02. Servlet & JSP

Servlet

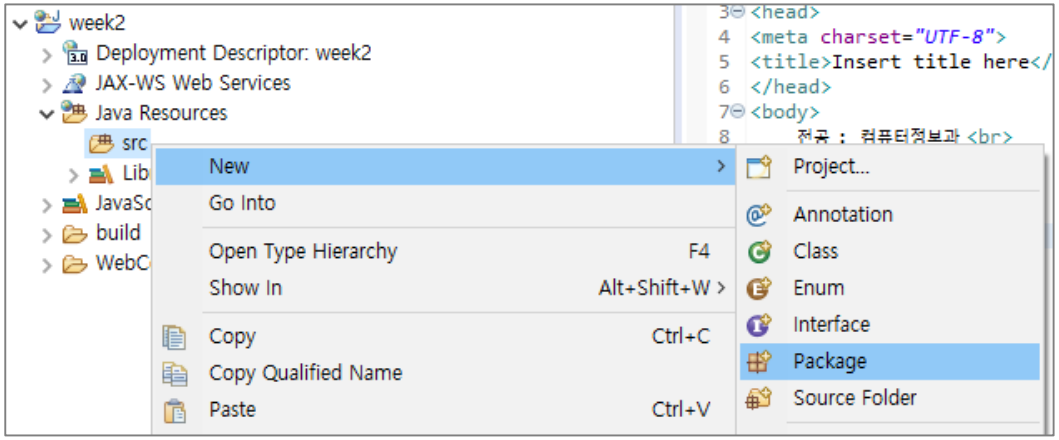
- **서블릿은**
  - **JSP 표준이 나오기 전에 만들어진 표준**
  - **Java**에서 **웹 어플리케이션을 개발**할 수 있도록 하기 위해 만들어짐
  - Java 클래스를 웹에서 호출 및 실행할 수 있도록 한 표준

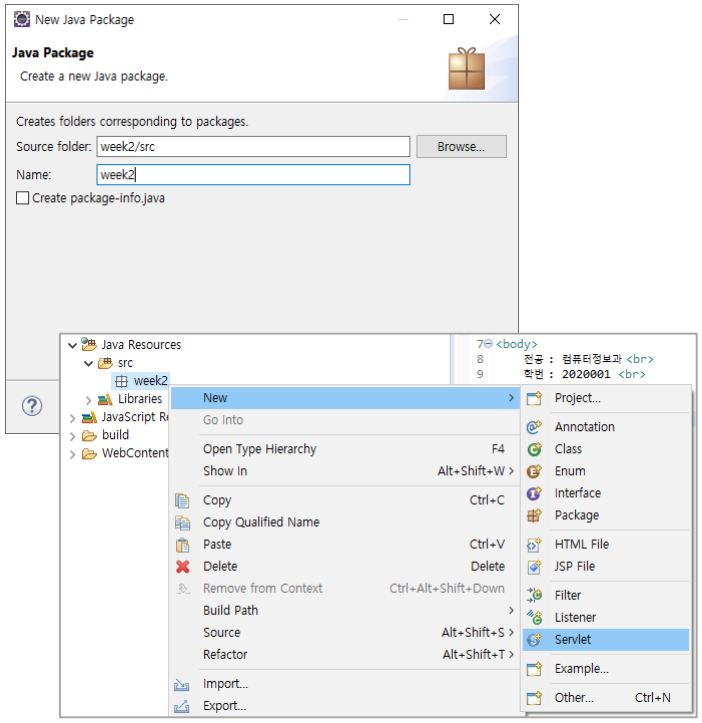
- **작성방법**
  - **javax.servlet.http.HttpServlet** 클래스로부터 **상속**받아서 작성
  - 위 클래스는 톰캣의 servlet-api.jar에 포함되어 있음

- **작성과정**
  - **서블릿 규칙**에 따라 **자바 코드를 작성**
  - 자바 코드를 컴파일해서 클래스 파일을 생성
  - 서블릿 3.0부터는 **@WebServlet 어노테이션**을 사용
  - 톰캣 등의 웹 컨테이너에서 실행

Servlet 예제 1

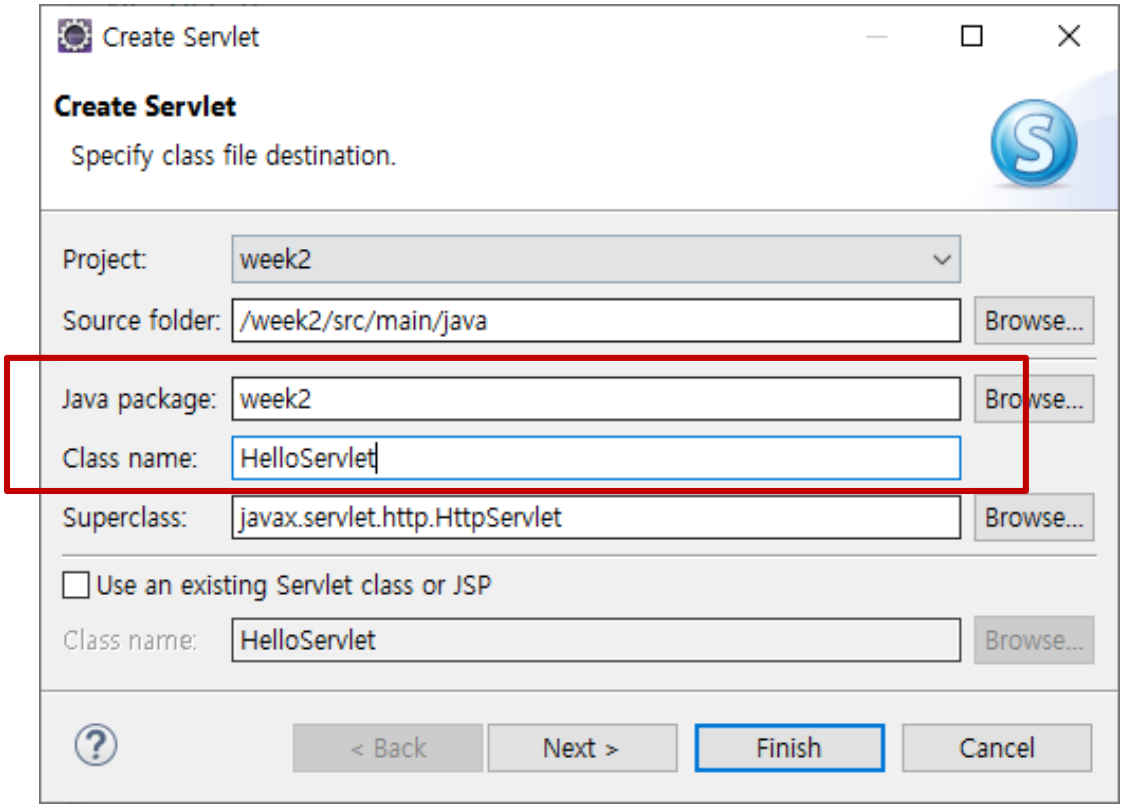
- 새로운 프로젝트 생성
  - File > New > Dynamic Web Project 선택 > week2 생성
  - Java Resources > src 아래에 package 생성
  - week2 package 선택 후 오른쪽 클릭해서 “HelloServlet” 이름으로 서블릿 생성





