



Servlet & JSP

Servlet

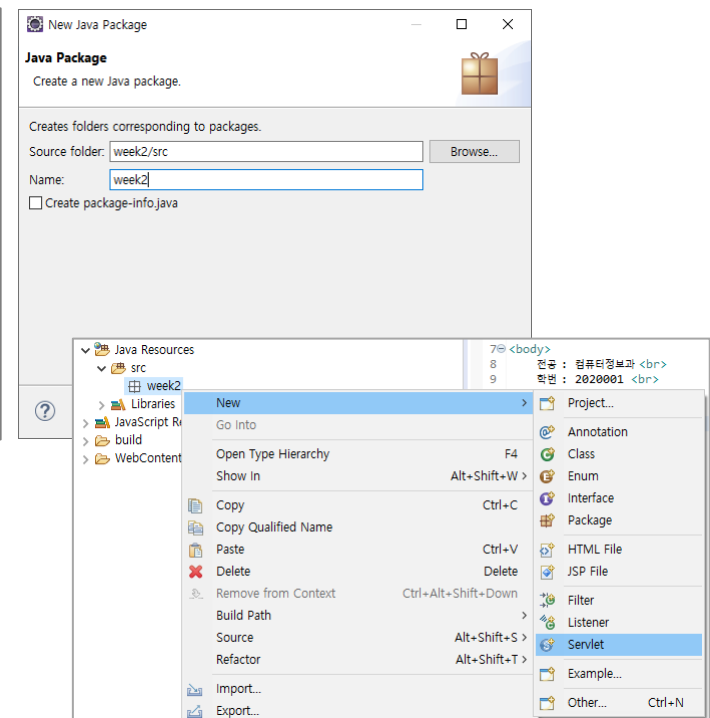
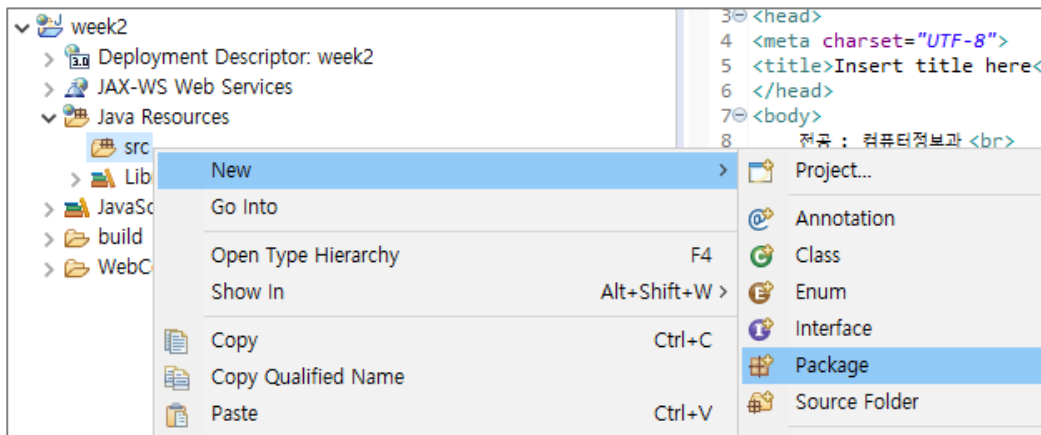
- 서블릿은
 - JSP 표준이 나오기 전에 만들어진 표준
 - Java에서 웹 어플리케이션을 개발할 수 있도록 하기 위해 만들어짐
 - Java 클래스를 웹에서 호출 및 실행할 수 있도록 한 표준
- 작성방법
 - `Javax.servlet.http.HttpServlet` 클래스로부터 상속받아서 작성
 - 위 클래스는 톰캣의 `servlet-api.jar`에 포함되어 있음
- 작성과정
 - 서블릿 규칙에 따라 자바 코드를 작성
 - 자바 코드를 컴파일해서 클래스 파일을 생성
 - 서블릿 3.0부터는 `@WebServlet` 어노테이션을 사용
 - 톰캣 등의 웹 컨테이너에서 실행

Servlet

- 서블릿 요청처리
 - 요청방식에 따라 doGet 이나 doPost 메소드를 재정의해서 처리
 - Service 메소드를 재정의해서 사용할 수도 있다.
- 세부사항
 - 서블릿 요청처리를 위해 오버라이딩 한 메소드는 request 객체를 이용해서 웹 브라우저의 요청 정보를 읽어온다.
 - 요청에 대한 응답을 전송할 때는 response 객체를 이용한다.
 - response 객체의 setContentType() 메소드를 이용해서 데이터 타입과 인코딩 방식을 지정해 준다.
 - 웹 브라우저에 데이터를 전송할 경우, PrintWriter 객체의 getWriter() 메소드를 호출해서 문자열 데이터를 출력한다.

