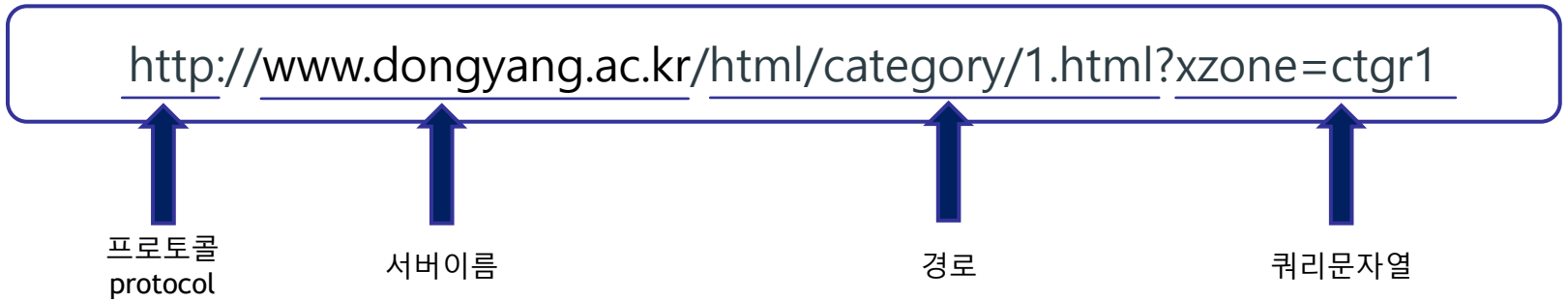
2강 자료

URL과 웹 페이지

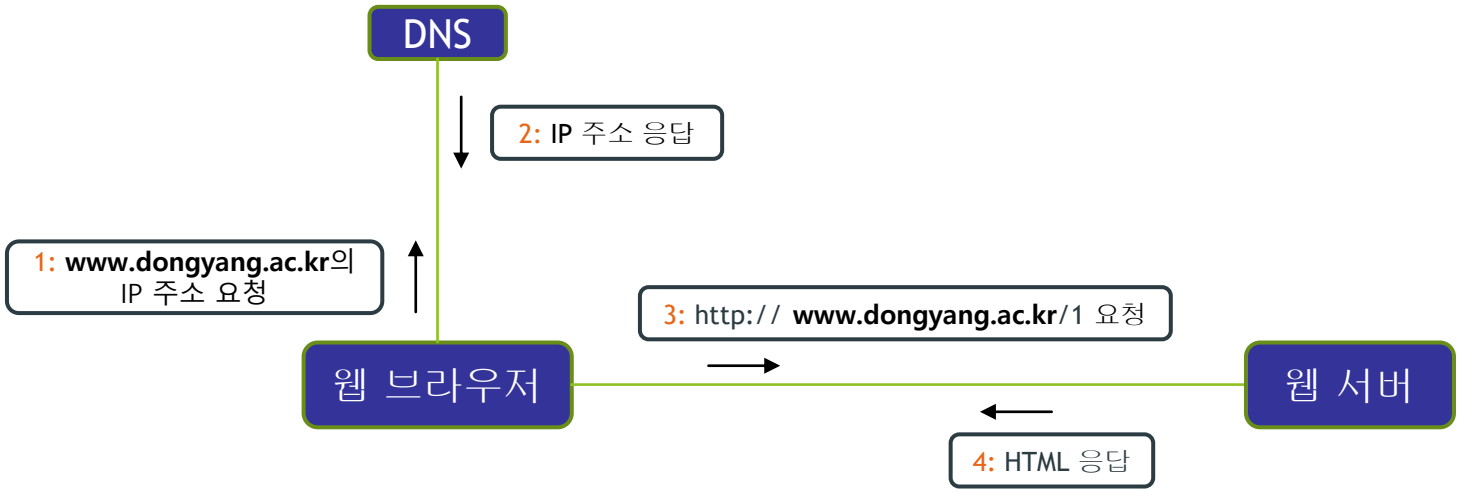
- URL Uniform Resource Locator
  - 웹 페이지(자원)의 주소/위치
  - 네트워크에서 자원이 어디 있는지를 알려주기 위한 규약
  - 맞는 프로토콜(protocol) 사용
- URI Uniform Resource Identifier
  - 웹 페이지(자원)의 식별자
  - URL의 상위 개념
  - 자원이 어디에 있는지 자원 자체를 식별하는 방법