Servlet 예제 1

- 새로운 프로젝트 생성
  - week2 선택 후 오른쪽 클릭해서 New > Servlet > "HelloServlet" 이름으로 서블릿 생성



Servlet 예제 1

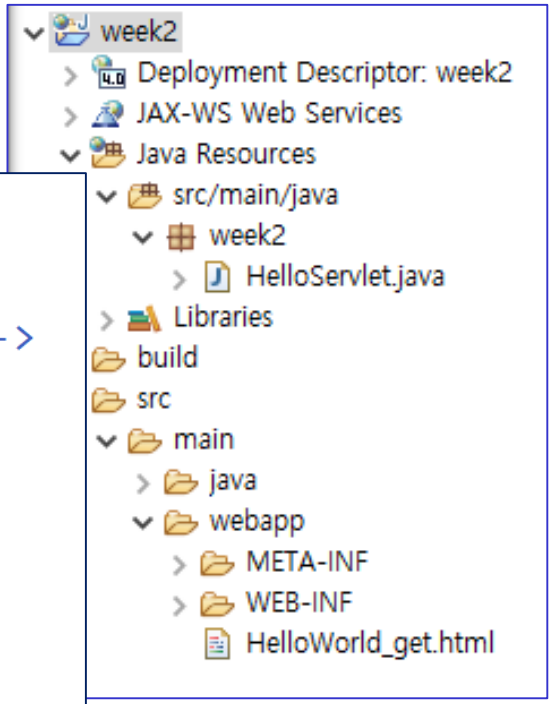
- 새로운 프로젝트 생성
  - File > New > Dynamic Web Project 선택 > week2 생성
  - Java Resources > src 아래에 package 생성
  - week2 package 선택 후 오른쪽 클릭해서 "HelloServlet" 이름으로 서블릿 생성

import java.io.PrintWriter;

```
@WebServlet("/HelloServlet")    //"/HelloServlet" 이라는 요청이 들어오면
public class HelloServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public HelloServlet() {
        super();
    }

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        // get 방식으로 요청한 경우 실행되는 메서드
        // 웹 페이지에 출력되는 내용을 HTML 코드를 포함하여 기술
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h1>Hello Servlet(get method)</h1>");
        out.println("</body></html>");
        out.close();
    }
```

Servlet 호출 예제-1

- **Get 방식** : 주소에 매개변수를 붙여서 호출하는 방식
  - HTML 문서에서 서블릿을 호출



< HelloWorld_get.html >

```
<body>

    <!-- 요청이 들어오면 form 태그의 action에 있는 부분을 호출 -->
    <form action="HelloServlet" method="get">
        <h1>get 방식으로 부르는 페이지입니다.</h1>

        <!-- 'submit' 이 서버로 요청을 보내는 동작 -->
        <input type="submit" value="확인">
    </form>

</body>
```

Servlet 호출 예제-1

- **Post 방식** : 매개변수를 본문에 포함시켜 호출하는 방식
  - HTML 문서에서 서블릿을 호출

< HelloWorld_post.html >

```
<body>

    <!-- 요청이 들어오면 form 태그의 action에 있는 부분을 호출 -->
    <form action="HelloServlet" method="post">
        <h1>post 방식으로 부르는 페이지입니다.</h1>

        <!-- 'submit' 이 서버로 요청을 보내는 동작 -->
        <input type="submit" value="확인">
    </form>

</body>
```

Servlet 예제 1

- HelloServlet 클래스에서 요청 처리

```
protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    // post 방식으로 요청한 경우 실행되는 메서드
    // 웹 페이지에 출력되는 내용을 HTML 코드를 포함하여 기술
    PrintWriter out = response.getWriter();
    out.println("<html><body>");
    out.println("<h1>Hello Servlet(post method)</h1>");
    out.println("</body></html>");
    out.close();
}
```

Servlet

- **서블릿 요청처리**
  - 요청방식에 따라 **doGet** 이나 **doPost 메소드를 재정의**해서 처리
  - Service 메소드를 재정의해서 사용할 수도 있다.

- **세부사항**
  - 서블릿 요청처리를 위해 오버라이딩 한 메소드는 **request 객체**를 이용해서 **웹 브라우저의 요청 정보**를 읽어온다.
  - 요청에 대한 **응답을 전송**할 때는 **response 객체**를 이용한다.
  - **response 객체의 setContentType() 메소드**를 이용해서 **데이터 타입**과 **인코딩 방식**을 지정해 준다.
  - 웹 브라우저에 데이터를 전송할 경우, **PrintWriter 객체**의 **getWriter() 메소드를 호출**해서 **문자열 데이터를 출력**한다.

Servlet 호출 방법

- **Get 방식** : **주소에 매개변수를 붙여서 호출하는 방식**
  - 주소와 매개변수를 붙여서 **주소 표시줄에 입력**하는 방법**(?로 구분**)
  - **<a> 태그**를 이용해서 페이지를 요청하는 경우
  - **자바 스크립트**를 이용해서 요청하는 경우
  - <form> 태그에서 **명시적으로 GET 방식**으로 요청하는 경우
  - **매개변수의 데이터는 255자 이내**이며 **보안이 취약함**

- **Post 방식** : **매개변수를 본문에 포함시켜 호출하는 방식**
  - <form> 태그에서 **명시적으로 POST 방식**으로 요청하는 경우
  - **데이터의 크기에 제한이 없다.**
  - URL에 표시가 되지 않으므로 **보안성이 우수**하다.

Servlet 호출 예제-1

- Get 방식 : 주소에 매개변수를 붙여서 호출하는 방식
  - HTML 문서에서 서블릿을 호출

< HelloWorld.java >

```
@WebServlet("/HelloWorld")
public class HelloWorld extends HttpServlet {
  private static final long serialVersionUID = 1L;
  public HelloWorld () {  super();  }
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
                                                  throws ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h1>Hello World Servlet doGet() 페이지입니다</h1>");
        out.println("</body></html>");
        out.close(); }

protected void doPost(HttpServletRequest request, HttpServletResponse response)
                                                  throws ServletException, IOException {}
}
```

응답할 때 한글 타입 설정

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< addrGet.html >

```
<body>
    <form action="Address" method="get">
        <p>이름 : <input type="text" name="name"></p>
        <p>주소 : <input type="text" name="addr"></p>
        <p><input type="submit" value="확인"></p>
    </form>
</body>
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

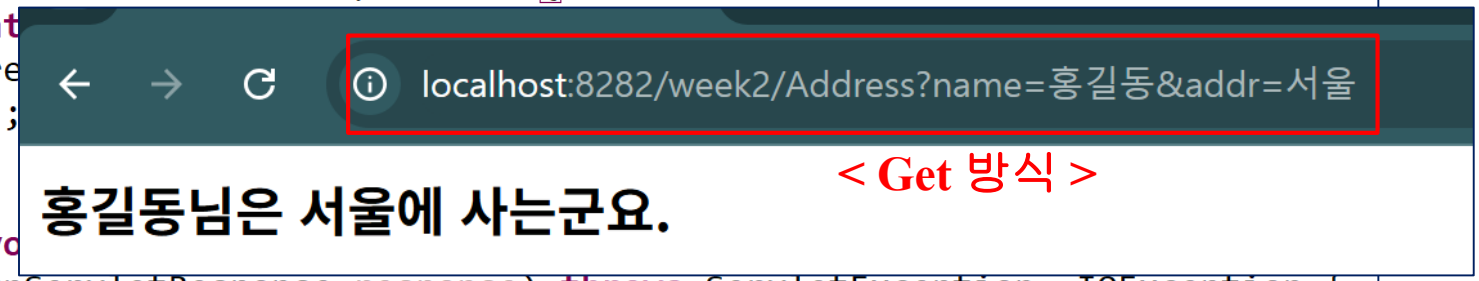
```
@WebServlet("/Address")
public class Address extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Address() {
        super();
    }

