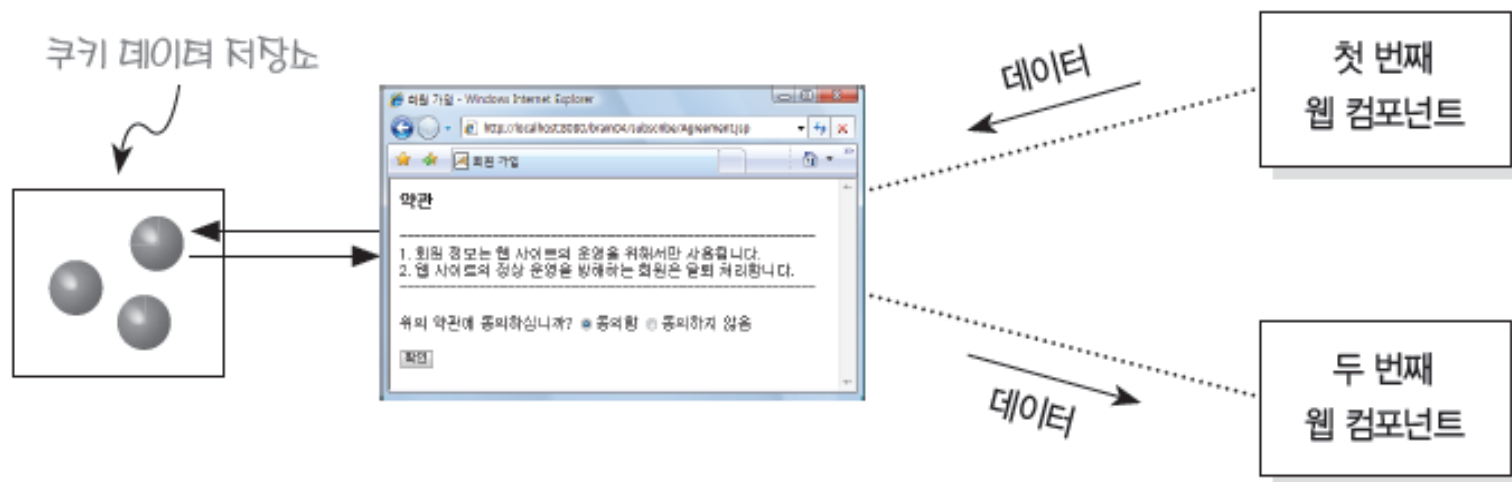


# 쿠키와 세션

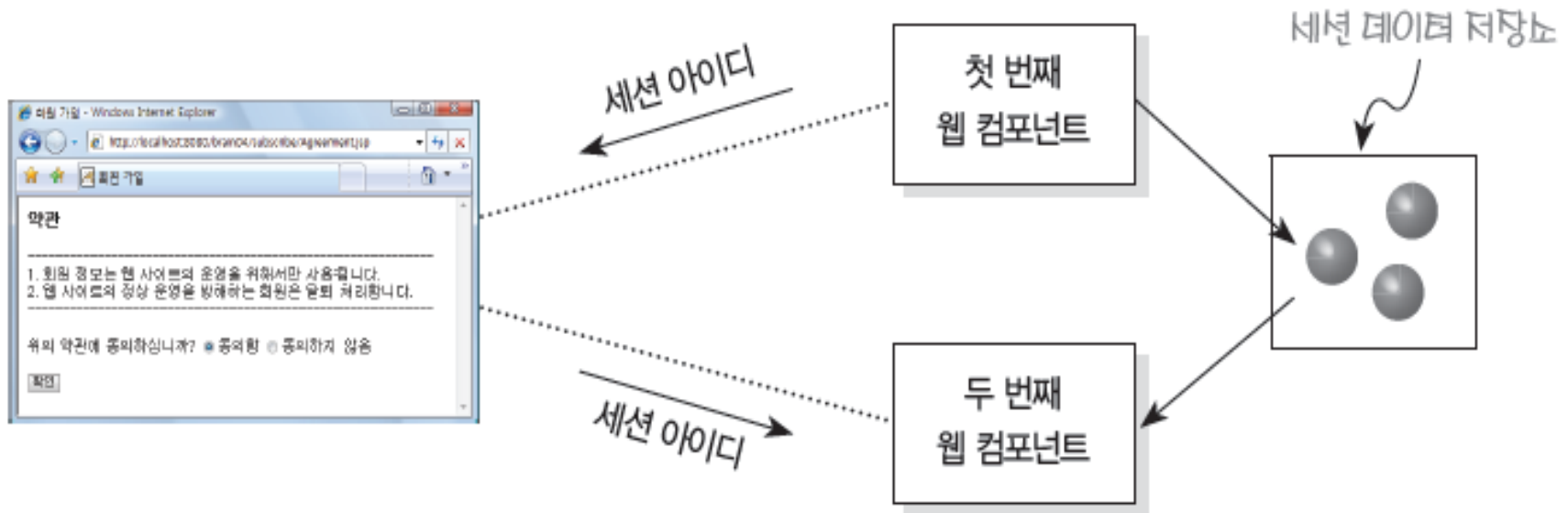
# 쿠키와 세션에 대하여

- 쿠키 기술은 웹 서버가 웹 브라우저로 데이터를 보냈다가 웹 서버 쪽으로 다시 되돌려 받는 방법을 사용한다.
- 첫 번째 웹 컴포넌트는 웹 브라우저로 HTML 문서를 보낼 때 전달한 데이터를 함께 보내며, 웹 브라우저는 그 데이터를 저장해 두었다가 두 번째 웹 컴포넌트를 호출할 때 URL과 함께 웹 서버로 보낸다



쿠키 기술을 이용한 웹 컴포넌트 간의 데이터 전달

- 세션 기술은 웹 브라우저를 거치지 않고 웹 서버에 있는 데이터 영역을 통해 데이터를 전달하는 방법이다.
- 첫 번째 웹 컴포넌트는 웹 서버 쪽에 데이터를 저장해 놓고, 그 데이터를 읽기 위해 필요한 세션 아이디만 웹 브라우저로 보낸다. 웹 브라우저는 아이디를 저장해 두었다가 두 번째 웹 컴포넌트를 호출할 때 웹 서버로 보내며, 그 아이디를 이용하면 저장된 데이터를 찾을 수 있다.



