
Enumeration<String> getParameterNames()	현재 요청에 포함된 매개변수의 이름을 열거(Enumeration) 형태로 넘겨준다.
String getParameter(String name)	문자열 name과 이름이 같은 매개변수의 값을 가져온다.
String[] getParameterValues (String name)	문자열 name과 이름이 같은 매개변수의 값을 배열 형태로 가져온다. checkbox, multiple list 등에 주로 사용한다.
Cookie[] getCookies()	모든 쿠키 값을 javax.servlet.http.Cookie의 배열 형태로 가져온다.
String getMethod()	현재 요청이 GET이나 POST 형태로 가져온다.
HttpSession getSession()	현재 세션 객체를 가져온다.
String getRemoteAddr()	클라이언트의 IP 주소를 알려준다.
String getProtocol()	현재 서버의 프로토콜을 문자열 형태로 알려준다.
void setCharacterEncoding(String env)	현재 JSP로 전달되는 내용을 지정한 캐릭터셋으로 변환해준다. HTML 폼에서 한글 입력을 정상적으로 처리해주려면 반드시 필요하다.

02. request

- [실습] request 테스트 폼(request_form.html) - 교재 p.200 ~ 201 참고



[그림 6-1] 입력 화면

- [실습] request 테스트 결과(request_result.jsp) - 교재 p.202 ~ 203 참고



이름	황희정
직업	무직
관심분야	정치 사회

request 테스트 결과 - 2

1. 클라이언트 IP 주소 : 0:0:0:0:0:0:1
2. 요청 메서드 : POST
3. test 에 설정된 쿠키값 : OK.
4. 프로토콜 : HTTP/1.1

[그림 6-2] 실행 결과

[예제 6-1] request_form.html

```
request_form.html
1 <HTML>
2 <HEAD>
3 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4 <TITLE>ch06 : request_form.html</TITLE>
5 <script type="text/javascript">
6     document.cookie = "test=OK.";
7 </script>
8
9 </HEAD>
10 <BODY>
11 <div align=center>
12 <H2>request 테스트 폼 </H2>
13 <HR>
14 <form name=form1 method=post action=request_result.jsp>
15 <table border=1 cellspacing="1" cellpadding="5">
16 <tr>
17 <td>이름</td>
18 <td><input type=text size=10 name=username></td>
19 <tr>
20 <td>직업</td>
21 <td>
22 <select name=job >
23     <option selected>무직</option>
24     <option>회사원</option>
25     <option>전문직</option>
26     <option>학생</option>
27 </select>
28 </td>
29 <tr>
30 <td>관심분야</td>
31 <td>
```


[예제 6-1] request_form.html (cont'd)

```
29 <tr>
30 <td>관심분야</td>
31 <td>
32 <input type=checkbox name=favorite value="정치">정치
33 <input type=checkbox name=favorite value="사회">사회
34 <input type=checkbox name=favorite value="정보통신">정보통신
35 </td>
36 <tr><td colspan=2 align=center><input type=submit value="확인">
37 <input type=reset value="취소"></td></tr>
38 </table>
39 </form>
40 </div>
41
42 </BODY>
43 </HTML>
```

[예제 6-2] request_result.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <% request.setCharacterEncoding("UTF-8"); %>
3 <HTML>
4 <HEAD>
5 <TITLE>ch06 : request_result.jsp</TITLE></HEAD>
6 <BODY>
7 <div align="center">
8 <H2>request 테스트 결과 - 1</H2>
9
10 <HR>
11 <table border=1 cellspacing="1" cellpadding="5">
12 <tr>
13 <td>이름</td>
14 <td><%=request.getParameter("username")%> </td>
15 <tr>
16 <td>직업</td>
17 <td><%=request.getParameter("job")%></td>
18 <tr>
19 <td>관심분야</td>
20 <td>
21 <%
22     // getParameterValues 메서드를 이용해 "favorite" 로 설정된 form 의 체크박스 값들을 모두 읽어옴.
23     String favorites[] = request.getParameterValues("favorite");
24
25     // 배열의 크기만큼 루프를 돌면서 값을 출력함.
26     for(String favorite: favorites) {
27         out.println(favorite+"<BR>");
28     }
29 <%></td>
30 </table>
31 <HR>
```

[예제 6-2] request_result.jsp (cont'd)

```
31 <HR>
32 <H2>request  테스트 결과 - 2</H2>
33 <table border=0><tr><td>
34 1. 클라이언트 IP 주소 : <%= request.getRemoteAddr() %> <BR>
35 2. 요청 메서드 : <%= request.getMethod() %> <BR>
36 <%
37     Cookie cookie[] = request.getCookies();
38 %>
39 3. <%= cookie[0].getName() %> 에 설정된 쿠키값 : <%=cookie[0].getValue() %><BR>
40 4. 프로토콜 : <%= request.getProtocol() %>
41 </td></tr>
42 </table>
43 </div>
44
45 </BODY>
46 </HTML>
```

request.getParameterNames()

- `interface javax.servlet.ServletException`
- `class javax.servelet.http.HttpServletRequest` implements `javax.servlet.ServletException`
- `java.util.Enumeration<java.lang.String> getParameterNames()`
- `<%@ page import = "java.util.Enumeration" %>`
- `interface java.util.Enumeration<E>`
 - `boolean hasMoreElements()`
 - `E nextElement()`

[예제 6-2] request_result.jsp (추가)