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        // get 방식으로 요청이 들어온 경우 실행되는 메서드
        // input 태그에서 입력되어 넘어온 데이터(요청데이터, request에 저장됨)를 받아야 함
        response.setContentType("text/html;charset=utf-8");

        String name = request.getParameter("name");
        String addr = request.getParameter("addr");

        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>" + name + "님은 " + addr + "에 사는군요.</h3>");
        out.println("</body></html>");
        out.close();
    }
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

```
@WebServlet("/Address")
public class Address extends HttpServlet {
    private stat
    public Addre
        super();
    }

    protected vo
            HttpServletResponse response) throws ServletException, IOException {
        // get 방식으로 요청이 들어온 경우 실행되는 메서드
        // input 태그에서 입력되어 넘어온 데이터(요청데이터, request에 저장됨)를 받아야 함
        response.setContentType("text/html;charset=utf-8");

        String name = request.getParameter("name");
        String addr = request.getParameter("addr");

        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>" + name + "님은 " + addr + "에 사는군요.</h3>");
        out.println("</body></html>");
        out.close();
    }
```



Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< addrPost.html >

```
<body>
    <form action="Address" method="post">
        <p>이름 : <input type="text" name="name"></p>
        <p>주소 : <input type="text" name="addr"></p>
        <p><input type="submit" value="확인"></p>
    </form>
</body>
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    // get 방식으로 요청이 들어온 경우 실행되는 메서드
    // input 태그에서 입력되어 넘어온 데이터(요청데이터, request에 저장됨)를 받아야 함
    response.setContentType("text/html;charset=utf-8");

    String name = request.getParameter("name");
    String addr = request.getParameter("addr");

    PrintWriter out = response.getWriter();
    out.println("<html><body>");
    out.println("<h3>" + name + "님은 " + addr + "에 사는군요.</h3>");
    out.println("</body></html>");
    out.close();
}
```

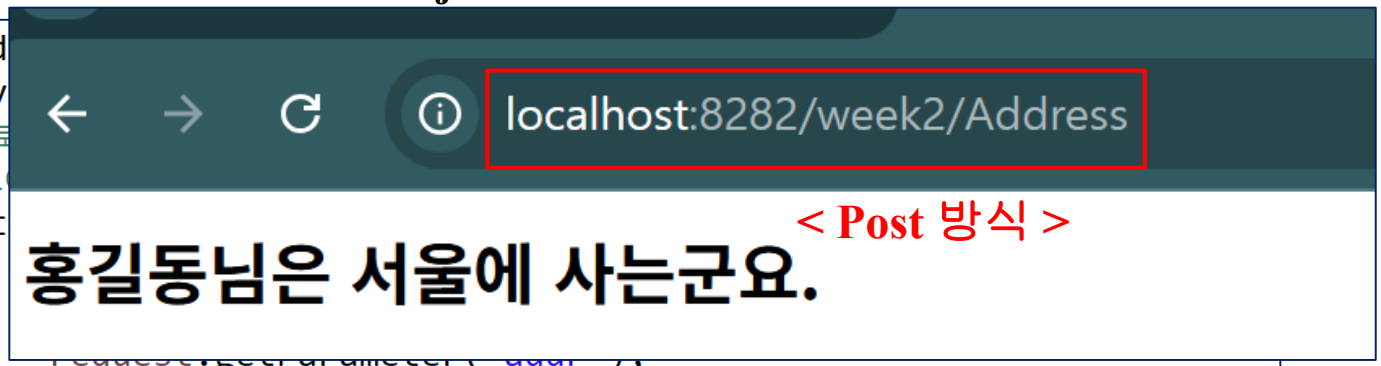
```
protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    // post 방식으로 요청이 들어온 경우, get 방식과 동일한 처리 내용이라면
    // doGet() 메서드를 호출
    doGet(request, response);
}
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

```
protected void d
        HttpServ
    // get 방식으로
    // input 태그
    response.set

    String name
    String addr

    PrintWriter out = response.getWriter();
    out.println("<html><body>");
    out.println("<h3>" + name + "님은 " + addr + "에 사는군요.</h3>");
    out.println("</body></html>");
    out.close();
}
```




```
protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    // post 방식으로 요청이 들어온 경우, get 방식과 동일한 처리 내용이라면
    // doGet() 메서드를 호출
    doGet(request, response);
}
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< addrGet.html >

@WebServlet 어노테이션(annotation)으로
URL mapping - 요청할 서블릿

```
<body>
    <form action="Address" method="get">
        <p>이름 : <input type="text" name="name"></p>
        <p>주소 : <input type="text" name="addr"></p>
        <p><input type="submit" value="확인"></p>
    </form>
</body>
```

해당 버튼을 클릭하면 서블릿이 요청된다

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< addrGet.html >

```
<body>
    <form action="Address" method="get">
        <p>이름 : <input type="text" name="name"></p>
        <p>주소 : <input type="text" name="addr"></p>
        <p><input type="submit" value="확인"></p>
    </form>
</body>
```

해당 매개변수명에 값이 저장되어 서블릿으로 데이터가 전송

전송되는 값의 데이터 타입
"text" 인 경우 String 타입으로 전송

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

```
@WebServlet("/Address")
public class Address exter
    private static final long
    public Address() {
        super();
    }

    protected void doGet(HttpS
            HttpServletRespons
        // get 방식으로 요청이 들어
        // input 태그에서 입력되어
        response.setContentType("text/html;charset=utf-8");

        String name = request.getParameter("name");
        String addr = request.getParameter("addr");

        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>" + name + "님은 " + addr + "에 사는군요.</h3>");
        out.println("</body></html>");
        out.close();
    }
```

@WebServlet("/Address")
서블릿 클래스의 요청을 위한 URL 매핑을 보다 쉽게 자바 클래스에서 설정할 수 있도록 제공되는 어노테이션

@어노테이션
- 문장이나 문서에 추가적인 정보를 기입하는 것
- 자바 프로그램에 영향을 주는 것이 아니라 컴파일할 때 환경설정을 변경해 줄 것을 알려주는 주석형태

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

```
@WebServlet("/Address")
public class Address extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Address() {
        super();
    }

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws Servlet
        // get 방식으로 요청이 들어온 경우 실행되는 메서드
        // input 태그에서 입력되어 넘어온 데이터(요청데이터, request
        response.setContentType("text/html;charset=utf-8");

        String name = request.getParameter("name");
        String addr = request.getParameter("addr");

        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>" + name + "님은 " + ad
        out.println("</body></html>");
        out.close();
    }
```

서블릿 클래스의 요청을 위한 URL 매핑을 보다 쉽게 자바 클래스에서 설정할 수 있도록 제공되는 어노테이션

클라이언트에 응답할 페이지에 대한 환경설정

모든 요청 정보는 request 객체에 저장되어 넘어옴.
'name', 'addr' 매개변수명으로 읽음

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< Address.java >

```
@WebServlet("/Address")
public class Address extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Address() {
        super();
    }

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws Servlet
        // get 방식으로 요청이 들어온 경우 실행되는 메서드
        // input 태그에서 입력되어 넘어온 데이터(요청데이터, request
        response.setContentType("text/html;charset=utf-8");

        String name = request.getParameter("name");
        String addr = request.getParameter("addr");
```

