03장. JSP 기초

강사 강태광

JSP를 배우기 위한 필수 기술

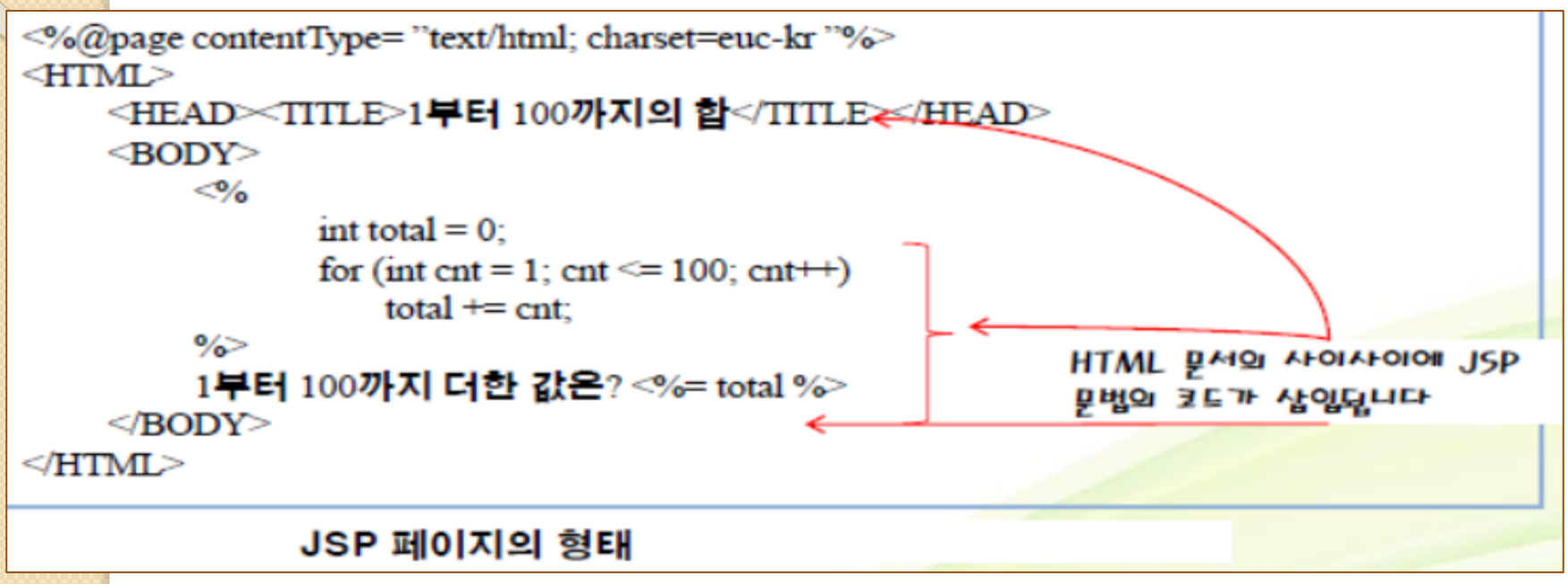
필수 기술	프로그램 경험	비 고
자바	· 자바 언어 기본 · 객체지향 개념 · 상속, 오버로딩, 오버라이딩 · 인터페이스 구현 · java.util, java.io 패키지 · 스레드 · 예외 핸들링	· 패키지와 클래스 이해 · 클래스 DOC을 참조하여 프로그래밍이 가능한 수준 · 자바 개발환경 설치 및 사용
JDBC	· JDBC 드라이버 세팅 · ResultSet · PreparedStatement · 데이터 핸들링 · 기초 SQL문	· 오라클, MySQL 등 원격지 데이터베이스 연결 처리 경험
서블릿	· 서블릿 구조 이해 · 간단한 서블릿 프로그래밍 · request, response 처리 · GET/POST 처리	· 서블릿 생명주기 이해

JSP를 배우기 위한 연관 기술

관련 기술	프로그램 경험	최소 요구사항
HTML	· HTML 기초 태그 사용 · FORM 관련 태그 사용	· 전용 편집기가 아닌 수작업으로 코딩이 가능한 수준 · CSS, 레이어 이해
자바스크립트	· 함수(Function) 만들기 · FORM 연계 · 이벤트 처리	· 자바스크립트 문법 이해 · 브라우저 객체 모델 이해
데이터베이스	· 다양한 SQL문의 사용 · 데이터베이스 연계 프로그래밍 경험 · 데이터베이스 함수 및 내장프로시저	· 테이블 생성 및 키에 대한 이해와 관계 설정
웹 프로그래밍	· 웹 서버 세팅 · CGI, ASP, PHP 등 웹 프로그래밍 경험	· 유닉스에서 웹 서버 세팅 경험
XML	· XML 스키마 및 DTD 이해 · XML DOM 개요	· 스키마와 DTD 기반의 XML 문서 작성 및 파싱 능력

JSP 문서의 기본 구조

1.JSP 기술에서 웹 애플리케이션을 구현 할 때 작성하는 코드를 JSP
2.JSP 페이지는 HTML 문서의 사이에 Java 문법의 코드가 삽입되는 형태로 작성
3.JSP 페이지에 있는 HTML 코드는 웹 브라우저로 그대로 전송되지만, JSP 문법의 코드는 웹 컨테이너 쪽에서 실행되고 그 결과만 웹 브라우저로 전송