http://<host>:<port>?<query>#fragment

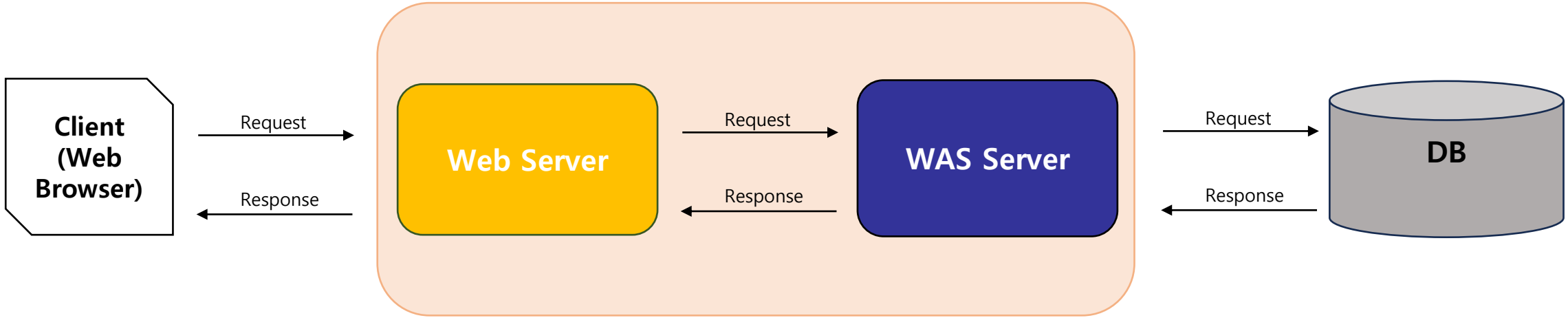
URL 형식


http://www.dongyang.ac.kr/html/category/1.html?xzone=ctgr1
프로토콜
protocol
서버이름
경로
쿼리문자열