서블릿 클래스의 요청을 위한 URL 매핑을 보다 쉽게 자바 클래스에서 설정할 수 있도록 제공되는 어노테이션

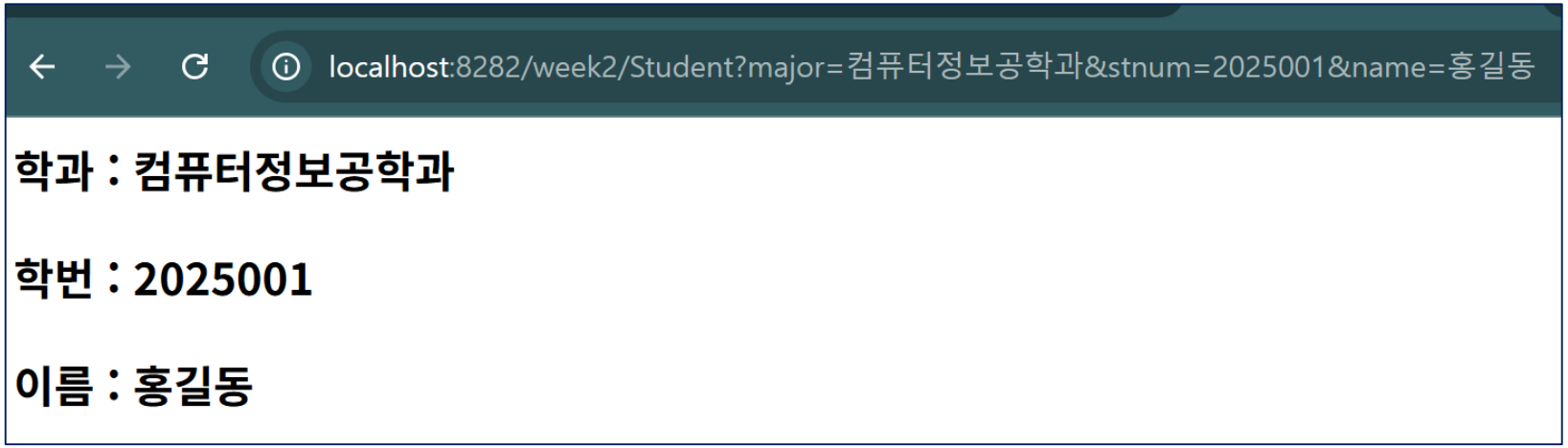
클라이언트에 응답할 페이지에 대한 환경설정

모든 요청 정보는 request 객체에 저장되어 넘어옴.
'name', 'addr' 매개변수명으로 읽음

```
<form action="Address"  method="post">
    이름 : <input type="text" name="name"><br>
    주소 : <input type="text" name="addr"><br>
    <input type="submit" value="확인">
</form>
```

Servlet 연습문제

- **학생 정보 출력**
  - HTML 문서에서 **학과, 학번, 이름** 입력받기
  - 입력된 정보를 서블릿으로 넘겨서 아래 <결과 화면>과 같이 출력하기
  - **student.html**
  - **Student 서블릿**에서 요청 처리
  - **get 방식**으로 요청하기



Servlet 연습문제

- **학생 정보 출력**
  - HTML 문서에서 **학과, 학번, 이름** 입력받기
  - 입력된 정보를 서블릿으로 넘겨서 아래 <결과 화면>과 같이 출력하기
  - **student.html**

```
<body>
    <form action="Student" method="get">
        <p>학과 : <input type="text" name="major"></p>
        <p>학번 : <input type="text" name="stnum"></p>
        <p>이름 : <input type="text" name="name"></p>
        <p><input type="submit" value="확인"></p>
    </form>
</body>
```

Servlet 연습문제

- **학생 정보 출력**
  - HTML 문서에서 **학과, 학번, 이름** 입력받기
  - 입력된 정보를 서블릿으로 넘겨서 아래 <결과 화면>과 같이 출력하기
  - **student.html**
  - **Student 서블릿**에서 요청 처리

```
@WebServlet("/Student")
public class Student extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Student() {
        super();
    }
    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        String major = request.getParameter("major");
        String stnum = request.getParameter("stnum");
        String name  = request.getParameter("name");

        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h3>학과 : " + major + "</h3>");
        out.println("<h3>학번 : " + stnum + "</h3>");
        out.println("<h3>이름 : " + name + "</h3>");
        out.println("</body></html>");
    }
```

Servlet 연습문제

- **구구단 출력**
  - HTML 문서에서 **출력할 구구단 숫자를 선택**한다.
  - 선택한 숫자를 서블릿으로 넘겨서 **해당 숫자의 구구단을 출력**한다.

Servlet 연습문제

- 구구단 출력
  - HTML 문서에서 출력할 구구단 숫자를 선택한다
  - 선택한 숫자를 서블릿으로 넘겨서 해당 숫자의 구구단을 출력한다

< gugu.html >

```
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>Insert title here</title></head>
<body>  <h3>구구단 선택</h3>
    <form action="Gugu" method="get">
        숫자 : <select name="number">
                <option>2</option>
                <option>3</option>
                <option>4</option>
                <option>5</option>
                <option>6</option>
                <option>7</option>
                <option>8</option>
                <option>9</option>
              </select>   <br><br>
        <input type="submit" value="확인">
    </form>
</body></html>
```

Servlet 연습문제

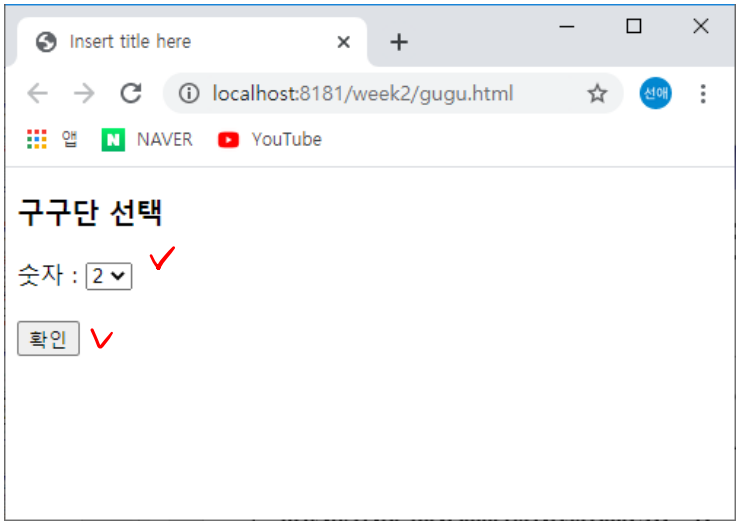
- 구구단 출력
  - HTML 문서에서 출력할 구구단 숫자를 선택한다
  - 선택한 숫자를 서블릿으로 넘겨서 해당 숫자의 구구단을 출력한다