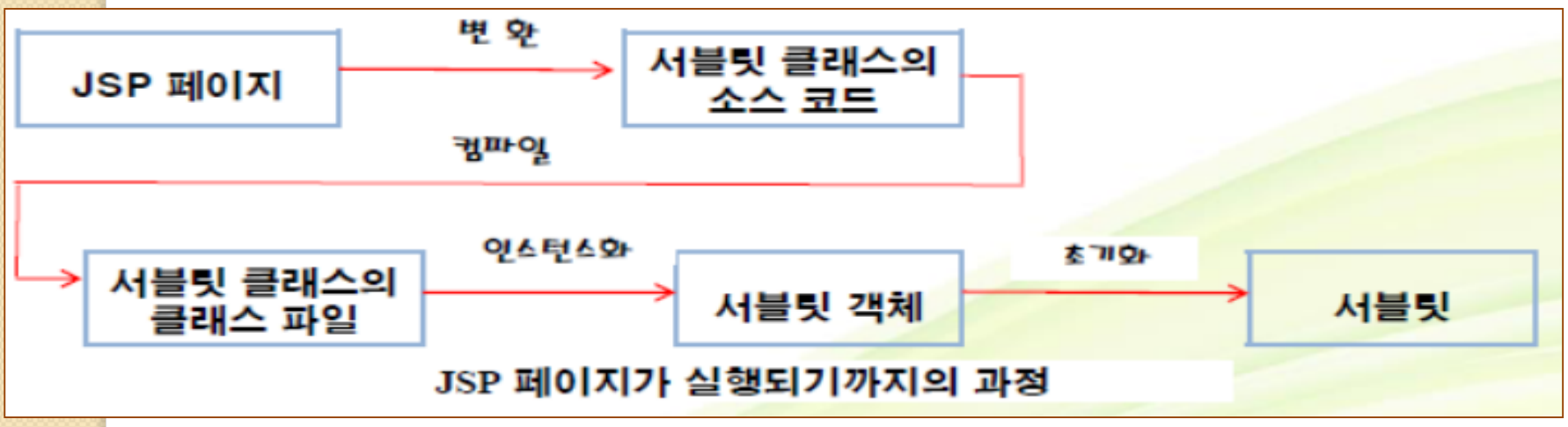

```
<%@page contentType= "text/html; charset=euc-kr "%>
<HTML>
    <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
    <BODY>
        <%
            int total = 0;
            for (int cnt = 1; cnt <= 100; cnt++)
                total += cnt;
        %>
        1부터 100까지 더한 값은? <%= total %>
    </BODY>
</HTML>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

JSP 페이지의 형태

<!DOCTYPE ...> 이전 까지는 JSP 페이지에 대한 정보를 이용하는 설정부분
JSP 페이지가 생성하는 문서의 타입 및 사용할 커스텀 태그, 사용할 자바 클래스를 지정하는 부분
<!DOCTYPE ..> 이후 부분은 문서를 생성하는 생성부분
문서의 데이터와 문서를 생성하는데 필요한 스크립트 코드를 작성하는 부분
<% ... %> 부분이 스크립트 코드

Servlet 수행 Process

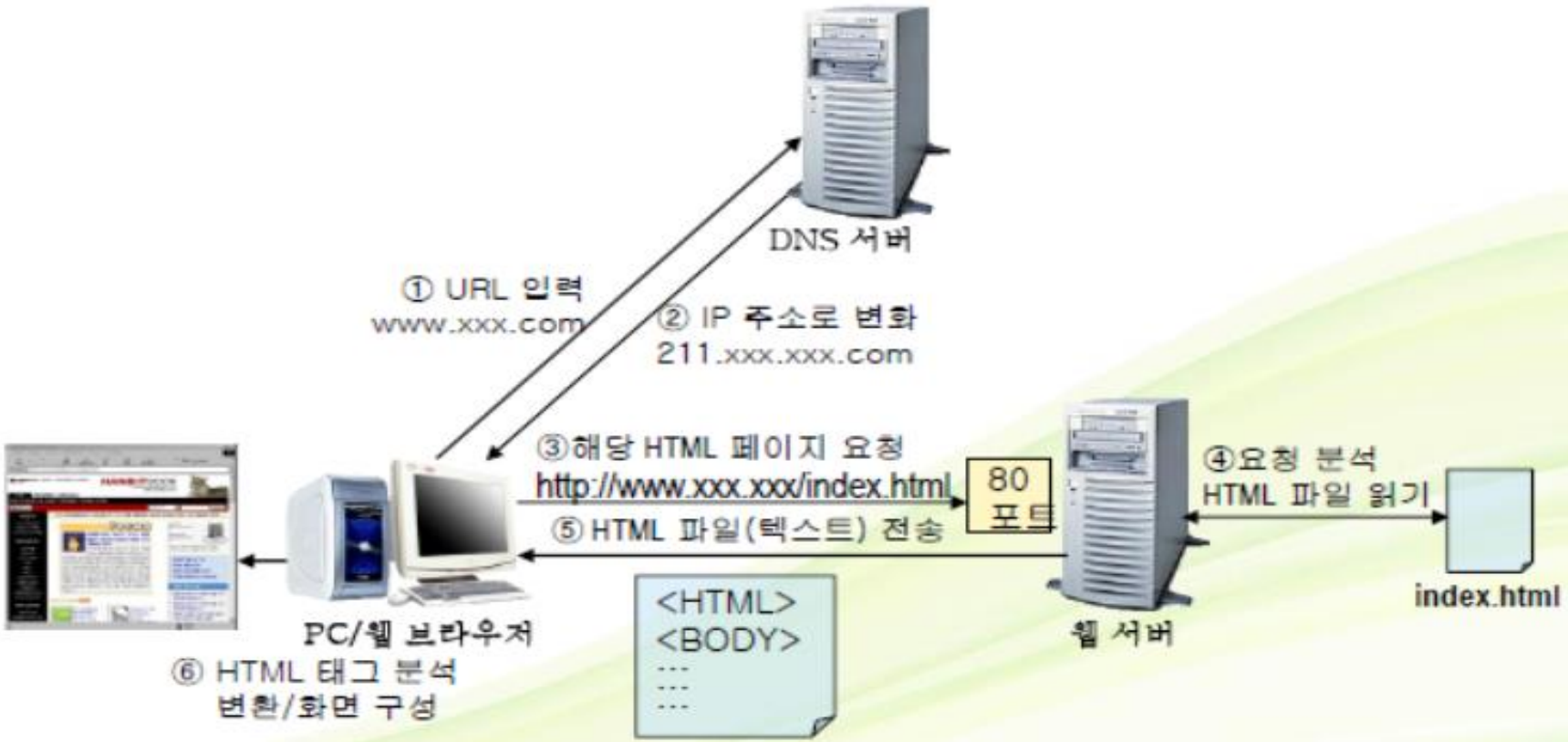
1.웹 컨테이너는 JSP 페이지 전체를 서블릿 클래스의 소스 코드로 변환한 다음에, 그 소스 코드를 컴파일해서 그 결과를 가지고 서블릿 객체를 만들고, 그 서블릿 객체를 초기화해서 서블릿을 만든다.