웹 브라우저와 웹 서버

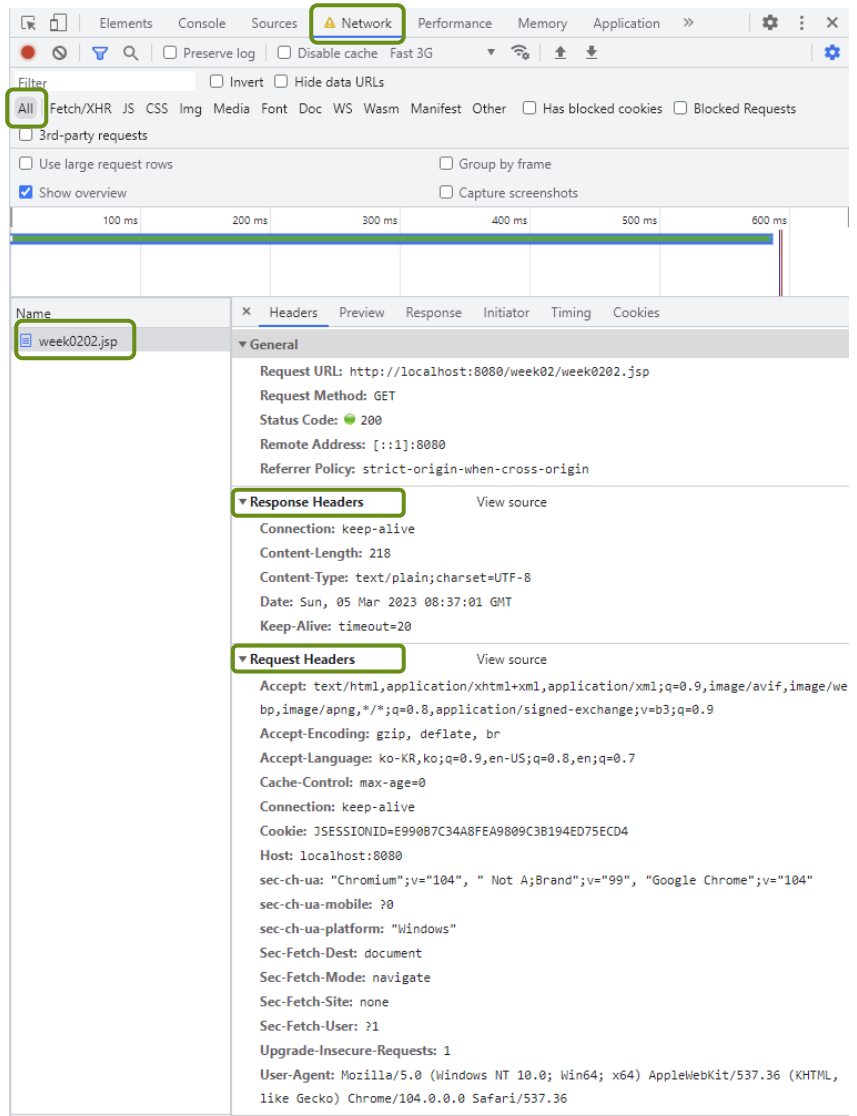

DNS
2: IP 주소 응답
1: www.dongyang.ac.kr의
IP 주소 요청
3: http:// www.dongyang.ac.kr/1 요청
웹 브라우저
웹 서버
4: HTML 응답

웹 브라우저와 웹 서버


Client
(Web
Browser)
Request
Response
Web Server
Request
Response
WAS Server
Request
Response
DB

웹 브라우저와 웹 서버

Chrome 개발자도구(F12)



요청 데이터

```
GET / HTTP/1.1
Host: www.daum.net
Connection: keep-alive
Accept: text/html
User-Agent: Mozilla/5.0 ...생략
Accept-Encoding: gzip, deflate, sdch
Accept-Language: ko,en-US;q=0.8,en;q=0.6
```

웹 브라우저 ⇄ 웹 서버

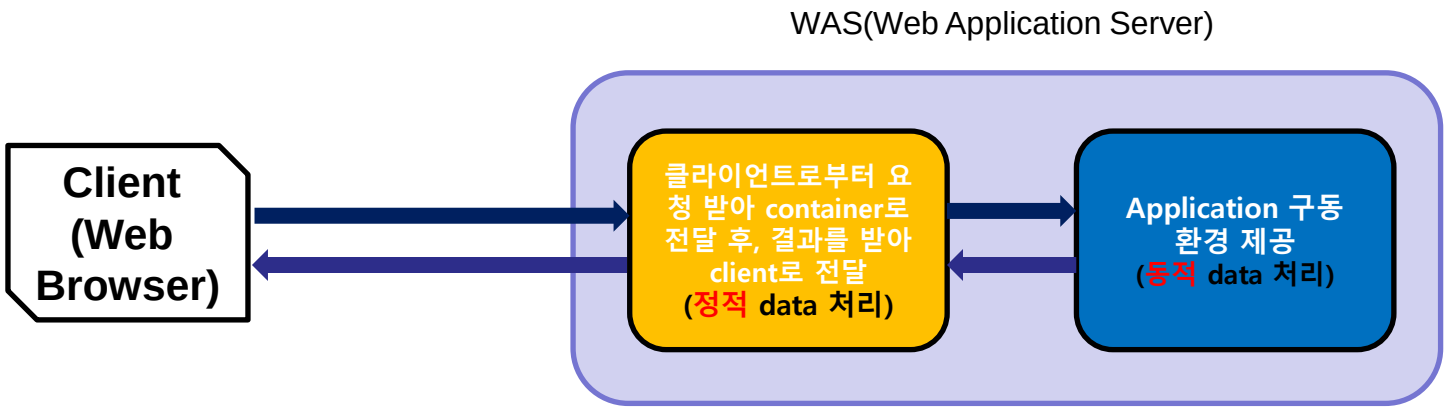
응답 데이터

```
HTTP/1.1 200 OK
Date: Wed, 22 Apr 2015 12:53:38 GMT
Expires: Sat, 01, Jan 1970 22:00:00 GMT
Content-Type: text/html;charset=UTF-8
Content-Language: ko

<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="utf-8">
...생략
</html>
```