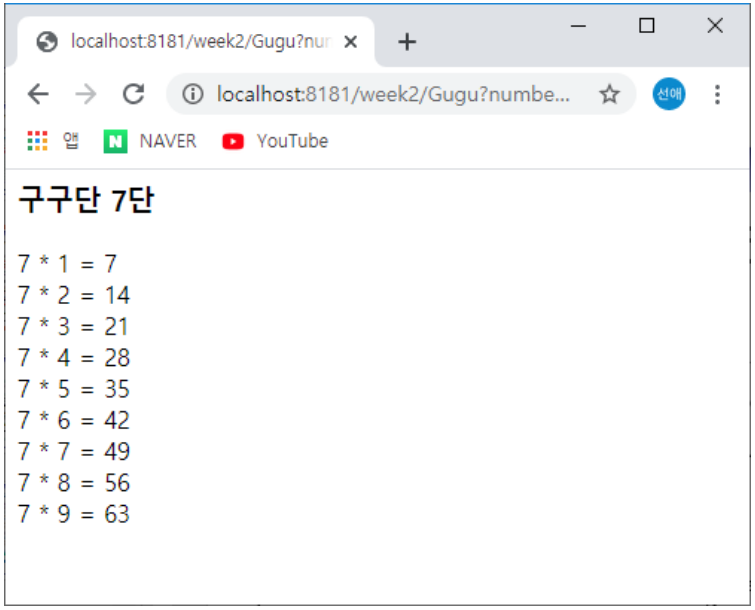
< Gugu.java >

```
@WebServlet("/Gugu")
public class Gugu extends HttpServlet {
  private static final long serialVersionUID = 1L;
  public Gugu() {  super();  }
   protected void doGet(HttpServletRequest request, HttpServletResponse response)
                                                    throws ServletException, IOException {
            response.setContentType("text/html;charset=utf-8");
            int num = Integer.parseInt(request.getParameter(“number”));
            PrintWriter out = response.getWriter();
            out.println("<html><body>");
            out.println(“<h3> 구구단 “ + num + “단</h3>”);
            for (int i=1; i<=9; i++) {
               out.println(num + “ * “ + i + “ = “ + num*i + “<br>”);
            }
            out.println("</body></html>");
            out.close();
            }}
```

Servlet 연습문제

- 구구단 출력
  - HTML 문서에서 출력할 구구단 숫자를 선택한다
  - 선택한 숫자를 서블릿으로 넘겨서 해당 숫자의 구구단을 출력한다





JSP 개요

- **JSP는**
  - **서블릿의 단점을 보완**하기 위한 **스크립트 방식**의 **표준 기술**
  - 서버 쪽 모듈을 개발하기 위한 기술
- **JSP 기본 구조**
  - **HTML 문서 사이**에 **Java 문법의 코드가 삽입**되는 형태로 작성

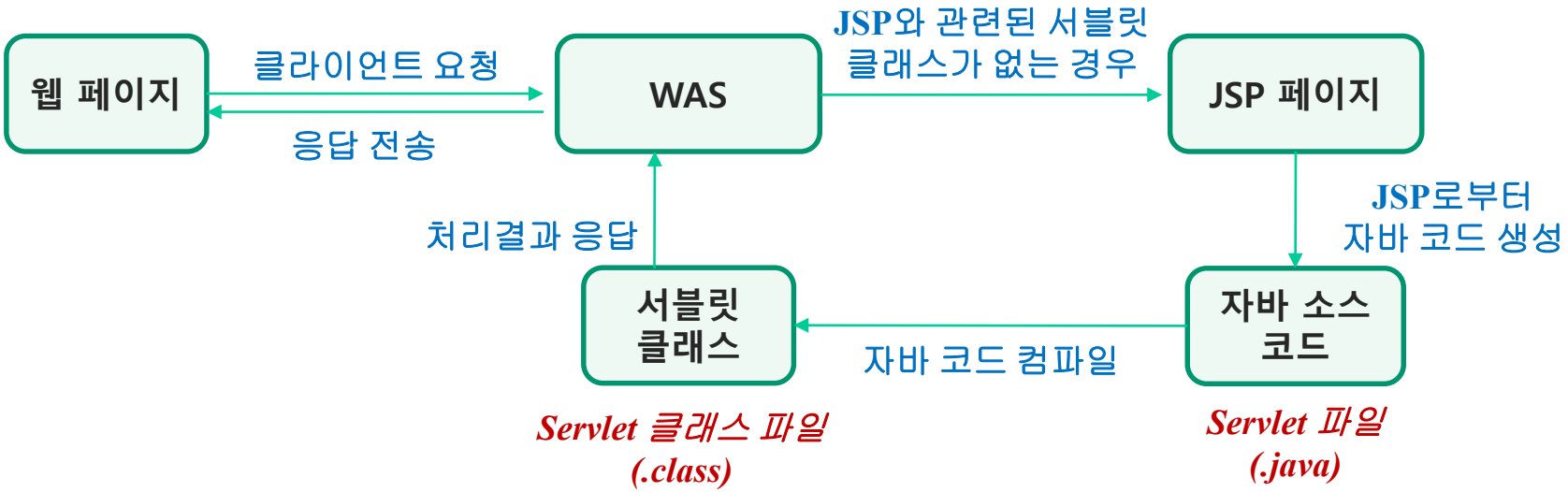
```
<%@page contentType= ”text/html; charset=euc-kr ”%>
<HTML>
        <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
        <BODY>
                <%
                        int total = 0;
                        for (int cnt = 1; cnt <= 100; cnt++)
                                total += cnt;
                %>
                1부터 100까지 더한 값은? <%= total %>
    </BODY>
</HTML>
```

HTML 문서 사이에
JSP 문법의 코드가 삽입

- **JSP 실행**
  - JSP 페이지에 있는 **HTML 코드**는 **웹 브라우저로 그대로 전송**
  - **JSP 문법의 코드**는 **웹 컨테이너** 쪽에서 **실행**되고 그 결과만 웹 브라우저로 전송

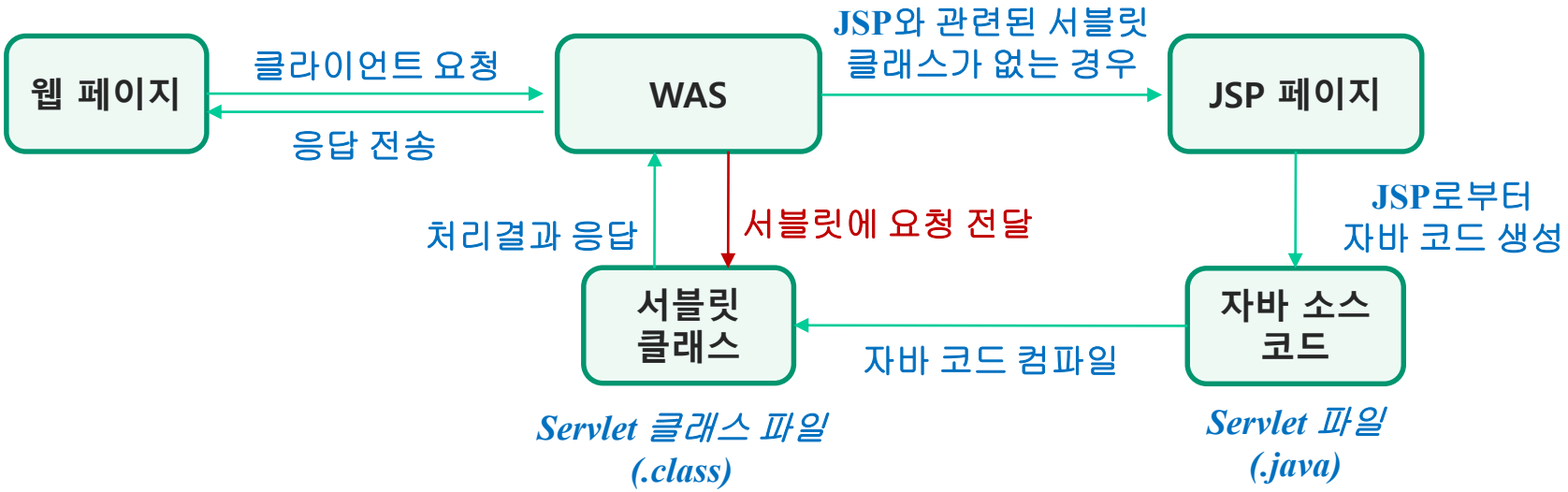
JSP 처리 과정

- WAS는 **JSP 페이지에 대한 요청**이 들어오면 다음과 같이 처리



JSP 처리 과정

- WAS는 **JSP 페이지에 대한 요청**이 들어오면 다음과 같이 처리



JSP 기본 구조

- **<!DOCTYPE ……> 이전**
  - **JSP 페이지에 대한 정보**를 이용하는 설정 부분
  - JSP 페이지가 생성하는 문서의 타입 및 사용할 커스텀 태그, 자바 클래스 등을 지정