웹 브라우저로부터 URL이 왔을 때 실행되는 것은 서블릿이다



JSP 페이지가 실행되기까지의 과정

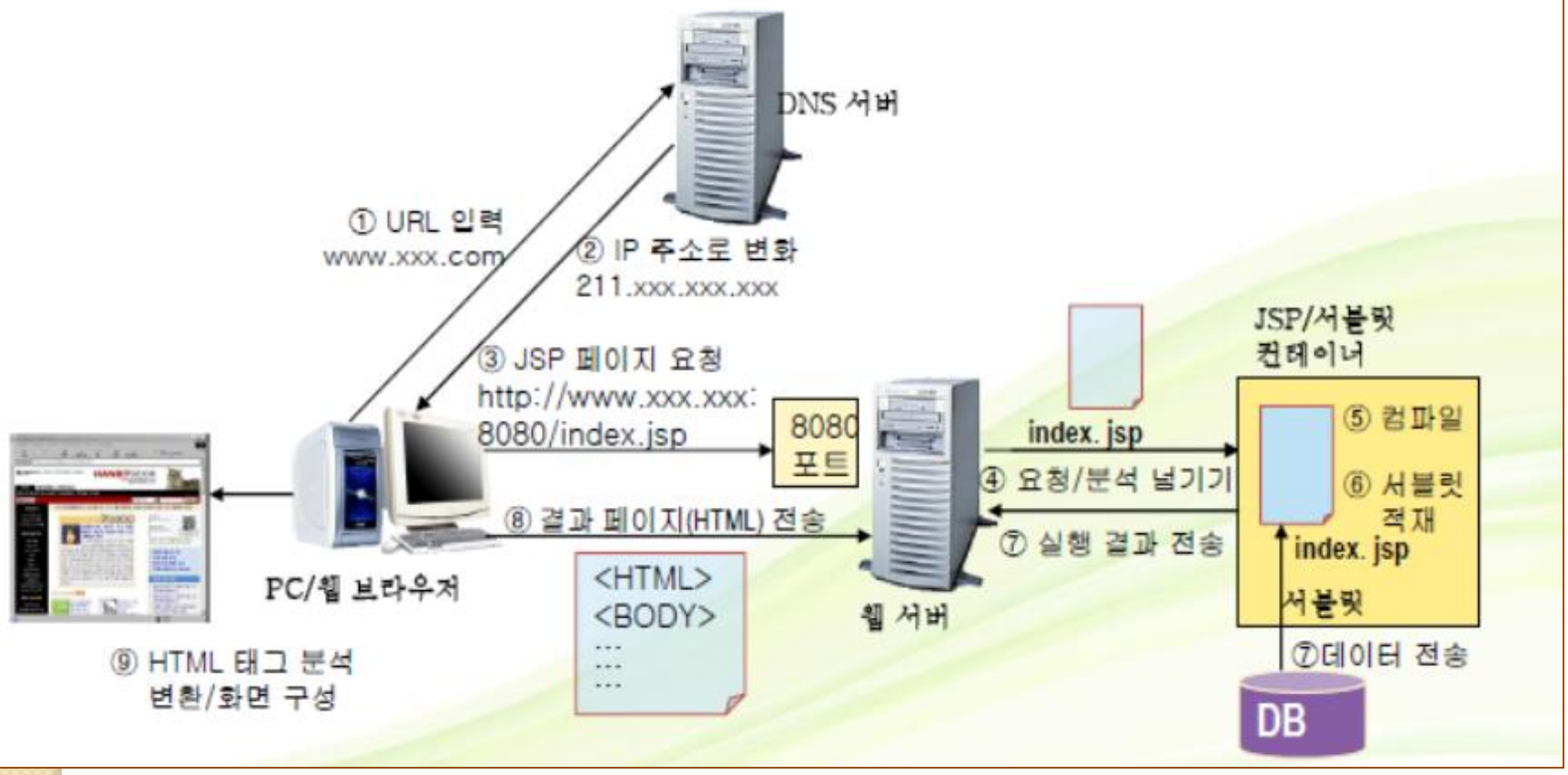
JSP 동작원리

- 일반적인 웹(www) 서비스 동작과정



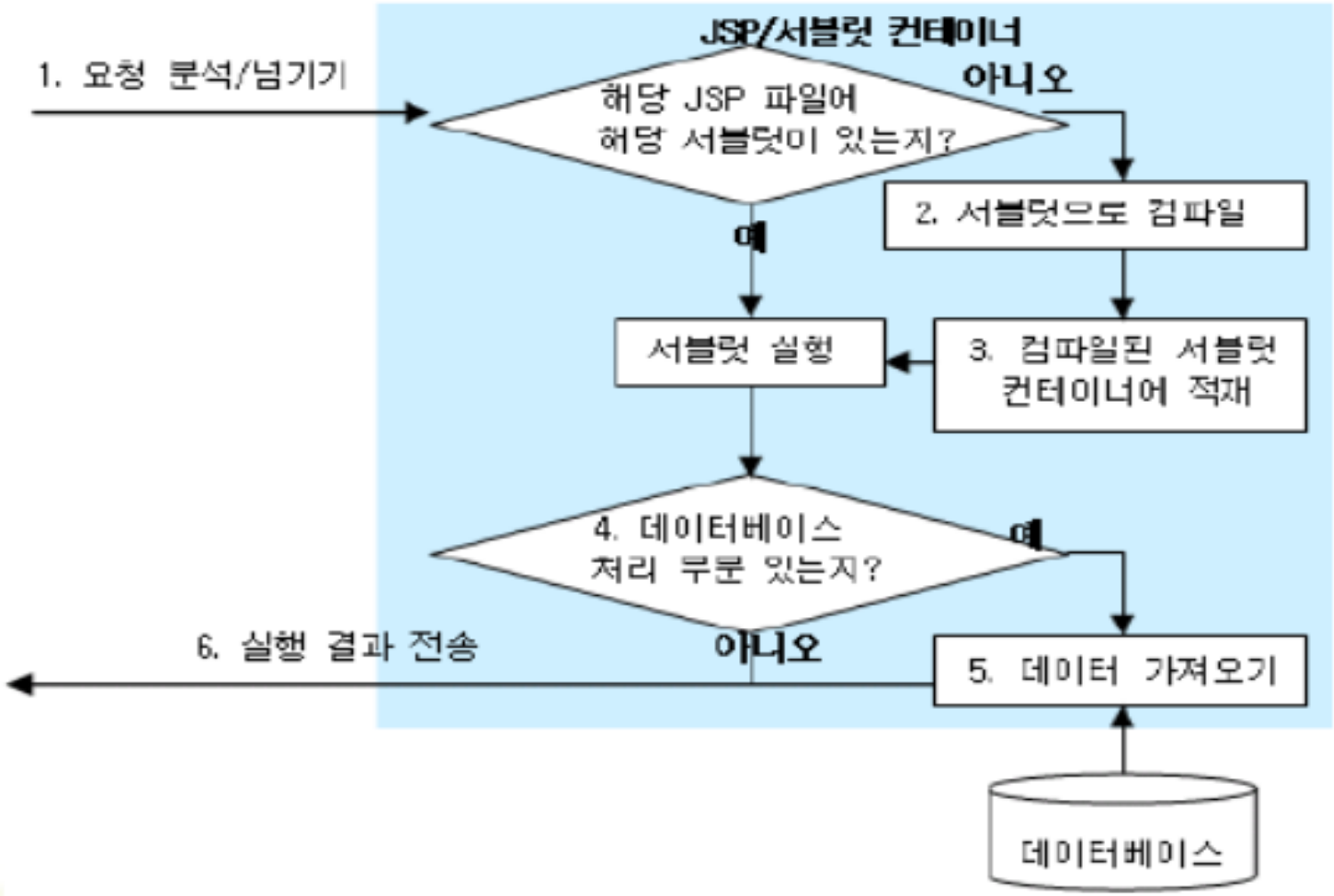
JSP 동작원리

- JSP 동작과정



JSP 동작원리

- JSP 서블릿 변환 처리 및 데이터 연동 과정

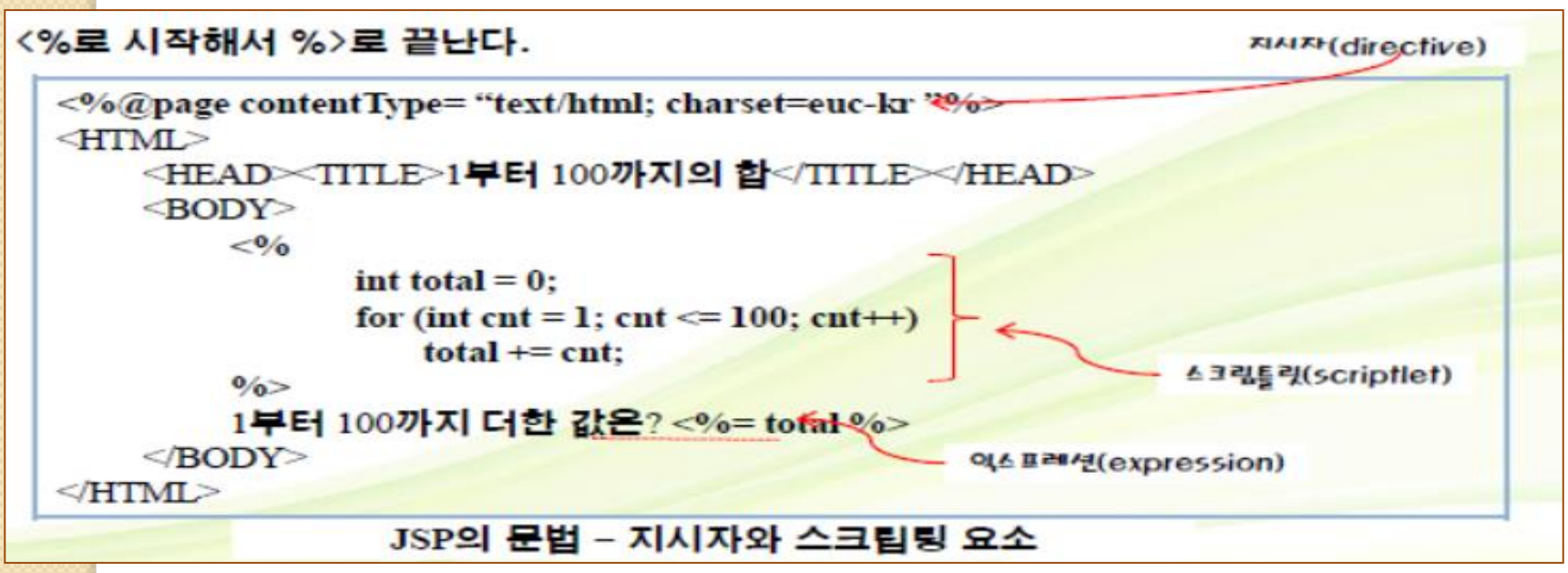


JSP의 기초 문법(스크립팅 요소, 지시자, 주석)

1)스크립틀릿(scriptlet)
- 스크립틀릿(scriptlet)은 <%로 시작해서 %> 로 끝나고, 그 사이 자바 명령문이 들어갈 수 있다

2)익스프레션(expression)

3)선언부(declaration)

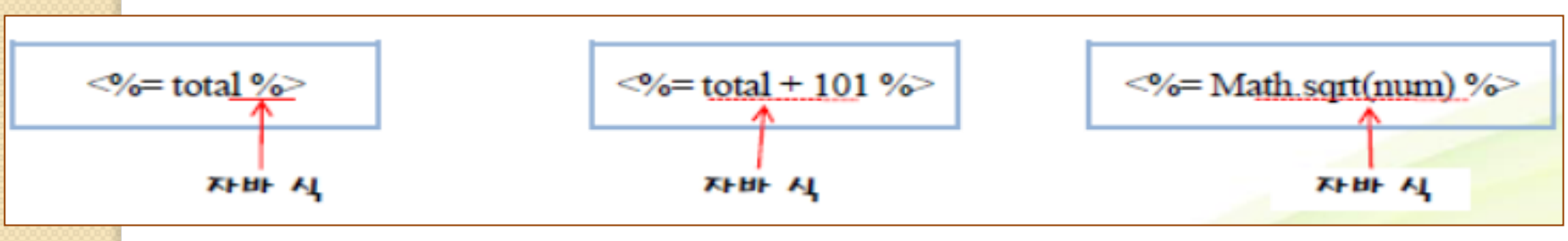


JSP의 문법 – 지시자와 스크립팅 요소

- JSP의 문법에는 세가지 형태가 있다.
  - <%로 시작해서 %>로 끝나는 형태 -> 스크립틀릿(scriptlet)
  - ${로 시작해서}로 끝나는 형태 : 익스프레션 언어(expression language)
  - <jsp:forward>와 같은 XML : 태그 형태 액션(action)

스크립팅 요소(scripting elements)

2)익스프레션(expression)
- <%=로 시작해서 %>로 끝나고 그 사이에 자바 식이 들어갈 수 있다.
