

# 3장. 내장 객체

JSP 내장 객체



# 내장 객체

## 내장 객체(implicit object)

JSP 페이지에서 사용할 수 있도록 JSP 컨테이너에 미리 정의된 객체이다.

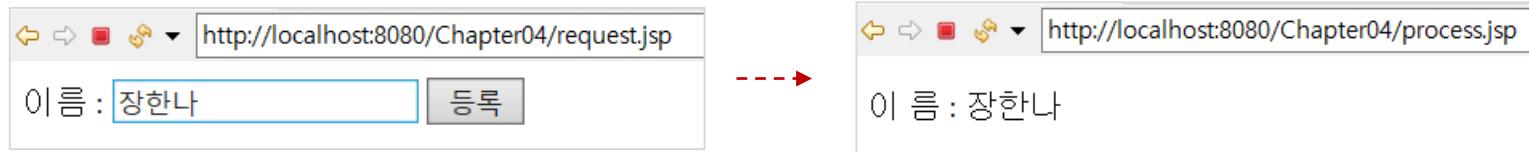
- **request** : 웹 브라우저에서 서버의 JSP 페이지로 전달하는 정보를 저장한다.
- **response** : 사용자의 요청을 처리한 결과를 서버에서 웹 브라우저로 전달하는 정보를 저장한다.

### request 객체

메소드	반환유형	설 명
<b>getParameter(String name)</b>	String	요청 파라미터 이름이 name인 값을 전달받음
<b>getParameterValues(String name)</b>	String[ ]	모든 요청 파라미터 이름이 name인 값을 배열 형태로 전달 받음
<b>getParameterNames()</b>	Java.util.Enumeration	모든 요청 파라미터의 이름과 값을 Enumeration 객체 타입으로 전달 받음

# request 객체

- request 객체를 이용한 이름 등록하기



```
<form action="process.jsp" method="post">
    <label for="name">이름 : </label>
    <input type="text" id="name" name="name">
    <input type="submit" value="등록">
</form>
```

# request 객체

- request객체를 이용한 이름 등록하기

```
<%  
    //폼 페이지에 전달받은 한글 인코딩 처리  
    request.setCharacterEncoding("utf-8");  
  
    //폼 페이지에 입력받은 이름을 전달받음  
    String name = request.getParameter("name");  
%>  
<p>이 름 : <%=name %></p>
```



# request 객체

- 아이디와 비밀번호를 전송받아 출력하기



아이디 : myuser

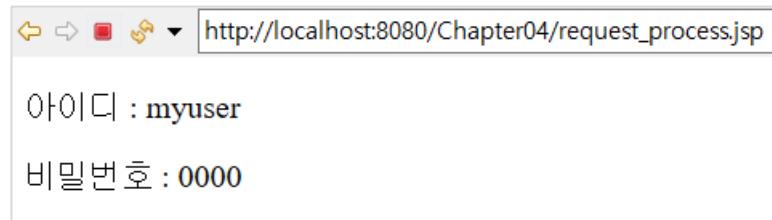
비밀번호 : .....

전송

```
<form action="request_process.jsp" method="post">
    <p>
        <label for="id">아이디 : </label>
        <input type="text" id="id" name="id">
    </p>
    <p>
        <label for="passwd">비밀번호 : </label>
        <input type="password" id="passwd" name="passwd">
    </p>
    <p><input type="submit" value="전송"></p>
</form>
```

# request 객체

- 아이디와 비밀번호를 전송받아 출력하기



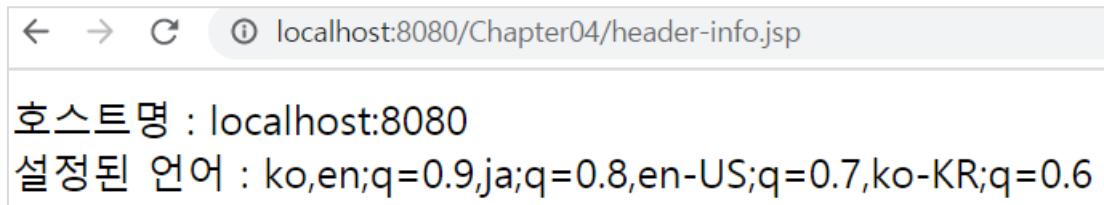
```
<%  
    request.setCharacterEncoding("utf-8");  
  
    //폼 페이지에 입력받은 아이디와 비밀번호를 가져옴  
    String id = request.getParameter("id");  
    String pwd = request.getParameter("passwd");  
%>  
<p>아이디 : <%=id %></p>  
<p>비밀번호 : <%=pwd %></p>
```

