

JSP 웹 프로그래밍

5 내장 객체

Contents

1

내장 객체의 개요

2

request 내장 객체의 기능과 사용법

3

response 내장 태그의 기능과 사용법

4

out 내장 객체의 기능과 사용법

5

[웹 쇼핑몰] 상품 상세 정보 표시하기

학습목표

- 내장 객체의 개념과 특징을 이해합니다.
- 내장 객체 구성 요소의 사용법을 익힙니다.
- 내장 객체를 이용하여 웹 쇼핑몰의 상품 상세 정보를 출력합니다.

1. 내장 객체의 개요

❖ 내장 객체(implicit object)

- JSP 페이지에서 사용할 수 있도록 JSP 컨테이너에 미리 정의된 객체
- JSP 페이지가 서블릿 프로그램으로 번역될 때 JSP 컨테이너가 자동으로 내장 객체를 멤버 변수, 메소드의 매개변수 등에 대한 각종 참조 변수(객체)로써 포함시킴.
- JSP 페이지에 별도의 import 문 없이 자유롭게 사용 가능
- 스크립틀릿 태그나 표현문 태그에 선언을 하거나 객체를 생성하지 않고도 직접 호출하여 사용 가능

1. 내장 객체의 개요

❖ 내장 객체의 종류

내장 객체	반환 유형	설명
request	javax.servlet.http.HttpServletRequest	웹 브라우저의 HTTP 요청 정보를 저장합니다.
response	javax.servlet.http.HttpServletResponse	웹 브라우저의 HTTP 요청에 대한 응답 정보를 저장합니다.
out	javax.servlet.jsp.jspWriter	JSP 페이지에 출력할 내용을 담고 있는 출력 스트림입니다.
session	javax.servlet.http.HttpSession	웹 브라우저의 정보를 유지하기 위한 세션 정보를 저장합니다 (13장 참고).
application	javax.servlet.ServletContext	웹 애플리케이션의 컨텍스트 정보를 저장합니다.
pageContext	javax.servlet.jsp.PageContext	JSP 페이지의 정보를 저장합니다.
page	java.lang.Object	JSP 페이지를 구현한 자바 클래스로 JSP 페이지 자체를 나타냅니다.
config	javax.servlet.ServletConfig	JSP 페이지의 설정 정보를 저장합니다.
exception	java.lang.Throwable	JSP 페이지의 예외 발생을 처리합니다(11장 참고).

1. 내장 객체의 개요

...(생략)...

sprinting_jsp.java

```
public final class scripting_jsp extends org.apache.jasper.runtime.HttpJspBase  
    implements org.apache.jasper.runtime.JspSourceDependent,  
        org.apache.jasper.runtime.JspSourceImports {
```

매개변수로 선언된
request 내장 객체

...(생략)...

```
public void _jspService(final javax.servlet.http.HttpServletRequest request,  
final javax.servlet.http.HttpServletResponse response) throws  
java.io.IOException, javax.servlet.ServletException {
```

매개변수로 선언된
response 내장 객체

...(생략)...

```
    final javax.servlet.jsp.PageContext pageContext;
```

```
    javax.servlet.http.HttpSession session = null;
```

```
    final javax.servlet.ServletContext application;
```

```
    final javax.servlet.ServletConfig config;
```

```
    javax.servlet.jsp.JspWriter out = null;
```

```
    final java.lang.Object page = this;
```

미리 선언된
기타 내장 객체

```
    javax.servlet.jsp.JspWriter _jspx_out = null;
```

```
    javax.servlet.jsp.PageContext _jspx_page_context = null;
```

...(생략)...

```
}
```

```
}
```

1. 내장 객체의 개요

❖ 속성 처리 메소드의 종류(1)

- 모든 내장 객체는 JSP 컨테이너가 관리하는 객체.
- 내장 객체들 중에서 request, session, application, pageContext 객체를 이용하여 속성을 관리함.
- 속성은 각각의 내장 객체가 존재하는 동안 JSP 페이지 사이에서 정보를 주고 받거나 공유하는데 사용됨.

1. 내장 객체의 개요

❖ 속성 처리 메소드의 종류(2)

메소드	반환 유형	설명
setAttribute(String name, Object value)	void	해당 내장 객체의 속성 이름이 name인 속성 값을 value로 저장합니다.
getAttribute(String name)	Object	해당 내장 객체의 속성 이름이 name인 속성 값을 가져옵니다.
removeAttribute(String name)	void	해당 내장 객체의 속성 이름이 name인 속성을 삭제합니다.
getAttributeNames()	java.util.Enumeration	해당 내장 객체의 모든 속성 이름을 가져옵니다(단, pageContext 내장 객체는 이 메소드를 제공하지 않습니다).

2. request 내장 객체의 기능과 사용법

❖ request 내장 객체

- JSP 페이지에서 가장 많이 사용되는 기본 내장 객체
- 웹 브라우저에서 서버의 JSP 페이지로 전달하는 정보를 저장
 - 폼 페이지로부터 입력된 데이터를 전달하는 요청 파라미터 값을 JSP 페이지로 가져옴
- JSP 컨테이너는 웹 브라우저에서 서버로 전달되는 정보를 처리하기 위해 `javax.servlet.http.HttpServletRequest` 객체 타입의 request 내장 객체를 사용하여 사용자의 요구 사항을 얻어냄

2. request 내장 객체의 기능과 사용법

❖ 요청 파라미터 관련 메소드(1)

- 요청 파라미터는 사용자가 폼 페이지에 데이터를 입력한 후 서버에 전송할 때 전달되는 폼 페이지의 입력된 정보 형태를 말함
- 요청 파라미터는 **<name=value>** 형식으로 웹 브라우저에서 서버의 JSP 페이지로 전송
- 요청 파라미터는 폼(form) 페이지에서 `<input type="text" ...>` 처럼 입력 양식이 텍스트 유형인 경우 값을 입력하지 않으면 서버로 빈 문자열이 전송됨.
- 체크 박스와 레디오 버튼 유형인 경우 선택하지 않고 전송하면 요청 파라미터 자체가 전달되지 않음.