  - 이 식은 상수나 변수 이름 하나로 구성될 수도 있고, 연산자를 포함할 수도 있으며, 리턴 값이 있는 메서드 호출식이 될 수도 있다.
  - 이 식은 웹 서버 쪽에서 실행되고 그 결과만 웹 브라우저로 전송된다



JSP 문서의 구성 요소

1. 스크립트 요소: 자바 코드를 작성하고 코드에서 작성한 변수나 계산식 및 메소드의 결과를 출력하기 위해서 사용
   ① Expression(표현식): 값을 출력
   ② Scriptlet: 자바코드
   ③ Declaration: jsp 파일의 멤버변수 및 메서드 선언

2. Implicit Object: 웹 애플리케이션을 작성하는데 필요한 기능을 제공해 주는 객체로 별도의 생성과정 없이 사용할 수 있는 객체
   - request
   - response
   - session
   - application
   - page
   - 기타

JSP 문서의 구성 요소

1. 스크립트 요소: 자바 코드를 작성하고 코드에서 작성한 변수나 계산식 및 메소드의 결과를 출력하기 위해서 사용
   ① Expression(표현식): 값을 출력
   ② Scriptlet: 자바코드
   ③ Declaration: jsp 파일의 멤버변수 및 메서드 선언

2.Implicit Object: 웹 애플리케이션을 작성하는데 필요한 기능을 제공해 주는 객체로 별도의 생성과정 없이 사용할 수 있는 객체
   - request
   - response
   - session
   - application
   - page
   - 기타

JSP 문서의 구성 요소 사용예시

▶ **JSP 페이지의 코드**