- **<!DOCTYPE ……> 이후**
  - **문서를 생성하는 부분**
  - **<% … %>** 등의 **기호를 이용**하여 스크립트 코드 작성

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>

</body>
</html>
```

JSP 구성 요소

- **디렉티브(Directive)** – 지시자
- **스크립트**
  - 스크립틀릿(scriptlet)
  - 표현식(expression)
  - 선언부(declaration)

JSP 구성 요소

- **디렉티브(Directive)** – 지시자

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>

</body>
</html>
```

지시자(directive)

- <%@ 로 시작해서 %>로 끝난다.
- JSP 페이지에 대한 정보를 지정한다.
- JSP가 생성하는 문서의 타입, 출력 버퍼의 크기, 에러 페이지 등
- JSP 페이지에서 필요로 하는 정보를 설정한다.

JSP 구성 요소

- **스크립트 요소**
  - **스크립틀릿(scriptlet)**
  - 표현식(expression)
  - 선언부(declaration)

▪ <% 로 시작해서 %>로 끝난다.
▪ JSP 페이지에서 **JAVA 코드**를 실행할 때 사용

< sciptlet >

```
<%@  page contentType= ”text/html; charset=euc-kr ”%>
<HTML>
      <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
      <BODY>
            <%
                  int total = 0;
                  for (int cnt = 1; cnt <= 100; cnt++)
                        total += cnt;
                  %>

            1부터 100까지 더한 값은? <%= total %>
   </BODY>
</HTML>
```

Java 명령문

JSP 구성 요소

- **스크립트**
  - 스크립틀릿(scriptlet)
  - **표현식(expression)**
  - 선언부(declaration)

- **<%=** 로 시작해서 **%>**로 끝난다.
- 사이에 **자바식**이 들어갈 수 있다.
- 상수나 변수이름 하나로 구성될 수도 있다.
- 연산자를 포함할 수도 있다.
- 리턴값이 있는 메소드 호출이 가능하다.

< expression >

```
<%@page contentType= ”text/html; charset=euc-kr ”%>
<HTML>
      <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
      <BODY>
            <%
                  int total = 0;
                  for (int cnt = 1; cnt <= 100; cnt++)
                        total += cnt;
                  %>

            1부터 100까지 더한 값은? <%= total %>
   </BODY>
</HTML>
```

- 표현식(expression)
- 다음 표현도 가능하다.

```
<%= total + 100 %>
        or
<%= Math.sqrt(total) %>
```

JSP 구성 요소

- **스크립트**
  - 스크립틀릿(scriptlet)
  - 표현식(expression)
  - **선언부(declaration)**

- **<%!** 로 시작해서 **%>**로 끝난다.
- 사이에 **자바 메소드** 작성이 가능하다.
- 선언부의 함수는 자바 메소드 문법 구조와 동일하다.

< 선언부 >

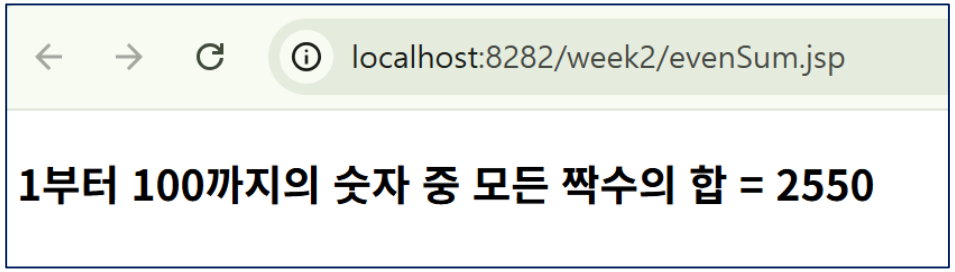
```
<%@page contentType= ”text/html; charset=euc-kr ”%>
<HTML>
      <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
      <BODY>
      <%!
            public int sum(int x, int y) {
               return x+y;
            }
      %>
            <h3> 두 수의 합 구하기</h3>
            <%=  sum(10, 15) %>
   </BODY>
</HTML>
```

선언부(declaration)

JSP 실습

- **스크립틀릿 & 표현식** 실습
- **1부터 100까지의 숫자** 중 **모든 짝수의 합** 구하기

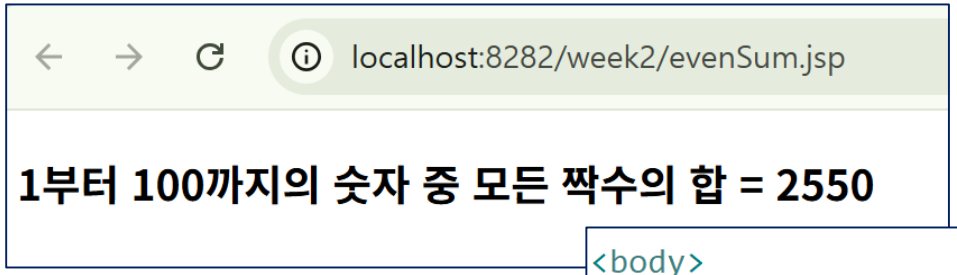
< evenSum.jsp >



JSP 실습

- **스크립틀릿 & 표현식** 실습
- **1부터 100까지의 숫자** 중 **모든 짝수의 합** 구하기

< evenSum.jsp >

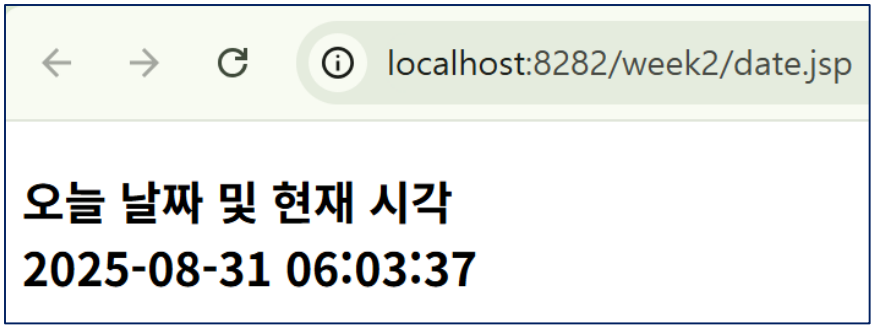



```jsp
<body>
    <%
        int total = 0;

        for (int i=1; i<=100; i++) {
            if (i%2 != 0)
                continue;
            total += i;
        }

    %>

    <h3>1부터 100까지의 숫자 중 모든 짝수의 합 = <%=total %></h3>

</body>
```