세션 기술을 이용한 웹 컴포넌트 간의 데이터 전달

# 쿠키

- ❖ 웹 서버가 웹 브라우저와 정보를 주고받는 방법 중의 하나
- ❖ 쿠키는 웹 브라우저가 보관하고 있는 데이터로 웹 서버에 요청을 보낼 때 함께 전송
- ❖ 웹 브라우저가 전송한 쿠키를 사용하여 필요한 데이터를 읽을 수 있습니다.
- ❖ 쿠키의 동작 방식
  - ❖ 쿠키 생성 단계 : 웹 서버 측에서 생성
  - ❖ 쿠키 저장 단계 : 웹 브라우저는 응답 데이터에 포함된 쿠키를 쿠키 저장소에 보관하는데 종류에 따라 메모리나 파일로 저장
  - ❖ 쿠키 전송 단계 : 서버 측에서 요청할 때 마다 웹 서버에 전송
- ❖ 쿠키의 구성
  - ❖ 이름 : 쿠키를 구별하는데 사용하는 이름
  - ❖ 값 : 쿠키의 이름과 관련된 값
  - ❖ 유효 시간 : 쿠키의 유지 시간
  - ❖ 도메인 : 쿠키를 전송할 도메인
  - ❖ 경로 : 쿠키를 전송할 요청 경로