2. request 내장 객체의 기능과 사용법

❖ 요청 파라미터 관련 메소드(2)

■ 요청 파라미터 관련 메소드의 종류

요청 파라미터 관련 메소드	반환 유형	설명
<code>getParameter(String name)</code>	String	요청 파라미터 이름이 name인 값을 전달받습니다. 요청 파라미터 값이 없으면 null을 반환합니다.
<code>getParameterValues(String name)</code>	String[]	모든 요청 파라미터 이름이 name인 값을 배열 형태로 전달받습니다. 요청 파라미터 값이 없으면 null을 반환합니다.
<code>getParameterNames()</code>	java.util.Enumeration	모든 요청 파라미터의 이름과 값을 Enumeration 객체 타입으로 전달받습니다.
<code>getParameterMap()</code>	java.util.Map	모든 요청 파라미터의 이름과 값을 Map 객체 타입으로 전달받습니다.[Map 객체 타입은 (요청 파라미터 이름, 값) 형식으로 구성됩니다].

2. request 내장 객체의 기능과 사용법

[request 내장 객체 사용 예: 요청 파라미터 값 출력하기]

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
  <form action="process.jsp" method="post">
    <p>
      이 름 : <input type="text" name="name">
      <input type="submit" value="전송">
    </p>
  </form>
</body>
</html>
```

request.jsp

이 름 : 홍길순 전송

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
  <%
    request.setCharacterEncoding("utf-8");
    String name = request.getParameter("name");
  %>
  <p> 이 름 : <%=name%>
</body>
</html>
```

process.jsp

이 름:홍길순

폼에서 한글 입력을 정상적으로 처리하려면 반드시 필요함. 3절의 설명 참고

2. request 내장 객체의 기능과 사용법

예제 5-1 request 내장 객체로 폼 페이지로부터 아이디와 비밀번호를 전송받아 출력하기

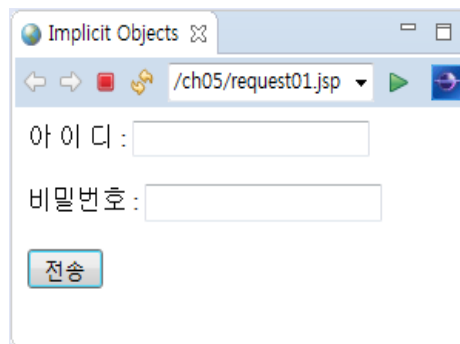
JSPBook/WebContent/ch05/request01.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <form action="request01_process.jsp" method="post">
08         <p>아 이 디 : <input type="text" name="id">
09         <p>비밀번호 : <input type="text" name="passwd">
10         <p><input type="submit" value="전송" />
11     </form>
12 </body>
13 </html>
```

2. request 내장 객체의 기능과 사용법

JSPBook/WebContent/ch05/request01_process.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <%
08         request.setCharacterEncoding("utf-8");
09         String userid = request.getParameter("id");
10         String password = request.getParameter("passwd");
11     %>
12     <p>아이디 : <%=userid%>
13     <p>비밀번호 : <%=password%>
14 </body>
15 </html>
```

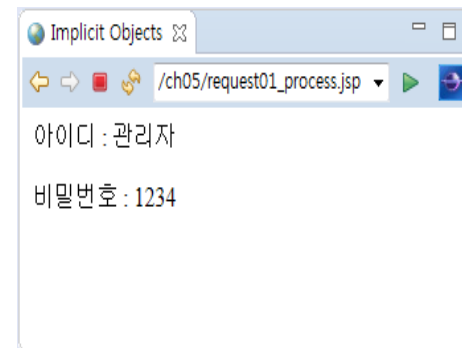


Implicit Objects

/ch05/request01.jsp

아이디 :

비밀번호 :



Implicit Objects

/ch05/request01_process.jsp

아이디 : 관리자

비밀번호 : 1234

2. request 내장 객체의 기능과 사용법

❖ 요청 HTTP 헤더 관련 메소드

- 웹 브라우저는 HTTP 헤더에 부가적인 정보를 담아 서버로 전송
- request 내장 객체는 헤더 정보나 쿠키 관련 정보를 얻을 수 있는 메서드를 제공함.
- 요청 HTTP 헤더 관련 메소드의 종류

요청 HTTP 헤더 관련 메소드	반환 유형	설명
getHeader(String name)	String	설정된 name의 헤더 값을 가져옵니다.
getHeaders(String name)	Enumeration	설정된 name의 헤더 목록 값을 가져옵니다.
getHeaderNames()	Enumeration	모든 헤더 이름을 가져옵니다.
getIntHeader(String name)	int	설정된 name의 헤더 값을 정수로 가져옵니다.
getDateHeader(String name)	long	설정된 name의 헤더 값을 시간 값으로 가져옵니다.
getCookies()	javax.servlet.http.Cookie	모든 쿠키 값을 가져옵니다.

2. request 내장 객체의 기능과 사용법

[request 내장 객체 사용 예: 요청 HTTP 헤더 정보 값 출력하기]

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <%
        String hostValue = request.getHeader("host");
        String alValue = request.getHeader("accept-language");

        out.print("호스트명 : " + hostValue + "<br>");
        out.print("설정된 언어 : " + alValue + "<br>");
    %>
</body>
</html>
```

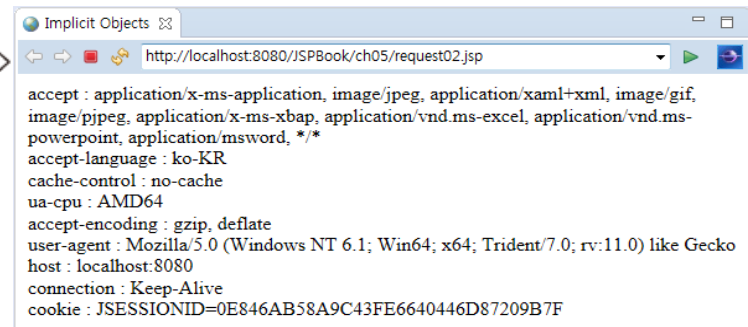
호스트명 : localhost:8080 설정된 언어 : ko-KR

2. request 내장 객체의 기능과 사용법

예제 5-2 request 내장 객체로 모든 HTTP 헤더 정보 값 출력하기

JSPBook/WebContent/ch05/request02.jsp