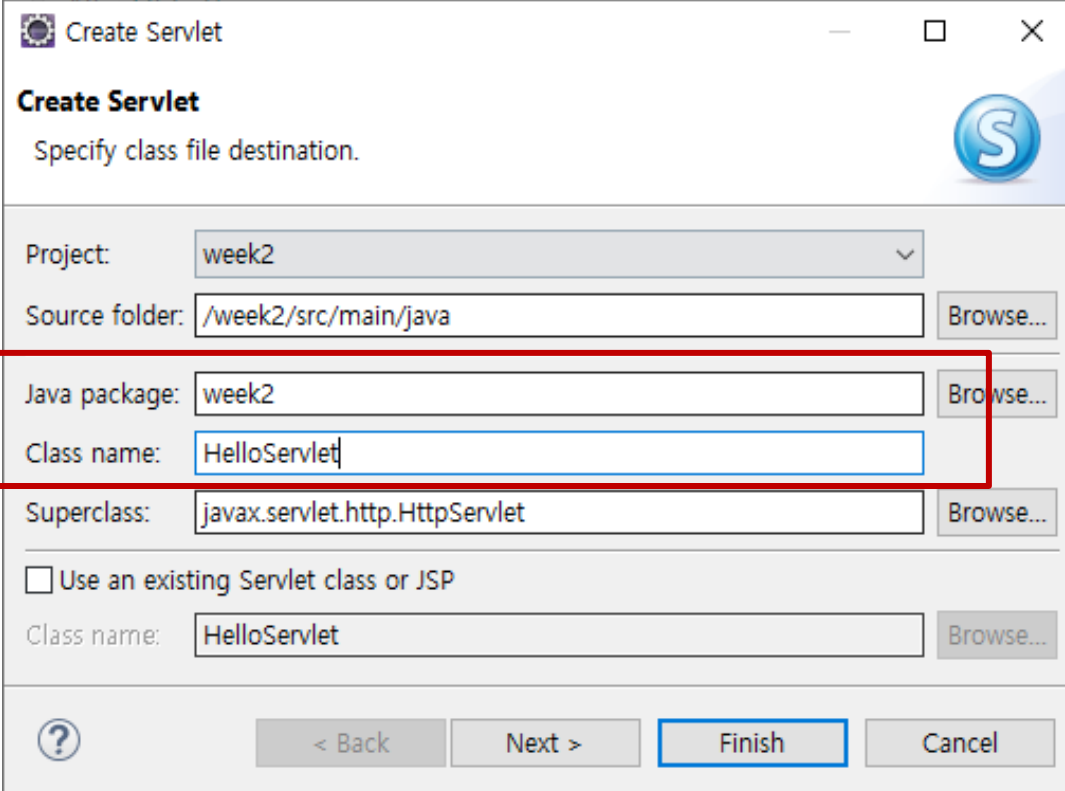
Servlet 예제 1

- 새로운 프로젝트 생성
 - File > New > Dynamic Web Project 선택 > week2 생성
 - 반드시 Build Path 실행
 - 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



The image shows the 'Create Servlet' dialog box in an IDE. The dialog has a title bar with a gear icon and the text 'Create Servlet'. Below the title bar, there's a section titled 'Create Servlet' with the instruction 'Specify class file destination.' and a blue 'S' icon. The main area contains several input fields and buttons:

- Project:** A dropdown menu showing 'week2'.
- Source folder:** A text field containing '/week2/src/main/java' with a 'Browse...' button to its right.
- Java package:** A text field containing 'week2' with a 'Browse...' button to its right. This field and the 'Class name' field below it are highlighted with a red rectangle.
- Class name:** A text field containing 'HelloServlet'.
- Superclass:** A text field containing 'javax.servlet.http.HttpServlet' with a 'Browse...' button to its right.
- ☐ **Use an existing Servlet class or JSP**
- Class name:** A text field containing 'HelloServlet' with a 'Browse...' button to its right.

At the bottom, there are four buttons: a help button (question mark icon), '< Back', 'Next >', and 'Finish' (which is highlighted with a blue border). A 'Cancel' button is also present.

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")
public class HelloServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
                                                throws ServletException, IOException
    {
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h1>Hello Servlet</h1>");
        out.println("</body></html>");
        out.close();
    }

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

Servlet 호출 방법

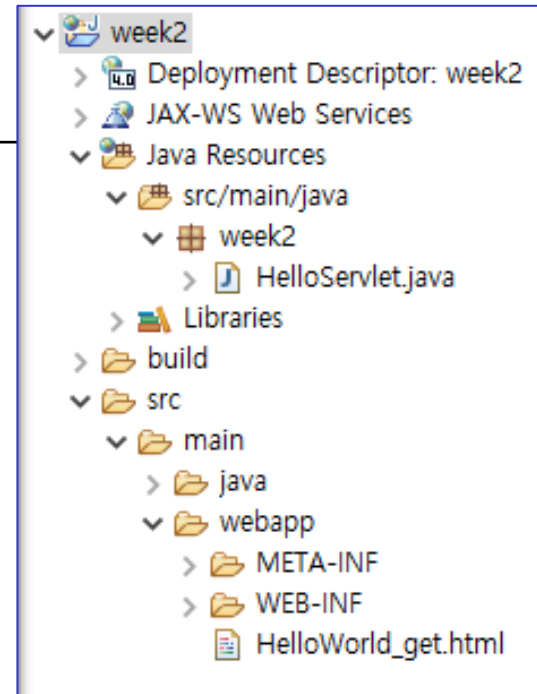
- **Get 방식** : 주소에 매개변수를 붙여서 호출하는 방식
 - 주소와 매개변수를 붙여서 주소 표시줄에 입력하는 방법(?로 구분)
 - <a> 태그를 이용해서 페이지를 요청하는 경우
 - 자바 스크립트를 이용해서 요청하는 경우
 - <form> 태그에서 명시적으로 GET 방식으로 요청하는 경우
 - 매개변수의 데이터는 255자 이내이며 보안이 취약함
- **Post 방식** : 매개변수를 본문에 포함시켜 호출하는 방식
 - <form> 태그에서 명시적으로 POST 방식으로 요청하는 경우
 - 데이터의 크기에 제한이 없다.
 - URL에 표시가 되지 않으므로 보안성이 우수하다.

Servlet 호출 예제-1

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

< HelloWorld_get.html >

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</title></head>
<body>
  <form action="HelloWorld" method="get">
    <br>
    <h1>get 방식으로 부르는 페이지입니다</h1><br>
    <input type="submit" value="확인">
  </form>
</body>
</html>
```



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 {  
        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 { }  
  
}
```

