# JSP 프로그래밍

(재)대덕인재개발원

# 03 JSP로 시작하는 웹 프로그래밍

# 3.1 JSP에서 HTML 생성하는 기본 코드 구조

```
<%@ page contentType="text/html;  charset=euc-kr" %>
<%@ page import="java.util.Date" %>
```

설정 부분
: JSP 페이지에 대한 설정 정보

```
<html>
<head><title>리스트1.1</title></head>
<body>

<%
   Date nowTime = new Date();
%>

현재 시간은 <%= nowTime %>입니다.

</body>
</html>
```

생성 부분
: HTML 코드 및 JSP 스크립트

# 3.1 JSP에서 HTML 생성하는 기본 코드 구조

```
<%@ page contentType="text/html;  charset=euc-kr" %>

<%@ page import="java.util.Date" %>
```

디렉티브

```
<%
    Date nowTime = new Date();
%>
```

스크립틀릿

```
<html>
<head><title>리스트1.1</title></head>
<body>

현재 시간은 <%= nowTime %>입니다.

</body>
</html>
```

표현식

# 3.2 JSP 페이지의 구성요소

- 정적인 데이터
- 디렉티브
- 스크립트 요소
  - 스크립트릿(Scriptlet), 표현식(Expression), 선언부(Declaration)
- 기본 객체(Implicit Language)
- 표준 액션 태그(Action Tag)
- 표현언어(Expression Language)
- 커스텀 태그(Custom Tag)와 표준 태그 라이브러리(JSTL)

# 3.2.1 JSP 페이지의 구성요소

- 디렉티브(Directive)
  - JSP 페이지에 대한 설정 정보를 지정할 때 사용한다.
  - <%@ 디렉티브이름 속성1="값1" 속성2="값2" …. %>
  - <%@ page contentType="text/html; charset=euc-kr" %>

| 디렉티브 | 설 명 |
|---|---|
| page | JSP 페이지에 대한 정보를 지정한다. JSP가 생성하는 문서의 타입, 출력 버퍼의 크기, 에러 페이지 등 JSP 페이지에서 필요로 하는 정보를 입력한다. |
| taglib | JSP 페이지에서 사용할 태그 라이브러리를 지정한다. |
| include | JSP 페이지의 특정 영역에 다른 문서를 포함시킨다. |
| | |

# 3.2.2 JSP 페이지의 구성요소

- 스크립트 요소
  - JSP 에서 실시간으로 문서의 내용을 생성하기 위해 사용되는 것이 스크리트 요소 이다.
  - 사용자가 폼에 입력한 정보를 데이터베이스에 저장 할 수 있으며, 데이터베이스로 부터 글을 읽어와 출력할 수도 있다.

| 스크립트 요소 | 설 명 |
|---|---|
| 표현식(Expression) | 값을 출력한다. |
| 스크립트릿(Scriptlet) | 자바 코드를 실행한다. |
| 선언부(Declaration) | 자바 메서드(함수)를 만든다. |
| | |

# 3.2.3 JSP 페이지의 구성요소

- 기본 객체(내장 객체)
  - JSP는 웹 어플리케이션 프로그래밍을 하는데 필요한 기능을 제공해 주는 '기본 객체(implicit object)'를 제공하고 있다.
  - request, response, session, application, page 등.
  - 각각 요청 패러미터 읽기, 응답 결과 전송, 세션 처리, 웹 어플리케이션 정보 읽어오기 등의 기능을 제공.

# 3.2.4 JSP 페이지의 구성요소

- 표현 언어(Expression Language; EL)
  - JSP 페이지 내부에서 사용되는 간단한 스크립트 언어이다. (JSP 2.0 이후)
  - JSP의 스크립트 요소(스크립트릿과 표현식)를 대신해 쉽고 간단하게 사용할 수 있다.

# 3.2.5 JSP 페이지의 구성요소

- 표준 액션 태그와 태그 라이브러리
  - 액션 태그는 XML의 태그와 같은 모양을 취하며, JSP 페이지에서 특별한 기능을 제공한다.
  - <jsp:액션태그이름>의 형태를 띠며 액션 태그 종류에 따라서 서로 다른 속성과 값을 갖는다.
  - <jsp:include page="header.jsp" flush="true" />
  - 위의 액션 태그는 특정 페이지의 실행 결과를 현재 위치에 포함시킬 때 사용한다.
- 커스텀 태그
  - JSP를 확장시켜 주는 기능으로서, 액션 태그와 마찬가지로 태그 형태로 기능을 제공한다.
  - 둘의 차이점은 커스텀 태그는 개발자가 직접 개발해 주어야 한다는 것이다.
  - 커스텀 태그는 JSP코드에서 중복되는 것들을 모듈화 하거나 스크립트 코드 사용시 소스 코드의 복잡함을 없애기 위해 사용된다.
  - 커스텀 태그 중 자주 사용되는 것들을 별도로 표준화한 태그 라이브러리가 바로 JSTL(Java Server Pages Standard Tag Library) 이다.