```
01 <%page import="java.util.Enumeration"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <%
08         Enumeration en = request.getHeaderNames();
09         while (en.hasMoreElements()) {
10             String headerName = (String) en.nextElement();
11             String headerValue = request.getHeader(headerName);
12         >%
13         <%=headerName%> : <%=headerValue%>\n<br>
14         <%
15             }
16         >%
17 </body>
18 </html>
```



2. request 내장 객체의 기능과 사용법

❖ 웹 브라우저의 요청 및 서버 관련 정보를 얻을 수 있는 메소드

웹 브라우저/서버 관련 메소드	반환 유형	설명
getRemoteAddr()	String	웹 브라우저의 IP 주소를 가져옵니다.
getContentTypeLength()	long	웹 브라우저의 요청 파라미터 길이를 가져옵니다.
getCharacterEncoding()	String	웹 브라우저의 문자 인코딩을 가져옵니다.
getContentType()	String	웹 브라우저의 콘텐츠 유형을 가져옵니다.
getProtocol()	String	웹 브라우저의 요청 프로토콜을 가져옵니다.
getMethod()	String	웹 브라우저의 HTTP 요청 메소드(GET, POST)를 가져옵니다.
getRequestURI()	String	웹 브라우저가 요청한 URI 경로를 가져옵니다.
getContextPath()	String	현재 JSP 페이지의 웹 애플리케이션 컨텍스트 경로를 가져옵니다.
getServerName()	String	서버 이름을 가져옵니다.
getServerPort()	int	실행 중인 서버 포트 번호를 가져옵니다.
getQueryString()	String	웹 브라우저의 전체 요청 파라미터 문자열(물음표(?) 다음 URL에 할당된 문자열)을 가져옵니다.

2. request 내장 객체의 기능과 사용법

[request 내장 객체 사용 예: 웹 브라우저/서버 정보 출력하기]

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <form action="process.jsp" method="post">
        <p>
            이름 : <input type="text" name="name">
            <input type="submit" value="전송">
        </form>
    </body>
</html>
```

request.jsp

이름 : 관리자

전송

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <%
        request.setCharacterEncoding("utf-8");
        String name = request.getParameter("name");
    %>
    <p>
        이름 : <%=name%><br>
        요청 정보 길이 : <%=request.getContentLength()%><br>
        클라이언트 전송 방식 : <%=request.getMethod()%><br>
        요청 URI : <%=request.getRequestURI()%><br>
        서버 이름 : <%=request.getServerName()%><br>
        서버 포트 : <%=request.getServerPort()%><br>
    </body>
</html>
```

process.jsp

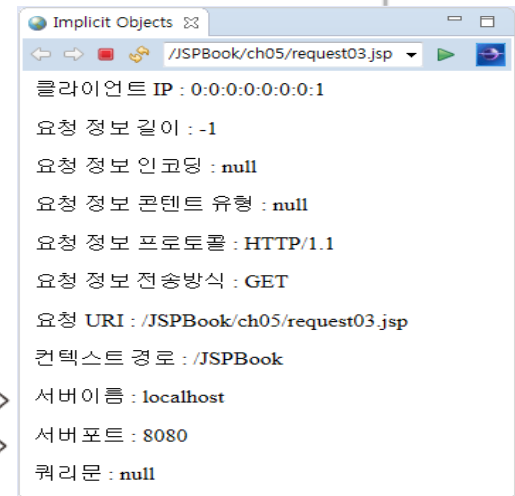
이름 : 관리자
요청 정보 길이 : 32
클라이언트 전송 방식 : POST
요청 URI : /JSPBook/process.jsp
서버 이름 : localhost
서버 포트 : 8080

2. request 내장 객체의 기능과 사용법

예제 5-3 request 내장 객체로 모든 웹 브라우저 및 서버 정보 값 출력하기

JSPBook/WebContent/ch05/request03.jsp

```
01 <%@ page contentType="text/html; charset=utf-8" %>
02 <html>
03 <head>
04     <title> Implicit Objects </title>
05 </head>
06 <body>
07     <p>클라이언트 IP : <%=request.getRemoteAddr() %></p>
08     <p>요청 정보 길이 : <%=request.getContentLength() %></p>
09     <p>요청 정보 인코딩 : <%=request.getCharacterEncoding() %></p>
10     <p>요청 정보 콘텐츠 유형 : <%=request.getContentType() %></p>
11     <p>요청 정보 프로토콜 : <%=request.getProtocol() %></p>
12     <p>요청 정보 전송 방식 : <%=request.getMethod() %></p>
13     <p>요청 URI : <%=request.getRequestURI() %></p>
14     <p>컨텍스트 경로 : <%=request.getContextPath() %></p>
15     <p>서버 이름 : <%=request.getServerName() %></p>
16     <p>서버 포트 : <%=request.getServerPort() %></p>
17     <p>쿼리문 : <%=request.getQueryString() %></p>
18 </body>
19 </html>
```



3. response 내장 객체의 기능과 사용법

❖ response 내장 객체

- 사용자의 요청을 처리한 결과를 서버에서 웹 브라우저로 전달하는 정보를 저장함.
- 서버는 응답 헤더와 요청 처리 결과 데이터를 웹 브라우저로 보냄.
- JSP 컨테이너는 서버에서 웹 브라우저로 응답하는 정보를 처리하기 위해 `javax.servlet.http.HttpServletResponse` 객체 타입의 response 내장 객체를 사용하여 사용자의 요청에 응답

3. response 내장 객체의 기능과 사용법

❖ 페이지 이동 관련 메소드

- 페이지 이동 = 리다이렉션(redirection)
 - 사용자가 새로운 페이지를 요청할 때와 같이 페이지를 강제로 이동하는 것
- 서버는 웹 브라우저에 다른 페이지로 강제 이동하도록 **response 내장 객체의 리다이렉션 메소드**를 제공
- 페이지 이동 시에는 문자 인코딩을 알맞게 설정해야 함
- 페이지 이동 관련 메소드의 종류

페이지 이동 관련 메소드	반환 유형	설명
<code>sendRedirect(String url)</code>	void	설정된 URL 페이지로 강제 이동합니다.

3. response 내장 객체의 기능과 사용법

[response 내장 객체 사용 예: 페이지 이동하기]