# request 객체

## ➤ HTTP 헤더 정보 값 출력하기

웹 브라우저는 HTTP 헤더에 부가적인 정보를 담아 서버로 전송한다.

request 내장 객체는 헤더 정보나 쿠키 관련 정보를 얻을수 있는 메서드를 제공한다.



Name	Headers	Preview	Response	Initiator	Timing	Cookies
header.jsp	<b>Request Headers</b> view source					
	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/a nge;v=b3;q=0.9 Accept-Encoding: gzip, deflate, br Accept-Language: ko,en;q=0.9,ja;q=0.8,en-US;q=0.7,ko-KR;q=0.6 Cache-Control: max-age=0 Connection: keep-alive Cookie: JSESSIONID=A103D16AAB676CCA6935AD722C9B36AF Host: localhost:8080					

# request 객체

## ➤ HTTP 헤더 정보 값 출력하기

```
<%  
    String hostValue = request.getHeader("host");  
    String alValue = request.getHeader("accept-language");  
  
    out.println("호스트명 : " + hostValue + "<br>");  
    out.println("설정된 언어 : " + alValue + "<br>");  
%>
```



# request 객체

## ➤ HTTP 헤더의 모든 정보 값 출력하기

```
host: localhost:8080
connection: keep-alive
sec-ch-ua: "Chromium";v="88", "Google Chrome";v="88", ";Not
sec-ch-ua-mobile: ?0
upgrade-insecure-requests: 1
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.150 Safari/537.36
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
sec-fetch-site: none
sec-fetch-mode: navigate
sec-fetch-user: ?1
sec-fetch-dest: document
accept-encoding: gzip, deflate, br
accept-language: ko,en;q=0.9,ja;q=0.8,en-US;q=0.7,ko-KR;q=0.6
cookie: JSESSIONID=6E6584C9DA1FC520C077E096DDC053D2
```

# request 객체

## ➤ HTTP 헤더의 모든 정보 값 출력하기

```
<%
    //Header 정보 모두 가져오기
    Enumeration<String> en = request.getHeaderNames();
    while(en.hasMoreElements()){
        String headerName = en.nextElement();
        String headerValue = request.getHeader(headerName);
        /* out.println(headerName + ":");
        out.println(headerValue + "<br>"); */
    } /*
%>
<%=headerName %> : <%=headerValue %><br>
<%
    }
%>
```



# request 객체

## ➤ 웹 브라우저 및 서버 정보 값 출력하기

request 내장 객체의 웹 브라우저와 서버 관련 메소드를 이용하여 서버 정보 출력

웹 브라우저 정보 – 클라이언트 IP, 요청 정보길이, 전송방식, 요청 URI 등

서버 정보 – 서버 이름과 포트

A screenshot of a web browser window. The address bar shows 'localhost:8080/Chapter04/request.jsp'. Below the address bar is a form with a single input field labeled '이름' containing the value '관리자' and a submit button labeled '등록'.

이 름 : 관리자  
요청 정보 길이 : 32  
클라이언트 IP : 0:0:0:0:0:0:1  
클라이언트 전송방식 : POST  
요청 URI: /Chapter04/process.jsp  
서버 이름 : localhost  
서버 포트 : 8080

# request 객체

## ➤ 웹 브라우저 및 서버 정보 값 출력하기

```
<%
    //폼 페이지에 전달받은 한글 인코딩 처리
    request.setCharacterEncoding("utf-8");

    //폼 페이지에 입력받은 이름을 전달받음
    String name = request.getParameter("name");
%>
<p>이 름 : <%=name %><br>
    요청 정보 길이 : <%=request.getLength() %><br>
    클라이언트 IP : <%=request.getRemoteAddr() %><br>
    클라이언트 전송방식 : <%=request.getMethod() %><br>
    요청 URI: <%=request.getRequestURI() %><br>
    서버 이름 : <%=request.getServerName() %><br>
    서버 포트 : <%=request.getServerPort() %>
</p>
```



# request 객체

## ➤ 모든 웹 브라우저 및 서버 정보 값 출력하기



The screenshot shows a browser window with the URL `localhost:8080/ImplicitObjects/request03.jsp`. The page content displays various pieces of information about the request object, each preceded by a descriptive label in blue.