Servlet 호출 예제-2

- Post 방식 : 매개변수를 본문에 포함시켜 호출하는 방식

< HelloWorld_post.html >

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</title></head>
<body>
  <form action="HelloWorld" method="post">
    <br>
    <h1>post 방식으로 부르는 페이지입니다</h1><br>
    <input type="submit" value="확인">
  </form>
</body>
</html>
```

Servlet 호출 예제-2

- Post 방식 : 매개변수를 본문에 포함시켜 호출하는 방식

< HelloWorld.java >

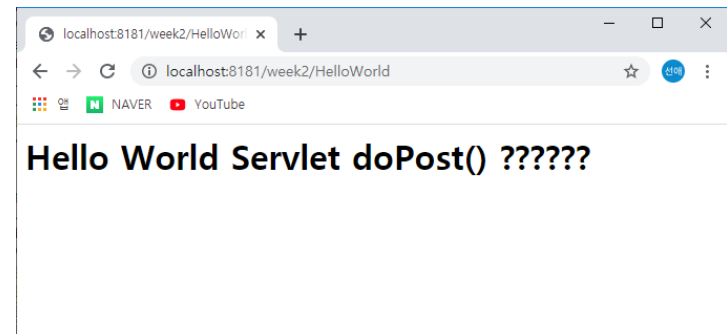
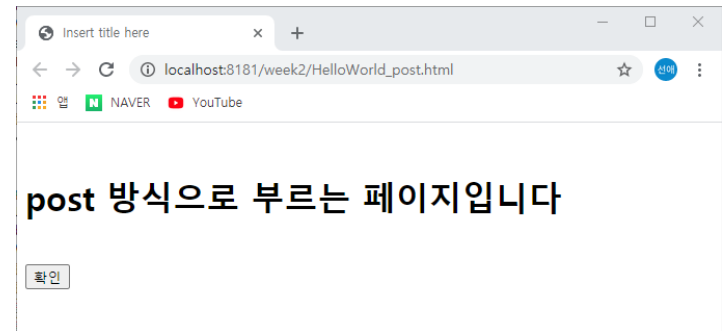
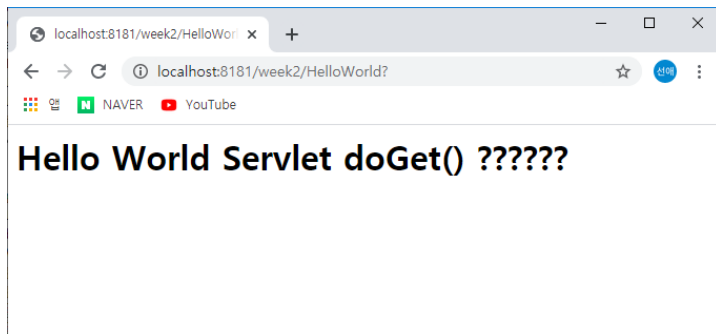
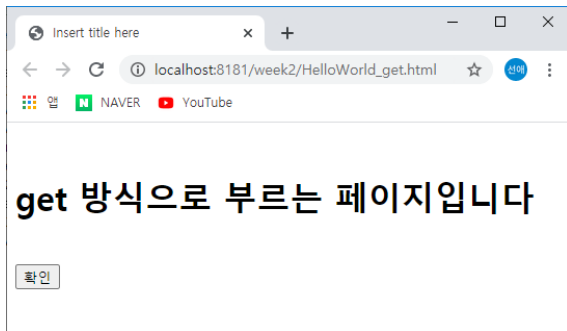
```
@WebServlet("/HelloWorld")
public class HelloWorld extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public HelloWorld () { super(); }

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

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
                                throws ServletException, IOException {
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h1>Hello World Servlet doPost() 페이지입니다</h1>");
        out.println("</body></html>");
        out.close(); }
}
```

Servlet 호출 예제-2

- 실행 결과



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 >

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</title></head>
<body>
  <form action="Address" method="get">
    이름 : <input type="text" name="name"><br>
    주소 : <input type="text" name="addr"><br>
    <input type="submit" value="확인">
  </form>
</body>
</html>
```

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 {
        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(name + "님은 " + addr + "에 사는군요");
        out.println("</body></html>"); out.close(); }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        doGet(request, response); }}
```

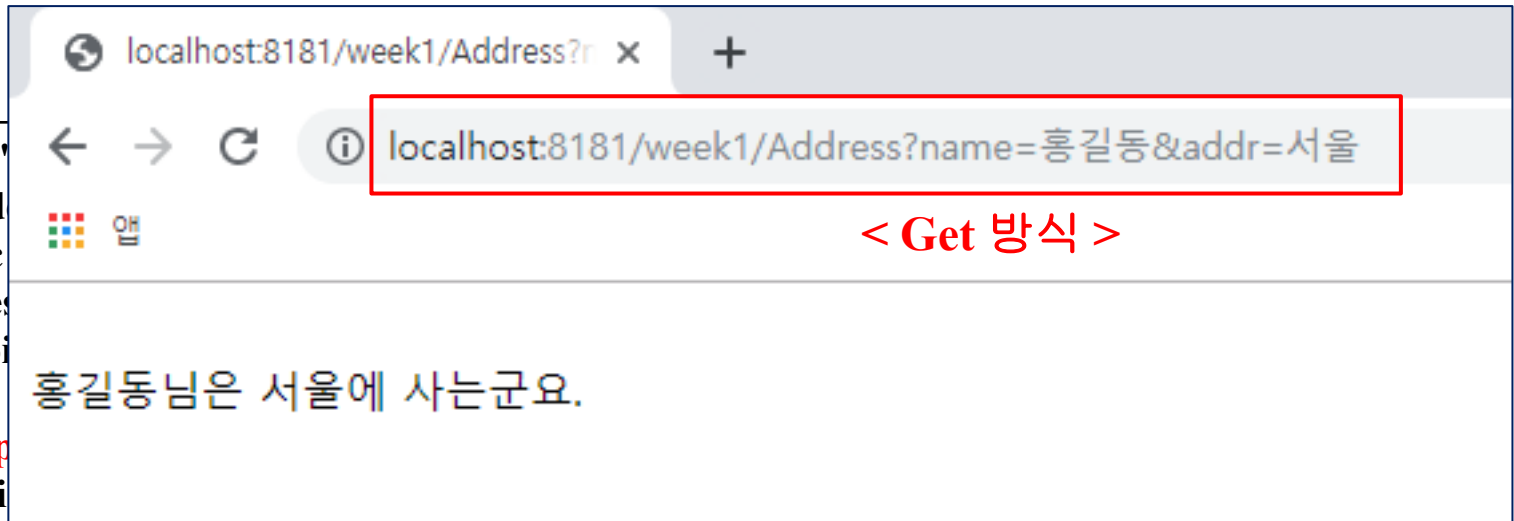
Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

```
@WebServlet("
public class Ad
private static
public Address
protected voi
```

