4장. 폼 태그를 이용한 값 입력

컴퓨터정보과
김진숙

이번 장에서 공부할 것

- 폼 태그를 통해 사용자 입력을 받는 방법

- 학습내용
  - 입력 폼 작성과 입력된 값 처리
  - 단일 값 입력 처리
  - 다중 선택 입력의 처리

4.1 입력 폼 작성과 입력된 값 처리

▪ 입력 폼 작성

▪ 웹에서 사용자 입력을 받으려면 **<form>** 태그로 만든 입력 폼을 이용

```
<form action="입력_값을_전달할_파일" method="데이터_전달하는_방식">
    입력 태그  ...
</form>
```

▪ **action** 속성

- 이 폼에 입력된 값들을 전달할 경로와 파일명
- 입력 폼이 담긴 HTML 파일과 입력된 값들을 처리할 JSP 프로그램이 다른 폴더에 있다면 그 프로그램이 있는 경로까지 적어 주어야 한다.

▪ **method** 속성

- 입력된 데이터를 전달하는 방식.
- **GET**과 **POST** 중 원하는 방식 설정

4.1 입력 폼 작성과 입력된 값 처리

- <form> 태그 내부에는 값을 입력 받을 수 있는 입력 태그들을 필요한 만큼 사용

입력 태그	입력 유형
<input>	텍스트 입력, 비밀번호 입력, 라디오 버튼, 체크 박스, 전송(submit) 버튼, 초기화(reset) 버튼, 일반 버튼 파일 업로드
<select>	드롭다운 리스트, 일반 리스트
<textarea>	여러 줄의 텍스트 입력

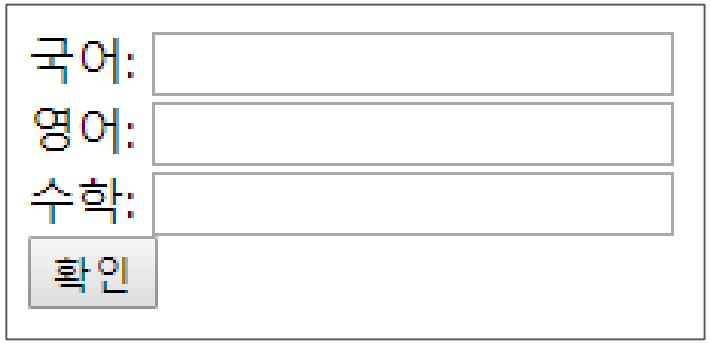
- <input type="text" name="abc">
  - 한 줄짜리 텍스트를 입력받는 입력란 생성. 이 입력란의 이름은 “abc”
- <input type="submit" value="확인">
  - “확인”이라는 문자열이 적혀 있는 전송(submit) 버튼을 생성
  - 사용자가 입력 폼에 값들을 입력한 뒤 이 버튼을 누르면, form 태그의 action 속성에 적힌 JSP 프로그램으로 페이지가 전환
  - 이때 입력된 값들도 request 객체에 담겨 해당 파일로 전달

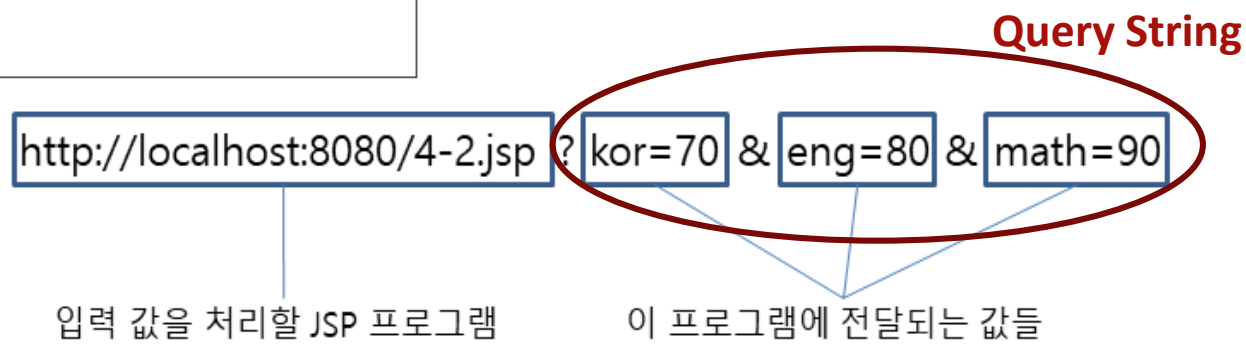
4.1 입력 폼 작성과 입력된 값 처리

- GET 방식으로 값 전달

[예제 4-1] GET 방식을 사용하는 입력 폼 (4-1.html)

```
 1: <!doctype html>
 2: <html>
 3: <head>
 4:     <meta charset="utf-8">
 5: </head>
 6: <body>
 7:
 8: <form action="4-2.jsp" method="get">
 9:     국어: <input type="text" name="kor"><br>
10:     영어: <input type="text" name="eng"><br>
11:     수학: <input type="text" name="math"><br>
12:     <input type="submit" value="확인">
13: </form>
14: </body>
15: </html>
```