요청 정보	값
클라이언트 IP	: 0:0:0:0:0:0:1
요청 정보 길이	: -1
요청 정보 인코딩	: null
요청 정보 콘텐츠 유형	: null
요청 정보 프로토콜	: HTTP/1.1
요청 정보 전송방식	: GET
요청 URI	: /ImplicitObjects/request03.jsp
컨텍스트 경로	: /ImplicitObjects
서버 이름	: localhost
서버 포트	: 8080

The right side of the screenshot shows the corresponding JSP code. It uses the `<%= %>` syntax to print the values of various request methods. The code is contained within a `<body></body>` block.

```
<body>
    <p>클라이언트 IP : <%=request.getRemoteAddr() %>
    <p>요청 정보 길이 : <%=request.getContentLength() %>
    <p>요청 정보 인코딩 : <%=request.getCharacterEncoding() %>
    <p>요청 정보 콘텐츠 유형 : <%=request.getContentType() %>
    <p>요청 정보 프로토콜 : <%=request.getProtocol() %>
    <p>요청 정보 전송방식 : <%=request.getMethod() %>
    <p>요청 URI : <%=request.getRequestURI() %>
    <p>컨텍스트 경로 : <%=request.getContextPath() %>
    <p>서버 이름 : <%=request.getServerName() %>
    <p>서버 포트 : <%=request.getServerPort() %>
    <p>쿼리문 : <%=request.getQueryString() %>
</body>
```



# response 객체

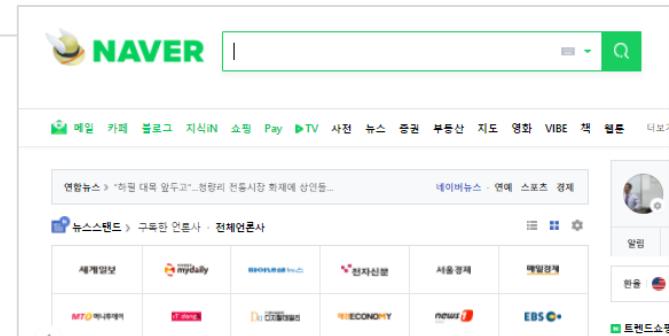
## response 내장 객체

사용자의 요청을 처리한 결과를 서버에서 웹 브라우저로 전달하는 정보를 저장한다 즉, 서버는 응답 헤더와 요청 처리 결과 데이터를 웹 브라우저로 보낸다.

### 1. 페이지 이동관련 메서드

메소드	반환유형	설명
sendRedirect(String url)	void	설정한 URL 페이지로 강제 이동합니다.

```
<!DOCTYPE html>
<html>
<head>
    <title>request 객체</title>
</head>
<body>
    <%
        response.sendRedirect("http://www.naver.com");
    %>
</body>
</html>
```



# response 객체

## response 객체로 페이지 이동하기

A screenshot of a web browser window. The address bar shows the URL `http://localhost:8080/Chapter04/login/loginForm.jsp`. The page contains a login form with two text input fields and one button. The first field is labeled "아이디 :" and contains the value "admin". The second field is labeled "비밀번호 :" and contains four redacted characters. Below the fields is a button labeled "전송".

A screenshot of a web browser window. The address bar shows the URL `http://localhost:8080/Chapter04/login/loginSuccess.jsp`. The page displays the message "로그인을 성공했습니다." (Login was successful.) in black text.

A screenshot of a web browser window. The address bar shows the URL `http://localhost:8080/Chapter04/login/loginFail.jsp`. The page displays the message "로그인을 실패했습니다." (Login failed.) in black text. Below the message is a purple link labeled "로그인 가기" (Login again).