# 3.3 page 디렉티브

- page 디렉티브(Directive)
  - JSP 페이지에 대한 설정 정보를 지정할 때 사용한다.
  - JSP 페이지가 어떤 문서를 생성하는지, 어떤 자바 클래스를 사용하는지, 세션에 참여여부, 출력 버퍼의 존재 여부와 같은 실행에 필요한 정보를 입력한다.
  - <%@ page contentType="text/html; charset=euc-kr" %>
  - <%@ page import="java.util.Date" %>

| 속성 | 설 명 | 기본값 |
|---|---|---|
| contentType | JSP페이지가 생성할 문서의 타입을 지정.<br>생성할 응답 문서의 MIME 타입을 입력한다.<br>"text/html","text/xml","text/plain" 등. | text/html |
| import | JSP 페이지에서 사용할 자바 클래스를 지정. | |
| trimDirectiveWhitespace | 출력 결과에서 템플릿 텍스트의 공백 문자를 제거할지의 여부를 지정한다. | false |
| pageEncoding | JSP 페이지 자체의 캐릭터 인코딩을 지정한다. | |

# 3.3 page 디렉티브

| 속성 | 설 명 | 기본값 |
|---|---|---|
| contentType | JSP페이지가 생성할 응답 데이터의 MIME 타입을 설정<br>“text/html;charset=UTF-8”, “text/xml”, “text/plain” 등 | text/html |
| import | JSP 페이지에서 사용할 자바 클래스를 지정 | |
| session | JSP 페이지에서 세션 사용 여부 설정 | true |
| buffer | JSP 페이지의 출력 버퍼 크기 설정<br>none 로 설정시 버퍼를 사용하지 않음 | 8kb |
| autoFlush | 출력버퍼가 다 찬 경우, 자동 방출 여부를 설정 | true |
| info | JSP 페이지에 대한 설명 | |
| errorPage | 에러가 발생할 경우 대체 페이지를 설정 | |
| isErrorPage | JSP 페이지가 에러를 처리할 에러페이지 인지 여부 설정<br>true 인 경우 exception 기본 객체를 사용할수 있음 | false |
| pageEncoding | JSP 페이지 자체의 캐릭터 인코딩을 지정 | |
| isELIgnored | JSP 페이지의 표현언어 지원 여부 설정 | false |
| deferredSyntaxAllowedAsLiteral | #{ } 문자를 deffered el 기호로 사용할지 여부 설정 | false |
| trimDirectiveWhitespace | 출력 결과에서 템플릿 텍스트의 공백 문자를 제거할지의 여부를 지정한다. | False |

# 3.4 스크립트 요소

- 스크립트 요소는 JSP 프로그래밍에서 로직을 수행하는 데 필요한 부분으로 프로그램이 수행해야 하는 기능을 구현할 수 있다.
  - 스크립트릿(Scriptlet)
  - 표현식(Expression)
  - 선언부(Declaration)

| 스크립트 요소 | 설 명 | 문법 구조 |
|---|---|---|
| 스크립트릿 | JSP 페이지에서 자바 코드를 실행할 때 사용되는 코드의 블록이다. | <%<br>자바 코드1;<br>자바 코드2;<br>자바 코드3;<br>%> |
| 표현식 | 어떤 값을 출력 결과에 포함시키고자 할 때 사용된다. | <%= 값 %> |
| 선언부 | JSP 페이지에서 스크립트릿이나 표현식에서 사용할 수 있는 함수를 작성할 때 사용된다. | <%!<br>public 리턴타입 메소드명(패러미터 목록){<br>자바 코드들…<br>}<br>%> |

# 3.5 request 기본 객체

- request 기본 객체
  - JSP 페이지에서 가장 많이 사용되는 기본 객체로서 웹 브라우저의 요청과 관련이 있다.
  - 클라이언트가 전송한 요청 정보를 제공하는 것이 바로 request 기본 객체이다.
- request 기본 객체가 제공하는 기능
  - 클라이언트(웹 브라우저)와 관련된 정보 읽기 기능
  - 서버와 관련된 정보 읽기 기능
  - 클라이언트가 전송한 요청 파라미터 읽기 기능
  - 클라이언트가 전송한 요청 헤더 읽기 기능
  - 클라이언트가 전송한 쿠키 읽기 기능
  - 속성 처리 기능