JSP 실습

- **스크립틀릿 & 표현식** 실습
- 오늘의 날짜와 현재 시각 출력하기

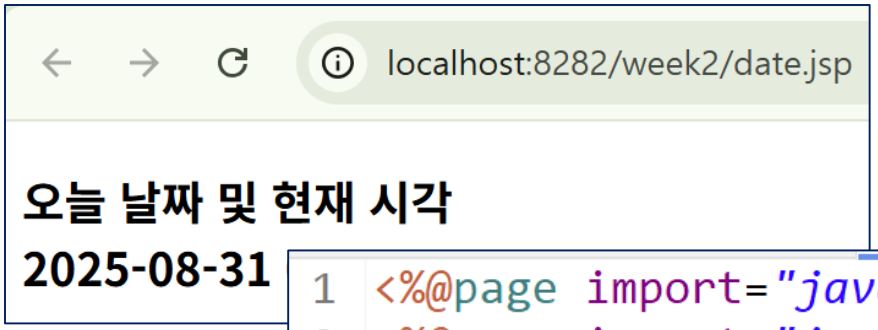
< date.jsp >



JSP 실습

- **스크립틀릿 & 표현식** 실습
- 오늘의 날짜와 현재 시각 출력하기

< date.jsp >



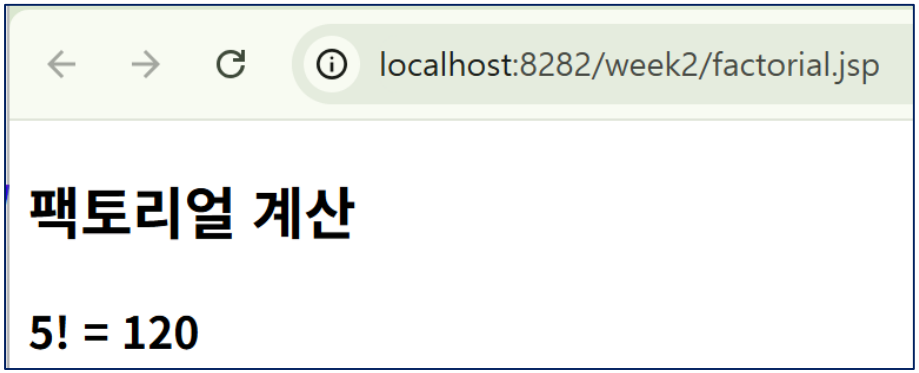

```jsp
1 <%@page import="java.text.SimpleDateFormat"%>
2 <%@page import="java.util.Date"%>
```

```jsp
<body>
    <%
        Date date = new Date();
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd hh:ss:mm");

        String dateStr = sdf.format(date);
    %>
    <h3>오늘 날짜 및 현재 시각 <br> <%= dateStr %> </h3>
</body>
```

JSP 실습

- **선언부 & 표현식** 실습
- 팩토리얼 계산 함수 이용하기

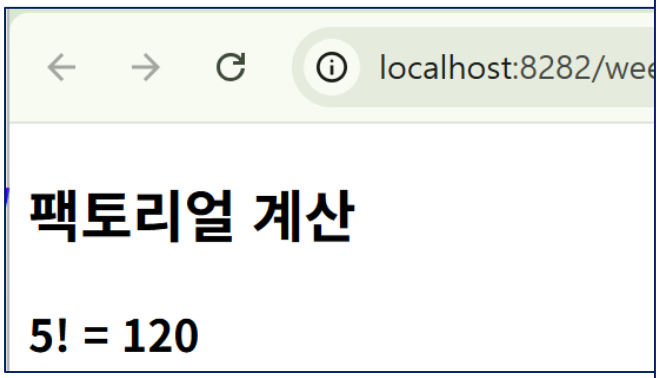
< factorial.jsp >



JSP 실습

- **선언부 & 표현식** 실습
- 팩토리얼 계산 함수 이용하기

< factorial.jsp >

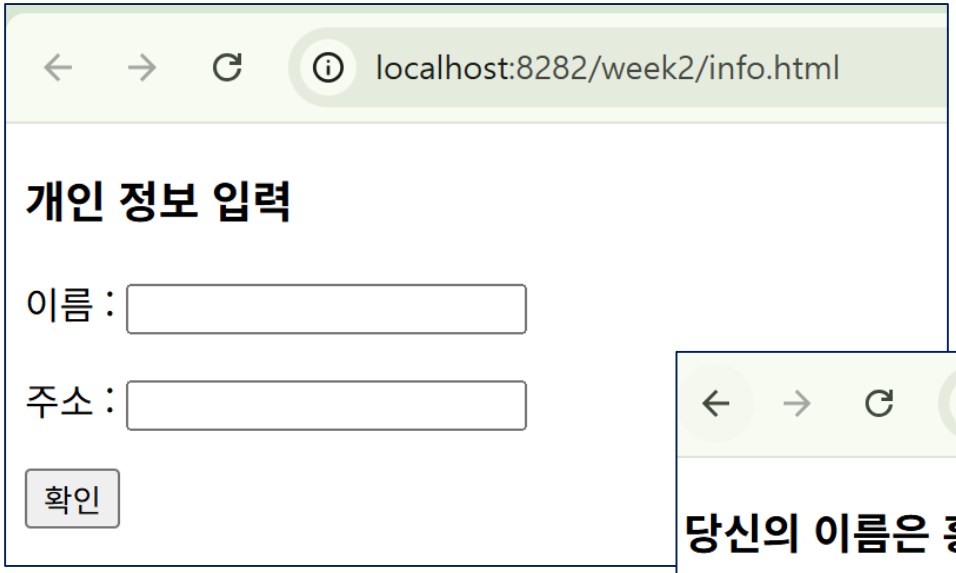



```jsp
<body>
    <%!
        //선언부 : 팩터리얼 계산 함수 선언
        int factorial(int n) {
            int result = 1;

            for (int i=n; i>1; i--) {
                result *= i;
            }
            return result;
        }
    %>
    <h2>팩토리얼 계산</h2>
    <h3>5! = <%=factorial(5) %></h3>
</body>
```

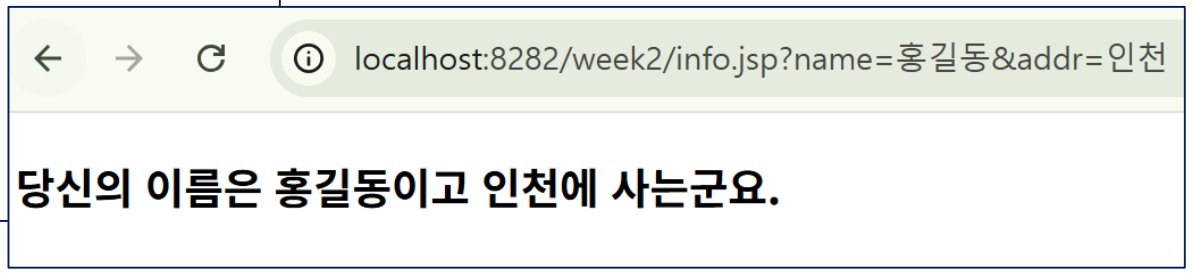
JSP 실습

- 개인 정보를 입력 받는다.
- 입력받은 정보를 넘겨받아 화면에 출력한다.

< info.html >



< info.jsp >



JSP 실습

- 개인 정보를 입력 받는다.
- 입력받은 정보를 넘겨받아 화면에 출력한다.

< info.html >

```
<body>

    <h3>개인 정보 입력</h3>
    <form action="info.jsp" method="get">
        <p>이름 : <input type="text" name="name"></p>
        <p>주소 : <input type="text" name="addr"></p>
        <p><input type="submit" value="확인"></p>
    </form>
</body>
```

JSP 실습

- 개인 정보를 입력 받는다.
- 입력받은 정보를 넘겨받아 화면에 출력한다.