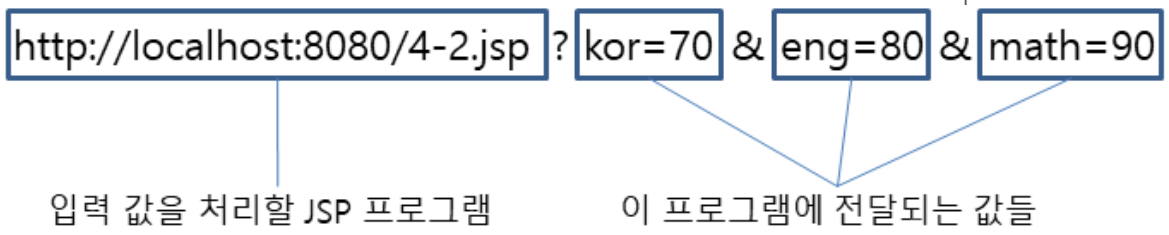


4.1 입력 폼 작성과 입력된 값 처리

- GET 방식으로 전달된 값 사용 (읽기)
  - **request.getParameter**("입력_태그의_name_속성")

[예제 4-2] GET 방식으로 전달받은 값의 사용 (4-2.jsp)

```
 1: <%@ page language="java" contentType="text/html; charset=UTF-8"
 2:     pageEncoding="UTF-8"%>
 3: <!DOCTYPE html>
 4: <html>
 5: <head>
 6:     <meta charset="UTF-8">
 7: </head>
 8: <body>
 9:
10: 국어: <%=request.getParameter("kor" )%><br>
11: 영어: <%=request.getParameter("eng" )%><br>
12: 수학: <%=request.getParameter("math")%> <br>
13:
14: </body>
15: </html>
```



[참고] 톰캣기반에서의 한글처리

1. 웹 브라우저 → 서버(Get방식)

- 별 다른 처리가 필요하지 않음

2. 웹 브라우저 → 서버(Post방식)

- 웹브라우저에서 POST방식으로 데이터가 넘어오는 파라미터 값에 한글이 있는 경우
- 폼으로부터 파라미터를 넘겨받는 페이지에는 반드시 아래 코드 작성

```
<% request.setCharacterEncoding("utf-8"); %>
```

3. 서버→ 웹 브라우저

- 서버로부터 클라이언트로 응답되는 페이지의 화면 출력 시 한글처리
- <%@ page **contentType**= " text/html; **charset=utf-8** " %>
- 서버페이지에 반드시 명시해야 함(지시자의 속성으로 명시)

4.1 입력 폼 작성과 입력된 값 처리

- POST 방식으로 값 전달된 값은 다음 두가지 사항만 GET 방식과 다름
  - 입력 폼의 form 태그에 method="post"라고 적는다.
  - 전달되는 값이 웹 브라우저의 주소 창에 보이지 않는다.

[예제 4-3] POST 방식을 사용하는 입력 폼 (4-3.html)

```
 1: <!doctype html>
 2: <html>
 3: <head>
 4:     <meta charset="utf-8">
 5: </head>
 6: <body>
 7:
 8: <form action="4-2.jsp" method="post">
 9:     국어: <input type="text" name="kor"><br>
10:     영어: <input type="text" name="eng"><br>
11:     수학: <input type="text" name="math"><br>
12:     <input type="submit" value="확인">
13: </form>
14:
15: </body>
16: </html>
```

전달된 값을 사용하는 방법은
GET 방식과 똑같음

4.1 입력 폼 작성과 입력된 값 처리

- GET 방식의 성질을 이용한 프로그램 테스트
  - 입력 양식을 만들지 않고 간단하게 테스트를 원할 때
    직접 URL에 전달될 값들을 적어 넣음(쿼리스트링 : Query String)
  - http://localhost:8080/4-2.jsp?kor=70&eng=80&math=90
  - JSP 프로그램은 URL에 딸려 온 값이 입력 폼으로부터 온 것인지 아니면 사람이 주소창에 직접 입력한 것인지 구분할 수 없음

4.2 단일 값 입력 처리[실습]