```
resp
Stri
```

```
String addr = request.getParameter("addr");
PrintWriter out = response.getWriter();
out.println("<html><body>");
out.println(name + "님은 " + addr + "에 사는군요");
out.println("</body></html>"); out.close(); }
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
    doGet(request, response); }}
```



Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

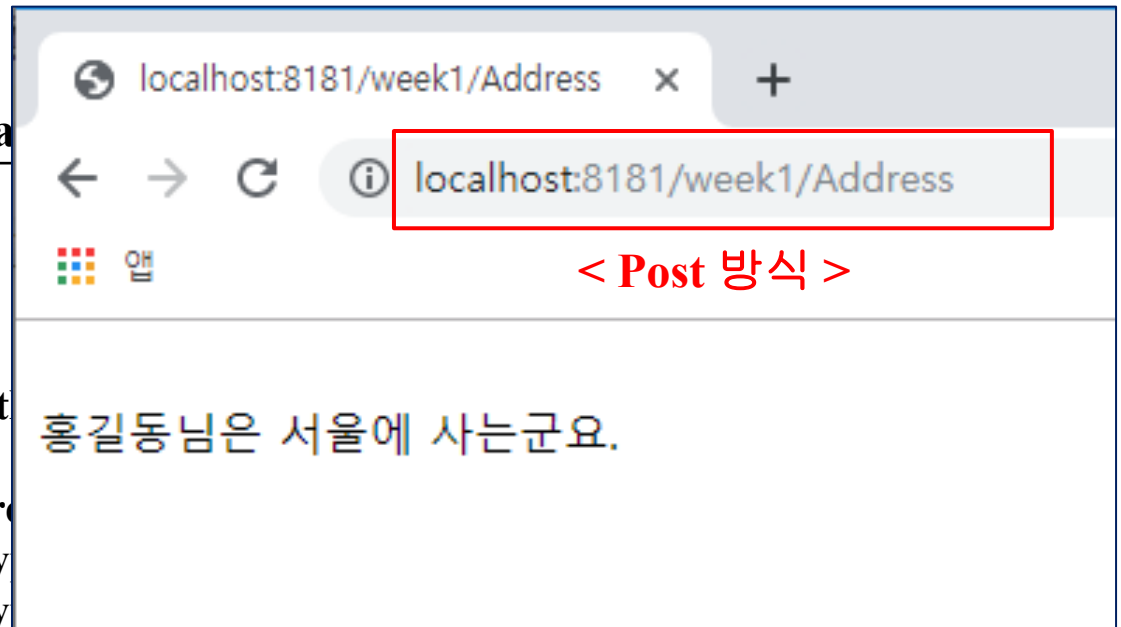
< addrPost.html >

```
<!DOCTYPE html>  
<html>  
<head>  
<meta charset="utf-8">  
<title>Insert title here</title></head>  
<body>  
    <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>  
</body>  
</html>
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

```
< a
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</tit
<body>
  <form action="Address"
    이름 : <input ty
    주소 : <input ty
    <input type="submit" value="확인">
  </form>
</body>
</html>
```



Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< addrGet.html >

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</title></head>
<body>
  <form action="Address" method="get">
    이름 : <input type="text" name="name"><br>
    주소 : <input type="text" name="addr"><br>
    <input type="submit" value="확인">
  </form>
</body>
</html>
```

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

해당 버튼을 클릭하면 서블릿이 요청된다

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

< addrGet.html >

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert title here</title></head>
<body>
  <form action="Address" method="get">
    이름 : <input type="text" name="name"><br>
    주소 : <input type="text" name="addr"><br>
    <input type="submit" value="확인">
  </form>
</body>
</html>
```

해당 변수명에 값이 저장되어 서블릿으로
데이터가 전송된다

전송되는 값의 데이터 타입
(text인 경우에는 String 타입으로 전송된다)

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

@WebServlet 어노테이션(annotation)으로 URL mapping

< 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 {
        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(name + "님은 " + addr + "에 사는군요");
        out.println("</body></html>"); out.close(); }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        doGet(request, response); }}
```

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

@WebServlet 어노테이션(annotation)으로 URL mapping

< 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 {
        response.setContentType("text/html");
        String name = request.getParameter("name");
        String addr = request.getParameter("addr");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println(name + "님은 " + addr + "에 사는군요");
        out.println("</body></html>"); out.close(); }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        doGet(request, response); }}

```

@WebServlet

- 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 주석형태이다

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

@WebServlet 어노테이션(annotation)으로 URL mapping

<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 {
        response.setContentType("text/html");
        String name = request.getParameter("name");
        String addr = request.getParameter("addr");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println(name + "님은 " + addr + "에");
        out.println("</body></html>"); out.close();
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        doGet(request, response); }
}
```

서블릿클래스명

※ 왜 URL Mapping 이 필요한가?

1. 실제 서블릿 클래스명을 공개하지 않기 위함이다
2. 프로그램 개발자는 자신이 만든 서블릿 클래스의 위치를 알아야 하지만
3. 클라이언트는 URL 을 입력하여 원하는 서비스만 얻으면 된다
4. 내부 경로나 파일명이 바뀌면 일일이 사용자에게 알려주어야 한다
5. URL 매핑을 이용하면 내부 구조가 바뀌더라도 매핑이름만 동일하면 URL 요청이 가능하다
6. 사용자에게 서블릿 구조와 파일명을 공개하는 것은 보안에 문제가 발생할 수 있다

Servlet 호출 예제-3

- HTML 문서에서 서블릿으로 데이터를 전송

<Address.java >

접근지정자는 반드시 public이어야 한다

```
@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 {
        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(name + "님은 " + addr + "에 사는군요");
        out.println("</body></html>"); out.close(); }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        doGet(request, response); }
}
```

HttpServlet 을 상속 받아야 한다

클라이언트에 응답할
페이지에 대한 환경설정

Post 방식인 경우,
입력되는 한글 인코딩 처리

Servlet 호출 예제-3

- HTML 문서에서 서블릿의

```
<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>
```

< 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 {
        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(name + "님은 " + addr + "에 사세요");
        out.println("</body></html>"); out.close(); }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        doGet(request, response); }}
```