```
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
  <%
    response.sendRedirect("http://www.google.com");
  %>
</body>
</html>
```

response.jsp

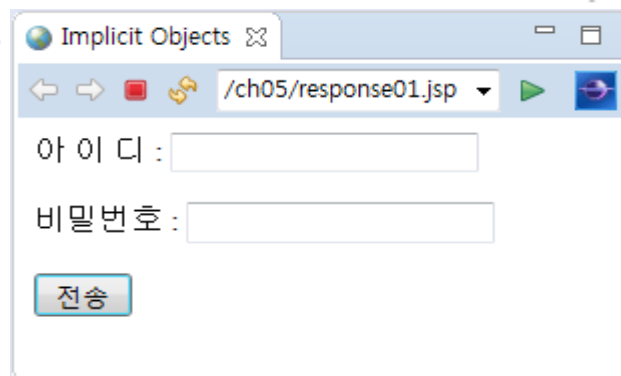


3. response 내장 객체의 기능과 사용법

예제 5-4 response 내장 객체로 페이지 이동하기

JSPBook/WebContent/ch05/response01.jsp

```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <form action="response01_process.jsp" method="post">
08         <p>아 이 디 : <input type="text" name="id">
09         <p>비밀번호 : <input type="text" name="pass">
10         <p><input type="submit" value="전송">
11     </form>
12 </body>
13 </html>
```



Implicit Objects

/ch05/response01.jsp

아 이 디 :

비밀번호 :

3. response 내장 객체의 기능과 사용법

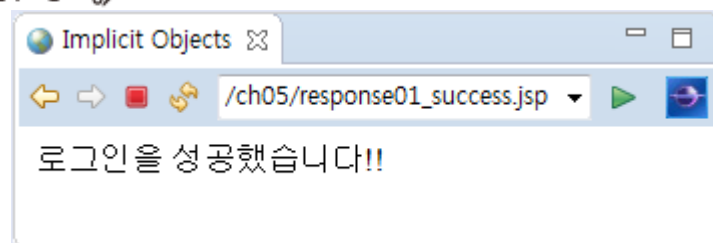
JSPBook/WebContent/ch05/response01_process.jsp

```
01  <% page contentType="text/html; charset=utf-8"%>
02  <html>
03  <head>
04  <title>Implicit Objects</title>
05  </head>
06  <body>
07      <%
08          request.setCharacterEncoding("utf-8");
09          String userid = request.getParameter("id");
10          String password = request.getParameter("passwd");
11
12          if (userid.equals("관리자") && password.equals("1234")) {
13              response.sendRedirect("response01_success.jsp");
14          } else {
15              response.sendRedirect("response01_failed.jsp");
16          }
17      %>
18  </body>
19  </html>
```


3. response 내장 객체의 기능과 사용법

JSPBook/WebContent/ch05/response01_sucess.jsp

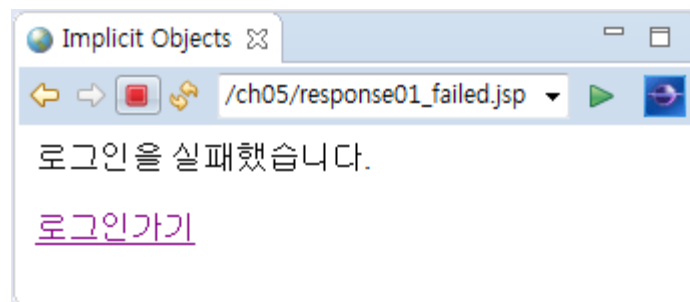
```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     로그인을 성공했습니다!!
08 </body>
09 </html>
```



3. response 내장 객체의 기능과 사용법

JSPBook/WebContent/ch05/response01_failed.jsp

```
01 %@ page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <p>로그인을 실패했습니다.
08     <p> <a href="./response01.jsp"> 로그인 가기</a>
09 </body>
10 </html>
```



3. response 내장 객체의 기능과 사용법

❖ 응답 HTTP 헤더 관련 메소드

- 응답 HTTP 헤더 관련 메소드는 서버가 웹 브라우저에 응답하는 정보에 헤더를 추가하는 기능을 제공
- 헤더 정보에는 주로 서버에 대한 정보가 저장되어 있음
- 응답 HTTP 헤더 관련 메소드의 종류

응답 HTTP 헤더 관련 메소드	반환 유형	설명
<code>addCookie(Cookie cookie)</code>	<code>void</code>	쿠키를 추가합니다.
<code>addDateHeader(String name, long date)</code>	<code>void</code>	설정된 헤더 이름 <code>name</code> 에 날짜/시간을 추가합니다.
<code>addHeader(String name, String value)</code>	<code>void</code>	설정된 헤더 이름 <code>name</code> 에 <code>value</code> 를 추가합니다.
<code>addIntHeader(String name, int value)</code>	<code>void</code>	설정된 헤더 이름 <code>name</code> 에 정수값 <code>value</code> 를 추가합니다.
<code>setDateHeader(String name, long date)</code>	<code>void</code>	설정된 헤더 이름 <code>name</code> 에 날짜/시간을 설정합니다.
<code>setHeader(String name, String value)</code>	<code>void</code>	설정된 헤더 이름 <code>name</code> 에 문자열 값 <code>value</code> 를 설정합니다.
<code>setIntHeader(String name, int value)</code>	<code>void</code>	설정된 헤더 이름 <code>name</code> 에 정수값 <code>value</code> 를 설정합니다.
<code>containsHeader(String name)</code>	<code>boolean</code>	설정된 헤더 이름 <code>name</code> 이 HTTP 헤더에 포함되었는지 여부를 확인합니다. 포함하고 있는 경우 <code>true</code> 를 반환하고, 그렇지 않은 경우 <code>false</code> 를 반환합니다.
<code>getHeader(String name)</code>	<code>String</code>	설정된 헤더 이름 <code>name</code> 값을 가져옵니다.

3. response 내장 객체의 기능과 사용법

[response 내장 객체 사용 예: 응답 HTTP 헤더에 정보 추가하기]