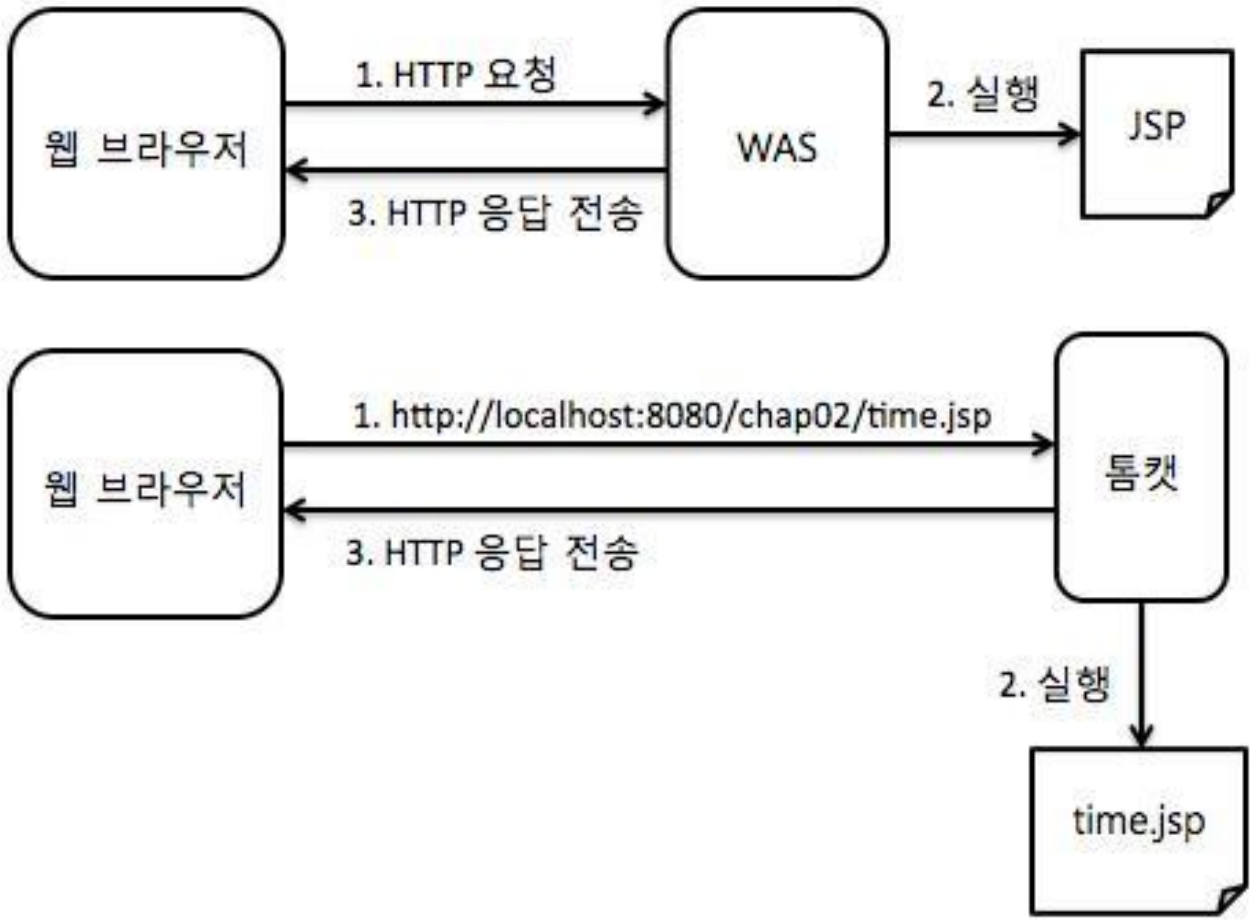
정적(static) 자원과 동적(dynamic) 자원

- 서버단에서 Application을 동작하여 비즈니스 로직을 수행하고 Database 접속 기능을 수행
- 일반적으로 container라는 용어로 쓰이고 동적인 파일을 서비스함



- WAS는 Web server 와 container의 결합으로 다양한 기능을 container에 구현하여 다양한 역할을 수행할 수 있음
- 즉, Client의 요청이 있을 때 내부의 프로그램(AP)을 통해 결과를 만들어내고 이것을 다시 client로 전달해주는 역할을 하는데 이 container 기능을 수행하느냐가 web server와의 차이점이라고 할 수 있음