# 쿠키 생성 및 읽기

- ❖ 쿠키 생성 및 쿠키 추가
  - ❖ `new Cookie("쿠키 이름", "쿠키의 값");`
  - ❖ `response.addCookie(쿠키 객체)`
- ❖ Cookie 클래스가 제공하는 메서드
  - ❖ `String getName()`
  - ❖ `String getValue()`
  - ❖ `void setValue(String value)`
  - ❖ `void setDomain(String pattern)`: 쿠키가 전송될 도메인을 설정
  - ❖ `String getDomain()`
  - ❖ `void setPath(String url)`: 쿠키가 전송할 경로 지정
  - ❖ `String getPath()`
  - ❖ `void setMaxAge(int expiry)`: 유효 시간을 초 단위로 설정(음수이면 웹 브라우저가 닫힐 때 쿠키가 삭제)
  - ❖ `int getMaxAge()`
- ❖ 쿠키 가져오기
  - ❖ `Cookie[] request.getCookies()`

# 쿠키 기술의 사용 방법

## ❖ 새로운 쿠키 데이터를 저장하는 방법 – 입력 기능

- 쿠키 데이터를 웹 브라우저 쪽에 저장하기 위해 해야 하는 두 가지 일
  - 첫째 : Cookie 클래스의 객체를 만든다.
  - 둘째 : addCookie 메서드를 호출한다.
- Cookie 클래스는 javax.servlet.http 패키지에 속하며, 이 클래스의 객체를 만들 때는 쿠키의 이름과 값을 파라미터로 넘겨줘야 한다.
  - 이 두 파라미터는 모두 String 타입이므로, **쿠키의 값이 수치일 경우는 문자 데이터로 만들어서 넘겨줘야 한다.**

```
Cookie cookie = new Cookie( "AGE ", "26 ");
```

쿠키 이름

쿠키 값

## ❖ 새로운 쿠키 데이터를 저장하는 방법 – 입력 기능

- addCookie 메서드는 웹 브라우저로 쿠키를 보내는 기능을 한다. JSP 페이지에서는 response 내장 객체에 대해, 서블릿 클래스에서는 doGet, doPost 메서드의 두 번째 파라미터에 대해 이 메서드를 호출해야 하며, Cookie 객체를 파라미터로 넘겨줘야 한다.

```
response.addCookie(cookie);
```

Cookie 객체

- addCookie 메서드를 통해 웹 브라우저로 전송된 쿠키를 실제로 저장하는 일은 웹 브라우저가 하도록 되어 있다.
- 웹 브라우저는 쿠키를 저장할 때 쿠키를 보낸 웹 서버의 주소도 함께 저장해 놓는다.

## ❖ 새로운 쿠키 데이터를 저장하는 방법 – 입력 기능

### ▪ 웹 브라우저 쪽에 쿠키 데이터를 저장하는 JSP 페이지

#### makeCookie.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<html>
<head> <title>쿠키를 생성하는 예제 </title> </head>
<%
    String cookieName = "id";
    Cookie cookie = new Cookie(cookieName, "hongkd");
    cookie.setMaxAge(60*2);
    cookie.setValue("kimkd");
    response.addCookie(cookie);
%>
<body>
<h2>쿠키를 생성하는 예제 </h2>
<P>
"<%=cookieName%>" 쿠키가 생성 되었습니다.<br>
<input type="button" value="쿠키의 내용확인"
onclick="javascript:window.location='useCookie.jsp'" >
</P>
</body>
</html>
```



## ❖ 쿠키 데이터를 읽는 방법 – 조회 기능

- 웹 브라우저는 웹 서버가 아무런 요청을 하지 않아도 웹 서버로 URL을 보낼 때 마다 그 URL에 포함된 웹 서버의 주소에 해당하는 모든 쿠키를 찾아서 웹 서버로 함께 보낸다.
- 쿠키를 받는 일은 `getCookies`라는 메서드를 이용해서 해야한다.
- `getCookies` 메서드는 JSP 페이지에서는 `request` 내장 변수에 대해, 서블릿 클래스에서는 `doGet`, `doPost` 메서드 첫 번째 파라미터에 대해 호출 해야 한다.
- `getCookies` 메서드는 웹 브라우저가 보낸 모든 쿠키를 Cookie 배열로 만들어서 리턴하기 때문에 다음과 같은 Cookie 배열 변수에 리턴값을 받아야 한다.