```
<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>
```

▶ **서블릿 클래스의 코드**

```
out.println( "<HTML> ");
out.println( "<HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD> ");
out.println( "<BODY> ");
int total = 0;
// 스크립틀릿 안에 있던 자바 명령문
for (int cnt = 1; cnt <= 100; cnt++)   total += cnt;
out.print( "1부터 100까지 더한 값은? ");
out.println(total);   // 익스프레션 안에 있던 자바 식
out.println( "</BODY> ");
out.println( "</HTML> ");
```

page 디렉티브

1. contentType 속성과 charaterset: 결과를 생성할 때의 문서 타입과 문자 인코딩 방식
   - TYPE 또는 TYPE; charset=캐릭터 셋
   - TYPE에는 MIME TYPE을 입력(text/html, text/xml, text/plain..)-기본값은 text/html
   - charset은 생략이 가능한데 생략하면 ISO-8859-1로 지정됩니다.
   - <%@ page contentType="text/html; charset=UTF-8" %>
   - 서블릿으로 변환될 때 response.setContentType("text/html; charset=UTF-8")로 변환됩니다

2. 자바 언어가 제공하거나 직접 작성한 패키지의 클래스를 사용하고자 할 때 경로를 줄여서 사용하기 위한 방법으로 import 속성 사용
   - <%@ page import ="java.util.Calendar, java.util.Date...." %>
   - jsp 페이지는 javax.servlet, javax.servlet.jsp, javax.servlet.http 이 3개의 패키지는 자동으로 import되어 있습니다.
3. include 지시자는 다른 jsp 페이지나 html 문서를 불러다가 현재 페이지의 일부로 만들기 위해서 사용하는 것으로 file 애트리뷰트를 이용하면 됩니다.
   - <%@include file="today.jsp">, <%@include file="sub/today.jsp">
4. html 문서로 변환될 때 불필요한 공백을 제거하기 위해서는 trimDirectiveWhitespaces 속성의 값을 true로 설정하면 됩니다.
5. pageEncoding 속성에 jsp 페이지를 구현한 파일의 인코딩 방식이나 파일을 읽어올 때 사용할 인코딩을 지정
6. 모든 지시자는 <%@으로 시작하고 %>로 끝나야 한다.
   <%@ 바로 다음에는 지시자 이름이 와야 하고, 지시자 이름 다음에는 여러 가지 애트리뷰트가 올 수 있다

page 디렉티브

- page 지시자는 JSP 페이지 전체에 적용되는 정보를 기술하기 위해 사용된다

page 디렉티브

애트리뷰트 이름	기술하는 정보/애트리뷰트의 역할
contentType	JSP 페이지가 생성하는 문서의 종류와 그 문서를 웹 브라우저로 전송할 때 사용되는 인코딩 타입
import	스크립팅 요소 안에서 사용할 자바 클래스와 인터페이스를 임포트하기 위해 사용하는 애트리뷰트
buffer	출력 버퍼의 크기, default = 8Kb
autoFlush	출력 버퍼가 모두 찼을 때의 동작
isThreadSafe	JSP 페이지가 싱글-스레드 모드로 작동하도록 만들기 위해 필요한 애트리뷰트
session	JSP 페이지의 세션 참여 여부
errorPage	에러를 처리할 JSP 페이지의 URL
isErrorPage	에러를 처리하는 JSP 페이지인지 여부
isELIgnored	익스프레션 언어의 무시/처리 여부
pageEncoding	JSP 페이지의 인코딩 타입
info	JSP 페이지에 대한 설명
extends	JSP 페이지로부터 생성되는 서블릿 클래스의 슈퍼클래스
language	스크립팅 요소 안에서 사용할 프로그래밍 언어. 현재는 'java' 라는 값만 지정할 수 있음
deferredSyntaxAllowedAsLiteral	익스프레션 언어의 예약 문자열인 '#{' 를 사용했을 때의 에러 발생 여부
trimDirectiveWhitespaces	지시자 바로 다음에 있는 공백 문자를 제거하기 위해 사용하는 애트리뷰트

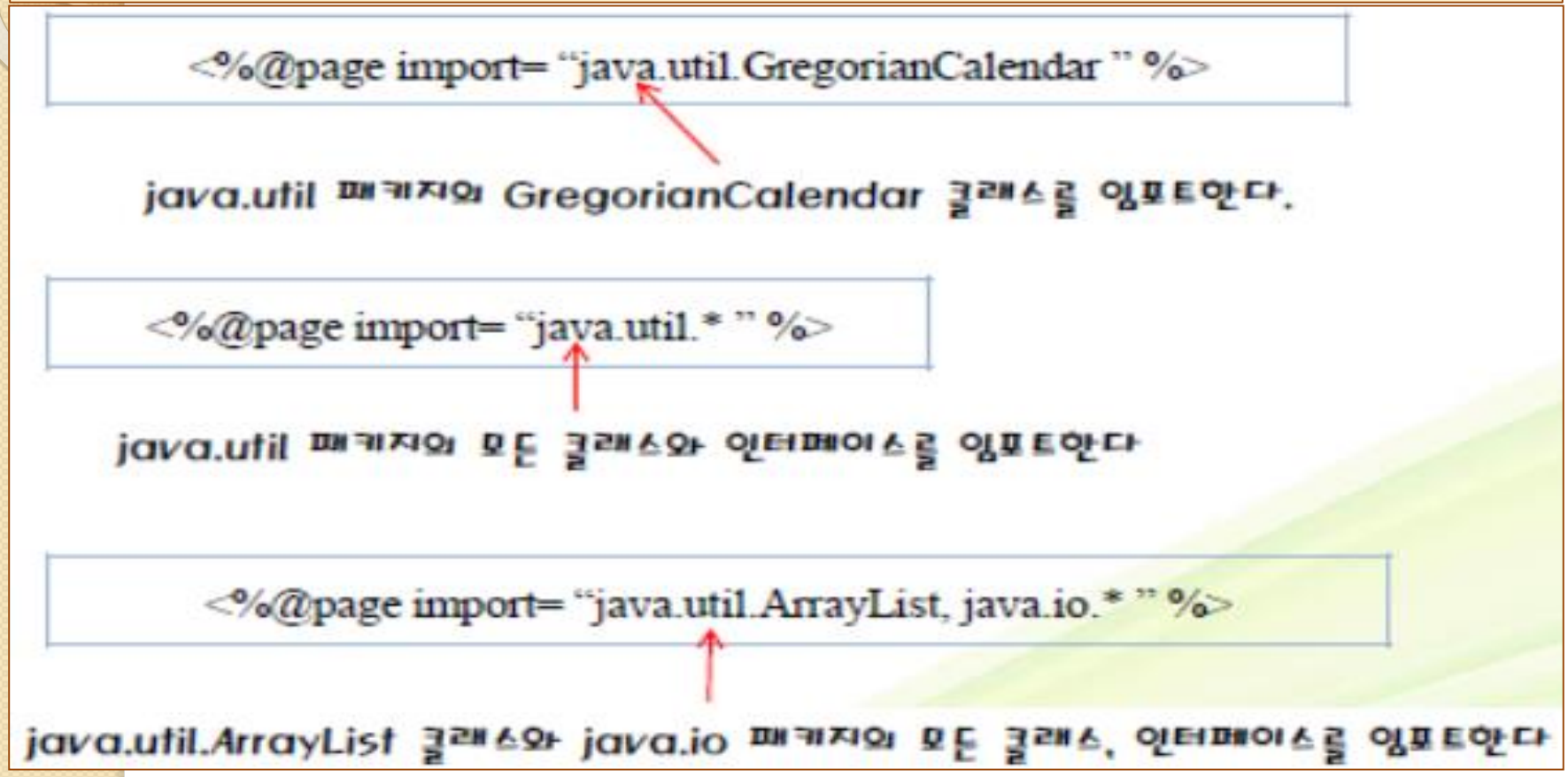
page 지시자의 애트리뷰트

JSP 문서의 구성 요소

1. Directive(디렉티브, 지시자)
   1) JSP 페이지에 대한 정보를 설정할 때 사용
   2) <%@ 디렉티브이름 속성="값" 속성="값" … %> 의 형태로 작성
   3) <%@ page language="java" contentType="text/html; charset=utf-8"
      pageEncoding="utf-8"%>
   4) page가 디렉티브 이름이고 contentType과 pageEncoding이 속성
   5) "text/html; charset=utf-8" , "utf-8" 이 값
   6) 제공되는 디렉티브
      ① page: JSP 페이지에 대한 정보를 지정하며 문서의 타입, 출력 버퍼의 크기, 에러 페이지 등의 정보를 입력
      ② taglib: 태그 라이브러리를 지정
      ③ include: 특정 영역에 다른 문서를 포함

JSP 문서의 구성 요소 사용예시

1. page 지시자의 import 애트리뷰트는 자바의 import 문과 마찬가지로 다른 패키지에 속하는 클래스나 인터페이스를 임포트하는 역할을 한다.



선언문 및 사용예시

선언문은 JSP 페이지 내에 안에서 필요한 멤버변수나 메소드가 필요할 때 쓰이는 요소
선언문에서 선언된 변수는 자바에서와 마찬가지로 전역변수 역할을 하는 멤버 변수가 된다. 선언문의 문법 <%! 문장 %>

1. 선언문에서 변수 선언
   선언문에서 선언된 변수는 JSP 페이지가 서블릿(Servlet) 으로 파싱(parsing)될 때 서블릿의 멤버변수가 된다. (예시)