# 3.5.1 클라이언트 정보 및 서버 정보 읽기

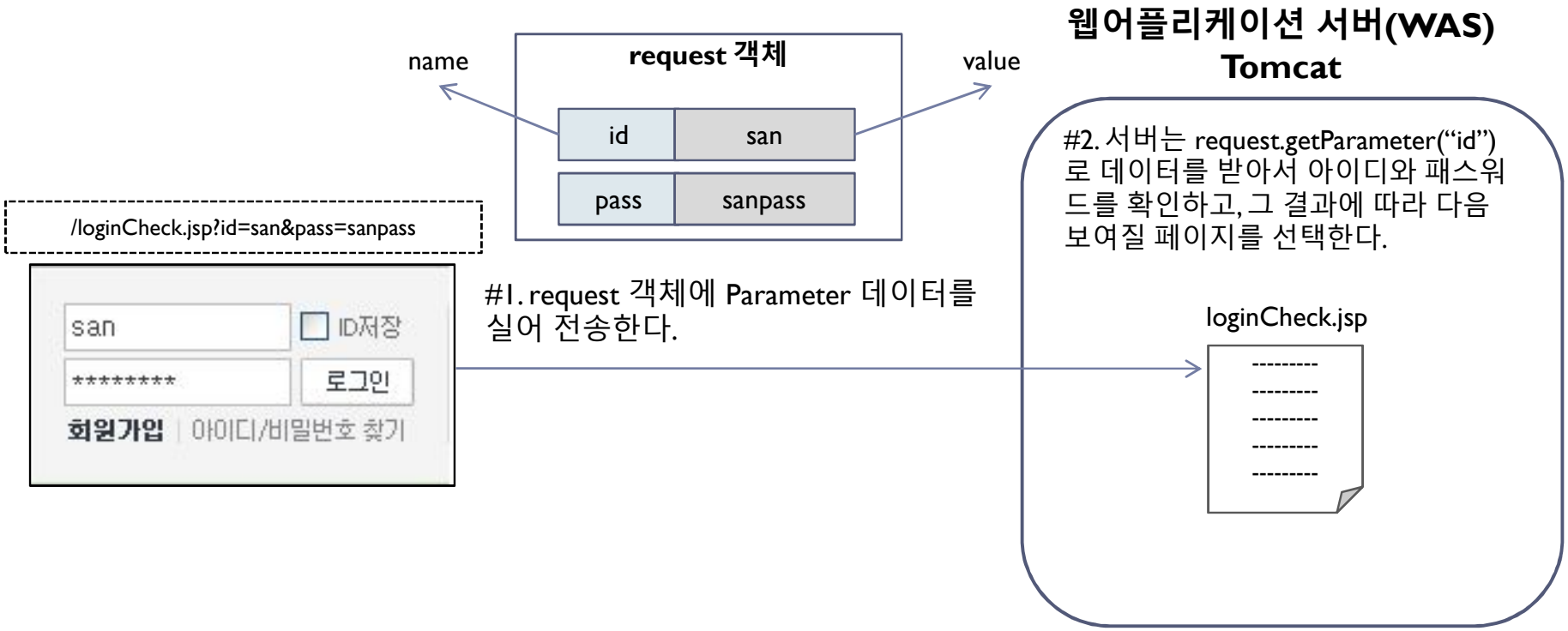
- request 기본 객체의 클라이언트 및 서버 정보 관련 메소드

| 속성 | 리턴 타입 | 설 명 |
|---|---|---|
| getRemoteAddr() | String | 웹서버에 연결한 클라이언트의 IP 주소를 구한다. 게시판이나 방명록등에서 글 작성자의 IP 주소가 자동으로 입력되기도 하는데, 이때 입력되는 IP 주소가 바로 이 메서드를 사용하여 구한 것이다. |
| getContentLength() | long | 클라이언트가 전송한 요청 정보의 길이를 구한다. 전송된 데이터의 길이를 알 수 없는 경우 -1을 리턴한다. |
| getCharacterEncoding() | String | 클라이언트가 요청 정보를 전송할 때 사용한 캐릭터의 인코딩을 구한다. |
| getContentType() | String | 클라이언트가 요청 정보를 전송할 때 컨텐트의 타입을 구한다. |
| getProtocol() | String | 클라이언트가 요청한 프로토콜을 구한다. |
| getMethod() | String | 웹 브라우저가 정보를 전송할 때 사용한 방식을 구한다. |
| getRequestURI() | String | 웹 브라우저가 요청한 URL에서 경로를 구한다. |
| getContextPath() | String | JSP 페이지가 속한 웹 어플리케이션의 컨텍스트 경로를 구한다. |
| getServerName() | String | 연결할 때 사용한 서버 이름을 구한다. |
| getServerPort() | int | 서버가 실행 중인 포트 번호를 구한다. |