정적(static) 자원과 동적(dynamic) 자원



동적페이지 예

Eclipse를 사용하여 프로젝트 생성
JSP 파일 추가, 다음 내용으로 파일 작성

```
<%@ page contentType="text/html; charset=UTF-8" %>
<!DOCTYPE html>
  <head>
    <title>현재 시간</title>
  </head>
  <body>
    지금 <%= new java.utils.Date() %>
  </body>
</html>
```

JSP 기본구조

```
<%@ page contentType="text/html; charset=UTF-8" %>
<!DOCTYPE html>
  <head>
    <title>HTML 문서 제목</title>
  </head>
  <body>
    <%
        String bookTitle = "JSP 웹 프로그래밍";
        String author = "홍길동";
    %>
    <b><%= bookTitle %></b>(<%= author %>)
</body>
</html>
```

설정부분: JSP 페이지에 대한 설정 정보

- contentType="text/plain" 일 때의 결과 확인!!!
- contentType에서 charset를 설정하지 않을 때의 결과 확인!!! – "iso-8859-1"
- trimDriectiveWhitespaces="true" 사용하는 결과 확인!!!

생성부분: HTML 코드 및 JSP 스크립트

JSP 페이지 구성 요소

- 디렉티브(directive)
- 스크립트
  - 스크립틀릿(scriptlet)
  - 표현식(expression)
  - 선언(declaration)
- 표현 언어(expression language)
- 기본 객체(implicit object)
- 정적인 데이터
- 표준 액션 태그(action tag)
- 커스텀 태그(custom tag)와 태그 라이브러리(JSTL)

JSP 페이지 구성 요소 - directive

- 디렉티브 형식
  - **<%@ 디렉티브이름 속성1="값1" 속성2="값2" ... %>**
- page
  - JSP 페이지에 대한 정보 지정
  - 예: 문서 형식 지정 contentType="text/html; charset=utf-8"
- taglib
  - 태그 라이브러리 지정
- include
  - JSP 페이지의 특정 영역에 다른 페이지 포함

```
<html>
  …..
  <jsp:include page="header.jsp" flush="true" />
  …..
<html>
```

JSP 페이지 구성 요소 – 스크립트

- **스크립틀릿**
  - Java 코드 실행
- 표현식
  - 값 출력
- 선언
  - Java 메서드 정의

```
<%@ page contentType="text/html; charset=UTF-8" %>
<!DOCTYPE html>
  <head>
    <title>HTML 문서 제목</title>
  </head>
  <body>
    <%
        int sum = ;
        for(int i = 1; I <= 10; ++i)
            sum += i;
    %>
    1부터 10까지의 합계는 <%= sum %>
</body>
</html>
```

스크립틀릿 scriptlet

표현식 expression

```
<%!
    public int multiply(int a, int b) {
        return a * b;
    }
%>
2 * 5 = <%= multiply(2, 5) %>
```

선언 declaration

JSP 페이지 구성 요소 – 기본 객체와 표현언어

```
<%
    int a = Integer.parseInt(request.getParameter("a");
    int b = Integer.parseInt(request.getParameter("b");
%>
a + b = <%= a + b %>
```

기본 객체 implicit object

표현식 expression

```
a + b = ${param.a + param.b}
```

표현언어 expression language

◆ 결과 확인 방법:

- ✓ Eclipse에서 실행후
- ✓ 웹 브라우저의 주소 입력 창에서

기본 URL?a=10&b=20 을 추가해서 확인한다.

참고 : encoding

- IDE/Tomcat에서
  - pageEncoding이 없을 경우:
    - contentType의 charset 적용
  - contentType의 charset도 없을 경우
    - 시스템 기본 encoding 설정 사용

- 브라우저에서
  - pageEncoding은 클라이언트(웹 브라우저)와 관계 없음
  - contentType의 charset이 없을 경우
    - <meta> 태그의 charset 사용
  - <meta> 태그의 charset도 없을 경우
    - 웹 브라우저의 기본 encoding 설정 사용