- 텍스트 박스와 전송 버튼 외의 입력 폼 작성
  - Github의 예제 내용 복사(미완성)하여 오른쪽 화면 완성할 것
    - 교재 73-75쪽 참조하여 작성
    - 부트스트랩 사용 권장
    - 참고사이트 :
      - https://www.w3schools.com/
  - Github의 예제 내용 복사하여 사용할 것
    - https://github..com/aicsdit/jspProject

4.2 단일 값 입력 처리

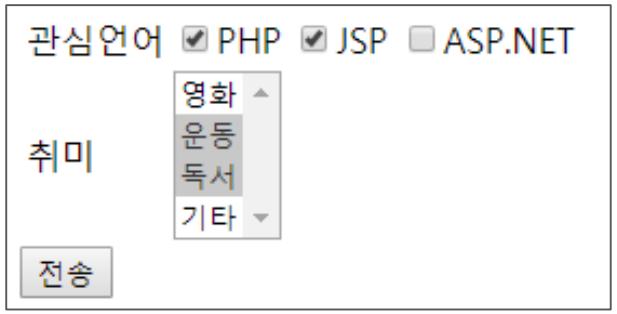
[예제 4-5] 단일 선택 입력값의 출력 (4-5.jsp)

```
 1: <%@ page language="java" contentType="text/html; charset=UTF-8"
 2:     pageEncoding="UTF-8"%>
 3: <!DOCTYPE html>
 4: <html>
 5: <head>
 6:     <meta charset="UTF-8">
 7: </head>
 8: <body>
 9:
10: <%
11:     request.setCharacterEncoding("utf-8");
12: %>
13:     아이디 :   <%=request.getParameter("id")    )%><br>
14:     비밀번호 : <%=request.getParameter("pw")    )%><br>
15:     성별 :     <%=request.getParameter("gender")%><br>
16:     가입경로 : <%=request.getParameter("intro" )%><br>
17:     주소 :     <%=request.getParameter("addr"  )%><br>
18:     메모 :     <%=request.getParameter("memo"  )%><br>
19: </body>
20: </html>
```

- request 객체에게 "이 프로그램으로 전달되어 네가 보관하고 있는 값들이 UTF-8로 인코딩되어 있어"라고 알려줌
- 이 코드가 없으면 한글로 입력된 내용이 깨져서 출력됨

4.3 다중 선택 입력의 처리

- 체크 박스, 다중 선택을 허용하는 리스트 박스로 부터 값을 전달받는 방법



4.3 다중 선택 입력의 처리

[예제 4-7] 다중 선택 입력값의 출력 (4-7.jsp)

```
 1: <%@ page language="java" contentType="text/html; charset=UTF-8"
 2:     pageEncoding="UTF-8"%>
 3: <!DOCTYPE html>
 4: <html>
 5: <head>
 6:     <meta charset="UTF-8">
 7: </head>
 8: <body>
 9:
10: <%
11:     request.setCharacterEncoding("utf-8");
12:
13:     String[] lang   = request.getParameterValues("lang" );
14:     String[] hobby = request.getParameterValues("hobby");
15: %>
16:
17:     관심언어 :
18:     <%
19:         for (int i = 0; i < lang.length; i++) {
20:             out.println(lang[i] + " ");
21:         }
22:     %>
23:     <br>
24:
25:     취미 :
26:     <%
27:         for (int i = 0; i < hobby.length; i++) {
28:             out.println(hobby[i] + " ");
29:         }
30:     %>
31:     <br>
32: </body>
33: </html>
```

- 다중 입력값을 꺼내오기 위해서는 request.getParameterValues() 메소드를 사용
- getParameter()는 하나의 값만 꺼내오면 되므로 문자열 하나를 반환하지만, getParameterValues()는 여러 개의 값을 반환해야 하므로 문자열의 배열을 반환

```
for (String s : lang){
  out.println(s + ", ");
}
```

배열 인덱스 대신 사용할 수 있음

```
for (String s : hobby){
  out.println(s + ", ");
}
```

배열 인덱스 대신 사용할 수 있음

[참조] 내장객체(Implicit Object)의 개요

- 내장 객체는 JSP의 특수 **레퍼런스**(참조) 변수
- 레퍼런스 변수는 선언과 객체 생성 없이 사용 가능
  - JSP 페이지가 서블릿으로 변환 시 JSP 컨테이너가 자동적으로 제공

```
public void _jspService(final javax.servlet.http.HttpServletRequest request, final
javax.servlet.http.HttpServletResponse response)
      throws java.io.IOException, javax.servlet.ServletException {

  final javax.servlet.jsp.PageContext pageContext;
  javax.servlet.http.HttpSession session = null;
  final javax.servlet.ServletContext application;
  final javax.servlet.ServletConfig config;
  javax.servlet.jsp.JspWriter out = null;
  final java.lang.Object page = this;
  javax.servlet.jsp.JspWriter _jspx_out = null;
  javax.servlet.jsp.PageContext _jspx_page_context = null;
```