# 3.5.2 HTML 폼과 요청 패러미터의 처리

- 웹 브라우저는 폼에 입력한 정보를 패러미터로 전송한다.
- request 기본 객체는 웹 브라우저가 전송한 패러미터를 읽을 수 있는 메서드를 제공한다.

| 속성 | 리턴 타입 | 설 명 |
|---|---|---|
| getParameter(String name) | String | 이름이 name인 패러미터의 값을 구한다. 존재하지 않을 경우 null을 리턴한다. |
| getParameterValues(String name) | String[] | 이름이 name인 모든 패러미터의 값을 배열로 구한다. 존재하지 않을 경우 null을 리턴한다. |
| getParameterNames() | java.util.Enumeration | 웹 브라우저가 전송한 패러미터의 이름을 구한다. |
| getParameterMap() | java.util.Map | 웹 브라우저가 전송한 패러미터의 맵을 구한다. 맵은 <패러미터 이름, 값> 쌍으로 구성된다. |


웹어플리케이션 서버(WAS)
Tomcat
request 객체
name
value
id
san
pass
sanpass
/loginCheck.jsp?id=san&pass=sanpass
san
ID저장
********
로그인
회원가입 | 아이디/비밀번호 찾기
#1. request 객체에 Parameter 데이터를 실어 전송한다.
#2. 서버는 request.getParameter("id") 로 데이터를 받아서 아이디와 패스워드를 확인하고, 그 결과에 따라 다음 보여질 페이지를 선택한다.
loginCheck.jsp

# 3.5.2 HTML 폼과 요청 패러미터의 처리

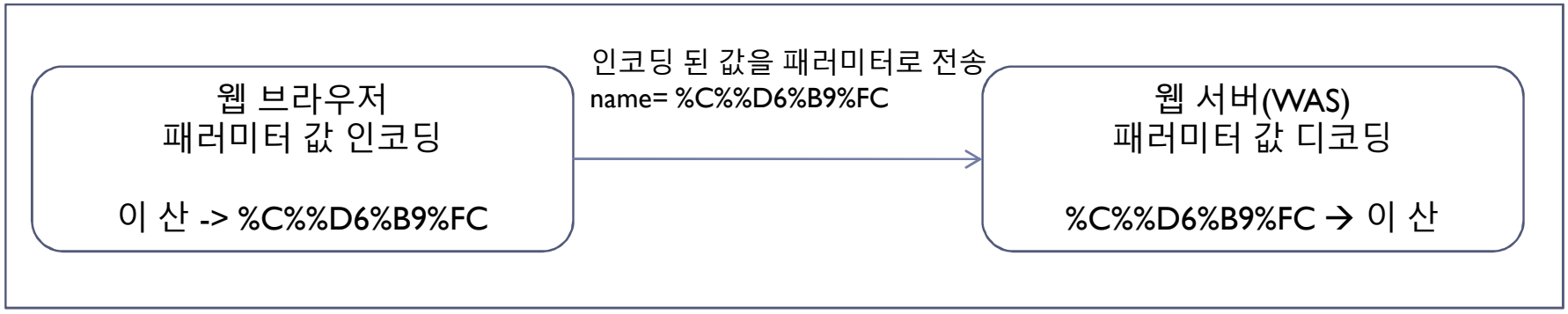
- GET 방식 전송과 POST 방식 전송

- GET 방식
  - 요청 URL에 물음표('?')와 함께 패러미터를 붙여서 전송.
  - ?이름1=값1&이름2=값2&….&이름n=값n
  - 폼을 사용하지 않더라도 패러미터를 전송할 수가 있다.
  - http://localhost:8080/chap03/viewParameter.jsp?name=san&addresss=deajeon
  - 웹 브라우저, 웹 서버, 웹 컨테이너에 따라 전송할 수 있는 길이에 제한됨.

- POST 방식
  - HTTP 프로토콜의 데이터(content) 영역을 이용해서 패러미터를 전송.
  - 전송할 패러미터 길이에 제한이 없다.

# 3.5.2 HTML 폼과 요청 패러미터의 처리

- 패러미터 값의 인코딩 처리
  - 웹 브라우저는 웹 서버에 패러미터를 전송할 때 알맞은 캐릭터 셋을 이용해서 파래미터 값을 인코딩 한다.
  - 반대로 웹 서버는 알맞은 캐릭터 셋을 이용해서 웹 브라우저가 전송한 패러미터 값을 디코딩 한다.
- POST 방식으로 패러미터 전송시
  - POST 방식에서는 입력 폼을 보여주는 응답 화면이 사용하는 캐릭터 셋을 사용한다.(euc-kr ==> euc-kr)
  - 서버에서는 request.setCharacterEncoding() 메서드를 이용해서 디코딩 캐릭터 셋을 지정할 수 있다.