# response 객체

## response 객체로 페이지 이동하기

```
<title>로그인 폼</title>
</head>
<body>
    <form action="LoginProcess.jsp" method="post">
        <p>
            <label for="id">아이디 : </label>
            <input type="text" id="id" name="id">
        </p>
        <p>
            <label for="passwd">비밀번호 : </label>
            <input type="password" id="passwd" name="passwd">
        </p>
        <p><input type="submit" value="전송"></p>
    </form>
</body>
```

loginForm.jsp



# response 객체

## response 객체로 페이지 이동하기

```
<%  
    request.setCharacterEncoding("utf-8");  
  
    String id = request.getParameter("id");  
    String pwd = request.getParameter("passwd");  
  
    if(id.equals("admin") && pwd.equals("1234")){  
        //out.println("로그인을 성공했습니다.");  
        response.sendRedirect("loginSuccess.jsp");  
    }  
    else{  
        //out.println("로그인을 실패했습니다.");  
        response.sendRedirect("loginFail.jsp");  
    }  
%>
```

loginProcess.jsp

```
<body>  
    <p>로그인을 성공했습니다.</p>  
</body>
```

loginSuccess.jsp

```
<body>  
    <p>로그인을 실패했습니다.</p>  
    <a href="LoginForm.jsp">로그인 가기</a>  
</body>
```

loginFail.jsp



# response 객체

## response 객체로 5초마다 JSP 페이지 갱신하기

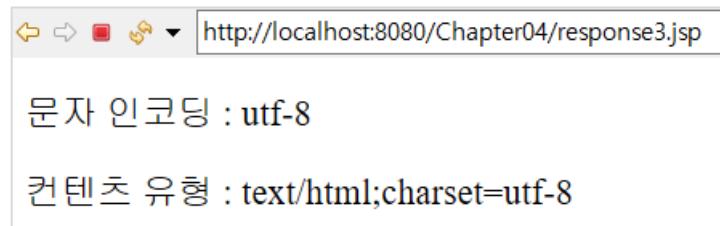
```
← → ⌂ ⓘ localhost:8080/ImplicitObjects/response02.jsp  
이 페이지는 5초마다 새로고침 됩니다.  
Mon Oct 12 10:47:41 KST 2020
```

```
<body>  
    <p>이 페이지는 5초마다 새로고침 됩니다.  
    <%  
        response.setIntHeader("Refresh", 5);  
    %>  
    <p><%=new Date() %>  
</body>
```



# 내장 객체

## response 객체로 응답 컨텐츠 설정하기



```
<%  
    response.setCharacterEncoding("uft-8");  
    response.setContentType("text/html; charset=utf8");  
%>  
<p> 문자 인코딩 : <%=response.getCharacterEncoding()%>  
<p> 컨텐츠 유형 : <%=response.getContentType() %>
```

# 내장 객체

## 계산기 프로그램 구현

0/Chapter04/calculator/calc.jsp

계산기

---

10  4

---

0/Chapter04/calculator/result.jsp

계산기

---

계산 결과 : 14

---

# 내장 객체

## 계산기 프로그램 구현 1. – JSP로만 구현

```
<body>
<%!
    //전역 멤버 변수 선언
    private int num1 = 0;
    private int num2 = 0;
    private String op = "";
    private int result = 0;

    public int calculate(){ //전역 메서드
        if(op.equals("+"))
            result = num1 + num2;
        else if(op.equals("-"))
            result = num1 - num2;
        else if(op.equals("*"))
            result = num1 * num2;
        else if(op.equals("/"))
            result = num1 / num2;

        return result;
    }
%>
```

calculator.jsp

```
<%
    //웹 페이지의 요청이 POST인 경우만 수행, 즉 폼을 통해 전달된 것만 수행
    if(request.getMethod().equals("POST")){
        num1 = Integer.parseInt(request.getParameter("num1"));
        num2 = Integer.parseInt(request.getParameter("num2"));
        op = request.getParameter("operator");
    }
%>
```