참고 : encoding

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%@ page pageEncodign="utf-8" %>
<%@ page import="java.util.Date" %>
<!DOCTYPE html>
  <head>
    <title>현재 시간</title>
  </head>
  <body>
     <%
       Date now = new Date();
     %>
     현재 시간은 <%= now %>
</body>
</html>
```

기본 객체: request

- JSP 페이지에서 가장 많이 사용되는 기본 객체
- 웹 브라우저의 요청과 관련
- 제공 기능
  - 클라이언트 관련 정보 읽기
  - 서버와 관련된 정보 읽기
  - 클라이언트가 전송한 파라미터 읽기
  - 클라이언트가 전송한 요청 헤더 읽기
  - 클라이언트가 전송한 쿠키 읽기
  - 속성(attribute) 처리

기본 객체: request

메서드	리턴 타입	설명
getRemoteAddr()	String	웹 서버와 연결된 클라이언트의 IP 주소
getContentLength()	Long	클라이언트가 전송한 요청 정보의 길이 길이를 알 수 없는 경우 -1을 반환
getCharacterEncoding()	String	클라이언트가 요청한 정보의 encoding 정보
getContentType()	String	클라이언트가 요청한 정보의 contentType
getProtocol()	String	클라이언트가 요청한 프로토콜
getMethod()	String	웹 브라우저가 정보를 전송할 때 사용한 method
getRequestURI()	String	웹 브라우저가 요청한 URL에서 경로 추출
getContextPath()	String	JSP 페이지가 속한 어플리케이션의 context 경로 추출
getServerName()	String	연결할 때 사용한 서버 이름
getServerPort()	Int	연결한 서버에서 사용하는 포트 번호

기본 객체: request 기본 정보 확인

- 리스트 3.13 chap03\requestInfo.jsp

```
...
<body>
  클라이언트 IP: <%= request.getRemoteAddr() %>
  요청 정보길이: <%= request. getContentLength () %>
  요청 정보 인코딩: <%= request.getCharacterEncoding() %>
  요청 정보 컨텐츠 타입: <%= request.getContentType() %>
  요청 정보 프로토콜: <%= request.getProtocol() %>
  요청 정보 전송 방식: <%= request.getMethod() %>
  요청 URI: <%= request.getRequestURI() %>
  컨텍스트 경로: <%= request.getContextPath() %>
  서버 이름: <%= request.getServerName() %>
  서버 포트: <%= request.getServerPort() %>
</body>
...
```

기본 객체: request

메서드	리턴 타입	설명
getParameter(String name)	String	이름이 'name'인 파리미터의 값을 반환 'name' 파라미터가 없으면 null을 반환
getParameterValues(String name)	String[]	이름이 'name'인 파라미터의 값을 배열로 반환 'name' 파라미터가 없으면 null을 반환
getParameterNames()	java.util.Enumeration	웹 브라우저가 전송한 파라미터의 이름 목록을 Enumeration 형식으로 반환
getParameterMap()	java.util.Map	웹 브라우저가 전송한 파라미터의 이름 목록을 Map 형식으로 반환

기본 객체: request 1

```
…
  <%
    request.setCharacterEncoding(“utf-8”);
  %>
<html>
  …
<body>
  <%
    String[] values = request.getParameterValues(“a”);
    if(values != null) {
      for(int i = 0; i < values.length(); ++i) {
  %>
        <%= values[i] %>
  <%
      }  // end of for
    } // end of if
  %>
</body>
…
```

기본 객체: request 2

```
...
<%
    // reqeust.getParameter() 사용
    String a = request.getParameter("a");
    if(a != null) {
      System.out.print("a is "); System.out.println(a);
  %>
        a is <%= a %><br />
        a is ${param.a}
  <%
    } else {
      System.out.println("파라미터 a가 없네요!");
  %>
      파라미터 a가 없네요!
  <%
    }
  %>
...
```

기본 객체: request 3

```
……
<%
    // request.getParameterNames() 사용
    Enumeration<String> paramEnum =
request.getParameterNames();
    while(paramEnum.hasMoreElements()) {
        String name = paramEnum.nextElement();
%>
          <%= name %>
<%
    }
%>
…
```

기본 객체: request 4