# 3.5.2 HTML 폼과 요청 패러미터의 처리

- GET 방식으로 패러미터 전송 시 인코딩 결정 규칙

| GET 방식 이용 시 패러미터 전송 방법 | 인코딩 결정 |
|---|---|
| <a> 태그의 링크 태그에 쿼리 문자열 추가 | 웹 페이지 인코딩 사용 |
| HTML 폼(FORM)의 method 속성값을 "GET" 으로 지정해서 폼을 전송 | 웹 페이지 인코딩 사용 |
| 웹 브라우저에 주소에 직접 쿼리 문자열 포함한 URL 입력 | 웹 브라우저 마다 다름. |

# 3.5.2 HTML 폼과 요청 패러미터의 처리

- 톰캣에서 GET 방식 패러미터를 위한 인코딩 처리하기
  - 1. 문제점
  - WAS 마다 GET 방식으로 전달되는 패러미터 값을 읽어올 때 사용하는 기본 캐릭터 셋이 다르다.
  - GET 방식으로 전송된 패러미터에 대해서는 request.setCharacterEncoding() 메서드로 적용이 안된다.
  - 2. 해결책
  - server.xml 파일에서 <Connector>의 useBodyEncodingForURI 속성의 값을 true로 지정하는 방법
  - server.xml 파일에서 <Connector>의 URIEncoding 속성의 값으로 원하는 캐릭터 셋을 지정하는 방법

# 3.5.2 HTML 폼과 요청 패러미터의 처리

- 톰캣에서 GET 방식 패러미터를 위한 인코딩 처리하기
  - useBodyEncodingForURI 속성값을 "true"로 지정하면 GET 방식으로 전달된 패러미터 값을 읽어올 때 request.setCharacterEncoding() 메서드로 지정한 캐릭터 셋이 적용된다.
  - URIEncoding 속성을 사용할 경우 GET 방식의 패러미터는 항상 지정한 캐릭터 셋을 지정한다. 이 경우, request.setCharacterEncoding() 메서드는 적용되지 않는다.

```
# fileName = conf\server.xml

<Connetor port="8080" protocol="HTTP/1.1"
          connectionTimeout="20000"
          redirectPor="8443"
          useBodyEncodingForURI="true"
          URIEncoding="utf-8" />
```

# 3.5.3 요청 헤더 정보의 처리

- HTTP 프로토콜은 헤더 정보에 부가적인 정보를 담도록 하고 있다.
- 웹 브라우저는 웹 브라우저의 종류에 대한 정보를 헤더에 담아서 전송한다.
- request 기본 객체는 이러한 헤더 정보를 읽어올 수 있는 기능을 제공한다.

| 메서드 | 리턴 타입 | 설 명 |
|---|---|---|
| getHeader(String name) | String | 지정한 이름의 헤더 값을 구한다. |
| getHeaders(String name) | Enumeration | 지정한 이름의 헤더 목록을 구한다. |
| getHeaderNames() | Enumeration | 모든 헤더의 이름을 구한다. |
| getIntHeader(String name) | int | 지정한 헤더의 값을 정수 값으로 읽어온다. |
| getDateHeader(String name) | long | 지정한 헤더의 값을 시간 값으로 읽어온다.(이때 시간은 1970년 1월 1일 이후로 흘러간 1/1000 초 단위의 값을 가진다.) |

# 3.6 response 기본 객체

- response 기본 객체는 request 기본 객체와 반대의 기능을 수행.
- response 기본 객체는 웹 브라우저에 보내는 응답 정보를 담는다.
- 웹 브라우저에 헤더 정보 전송하기
- 웹 브라우저 캐시 제어를 위한 응답 헤더 입력
- 리다이렉트를 이용해서 페이지 이동하기

# 3.6 response 기본 객체

- 웹 브라우저에 헤더 정보 전송하기
- request 기본 객체는 요청 정보에서 헤더를 읽어오는 기능을 제공하는데, response 기본 객체는 반대로 응답 정보에 헤더를 추가하는 기능을 제공하고 있다.