- name 변수명에 값이 저장되어 넘어온다
- text 인 경우에는 String 타입으로 받는다

- addr 변수명에 값이 저장되어 넘어온다
- text 인 경우에는 String 타입으로 받는다

Servlet 연습문제

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

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

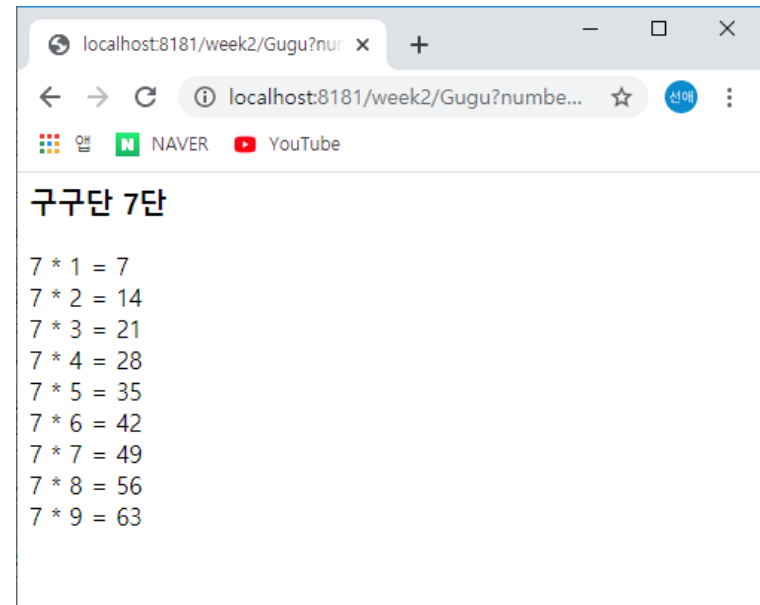
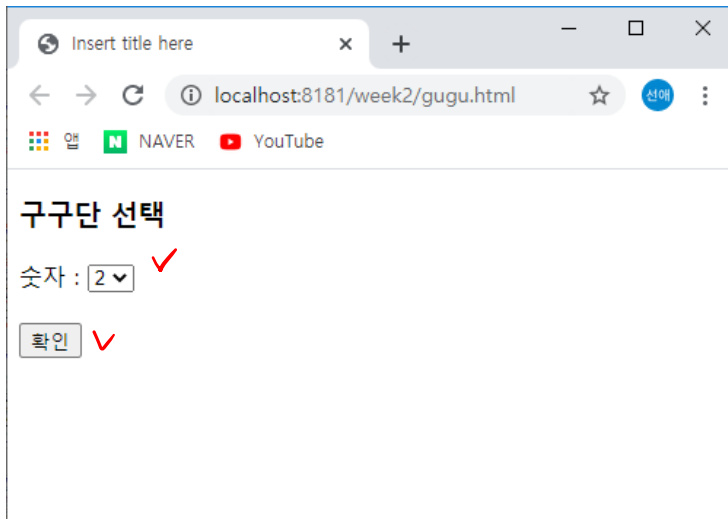
- 구구단 출력
 - HTML 문서에서 출력할 구구단 숫자를 선택한다
 - 선택한 숫자를 서블릿으로 넘겨서 해당 숫자의 구구단을 출력한다

< 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 보완하기 위한 스크립트 방식의 표준 기술
 - 서버 쪽 모듈을 개발하기 위한 기술
- JSP 기본 구조
 - HTML 문서 사이에 Java 문법의 코드가 삽입되는 형태로 작성된다

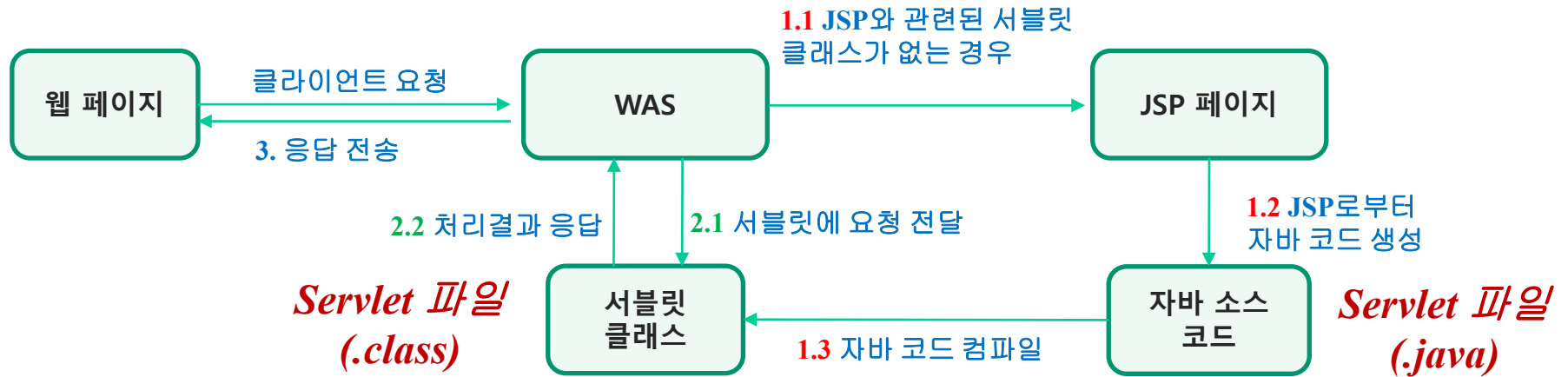
```
<%@page contentType= "text/html; charset=euc-kr" %>
<HTML>
  <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
  <BODY>
    <%
      int total = 0;
      for (int cnt = 1; cnt <= 100; cnt++)
        total += cnt;
    %>
    1부터 100까지 더한 값은? <%= total %>
  </BODY>
</HTML>
```

HTML 문서 사이에
JSP 문법의 코드가 삽입

- JSP 실행
 - JSP 페이지에 있는 HTML 코드는 웹 브라우저로 그대로 전송
 - JSP 문법의 코드는 웹 컨테이너 쪽에서 실행되고 그 결과만 웹 브라우저로 전송