```
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <%
        response.setHeader("Cache-control", "use_cache");
        response.addHeader("contentType", "text/html; charset=utf-8");
        response.setDateHeader("date", 1L);
    %>
    Cache-control : <%=response.getHeader("Cache-control")%>\n
    contentType : <%=response.getHeader("contentType")%>\n
    date : <%=response.getHeader("date")%>
</body>
</html>
```

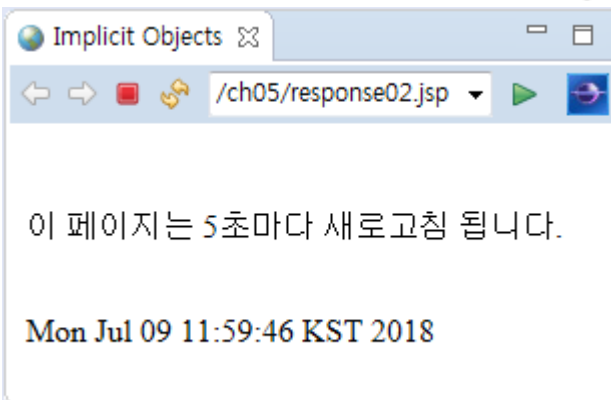
Cache-control : use_cache contentType : text/html; charset=utf-8 date : Thu, 01 Jan 1970 00:00:00 GMT

3. response 내장 객체의 기능과 사용법

예제 5-5 response 내장 객체로 5초마다 JSP 페이지 갱신하기

JSPBook/WebContent/ch05/response02.jsp

```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <p>이 페이지는 5초마다 새로고침 됩니다.
08     <%
09         response.setIntHeader("Refresh", 5);
10     %>
11     <p> <%= (new java.util.Date()) %>
12 </body>
13 </html>
```



3. response 내장 객체의 기능과 사용법

❖ 응답 콘텐츠 관련 메소드

- response 내장 객체는 웹 브라우저로 응답하기 위해 MIME 유형, 문자 인코딩, 오류 메시지, 상태 코드 등을 설정하고 가져오는 응답 콘텐츠 관련 메소드를 제공
- 응답 콘텐츠 관련 메소드의 종류

응답 콘텐츠 관련 메소드	반환 유형	설명
setContentType(String type)	void	웹 브라우저에 응답할 MIME 유형을 설정합니다.
getContentType()	String	웹 브라우저에 응답할 MIME 유형을 가져옵니다.
setCharacterEncoding(String charset)	void	웹 브라우저에 응답할 문자 인코딩을 설정합니다.
getCharacterEncoding()	String	웹 브라우저에 응답할 문자 인코딩을 가져옵니다.
sendError(int status_code, String message)	void	웹 브라우저에 응답할 오류(코드 및 오류 메시지)를 설정합니다.
setStatus(int statuscode)	void	웹 브라우저에 응답할 HTTP 코드를 설정합니다.

3. response 내장 객체의 기능과 사용법

❖ MIME (Multipurpose Internet Mail Extensions)

- MIME (Multipurpose Internet Mail Extensions)는 전자 우편을 위한 인터넷 표준 포맷.
- 오래전 이메일 서비스에서 메일에 첨부된 파일의 데이터타입을 구분하기위한용도로생겨남
- 전자우편은 7비트 ASCII 문자를 사용하여 전송되기 때문에, 8비트 이상의 코드를 사용하는 문자나 이진 파일들은 MIME 포맷으로 변환되어 SMTP로 전송됨.
- text/html:텍스트 데이터며,텍스트내용은HTML형태의 파일.
- video/mpeg:비디오 데이터며,MPEG영상 형태의 파일이다.
- image/jpeg:이미지 데이터며,JPEG 형태의 파일이다.
- audio/x-wav:오디오 데이터며,wav형태의 파일이다.

3. response 내장 객체의 기능과 사용법

[response 내장 객체 사용 예: 응답 콘텐츠 설정하기]

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <%
        response.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");
    %>
    <p>문자 인코딩 : <%=response.getCharacterEncoding()%>
    <p>콘텐츠 유형 : <%=response.getContentType()%>
</body>
</html>
```

문자 인코딩 : utf-8

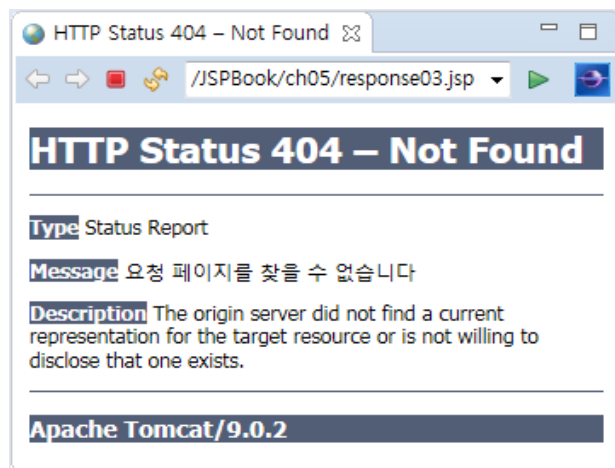
콘텐츠 유형 : text/html; charset=utf-8

3. response 내장 객체의 기능과 사용법

예제 5-6 response 내장 객체로 오류 응답 코드와 오류 메시지 보내기

JSPBook/WebContent/ch05/response03.jsp

```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <%
08         response.sendError(404, "요청 페이지를 찾을
09         수 없습니다");
10     %>
11 </body>
12 </html>
```