| 메서드 | 리턴 타입 | 설 명 |
|---|---|---|
| addDateHeader(String name, long date) | void | name 헤더에 date 를 추가한다. date는 1970년 1월 1일 이후 흘러간 시간을 1/1000 초 단위로 나타낸다. |
| addHeader(String name, String value) | void | name 헤더에 value를 값으로 추가한다. |
| addIntHeader(String name, int value) | void | name 헤더에 정수 값 value를 추가한다. |
| setDateHeader(String name, long date) | void | name 헤더에 date 를 지정한다. date는 1970년 1월 1일 이후 흘러간 시간을 1/1000 초 단위로 나타낸다. |
| setHeader(String name, String value) | void | name 헤더에 value를 값으로 지정한다. |
| setIntHeader(String name, int value) | void | name 헤더에 정수 값 value를 지정한다. |
| containsHeader(String name) | boolean | 이름이 name인 헤더를 포함하고 있을 경우 true, 그렇지 않을 경우 false를 리턴한다. |

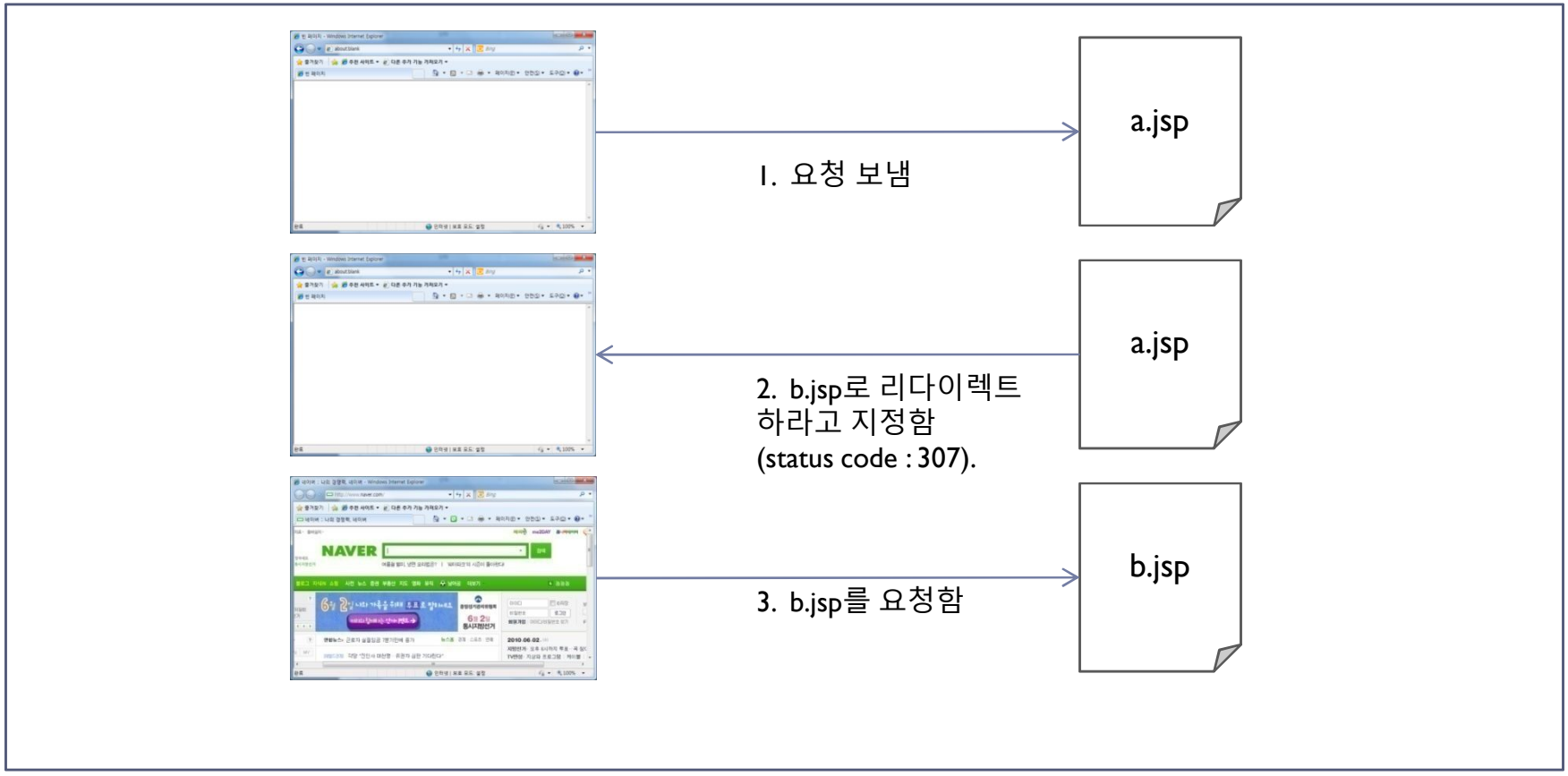
# 3.6 response 기본 객체

- 웹 브라우저 캐시 제어를 위한 응답 헤더 입력
  - HTTP는 특수한 응답 헤더를 통해서 웹 브라우저가 응답 결과를 캐시 할 것인지에 대한 여부를 설정할 수 있다.
  - Cache-Control 응답 헤더 : HTTP 1.1 버전에서 지원하는 헤더로서, 이 헤더의 값을 "no-cache"로 지정하면 웹 브라우저는 응답 결과를 캐시에 저장하지 않는다.
  - Pragma 응답 헤더 : HTTP 1.0 버전에서 지원하는 헤더로서, 이 헤더의 값을 "no-cache"로 지정하면 웹 브라우저는 응답 결과를 캐시에 저장하지 않는다.

```
<%
    response.setHeader("Pragma","no-cache");
    response.setHeader("Cache-Control","no-cache");
    // 일부 파이어폭스 버그 관련
    response.setHeader("Cache-Control","no-store");

    response.setDateHeader("Expires", 1L);
%>
```

# 3.6 response 기본 객체

- 리다이렉트를 이용해서 페이지 이동하기
  - 웹 서버가 웹 브라우저에게 다른 페이지로 이동하라고 지시하는 것을 의미한다.