[참조] JSP 내장객체(Implicit Object)

- 내장 객체의 종류
  - 내장 객체를 사용하여 JSP, 서블릿 간 정보 교환 가능

참조변수	자바클래스	역할	유효 영역
request	javax.servlet.http.HttpServletRequest	웹브라우저 요청 정보 저장	request
response	javax.servlet.http.HttpServletResponse	웹브라우저 요청에 대한 응답 정보 저장	page
pageContext	javax.servlet.jsp.PageContext	현재 JSP에 대한 페이지 컨텍스트 정보 저장	page
session	javax.servlet.http.HttpSession	웹브라우저 내의 정보를 유지하기 위한 세션정보 저장	session
application	javax.servlet.ServletContext	웹서버의 애플리케이션 처리화 관련된 정보 저장	application
out	javax.servlet.jsp.JspWriter	출력 스트림	page
config	javax.servlet.ServletConfig	현재 JSP의 초기화 설정 정보 저장	page
page	java.lang.Object	현재 JSP 페이지에 대한 클래스 정보	page
exception	java.lang.Throwable	에외처리에 사용되는 객체로 JSP페이지가 에러 페이지로 지정될 때 만들어지는 객체	page

[참조] JSP 내장객체(Implicit Object)

- request, session, application, pageContext 내장객체는 속성(attribute) 값을 저장하고 읽을 수 있는 공통 메소드를 제공

공통 메소드	리턴 타입	설명
setAttribute(String key, Object value)	void	해당 내장객체의 속성(attribute)값을 설정하는 메소드로 속성명 key매개변수에, 속성값 value매개변수 값을 지정한다.
getAttributeNames()	java.util.Enumeration	해당 내장객체의 속성(attribute)명을 읽어오는 메소드로 모든 속성의 이름을 얻어낸다.
getAttribute(String key)	Object	해당 내장객체의 속성(attribute)명을 읽어오는 메소드로, 주어진 key매개변수에 해당하는 속성명의 값을 얻어낸다.
removeAttribute(String key)	void	해당 내장객체의 속성(attribute)을 제거하는 메소드로, 주어진 key매개변수에 속성명을 제거한다.

Enumeration 객체 : 객체를 저장하는 객체 컬렉션
- hasMoreElements() : 객체가 있으면 true, 아니면 false 반환
- nextElement() : 처리할 객체를 가져옴

1. Request 객체

- 웹 클라이언트가 웹서버에 전송한 요청과 관련된 정보 제공
- 클래스 타입 : HttpServletRequest
- 제공하는 기능
  - 클라이언트 관련 정보 읽기
  - 서버 관련 정보 읽기
  - 클라이언트가 전송한 데이터 읽기
  - 클라이언트가 전송한 헤더, 쿠키 정보 읽기 등

request 메소드

- request 객체의 메소드들 중 웹 브라우저, 웹 서버 및 요청 헤더의 정보를 가져올 때 사용되는 메소드

메소드	설명
String **getProtocol**()	웹 서버로 요청 시, 사용 중인 프로토콜을 리턴
String **getServerName**()	웹 서버로 요청 시, 서버의 도메인 이름을 리턴
String **getMethod**()	웹 서버로 요청 시, 요청에 사용된 요청 방식(GET, POST, PUT등)을 리턴
String **getQueryString**()	웹 서버로 요청 시, 요청에 사용된 QueryString을 리턴
String **getRequestURI**()	웹 서버로 요청 시, 요청에 사용된 URL로부터 URI값을 리턴
String **getRemoteHost**()	웹 서버로 정보를 요청한 웹 브라우저의 호스트 이름을 리턴
String **getRemoteAddr**()	웹 서버로 정보를 요청한 웹 브라우저의 IP주소를 리턴
int **getServerPort**()	웹 서버로 요청 시, 서버의 Port번호를 리턴
String **getContextPath**()	해당 JSP페이지가 속한 웹 어플리케이션의 컨텍스트 경로를 리턴
String **getHeader**(name)	웹 서버로 요청 시, HTTP요청 헤더(header) 헤더이름 name에 해당하는 속성값을 리턴
Enumeration **getHeaderNames**()	웹 서버로 요청 시, HTTP요청 헤더(header)에 있는 모든 헤더이름을 리턴