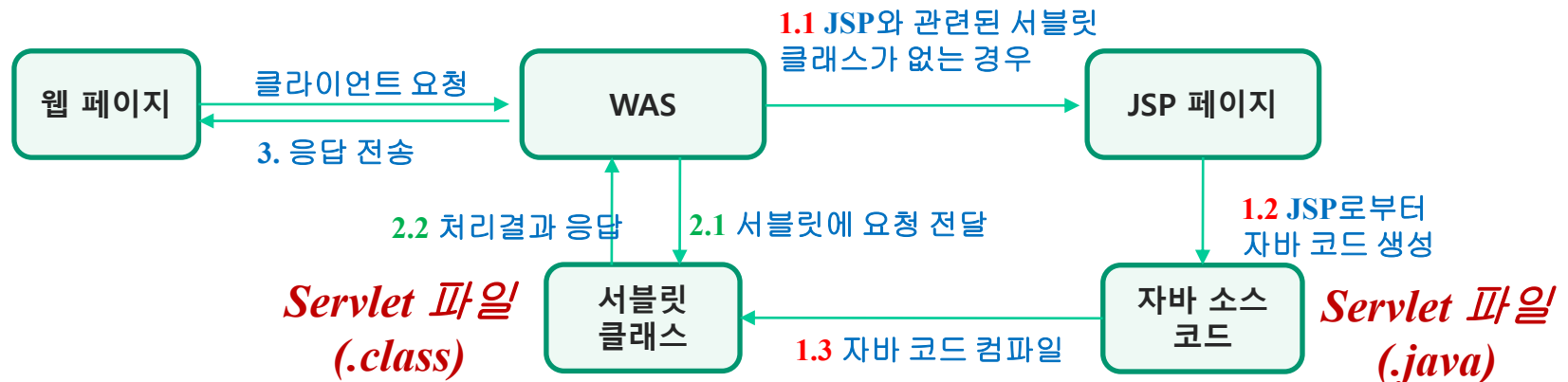
JSP 처리 과정

- WAS는 JSP 페이지에 대한 요청이 들어오면 다음과 같이 처리한다



JSP 처리 과정

- WAS는 JSP 페이지에 대한 요청이 들어오면 다음과 같이 처리한다
 - JSP에 해당하는 서블릿이 존재하지 않을 경우(과정 1.1)
 - JSP 페이지로부터 자바코드를 생성한다(과정 1.2)
 - 자바 코드를 컴파일해서 서블릿 클래스를 생성한다(과정 1.3)
 - 서블릿에 클라이언트 요청을 전달한다(과정 2.1)
 - 서블릿이 요청을 처리한 결과를 응답으로 생성한다(과정 2.2)
 - 응답을 웹 브라우저에 전송한다(과정 3)



JSP 처리 과정

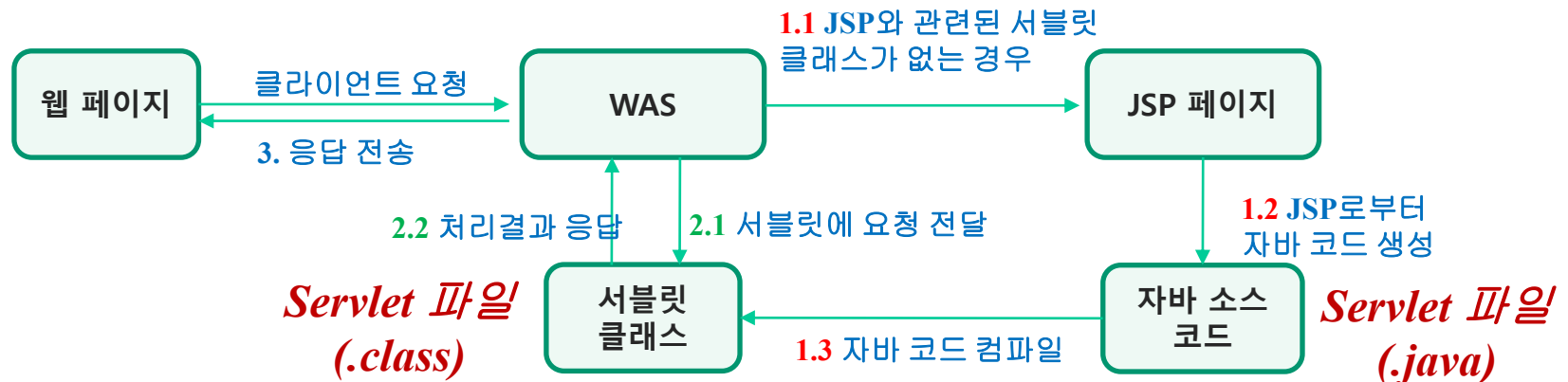
- WAS는 JSP 페이지에 대한 요청이 들어오면 다음과 같이 처리한다

- JSP에 해당하는 서블릿이 존재하지 않을 경우(과정 1.1)

- JSP 페이지로부터 자바코드를 생성한다(과정 1.2)
- 자바 코드를 컴파일해서 서블릿 클래스를 생성한다(과정 1.3)
- 서블릿에 클라이언트 요청을 전달한다(과정 2.1)
- 서블릿이 요청을 처리한 결과를 응답으로 생성한다(과정 2.2)
- 응답을 웹 브라우저에 전송한다(과정 3)

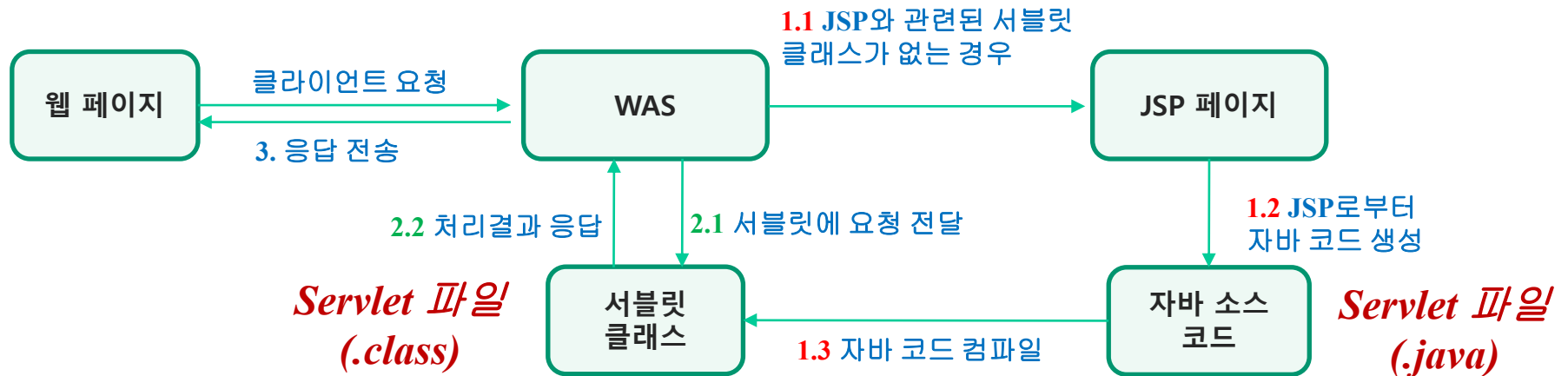
- JSP에 해당하는 서블릿이 존재하는 경우(1.1~1.3 과정을 이미 거친 경우)