4. out 내장 객체의 기능과 사용법

❖ out 내장 객체

- 웹 브라우저에 데이터를 전송하는 출력 스트림 객체
- JSP 컨테이너는 JSP 페이지에 사용되는 모든 표현문 태그와 HTML, 일반 텍스트 등을 out 내장 객체를 통해 웹 브라우저에 그대로 전달
- 스크립틀릿 태그에 사용하여 단순히 값을 출력하는 표현문 태그 (<%= ...%>)와 같은 결과를 얻을 수 있음

4. out 내장 객체의 기능과 사용법

■ out 내장 객체 메소드의 종류

out 내장 객체 메소드	반환 유형	설명
print(String str)	void	설정된 str 값을 웹 브라우저에 출력합니다.
println(String str)	void	설정된 str 값을 웹 브라우저에 출력합니다. 이때 줄 바꿈(WrWn 또는 Wn)이 적용되지 않습니다.
newLine()	void	줄바꿈(WrWn 또는 Wn)을 출력합니다.
getBufferSize()	int	현재 출력 버퍼의 크기를 가져옵니다.
getRemaining()	int	현재 남아 있는 출력 버퍼의 크기를 가져옵니다.
clear()	void	현재 출력 버퍼에 저장되어 있는 내용을 웹 브라우저에 전송하지 않고 비웁니다. 만약 버퍼가 이미 플러시되었다면 IOException이 발생합니다.
clearBuffer()	void	현재 출력 버퍼에 저장되어 있는 내용을 웹 브라우저에 전송하지 않고 비웁니다. 만약 버퍼가 이미 플러시되었다면 IOException이 발생하지 않습니다.
flush()	void	현재 출력 버퍼에 저장되어 있는 내용을 웹 브라우저에 전송하고 비웁니다.
isAutoFlush()	boolean	출력 버퍼가 채워졌을 때의 처리를 결정합니다. 자동으로 플러시하는 경우 true를 반환하고, 그렇지 않은 경우 false를 반환합니다.

4. out 내장 객체의 기능과 사용법

[out 내장 객체 사용 예]

```
<% page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <%
        out.println("Hello!");
        out.println("Java Server Pages 입니다.");
    %>
</body>
</html>
```

Hello! Java Server Pages 입니다.

```
<% page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>Implicit Objects</title>
</head>
<body>
    <%
        out.println("Hello!"+"<br>");
        out.println("Java Server Pages 입니다.");
    %>
</body>
</html>
```

Hello!
Java Server Pages 입니다.

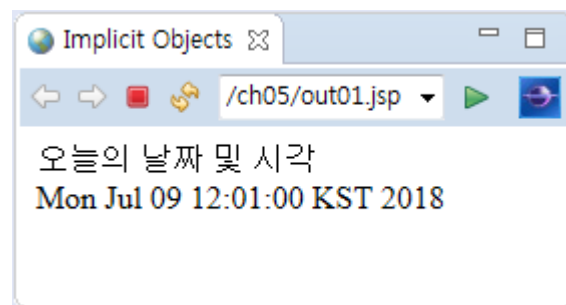
4. out 내장 객체의 기능과 사용법

예제 5-7 out 내장 객체로 오늘의 날짜 및 시각 출력하기

JSPBook/WebContent/ch05/out01.jsp

```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>

03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <%
08         out.println("오늘의 날짜 및 시각 " + "<br>");
09         out.println(java.util.Calendar.getInstance().getTime());
10     %>
11 </body>
12 </html>
```