```
<hr>
<H2>request 테스트 결과 - 3</H2>
<%
    out.println("전달된 파라미터 이름들: " + "<br>");

    Enumeration<String> e = request.getParameterNames();
    while (e.hasMoreElements()) {
        String s = e.nextElement();
        out.println(s + "<br>");
    }
%>
```

request 테스트 결과 - 1

이름	김동덕
직업	학생
관심분야	정보통신

request 테스트 결과 - 2

1. 클라이언트 IP 주소 : 0:0:0:0:0:0:1
2. 요청 메서드 : POST
3. test 에 설정된 쿠키값 : OK.
4. 프로토콜 : HTTP/1.1

request 테스트 결과 - 3

전달된 파라미터 이름들:

- username
- job
- favorite

GET 방식 vs. POST 방식

교재 p. 121



GET 방식 vs. POST 방식 (교재 p. 121)

■ GET 방식

- 서버에 있는 정보를 클라이언트로 가져오기 위한 방법이다.
 - 예를 들어 HTML, 이미지 등을 웹 브라우저에서 보기 위한 요청.
- 서버에는 최대 240Byte까지 데이터를 전달할 수 있다.
- QUERY_STRING 환경변수를 통해서 서버로 전달되는데, 다음 형식을 따른다.
 - `http://www.xxx.co.kr/servlet/login?id=hj&name=hong`
- ‘?’ 이후의 값들은 서버에서 QUERY_STRING을 통해 전달된다. ‘속성=값’ 형태로 사용해야 하며 ‘&’는 여러 속성 값을 전달할 때 연결해주는 문자열이다.
- URL이 노출되기 때문에 보안에 문제가 생길 수 있다.

■ POST 방식

- 서버로 정보를 올리기 위해 설계된 방법이다.
 - 예를 들어 HTML 폼에 입력한 내용을 서버에 전달하기 위한 요청.
- 서버에 전달 할 수 있는 데이터 크기에는 제한이 없다.
- URL에는 매개변수가 표시되지 않는다.

■ GET 방식과 POST 방식 모두 클라이언트가 서버로 정보는 보내는 용도로 사용 가능

03. response

■ response 내장객체

- response는 request와 반대되는 개념으로, 사용자 응답과 관련된 기능을 제공
- 사용자 요청(request)을 처리하고 응답을 다른 페이지로 전달하는 등의 기능을 제공한다.
- javax.servlet.http.HttpServletResponse 객체에 대한 참조 변수로, request에 만큼 많이 사용되지는 않으나 setContentType, sendRedirect와 같은 메소드는 잘 알아두어야 한다.

[표 6-3] response 주요 메소드

메서드	설명
setContentType(type)	문자열 형태의 type에 지정된 MIME Type으로 contentType을 설정한다.
setHeader(name,value)	문자열 name의 이름으로 문자열 value의 값을 헤더로 세팅한다.
setDateHeader(name, date)	문자열 name의 이름으로 date에 설정된 밀리세컨드 시간 값을 헤더에 설정한다.
sendError(status,msg)	오류 코드를 세팅하고 메시지를 보낸다.
sendRedirect(url)	클라이언트 요청을 다른 페이지로 보낸다.

03. response

■ [실습] forward 액션과의 차이 알아보기(page_control.jsp) – 교재 p.207 참고