- 서블릿에 클라이언트 요청을 전달한다(과정 2.1)
- 서블릿이 요청을 처리한 결과를 응답으로 생성한다(과정 2.2)
- 응답을 웹 브라우저에 전송한다(과정 3)



JSP 처리 과정

- WAS는 JSP 페이지에 대한 요청이 들어오면 다음과 같이 처리한다



생성된 자바 소스 코드 & 서블릿 클래스 위치 :

```
workspace\metadata\plugins\org.eclipse.wst.server.core\workspace\work\Catalina\localhost\week4\org.apache.jsp
```

JSP 기본 구조

- <!DOCTYPE> 이전
 - JSP 페이지에 대한 정보를 이용하는 설정 부분
 - JSP 페이지가 생성하는 문서의 타입 및 사용할 커스텀 태그, 자바 클래스 등을 지정
- <!DOCTYPE> 이후
 - 문서를 생성하는 부분
 - <% ... %> 등의 기호를 이용하여 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)
- 표현언어(Expression Language)
- 기본 객체
- 정적인 데이터
- 표준 액션 태그
- 커스텀 태그와 표준 태그 라이브러리(JSTL)

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 코드를 실행할 때 사용

< scriptlet >

```
<%@ 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 자바식이 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 >

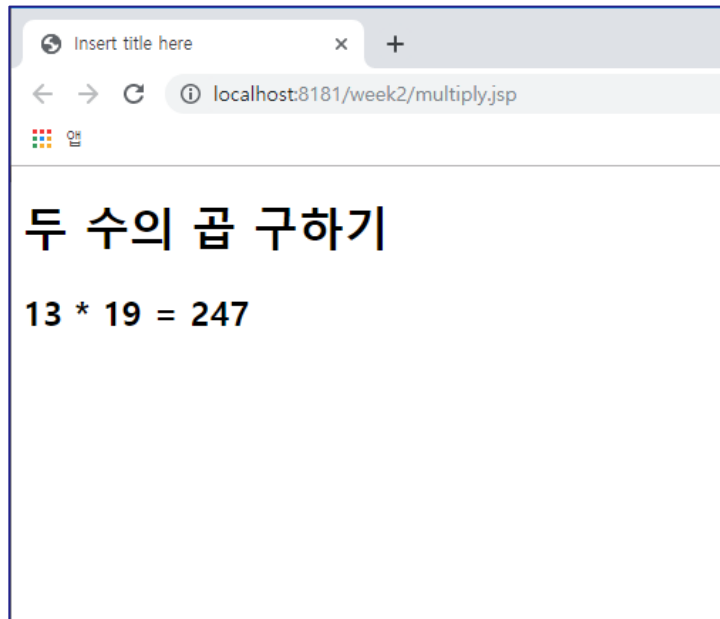
```
<%@page contentType= "text/html; charset=euc-kr" %>
<HTML>
  <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
  <BODY>
    <%!
      public int sum(int x, int y) {
        return x+y;
      }
    %>
    <h3> 두 수의 합 구하기</h3>
    <%= sum(10, 15) %>
  </BODY>
</HTML>
```

선언부(declaration)

JSP 실습

- 두 수($13 * 19$)의 곱을 선언부를 이용하여 출력

< multiply.jsp >



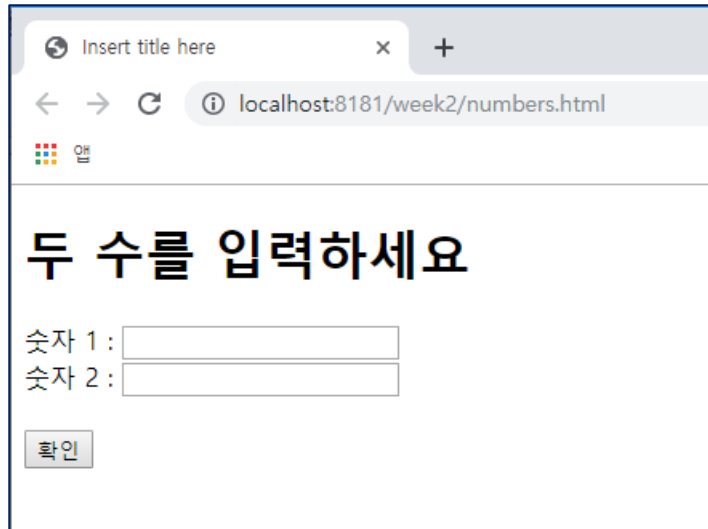
< multiply.jsp >

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html><html><head><meta charset="UTF-8">
4 <title>Insert title here</title></head>
5 <body>
6 <%!
7     public int multiply(int x, int y){
8         return x*y;
9     }
10 %>
11
12     <h1>두 수의 곱 구하기</h1>
13     <h2>13 * 19 = <%= multiply(13, 19) %></h2>
14 </body></html>
```

JSP 실습 2

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

< numbers.html >



Insert title here x +

localhost:8181/week2/numbers.html

앱

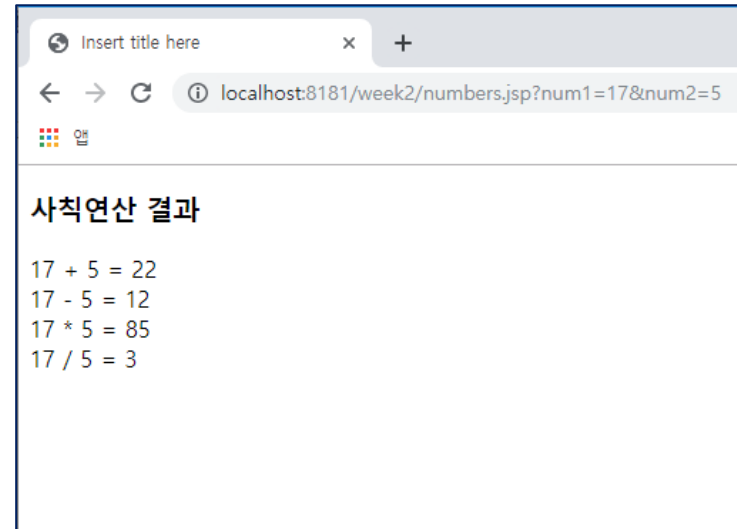
두 수를 입력하세요

숫자 1 :