```
…
<%
    // request.getParameterMap() 사용
    Map<String, String[]> paramMap =
request.getParameterMap();
    String[] aParam = (String[])paramMap.get("a");
    String[] bParam = (String[])paramMap.get("b");
    if(aParam != null) {
        for(int i = 0; i < aParam.length; ++i){
%>
            <%= aParam[i] %>
<%
        }
    }
    if(bParam != null) {
        for(int i = 0; i < bParam.length; ++i) {
%>
            <%= bParam[i] %>
<%
        }
    }
%>
…
```

기본 객체: request 5

- Chap03/form.jsp
- Chap03/viewParameter.jsp
- POST 방식의 parameter 확인

기본 객체: response

- 웹 브라우저에 보내는 응답 정보
  - 헤더 정보 설정/추가
    - Cache-Control
- 리다이렉트(페이지 이동) 하기 1

```
<%
    response.setHeader("Cache-Control", "no-cache");
%>
```

firstPage.jsp

```
<%
    // 1.
    response.sendRedirect("secondPage.jsp");
%>
```

여기는 firstPage.jsp에서 이동한 secondPage.jsp

기본 객체: response

- 리다이렉트(페이지 이동) 하기 2

여기는 firstPage.jsp

```
<%
    // 2.
    String ecodedName = URLEncoder.encode("홍길동", "utf-8");
    response.sendRedirect("secondPage.jsp?name=" + ecodedName);
%>
```

여기는 secondPage.jsp

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

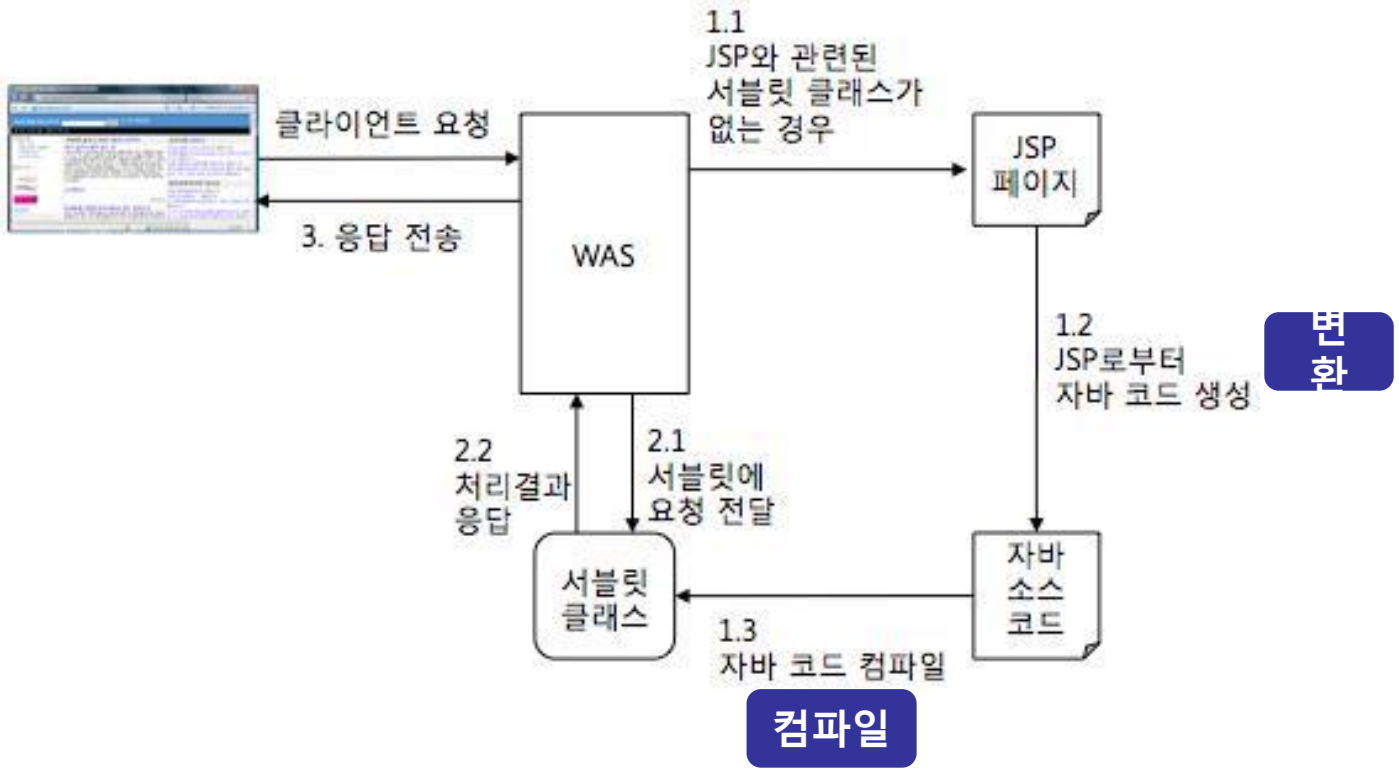
전달 받은 name 파라미터의 값은 [<%= name %>]이다.

JSP 주석

- HTML 주석 사용
- JSP 주석 사용
  - <%-- 주석 내용 --%>
  - <%-- 주석 내부에 <%-- 또다른 주석 --%> 이 있다. --%>

JSP 필수 이해 요소

- JSP 처리 과정



…₩workspace₩.metadata₩.plugins₩org.eclipse.wst.server.core₩tmp4₩work₩Catalina₩localhost₩week02₩org₩apache₩jsp

JSP 필수 이해 요소

- page 디렉티브에서 버퍼 설정
- <%@ page buffer="4kb" %> <%-- kb 단위로 지정, 기본 8kb 이상 --%>
  - <%@ page buffer="none" %> <%-- 버퍼를 사용하지 않도록 설정 --%>
  - <jsp:forward> 기능을 사용할 수 없다.
- 출력 내용, 즉 응답 내용을 수정할 수 없다.
- <%@ page autoFlush="true" %> <%-- 버퍼가 차면, 자동 전송 --%>
- <%@ page authFlush="false" %> <%-- 버퍼가 차면, 예외 Exception 발생으로 실행 중지 --%>

JSP 필수 이해 요소

- 예제 [리스트 4.1] 확인

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ page buffer="1kb" autoFlush="true" %>
<html>
    <head>
        <title>autoFlush 속성값 true 예제</title>
    </head>
    <body>
        <%  for (int i = 0 ; i < 1000 ; i++)
                {
        %>
                    1234
        <%  }  %>
    </body>
</html>
```