[그림 6-3] 테스트 양식

■ [실습] forward 액션방식의 호출(forward_action2.jsp), sendRedirect 메서드를 이용한 호출(response_sendRedirect.jsp), 호출된 화면 (page_control_end.jsp) – 교재 p.208 참고

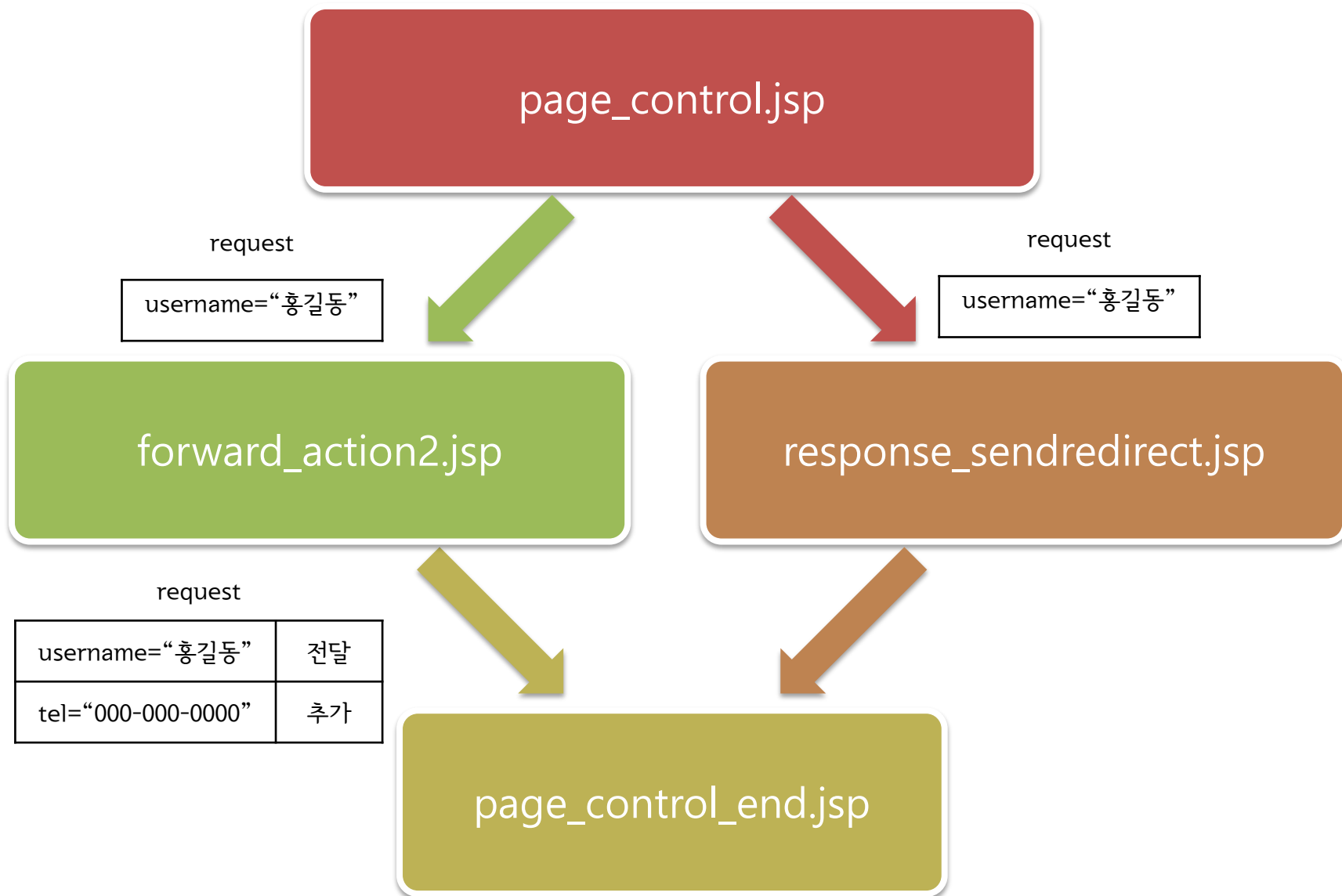


[그림 6-4] forward 액션 실행 결과



[그림 6-5] response.sendRedirect() 실행 결과

Forward action vs. sendRedirect method call



[예제 6-3] page_control.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2
3 <HTML>
4 <HEAD><TITLE>ch06 : page_control.jsp</TITLE></HEAD>
5 <BODY>
6 <H2>forward, sendRedirect 테스트</H2>
7 <HR>
8 <form method=post action=forward_action2.jsp>
9     forward action : <input type=text name=username>
10    <input type=submit value="확인">
11 </form>
12
13 <form method=post action=response_sendRedirect.jsp>
14     response.sendRedirect : <input type=text name=username>
15    <input type=submit value="확인">
16 </form>
17 </BODY>
18 </HTML>
```

[예제 6-4, 6-5] forward action vs. sendRedirect method

■ forward_action2.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <% request.setCharacterEncoding("UTF-8"); %>
3 <jsp:forward page="page_control_end.jsp">
4     <jsp:param name="tel" value="000-000-0000" />
5 </jsp:forward>
```

■ response_sendRedirect.jsp

```
1 <% response.sendRedirect("page_control_end.jsp"); %>
```

[예제 6-6] page_control_end.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2
3 <html>
4 <head><TITLE>ch06 : page_control_end.jsp</TITLE></HEAD>
5 <BODY>
6 <div align="center">
7 <H2>forward action 및 sendRedirect() 결과</H2>
8 <HR>
9   지금 보이는 화면은 page_control_end.jsp 에서 출력한 결과 입니다.
10 <HR>
11 이름 : <%= request.getParameter("username") %> <BR>
12 전화번호 : <%= request.getParameter("tel") %>
13 </div>
14 </BODY>
15 </HTML>
```



프로젝트로 배우는 자바 웹 프로그래밍

Servlet, JSP, JDBC

황희정 지음