실습

웹브라우저와 웹 서버 정보 표시

- 서버명 : localhost
- 전송방식 : GET
- 컨텍스트 경로 : /studyjsp
- URI : /studyjsp/4%EC%A3%BC%EC%B0%A8/requestTest2.jsp
- 클라이언트IP : 0:0:0:0:0:0:0:1
- 프로토콜 : HTTP/1.1
- 서버포트 : 8080

header 정보 표시

- host : localhost:8080
- connection : keep-alive
- upgrade-insecure-requests : 1
- user-agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.149 Safari/537.36
- sec-fetch-dest : document
- accept : text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
- sec-fetch-site : none
- sec-fetch-mode : navigate
- sec-fetch-user : ?1
- accept-encoding : gzip, deflate, br
- accept-language : ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
- cookie : JSESSIONID=B4ADCA380E1768BF4873A4DD51BAD333

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"
    import = "java.util.Enumeration"  %>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>웹브라우저와 웹 서버 정보 표시</title>
</head>
<body>
  <h2>웹브라우저와 웹 서버 정보 표시</h2>
  <hr>
  <ul>
    <li>서버명 : <%=request.getServerName() %></li>
    <li>전송방식 : <%=request.getMethod() %></li>
    <li>컨텍스트 경로 : <%=request.getContextPath() %></li>
    <li>URI : <%=request.getRequestURI() %></li>
    <li>클라이언트IP : <%=request.getRemoteAddr() %></li>
    <li>프로토콜 : <%=request.getProtocol() %></li>
    <li>서버포트 : <%=request.getServerPort() %></li>
  </ul>

  <h2>header 정보 표시</h2>
  <hr>
  <ul>
  <%
    Enumeration<String> en = request.getHeaderNames();
    String headerName="";
    String headerValue="";