```
<%!
      private String name= HongGilDong";
      private int year = 2007;
%>
```

2. 선언문에서 메소드 선언
   선언문에서 선언된 메소드는 JSP 페이지 내에서 일반적인 메소드로 사용(예시)

```
<%!
      String id = "HongGilDong";
      public String getId( ) {
            return id;
      }
%>
```

스크립트릿(Scriptlet)

1. 스크립트릿(Scriptlet)은 JSP페이지에서 가장 일반적으로 많이 쓰이는 스크립트 요소로 주로 프로그래밍의 로직을 기술할 때 많이 쓰인다.
2.JSP 페이지가 Servlet으로 변환되고 이 페이지가 호출 될 때 _jspService 메소드 안에 선언
3.스크립트릿(Scriptlet)에서 선언된 변수는 지역변수로 선언
4.스크립트릿의 문법
<% 문장%>
5. 스크립트릿에서 선언한 변수는 지역 변수이므로 별도로 선언한 메소드부분에서는 해당변수를 사용할 수 없다.
6.자동 초기화가 안되므로 반드시 초기화를 해 주어야 한다.

```
<%
String id = “HongGilDong";
String pass="";
%>
```

스크립트릿(Scriptlet) 예시

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
	<head>
		<%@ page language="java" contentType="text/html; charset=utf-8"
		pageEncoding="utf-8"%>
		<%@ page import="java.util.*" %>
		<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
		<title>스크립트릿 예제 - 활용</title>
	</head>
	<body>
		<h2>스크립트릿 예제 - 활용</h2>
		<%
		Calendar getDate = Calendar.getInstance();
		Date nowTime = getDate.getTime();
		%>
		현재는 <%=nowTime.getHours()%>시 <%=nowTime.getMinutes()%>분입니다.
	</body>
</html>
```

표현식(Expression)

1.표현식(expression)은 JSP 페이지에서 웹 브라우저에 출력할 부분을 표현하기 위한 것.
2.표현식은 웹 브라우저에 출력을 목적으로 하는 변수의 값 및 메소드의 결과값도 출력.
- 스크립트릿 코드 내에서 표현식을 쓸 수 없다. 대신 스크립트릿내에서 출력할 부분은 내장객체의 out객체의 print()또는 println()메소드를 사용해서 출력.

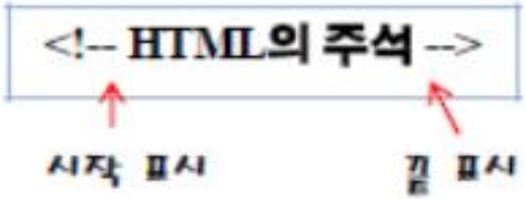
3.표현식의 일반적인 문법
<%=문장%>
4.표현식내에서는 세미콜론(;)은 생략하는데, 이는 JSP페이지가 서블릿으로 변환될 때 표현식부분은 out.print();메소드로 변환되어 자동적으로 세미콜론이 붙여지기 때문.
<%=name[i]%>

표현식(Expression) 예시(expressionTest2.jsp)

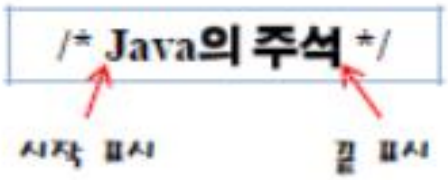
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
	<head>
		<%@ page language="java" contentType="text/html; charset=utf-8"
		pageEncoding="utf-8"%>
		<%@ page import="java.util.*" %>
		<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
	   <title>표현식 예제 - 레퍼런스타입출력</title>
	</head>
	<body>
		<h2>표현식 예제 - 레퍼런스타입출력</h2>
		<%
			Date date = new Date();
		%>
		현재 날짜와 시간: <%=date.toString()%><p>
	</body>
</html>
```

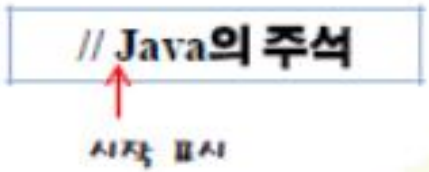
주석을 기술하는 방법

JSP 페이지에 주석을 다는 방법은 다양하다.
JSP 페이지의 HTML 코드 부분 : <!--로 시작해서 -->로 끝나는 HTML 주석을 쓸 수 있다

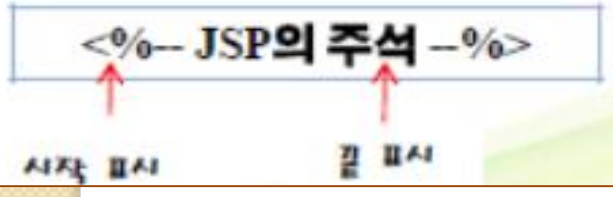


- JSP 페이지의 스크립팅 요소 안 : 자바 문법을 따르는 주석을 쓸 수 있다.





- JSP 고유의 주석을 사용할 수 있다.



JSP 페이지의 내장 변수(implicit variable)1

JSP 페이지 안에 선언을 하지 않고도 사용할 수 있는 변수(예시)