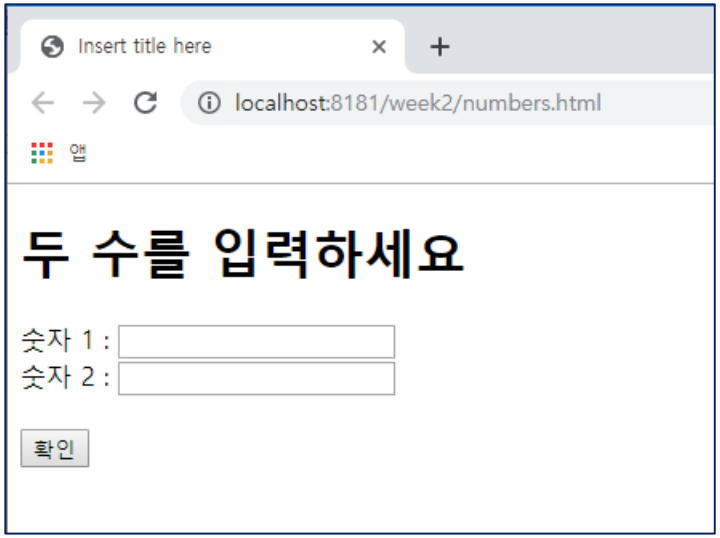
< info.jsp >

```
<body>
    <%
        String name = request.getParameter("name");
        String addr = request.getParameter("addr");
    %>

    <h3>당신의 이름은 <%=name %>이고 <%=addr %>에 사는군요.</h3>

</body>
```

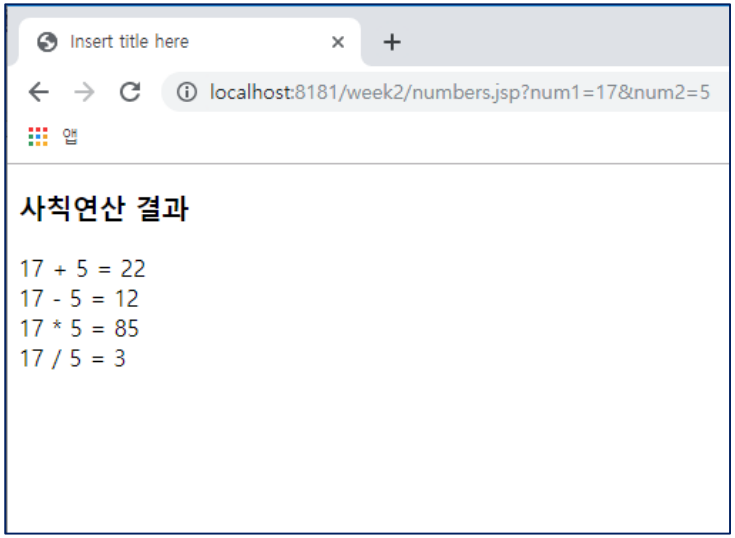
JSP 실습

- 화면에서 두 수를 입력 받는다.
- 입력 받은 두 수를 이용하여 사칙연산 결과를 출력한다.

< numbers.html >



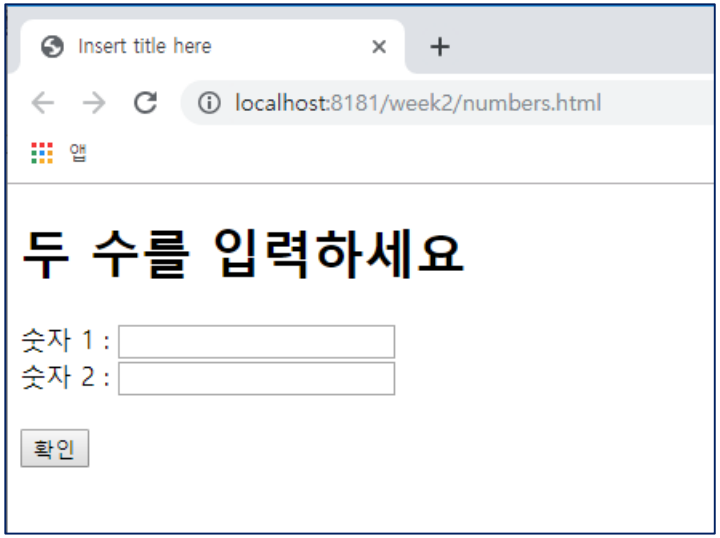
< numbers.jsp >



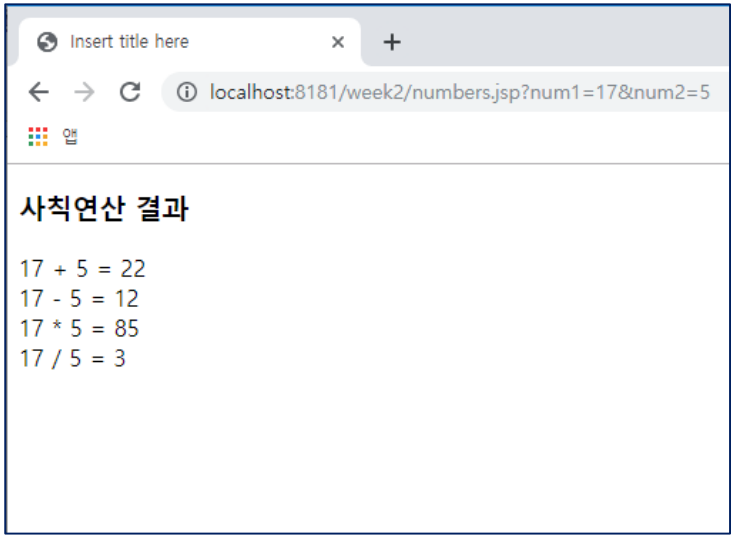
JSP 실습

- 화면에서 두 수를 입력 받는다.
- 입력 받은 두 수를 이용하여 사칙연산 결과를 출력한다.

< numbers.html >



< numbers.jsp >



문자형으로 넘어오는 데이터를 숫자로 변환

int num1 = Integer.parseInt(request.getParameter("num1"));

JSP 실습 2

```
1  <!DOCTYPE html><html><head><meta charset="UTF-8">
2  <title>Insert title here</title>
3  </head>
4  <body>
5      <h1>두 수를 입력하세요</h1>
6      <form action="numbers.jsp">
7          숫자 1 : <input type="text" name="num1"> <br>
8          숫자 2 : <input type="text" name="num2"> <br> <br>
9          <input type="submit" value="확인">
10     </form>
11 </body></html>
```

< numbers.html >

JSP 실습 2

< numbers.html >

```html
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h1>두 수를 입력하세요</h1>
    <form action="numbers.jsp">
        숫자 1 : <input type="text" name="num1"> <br>
        숫자 2 : <input type="text" name="num2"> <br> <br>
        <input type="submit" value="확인">
    </form>
</body></html>
```

< numbers.jsp >

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>Insert title here</title></head>
<body>
<%
    int num1 = Integer.parseInt(request.getParameter("num1"));
    int num2 = Integer.parseInt(request.getParameter("num2"));
%>
    <h3>사칙연산 결과</h3>
    <%=num1%> + <%=num2%> = <%=num1+num2%><br>
    <%=num1%> - <%=num2%> = <%=num1-num2%><br>
    <%=num1%> * <%=num2%> = <%=num1*num2%><br>
    <%=num1%> / <%=num2%> = <%=num1/num2%><br>
</body></html>
```