    while(en.hasMoreElements()){
      headerName=en.nextElement();
      headerValue = request.getHeader(headerName);
      out.println("<li>" + headerName + " : " +headerValue +"</li>");
    }
  %>
  </ul>
</body>
</html>
```

요청 데이터 얻기

- 사용되는 폼 요소 :
  - 텍스트박스 : text
  - 체크박스 : checkbox
  - 라디오 버튼 : radio
  - 리스트상자 : select

메소드	리턴 타입	설명
getParameter(name)	String	• 파라메터 변수 name에 저장된 변수값을 얻어내는 메소드 • 파라메터 변수 name에 해당하는 변수명이 없으면 null값 리턴
getParameterValues(name)	String[]	• 파라메터 변수 name에 저장된 모든 변수값을 얻어내는 메소드 • 변수값은 String 배열로 리턴. checkbox에서 주로 사용.
getParameterNames()	Enumeration	• 요청에 의해 넘어오는 모든 파라미터 변수를 java.util.Enumeration 타입으로 리턴 • 변수가 가진 객체들을 저장해야 하기 때문에 컬렉션인 Enumeration 타입을 사용

2. response 내장객체

- 웹 브라우저로 응답할 응답 정보를 가지고 있음
- 클래스 타입 : HttpServletResponse
- 주요 기능
  - 응답 헤더 정보 설정
  - 다른 페이지로 강제 이동시키기
  - 쿠키 데이터 기록

주요 메소드	설명
void **setHeader**(name, value)	헤더정보의 값을 수정하는 메소드
void **setContentType**(type)	웹 브라우저의 요청의 결과로 보여질 페이지의 contentType을 설정한다.
void **sendRedirect(url)**	url로 주어진 페이지로 제어가 이동
addCookie(Cookie cookie)	쿠키 데이터 기록
addHeader(String name, String value)	Response 객체의 헤더 내용 기록
setStatus(int status-code)	HTTP response 상태 코드 설정
encodeURL(String url)	세션ID와 URL을 전송코드로 변화

3. out 내장객체

- JSP페이지의 결과를 웹 브라우저에 전송해 주는 출력 스트림
- 클래스 타입 : javax.servlet.jsp.JspWriter
- 웹 브라우저로 보내는 정보는 out객체로 통해서 전송
  - 모든 정보는 JSP 스크립트요소 뿐만 아니라 비 스크립트요소인 HTML, 일반 텍스트도 모두 포함된다.
- 스크립트릿 내에 표현식<%= %> 사용 불가
  - JSP에서는 out.println() 사용

주요 메서드	설명
print(content)	데이터 출력
println(content)	데이터 출력
getBufferSize()	버퍼 크기 리턴
ClearBuffer()	버퍼 내용 전송하지 않고 비움
close()	버퍼 내용을 브라우저에 전송하고 스트림을 닫아줌
flush()	버퍼 내용을 브라우저에 전송하고 비움
getRemaining()	현재 버퍼 크기 리턴

4. session 객체

- 웹 브라우저의 요청 시, 요청한 웹 브라우저에 관한 정보를 저장하고 관리하는 내장 객체
- 클래스 타입 : javax.servlet.http.HttpSession
- 웹 브라우저(클라이언트)당 1개가 할당
- 주로 **사용자 인증에 관련된 작업**을 수행할 때 사용

주요 메서드	설명
String **getId**()	해당 웹 브라우저에 대한 고유한 세션 ID를 리턴
long **getCreationTime**()	해당 세션이 생성된 시간을 리턴
long **getLastAccessedTime**()	웹 브라우저의 요청이 시도된 마지막 접근시간을 리턴
void **setMaxInactiveInterval**(time)	해당 세션을 유지할 시간을 초단위로 설정
int **getMaxInactiveInterval**()	기본값은 30분으로 setMaxInactiveInterval(time) 로 지정된 값을 리턴
boolean **isNew**()	현재의 웹 브라우저가 새로 불려진 즉, 새로 생성된 세션의 경우 true 값을 리턴
void **invalidate**()	현재 정보의 유지로 사용 시, 설정된 세션의 속성 값을 모두 제거한다. 주로 세션을 무효화시킬 때 사용

5. application 객체

- 웹 어플리케이션의 설정 정보를 가진 객체
- 클래스 타입 : javax.servlet.ServletContext
- 특정 웹어플리케이션에 포함된 모든 JSP페이지는 하나의 application객체 공유
  – 공유 변수로 사용됨
- 웹 어플리케이션당 1개의 application객체가 생성
- 웹 어플리케이션이 실행되는 **서버 설정 정보 및 자원에 대한 정보**를 얻어내거나 어플리케이션이 실행되고 있는 동안에 발생할 수 있는 **이벤트 로그 정보와 관련된 기능**들을 제공

주요 메소드	설명
String **getServerInfo**()	웹 컨테이너의 이름과 버전을 리턴