```
<%@page contentType= "text/html; charset=utf-8 "%>
<HTML>
        <%
            String str = request.getParameter( "MAX ");  // 내장변수
            int max = Integer.parseInt(str);
            for (int cnt = 1; cnt <= max; cnt++)
                out.println(cnt + "<BR> "); // 내장변수
        %>
    </BODY>
</HTML>
```

1. request 내장 변수는 서블릿 클래스의 doGet, doPost 메서드의 첫 번째 파라미터와 동일한 역할을 한다.
2. out 내장 변수는 서블릿 클래스에서 getWriter 메서드를 호출해서 얻은 PrintWriter 객체와 마찬가지의 역할

JSP 페이지의 내장 변수(implicit variable)2

1. JSP 페이지 안에서 내장 변수를 사용할 수 있는 이유는 웹 컨테이너가 JSP 페이지를 서블릿 클래스로 변환할 때 자동으로 내장 변수를 선언하기 때문이다.
2. JSP 페이지에서 사용할 수 있는 내장 변수들

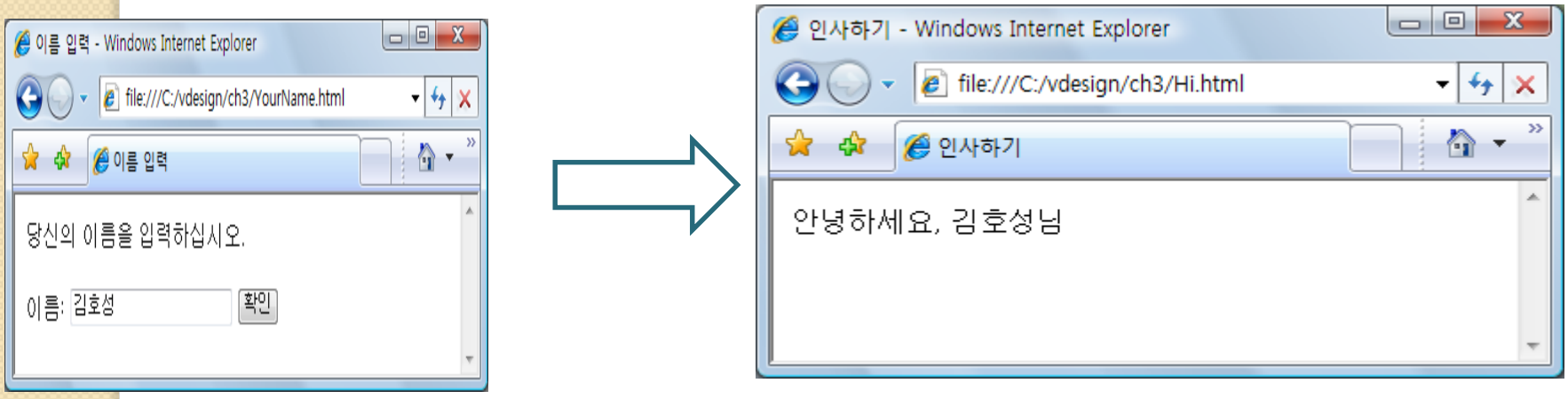
변수 이름	제공하는 기능/변수의 역할	변수 타입
request	doGet, doPost 메서드의 첫 번째 파라미터와 동일한 역할	javax.servlet.http.HttpServletRequest
response	doGet, doPost 메서드의 두 번째 파라미터와 동일한 역할	javax.servlet.http.HttpServletResponse
out	웹 브라우저로 HTML 코드를 출력하는 기능	javax.servlet.jsp.JspWriter
application	JSP 페이지가 속하는 웹 애플리케이션에 관련된 기능	javax.servlet.ServletContext
config	JSP 페이지의 구성 정보를 가져오는 기능	javax.servlet.ServletConfig
pageContext	JSP 페이지 범위 내에서 사용할 수 있는 데이터 저장 기능 등	javax.servlet.jsp.PageContext
session	세션에 관련된 기능	javax.servlet.http.HttpSession
page	JSP 페이지로부터 생성된 서블릿	java.lang.Object
exception	익셉션 객체	java.lang.Throwable

JSP 페이지의 내장 변수(implicit variable)3

1.request 내장 변수
request 내장 변수는 서블릿 클래스에 있는 doGet, doPost 메서드의 첫 번째 파라미터와 동일한 역할을 하고, 타입도 동일하게 javax.serlet.http.HttpServletRequest이다

String str = request.getParameter("NAME ");

인사말을 출력하는 웹 애플리케이션의 화면 설계



JSP 페이지의 내장 변수(implicit variable)3

1.request 내장 변수 적용예시(**브라우저로부터 이름을 입력받는 HTML 문서)**

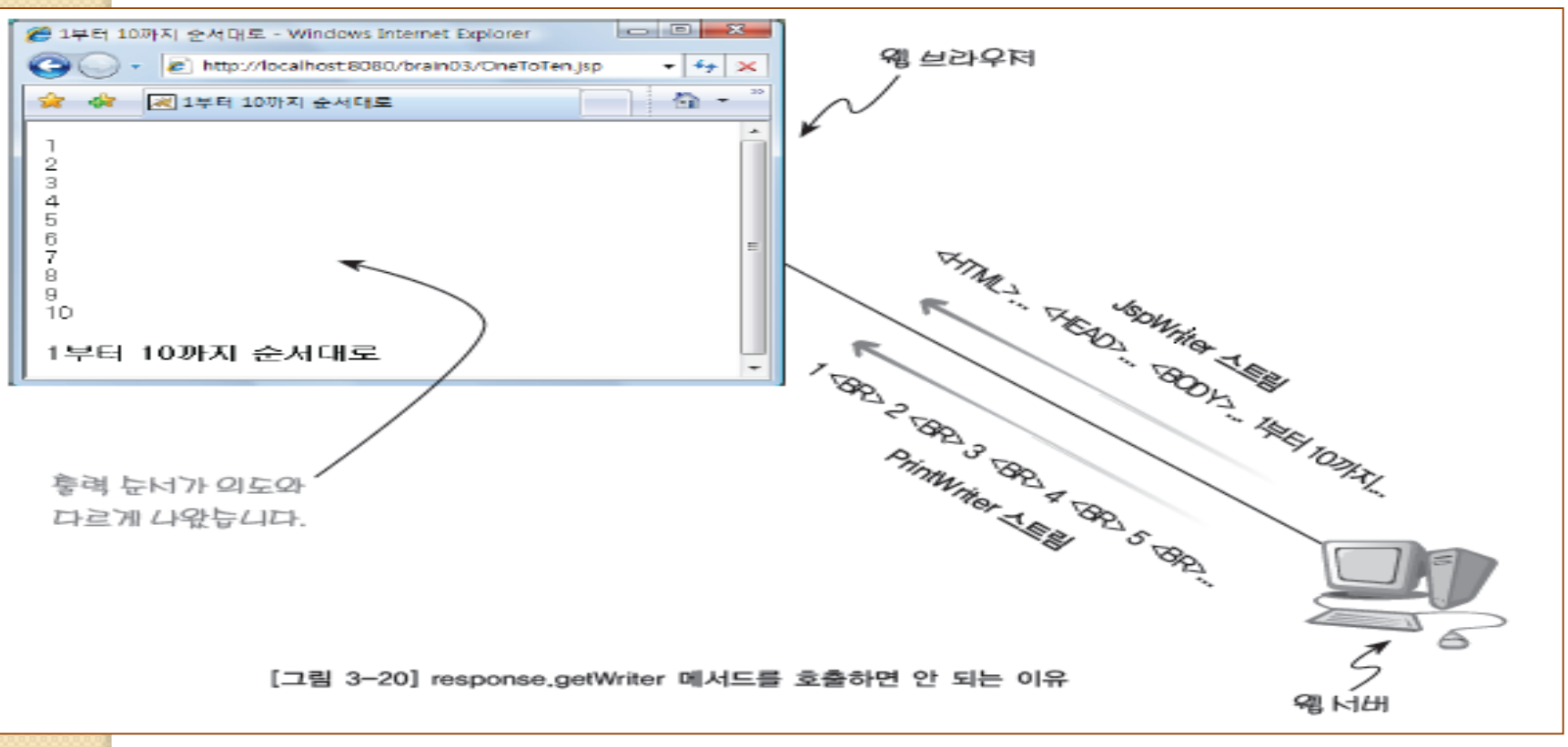
```
<HTML>
        <HEAD>
                <META http-equiv= "Content-Type " content= "text/html;charset=utf-8 ">
                <TITLE>이름 입력</TITLE>
        </HEAD>
        <BODY>
                당신의 이름을 입력하세요.
                <FORM ACTION=/ch03_web/Hi.jsp METHOD=GET>
                이름: <INPUT TYPE=TEXT NAME=YOURNAME>
                <INPUT TYPE=SUBMIT VALUE= „확인 '>
                </FORM>
        </BODY>
</HTML>
```

1.request 내장 변수 적용예시(**입력된 이름을 가지고 인사말을 출력하는 JSP 페이지)**