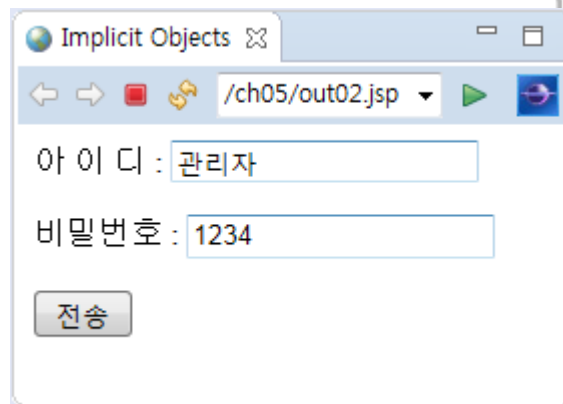


4. out 내장 객체의 기능과 사용법

예제 5-8 out 내장 객체로 폼 페이지에서 아이디와 비밀번호를 전송받아 출력하기

JSPBook/WebContent/ch05/out02.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <form action="out02_process.jsp" method="post">
08         <p>아 이 디 : <input type="text" name="id">
09         <p>비밀번호 : <input type="text" name="passwd">
10         <p><input type="submit" value="전송" />
11     </form>
12 </body>
13 </html>
```



Implicit Objects

/ch05/out02.jsp

아 이 디 : 관리자

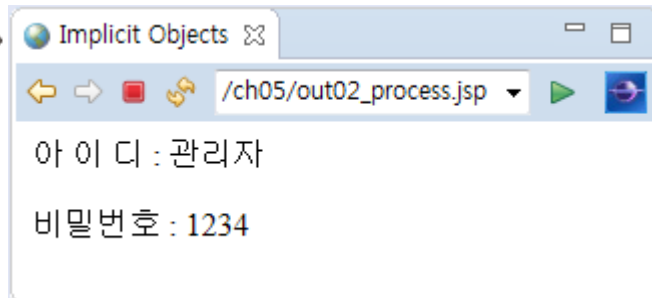
비밀번호 : 1234

전송

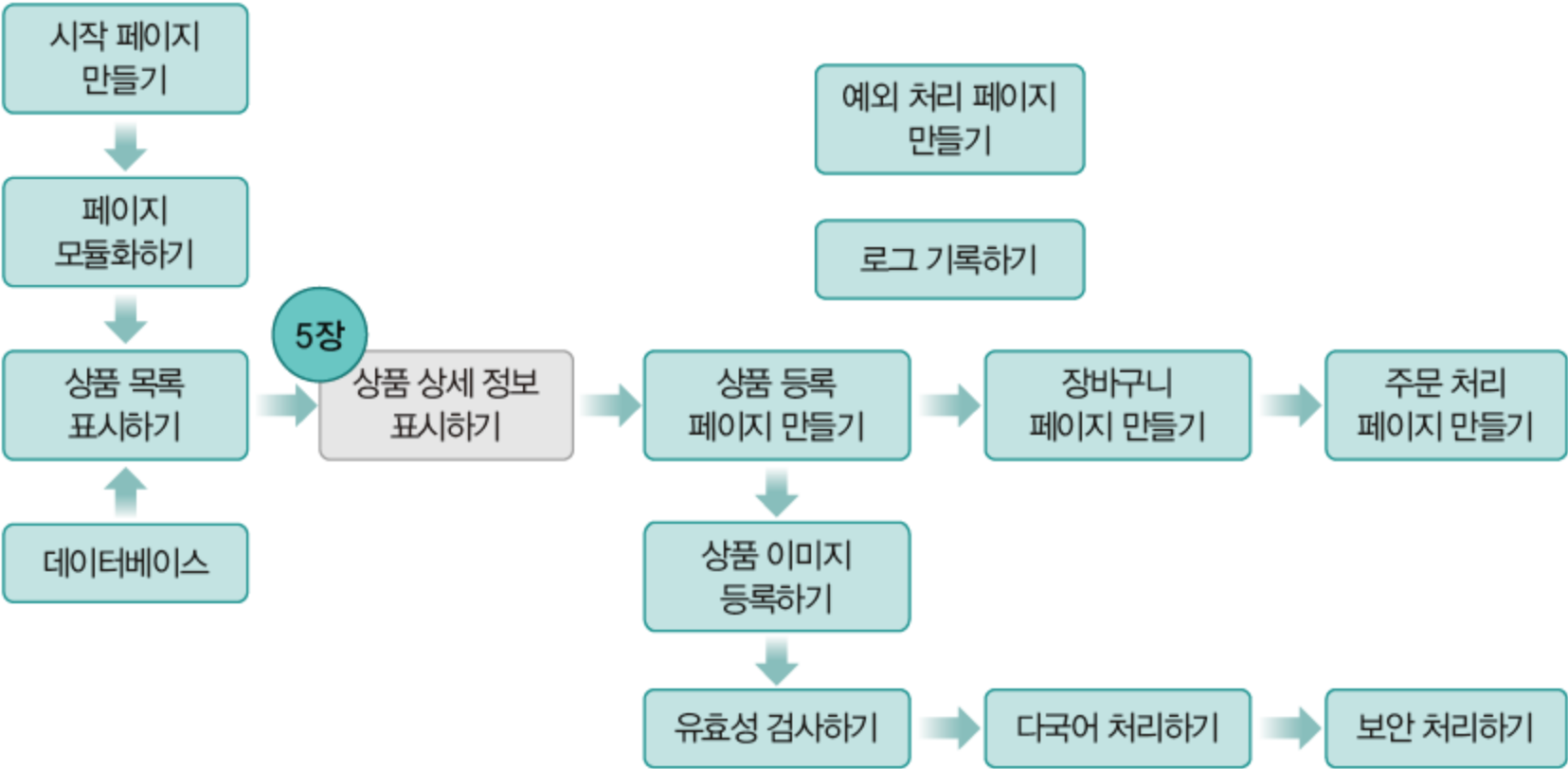
4. out 내장 객체의 기능과 사용법

JSPBook/WebContent/ch05/out02_process.jsp

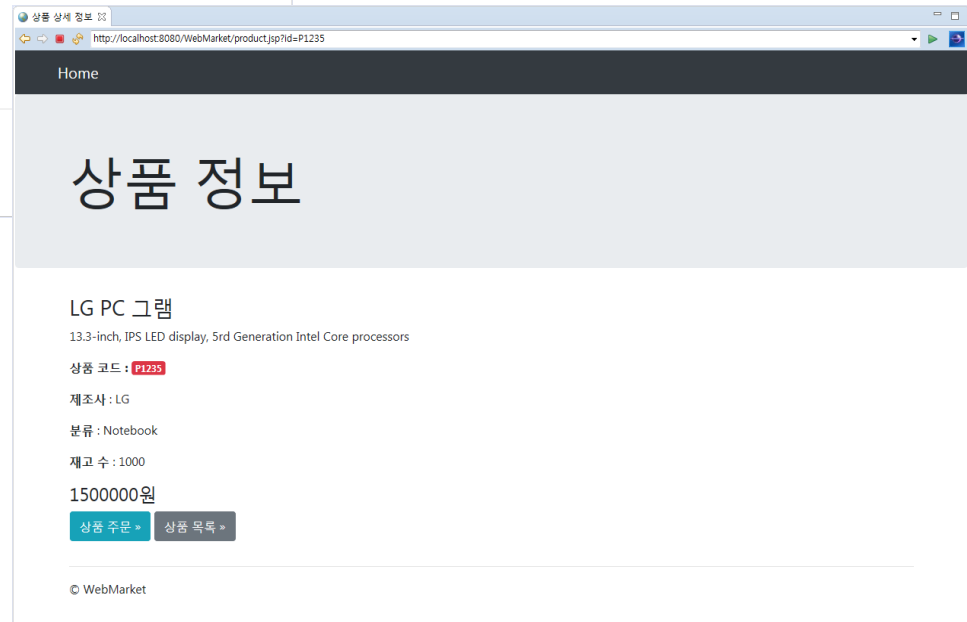
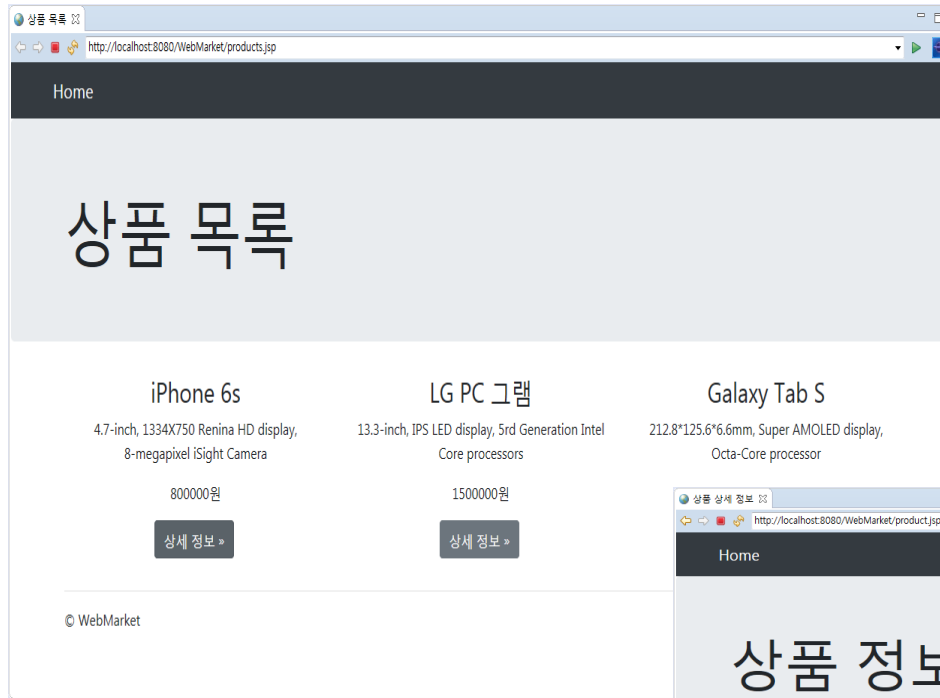
```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>Implicit Objects</title>
05 </head>
06 <body>
07     <%
08         request.setCharacterEncoding("utf-8");
09         String userid = request.getParameter("id");
10         String password = request.getParameter("passwd");
11     %>
12     <p>아 이 디 :<% out.println(userid); %>
13     <p>비밀번호 :<% out.println(password); %>
14 </body>
15 </html>
```



5. [웹 쇼핑몰] 상품 상세 정보 표시하기



5. [웹 쇼핑몰] 상품 상세 정보 표시하기



5. [웹 쇼핑몰] 상품 상세 정보 표시하기

예제 5-9 상품 상세 정보 표시하기

- ❖ [상품 데이터 접근 클래스 만들기]
 - 상품 상세 정보를 가져오는 메소드 만들기

WebMarket/src/dao/ProductRepository.java