String **getMimeType**(fileName)	지정한 파일의 MIME 타입을 리턴
String **getRealPath**(path)	지정한 경로를 웹 어플리케이션 시스템상의 경로로 변경하여 리턴
void **log**(message)	로그 파일에 message를 기록

[참고] 내장객체의 유효범위(scope)

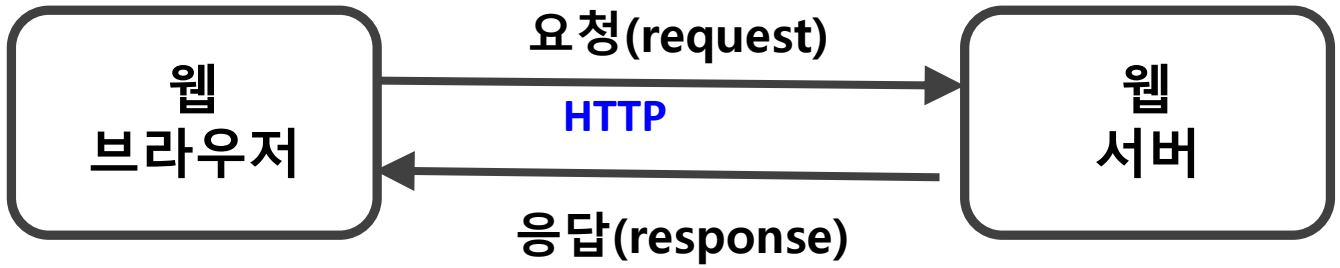
- 내장 객체의 유효 범위
- 스코프 필요한 이유
  - HTTP 프로토콜의 상태정보를 유지할 수 없음(stateless, connectionless)
  - 웹 상에서 상태정보를 유지하기 위한 방법

page < request < session < application

스코프의 종류	설명	유효 범위	JSP 내장 객체	활용 방법
page scope	pageContext 객체를 통해 접근할 수 있는 영역이다.	요청된 page 내부(현재 페이지)	pageContext	JSP 페이지의 지역변수
request scope	웹 브라우저의 한번의 요청에 단지 한 개의 페이지만 요청될 수 있고, 같은 request영역이면 두 개의 페이지가 같은 요청을 공유(forward)	요청부터 응답까지(한페이지만 넘겨받음)	request	forward를 통해 데이터 전달
session scope	세션이 유지되고 있는 동안 관련된 JSP 페이지들이 속성에 접근할 수 있다. 같은 웹 브라우저 내에서는 요청되는 페이지들은 같은 객체 공유.	- 해당 웹브라우저의 종료 - 설정 한 시간 (웹 브라우저)	session	사용자 별로 정보 저장
application scope	웹 애플리케이션이 실행되고 있는 동안 모든 JSP 페이지가 속성에 접근할 수 있다.	웹 애플리케이션의 종료	application	전체 접속자 수 등

[참고] HTTP 동작 방식

- 웹 서버와 웹클라이언트 간에 통신(요청, 응답)하기 위한 규약(프로토콜)
- TCP/IP 4계층에서 응용계층(Application Layer)에 해당
- 웹 클라이언트는 HTTP에 맞게 요청(**request**)를 웹 서버에 전송하고, 웹 서버는 응답(**response**)을 HTTP에 맞게 웹 클라이언트로 전송



[참고] HTTP 특징

- 인터넷 환경에 가장 적합한 통신 구조
  - 요청/응답
    - HTTP는 클라이언트가 서버에 접속한 후 요청 메시지를 보내면, 서버는 요청메세지를 근거로 서비스를 처리한 다음 결과인 응답메세지를 클라이언트에 보냄으로써 통신한다.
  - **무연결**(Connectionless)
    - HTTP는 클라이언트로부터 요청이 들어와 서버가 응답하면 클라이언트와 서버의 연결을 끊고 클라이언트가 새로 요청하면 또 다른 연결을 맺는다. 이때 연결은 이전과 아무 상관이 없는 새로운 연결이다.
  - **무상태**(Stateless)
    - 하나의 요청을 독립적 트랜잭션으로 취급하기 때문에 이전 연결에서 했던 작업내용을 다음 요청에서 사용할 수 없다.
    - 요청 마다 다른 연결로 인식되어 요청들 간 정보를 공유할 수 없는 상태, 즉 **상태 정보가 유지되지 않는 특성**을 가진다.
    - 이는 인터넷 상에서 서버의 오버헤드를 줄일 수 있는 장점이 있다.

[참고] HTTP 요청 방식

처리방식	설명
GET	• 웹서버에 데이터를 요청할 때 사용 • 가장 단순한 요청으로 서버에 빠른 속도로 요청시 사용 • 서버로 전달되는 정보는 URI 뒤에 ?기호와 함께 전달(Query String) • 전달되는 문자열의 크기에 제한 있음
POST	• 웹서버에 데이터를 전달하기 위해 사용 • 데이터는 HTTP 요청 정보의 엔티티 바디에 포함되어 전달 • 데이터의 크기에 제한이 없고 화면에 노출되지 않음 • GET보다 처리 속도 늦음
PUT	• 파일 업로드를 할 때 이용 • 서버의 리소스 수정에 사용
DELETE	• 서버 리소스 삭제 작업 요청에 사용
OPTIONS	• 요청 URI에 대해 허용되는 통신 옵션을 알고자 할 때 사용
HEAD	• GET과 같으나 요청정보의 몸체없이 헤더 정보만 요청
TRACE	• 요청정보가 웹서버에 도달하기까지의 경로 기록. 서버상태 확인을 위한 목적
CONNECT	• 프락시에 사용하기 위해 예약된 메소드로서 프락시가 동적으로 접속할 수 있게 지원

HTTP 요청정보

- 요청(request) 정보 구조(크롬 개발자도구 : F12)

요청라인 → <method><resource identifier><HTTP version><CRLF>
헤더 → [<header>:<value>] < CRLF >
.....
< CRLF >
엔티티 바디 → [entity body]

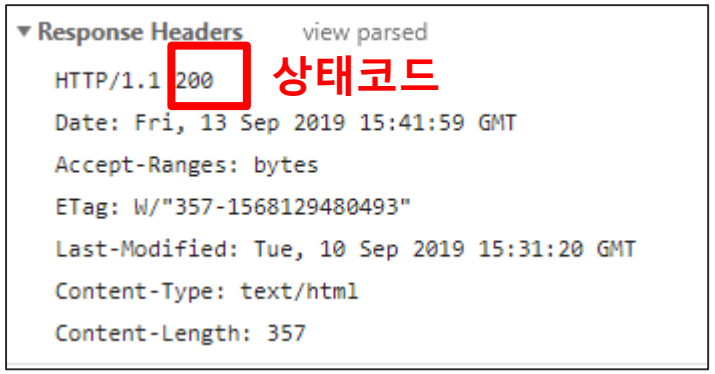
CRLF(carriage return line formfeed)
: 한줄 띠우기

크롬 개발자도구의 >>network<< 에서 확인

HTTP 응답 정보

- 응답(response) 정보 구조

응답라인 → <HTTP version><result code>[<explanation>]< CRLF >
헤더 → [<header>:<value>] < CRLF >
.....
< CRLF >
엔티티 바디 → [entity body]