# JSP 빈즈 프로그래밍

## 계산기 프로그램 구현

calculator.jsp

```
<div id="container">
    <h3>계산기</h3>
    <hr>
    <form name="form1" method="post">
        <input type="text" name="num1">
        <select name="operator">
            <option selected>+</option>
            <option>-</option>
            <option>*</option>
            <option>/</option>
        </select>
        <input type="text" name="num2">
        <input type="submit" value="계산">
        <input type="reset" value="다시 입력">
    </form>
    <hr>
    <p>계산 결과: <%=calculator() %>
</div>
```

```
<style>
    #container{
        width: 600px;
        margin: 0 auto;
        text-align: center;
    }
    p{font-size: 1.2em}
</style>
```



# JSP 빈즈 프로그래밍

## 계산기 프로그램 구현2 – 자바 빈즈로 구현

### 프로그램 소스 목록

파일 이름	역 할
calc.jsp	계산기 메인 화면
calc.css	전체 레이아웃 등의 스타일을 구현
result.jsp	calc.jsp에 입력된 내용을 Calculator 빈을 이용해 계산 기능 수행
Calculator.java	계산기 자료형을 정의한 빈즈 클래스



# JSP 빈즈 프로그래밍

## 계산기 프로그램 구현2 – 자바 빈즈로 구현

```
<title>계산기 프로그램</title>
<link rel="stylesheet" href="calc.css">
</head>
<body>
    <div id="container">
        <h2>계산기</h2>
        <hr>
        <form name="form1" action="result.jsp" method="post">
            <input type="text" name="num1">
            <select name="op">
                <option>+</option>
                <option>-</option>
                <option>*</option>
                <option>/</option>
            </select>
            <input type="text" name="num2">
            <input type="submit" value="계산">
            <input type="reset" value="다시입력">
        </form>
        <hr>
    </div>
</body>
```

calc.jsp

```
calc.css
1 @charset "UTF-8";
2 #container{
3     width: 600px;
4     margin: 0 auto;
5     text-align: center;
6 }
7 p{font-size: 1.2em}
```



# JSP 빈즈 프로그래밍

## 계산기 프로그램 구현2 – 자바 빈즈로 구현

calculator.java

```
package com.bean;
import java.io.Serializable;
public class Calculator implements Serializable{
    private static final long serialVersionUID = 1234;

    private int num1 = 0;
    private int num2 = 0;
    private String op = "";
    private int result = 0;

    public int calculate(){
        if(op.equals("+"))
            result = num1 + num2;
        else if(op.equals("-"))
            result = num1 - num2;
        else if(op.equals("*"))
            result = num1 * num2;
        else if(op.equals("/"))
            result = num1 / num2;

        return result;
    }
}
```

```
public int getNum1() {
    return num1;
}

public void setNum1(int num1) {
    this.num1 = num1;
}

public int getNum2() {
    return num2;
}

public void setNum2(int num2) {
    this.num2 = num2;
}

public String getOp() {
    return op;
}

public void setOp(String op) {
    this.op = op;
}

public int getResult() {
    return result;
}
```



# JSP 빈즈 프로그래밍

Calculator클래스 import해서 구현

```
<div id="container">
    <h2>계산기</h2>
    <hr>
    <%
        int num1 = Integer.parseInt(request.getParameter("num1"));
        int num2 = Integer.parseInt(request.getParameter("num2"));
        String op = request.getParameter("op");

        Calculator calc = new Calculator();
        calc.setNum1(num1);
        calc.setNum2(num2);
        calc.setOp(op);

        calc.calculate();
    %>
    <p>계산 결과 : <%=calc.getResult() %>
    <hr>
</div>
```

result.jsp



# JSP 빈즈 프로그래밍

useBean 태그로 구현하기

result2.jsp

```
<body>
<jsp:useBean id="calc" class="com.bean.Calculator" />
<jsp:setProperty property="num1" name="calc"/>
<jsp:setProperty property="num2" name="calc"/>
<jsp:setProperty property="op" name="calc"/>
<div id="container">
    <h2>계산기</h2>
    <hr>
    <%
        calc.calculate();
    %>

    <p>계산 결과 : <%=calc.getResult() %>
    <hr>
</div>
</body>
```