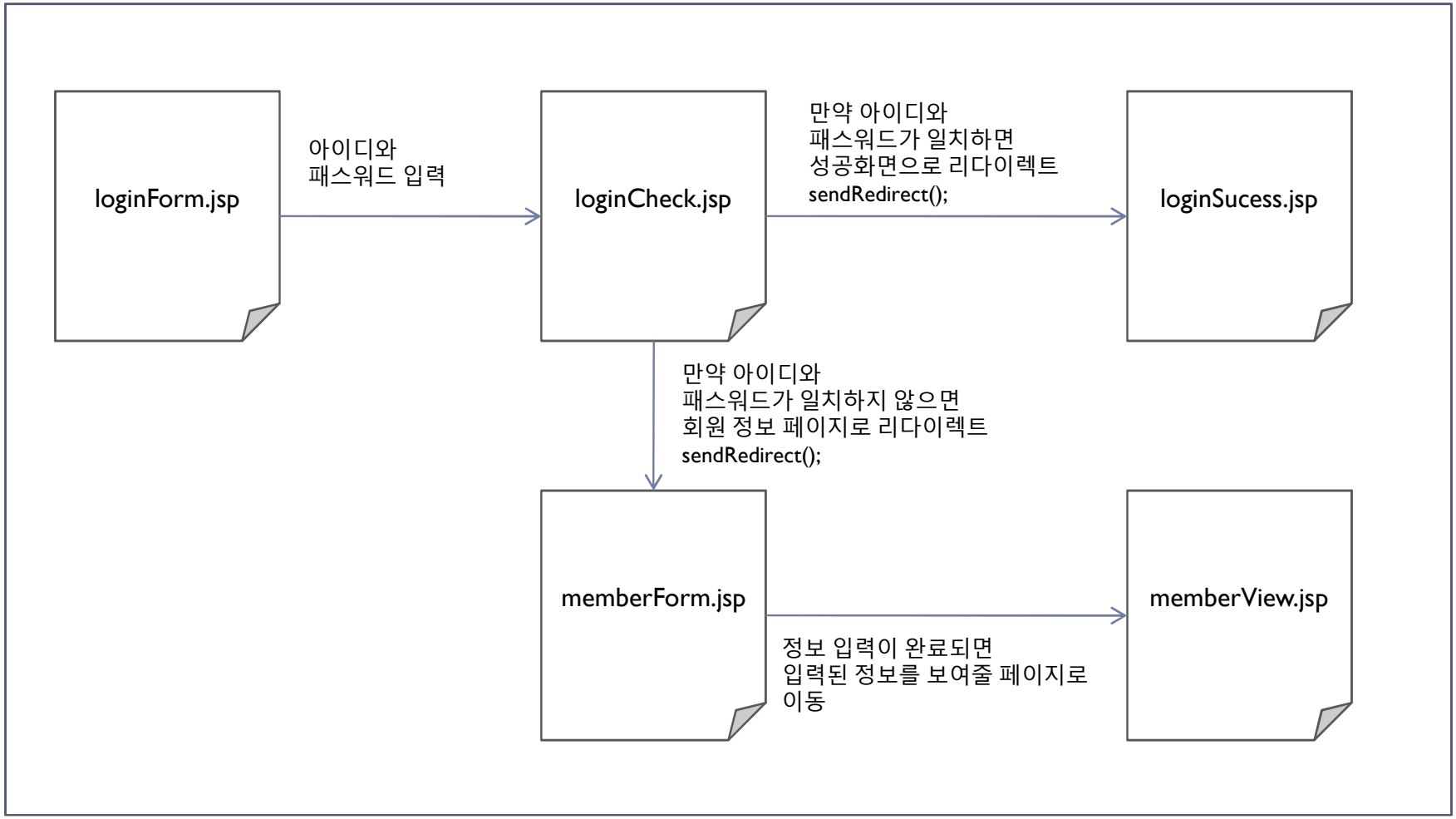
# ** Request Dispatch 방식

# 3.6 response 기본 객체

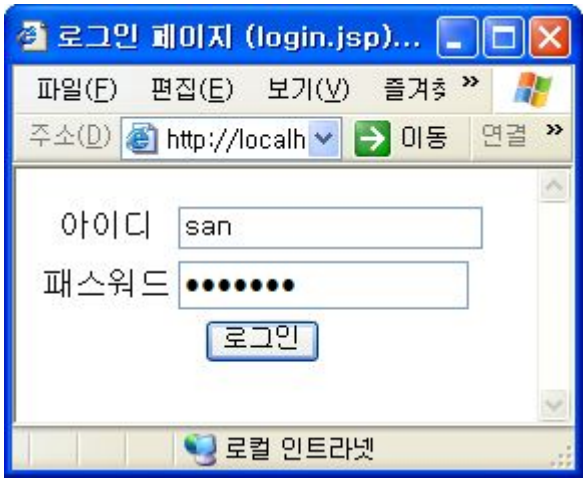
- 리다이렉트를 이용해서 페이지 이동하기
  - response.sendRedirect(String location) 메서드를 이용.

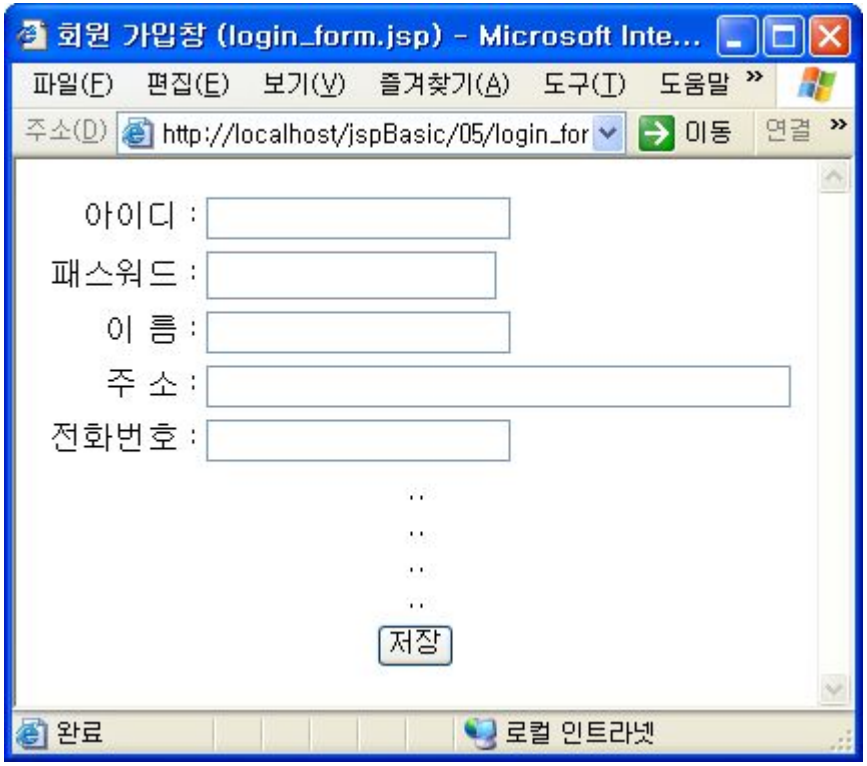
```
<%@ page import="java.net.URLEncoder" %>
<%@ page pageEncoding="utf-8" %>
<%
    // JSP 페이지에서 필요한 코드를 실행한다.
    …
    …
    response.sendRedirect("이동할 페이지");

    // URL에 패러미터 값 인코딩 필요시 URLEncoder 클래스 이용.
    String value = "이 산";
    String encodedValue = URLEncoder.encode(value,"utf-8");
    response.sendRedirect("/chap03/index.jsp?name=" + encodedValue);
%>
```

# 3.6 response 기본 객체-redirect 예제



# 3.6 response 기본 객체-redirect 예제




- 도구 => 사용자 도구 구성
- 인수 : -d ..\classes -cp .;C:\apache-tomcat-6.0.26\lib\servlet-api.jar $(FileName)