```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
</head>
<body>
        <form action ="calc" method="post">
        x : <input type="text" name ="x"><br>
        y : <input type="text" name ="y"><br>
        <input type="submit" value="덧셈" name="op">
        <input type="submit" value="뺄셈" name="op">
    </form>
</body>
</html>
```

HTTP 응답 정보

- 주요 상태 코드(result code)

HTTP 상태코드	코드	설명
1xx	정보	클라이언트로부터 일부분만 받았으니 나머지 요청 정보 요청
200	OK	에러 없이 전송함
3xx	경로변경	요청을 완전히 처리하기 위해 추가적 액션이 수행되어야 함을 의미
400	Bad Request	요청 실패- 문법상 오류로 서버가 요청사항을 이해하지 못함
401	Unauthorized	인증 오류로 클라이언트가 잘못된 정보를 헤더에 넣었음을 의미
403	Forbidden	사용자 허가 모드 오류로 클라이언트의 인증 정보에 상관없이 페이지 접근을 거부
404	Not found	클라이언트가 요청한 문서가 존재하지 않음
405	Method Not Allowed	클라이언트가 요청한 서비스 요청 방식을 웹 서버에서 지원하지 않음
5xx	Internal Server Error	정당한 요청을 서버가 처리하지 못함을 의미

[참고]

- **URL**(Uniform Resource Locator) : 네트워크 상에 존재하는 자원을 찾아가기위한 정보
- **URI**(Uniform Resource Identifier) : 인터넷에 있는 자원을 식별하기 위한 문자열 구성으로 URL+URN

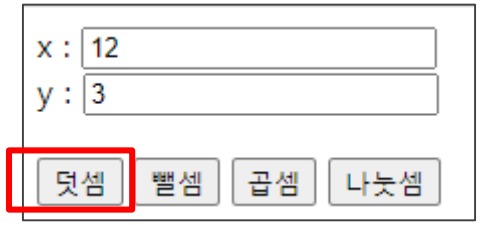
http://test.com:8080/test.jsp?id=111

URL **URI**

- **프로토콜** :
  - 서버와 통신하기 위한 규약으로 서버마다 사용하는 프로토콜이 정해져 있음.
  - 웹 서버는 http:// 로 표현
- **서버주소** : IP주소 또는 도메인 이름
- **포트번호** :
  - 컴퓨터에서 동작하고 있는 서버 접속을 위한 정보.
  - 0~65,535까지 사용 가능.
  - 보통 0~1023의 번호는 well-known port로 예약되어 있음

실습문제

- 다음은 간단한 계산기이다. 그 결과화면을 구현하시오.



덧셈 결과는 : 15

나눗셈 불능입니다. 0으로 나눌 수 없습니다.

나눗셈의 경우는 결과값이 실수(float)이므로 이에 대한 형변환이 필요

[과제]

마감일은 항상 과제가 나간 날에서 일주일 후 밤 12:00까지!!!
마감 기한을 지나면 하루에 20%씩 점수 깎임

- 각자 자신의 github repository에 각자의 이름과 제출일을 폴더로 만들어 파일들을 제출할 것
  - 예 : WebContent/홍길동-220310/숙제한파일들...

1. 한 학생의 국어, 영어, 수학 점수를 입력받아 세 과목의 점수와, 총점, 평균을 출력하는 프로그램을 작성하시오.
   - 학생의 점수를 입력받는 폼은 예제 4-1.html을 그대로 사용한다.
   - 넘겨받은 점수, 그리고 총점과 평균을 출력하는 프로그램은 예제 4-2.jsp를 수정해서 작성한다.
   - request.getParameter()로 얻어낸 입력값은 문자열이다. 따라서 이렇게 얻어낸 문자열에, 정수는 Interger.parseInt(), 실수는 Float.parseFloat() 등의 메서드를 사용해서 숫자로 바꾸어 계산하여야 한다.
   - 평균은 소수점 아래 2자리까지 출력합니다. 다음과 같이 하면 소수점 3번째 자리에서 반올림되어 소수점 2번째 자리까지 숫자가 있는 문자열을 얻을 수 있다.

     String.format("%.2f", 평균값)
2. 게시판 글쓰기를 위한 폼을 설계하고 입력된 값을 출력하는 JSP 프로그램을 작성하시오.
3. 회원 가입 양식(단일 값 입력 태그들과 다중 선택 입력 태그를 모두 활용)을 만들고, 입력된 값을 출력하는 JSP 프로그램을 작성하시오.