```
01 package dao;
02 ... (생략) ...
03 public class ProductRepository {
04     ... (생략) ...
05     public Product getProductById(String productId) {
06         Product productById = null;
07
08         for (int i = 0; i < listOfProducts.size(); i++) {
09             Product product = listOfProducts.get(i);
10             if (product != null && product.getProductid() != null && product.
                getProductid().equals(productId)) {
11                 productById = product;
12                 break;
13             }
14         }
15         return productById;
16     }
17 }
```

5. [웹 쇼핑몰] 상품 상세 정보 표시하기

- 상품 상세 정보 버튼 만들기

WebMarket/WebContent/products.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02     ...(생략)...

03         <p>×%=product.getUnitPrice()%원
04         <p> <a href="./product.jsp?id=×%=product.getProductId()%"
           class="btn btn-secondary" role="button"> 상세 정보 &raquo;×/a>
05     </div>
06     ...(생략)...
```

5. [웹 쇼핑몰] 상품 상세 정보 표시하기

■ 상품 상세 정보 버튼 만들기

WebMarket/WebContent/product.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02 <%@ page import="dto.Product"%>
03 <jsp:useBean id="productDAO" class="dao.ProductRepository" scope="session" />
04 <html>
05 <head>
06 <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
07 <title>상품 상세 정보</title>
08 </head>
09 <body>
10     <jsp:include page="menu.jsp" />
11     <div class="jumbotron">
12         <div class="container">
13             <h1 class="display-3">상품 정보</h1>
14         </div>
15     </div>
16     <%
17         String id = request.getParameter("id");
18         Product product = productDAO.getProductById(id);
19     %>
20     <div class="container">
21         <div class="row">
22             <div class="col-md-6">
23                 <h3><%=product.getPname()%></h3>
24                 <p><%=product.getDescription()%>
25                 <p><b>상품 코드 : </b><span class="badge badge-danger">
                    <%=product.getProductid()%></span>
```

5. [웹 쇼핑몰] 상품 상세 정보 표시하기

```
26         <p> <b>제조사</b> : <%=product.getManufacturer()%>
27         <p> <b>분류</b> : <%=product.getCategory()%>
28         <p> <b>재고 수</b> : <%=product.getUnitsInStock()%>
29         <h4><%=product.getUnitPrice()%>원</h4>
30         <p> <a href="#" class="btn btn-info"> 상품 주문 &raquo;</a>
           <a href="./products.jsp" class="btn btn-secondary"> 상품
           목록 &raquo;</a>
31     </div>
32 </div>
33 <hr>
34 </div>
35 <jsp:include page="footer.jsp" />
36 </body>
37 </html>
```

```
<h3><%= out.println(product.getPname() );%></h3>
<p><%= out.println(product.getDescription() ); %>
<p><b>상품 코드 : </b>
<span class="badge badge-danger"> <%=product.getProductId()%></span>
<p><b>제조사</b> : <%=out.println( product.getManufacturer() ); %>
<p><b>분류</b> : <%=out.println( product.getCategory() ); %>
<p><b>재고 수</b> : <%=out.println(product.getUnitsInStock() ); %>
<h4><%=out.println(product.getUnitPrice()) ; %>원</h4>
```

5. [웹 쇼핑몰] 상품 상세 정보 표시하기

예제 5-10 시작 페이지의 접속 시각 자동 갱신하기

WebMarket/WebContent/welcome.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02     ...(생략)...
03         response.setIntHeader("Refresh", 5);
04         Date day = new java.util.Date();
05         String am_pm;
06     ...(생략)...
```