숫자 2 :

확인

< numbers.jsp >



Insert title here x +

localhost:8181/week2/numbers.jsp?num1=17&num2=5

앱

사칙연산 결과

17 + 5 = 22

17 - 5 = 12

17 * 5 = 85

17 / 5 = 3

< numbers.html >

```
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>
```

JSP 실습 2

< numbers.html >

```
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>
```

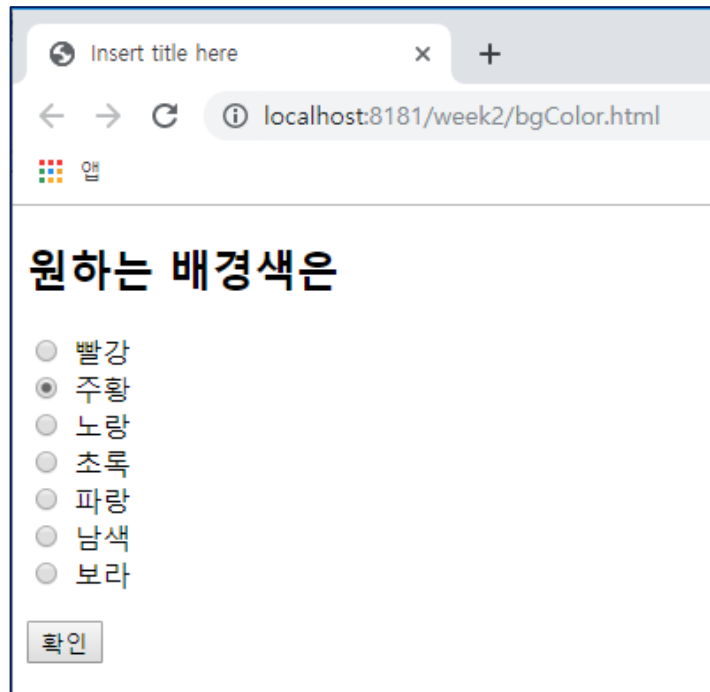
```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <!DOCTYPE html><html><head><meta charset="UTF-8">
4 <title>Insert title here</title></head>
5 <body>
6 <%
7     int num1 = Integer.parseInt(request.getParameter("num1"));
8     int num2 = Integer.parseInt(request.getParameter("num2"));
9 %>
10    <h3>사칙연산 결과</h3>
11    <%=num1%> + <%=num2%> = <%=num1+num2%><br>
12    <%=num1%> - <%=num2%> = <%=num1-num2%><br>
13    <%=num1%> * <%=num2%> = <%=num1*num2%><br>
14    <%=num1%> / <%=num2%> = <%=num1/num2%><br>
15 </body></html>
```

< numbers.jsp >

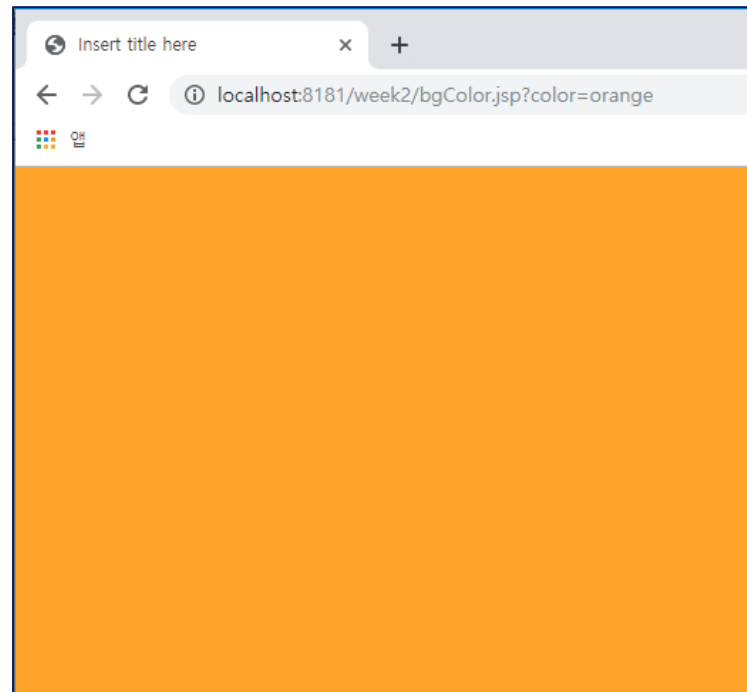
JSP 실습 3

- 웹 브라우저 배경색 바꾸기
- 무지개 색깔 중 하나를 선택해서 배경색을 바꾸도록 한다.

< bgColor.html >



< bgColor.jsp >



JSP 실습 3

< bgColor.html >

```
1<!DOCTYPE html><html><head><meta charset="UTF-8">
2 <title>Insert title here</title></head>
3<body>
4     <h2>원하는 배경색은</h2>
5     <form action="bgColor.jsp">
6         <input type="radio" name="color" id="c1" value="red" checked="checked">
7         <label for="c1">빨강</label><br>
8         <input type="radio" name="color" id="c2" value="orange">
9         <label for="c2">주황</label><br>
10        <input type="radio" name="color" id="c3" value="yellow">
11        <label for="c3">노랑</label><br>
12        <input type="radio" name="color" id="c4" value="green">
13        <label for="c4">초록</label><br>
14        <input type="radio" name="color" id="c5" value="blue">
15        <label for="c5">파랑</label><br>
16        <input type="radio" name="color" id="c6" value="navy">
17        <label for="c6">남색</label><br>
18        <input type="radio" name="color" id="c7" value="violet">
19        <label for="c7">보라</label><p>
20        <input type="submit" value="확인"></form>
21 </body></html>
```

< bgColor.jsp >

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3<!DOCTYPE html><html><head><meta charset="UTF-8">
4 <title>Insert title here</title></head>
5<%
6     String color = request.getParameter("color");
7 %>
8<body bgcolor="<%=color%>">
9
10 </body></html>
```