```
<%@page contentType="text/html; charset=utf-8"%>
<HTML>
        <HEAD><TITLE>인사하기</TITLE></HEAD>
        <BODY>
                <%
                        String str = request.getParameter("YOURNAME");
                        String st= new String (str.getBytes("8859_1"), "utf-8");
                %>
                안녕하세요, <%=st %>님
        </BODY>
</HTML>
```

JSP 페이지의 **out 내장 변수**

1.JSP 페이지에서는 HTML 코드와 익스프레션만 가지고도 원하는 HTML 문서를 만들어서 출력할 수 있기 때문에, 서블릿 클래스의 경우처럼 println, print, printf 메서드를 굳이 호출해야 할 필요가 없다
2.out 내장 변수는 서블릿 클래스에서 getWriter 메서드를 호출해서 얻은 PrintWriter 객체와 비슷한 역할을 한다
3.JspWriter 나 PrintWriter처럼 스트림 형태로 데이터를 출력하는 클래스는 송신측과 수신측 사이에 가상의 통로를 만든다
4. JSP 페이지에서 PrintWriter 객체를 새로 만들면 기존의 out 내장 변수가 관리하던 통로와 더불어 두 개의 통로가 공존하게 된다



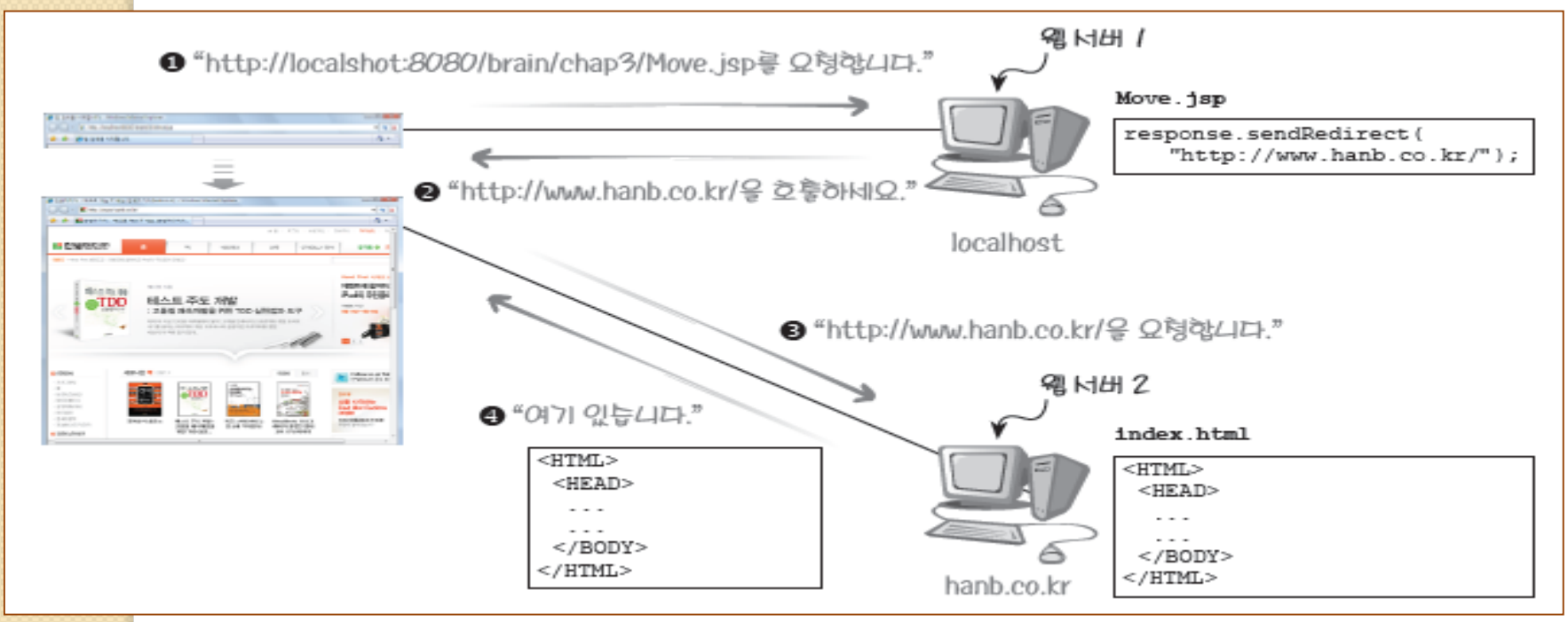
[그림 3-20] response.getWriter 메서드를 호출하면 안 되는 이유

JSP 페이지의 Response 내장 변수

1. response 내장 변수는 서블릿 클래스에 있는 doGet, doPost 메서드의 두 번째 파라미터와 동일한 역할을 한다. 이 변수는 javax.servlet.http.HttpServletResponse 타입이기 때문에 이 인터페이스에 속하는 여러 가지 메서드들을 호출
예시) response.sendRedirect("http://www.daum.net/ ");

2. sendRedirect 메서드를 호출할 때 주의할 점: 이 메서드를 호출하기 전과 후에 웹 브라우저로 데이터를 출력하면 안 된다

3.sendRedirect 메서드는 파라미터로 지정한 URL을 직접 호출하는 것이 아니라 그 URL 을 이용해서 다시 웹 자원을 호출하라는 메시지를 웹 브라우저로 보낼 뿐이다



JSP 페이지의 application 내장 변수

1. application 내장 변수는 웹 애플리케이션에 관련된 여러 가지 기능을 제공한다.
   application 내장 변수에 대해 호출할 수 있는 getContextPath 메서드는 웹 애플리케이션의 URL 경로명을 리턴하는 메서드이다.

```
// 웹 애플리케이션의 URL 경로명을 리턴하는 메서드
String appPath = application.getContextPath();
```

application 내장 변수에 대해 호출할 수 있는 getRealPath 메서드는 웹 애플리케이션 내에서의 파일 경로명을 파일시스템 전체에 대한 절대 경로명으로 바꾸는 메서드이다

```
 // 웹 애플리케이션 내에서의 파일의 경로명
String absolutePath = application.getRealPath( "/sub1/Intro.html ");
```