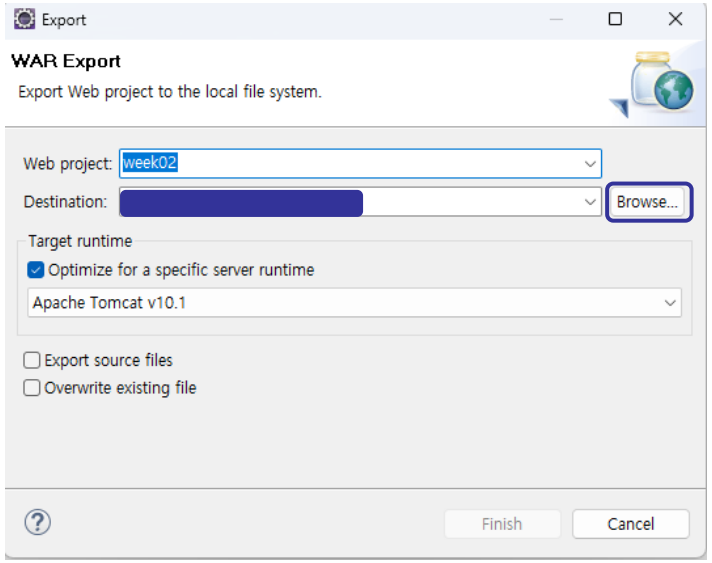
JSP 필수 이해 요소

- request.getContextPath()

```
<%@ page contentType = "text/html; charset=utf-8" %>
<html>
    <head>
        <title>request.contextPath</title>
    </head>
    <body>
        웹 어플리케이션 경로는 <%= request.getContextPath() %>
    </body>
</html>
```

JSP 필수 이해 요소

- 웹 어플리케이션 배포
  - war web application archive
- war 파일 생성
  - Eclipse 프로젝트 이름을 마우스 오른쪽 버튼으로 클릭
  - 표시되는 단축 메뉴에서
    - Export -> WAR file



End of Document

Thank you