```
Cookie cookies[] = request.getCookies();
```

웹 브라우저가 보낸 모든 쿠키를  
Cookie 배열로 만들어서 리턴하는 메서드

- `getCookies` 메서드가 리턴한 `Cookie` 배열에서 특정 쿠키를 찾기 위해서는 그 배열에 있는 `Cookie` 객체를 하나씩 가져다가 이름을 비교해서 찾을 수 밖에 없다. 쿠키의 이름은 `Cookie` 객체에 대해 `getName`이라는 메서드를 호출해 구할 수 있다.

```
String name = cookie.getName();
```

↑  
쿠키 이름을 가져오는 메서드

- 원하는 이름의 `Cookie` 객체를 찾은 다음에는 그 객체에 대해 `getValue` 메서드를 호출해서 쿠키 값을 가져올 수 있다.

```
String value = cookie.getValue();
```

↑  
쿠키 값을 가져오는 메서드

## useCookie.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<html>
<head> <title>웹 브라우저에 저장된 쿠키를 가져오는 예제 </title> </head>
<body>
<h2>웹 브라우저에 저장된 쿠키를 가져오는 예제 </h2>
<%
    Cookie[] cookies = request.getCookies();
    if(cookies!=null){
        for(int i=0; i<cookies.length;++i){
            if(cookies[i].getName().equals("id")){
                쿠키의 이름은 "<%=cookies[i].getName()%>" 이고
                쿠키의 값 "<%=cookies[i].getValue()%>" 입니다.
            }
        }
    }
%>
</body>
</html>
```

# 쿠키의 유효시간

## ❖ 유효 시간

### ❖ setMaxAge(시간)을 이용해서 설정 가능

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%
    Cookie cookie = new Cookie("name", "park");
    cookie.setMaxAge(60 * 60); // 60초(1분) * 60 = 1시간
    response.addCookie(cookie);
%>
<html>
<head> <title>쿠키유효시간설정</title> </head>
<body>
```

유효시간이 1시간인 name 쿠키 생성.

```
</body>
</html>
```

# 쿠키 변경 및 삭제

- ❖ 쿠키의 값을 변경하기 위해서는 같은 이름의 쿠키를 새로 생성해서 응답 데이터로 전송하면 됩니다.
- ❖ 쿠키를 삭제하고자 하는 경우는 직접 삭제하는 것이 아니고 `setMaxAge()` 메서드를 호출할 때 인자 값으로 0을 대입해서 쿠키의 수명을 0으로 만드는 방법을 이용합니다.

## ❖ 쿠키 데이터를 삭제하는 방법 – 삭제 기능

- 쿠키 기술에서 데이터를 삭제하기 위해서는 **쿠키의 남은 수명을 0으로** 설정하는 방법을 사용해야 한다.
- 쿠키의 수명을 설정하기 위해서는 addCookie 메서드를 호출하기 전에 Cookie 객체에 대해 setMaxAge라는 메서드를 호출하면 된다. 이 메서드에는 초단위의 값을 넘겨줘야 하므로, 1시간 후에 쿠키가 지워지도록 만들려면 다음과 같은 값을 넘겨줘야 한다.

```
cookie.setMaxAge(3600);
```

↑  
쿠키의 최대 수명(초 단위)

- addCookie 메서드에 0이나 마이너스 값을 넘겨줄 수도 있다.

```
cookie.setMaxAge(0);
```

↑  
쿠키를 바로 삭제하도록  
만드는 값

```
cookie.setMaxAge(-1);
```

↑  
웹 브라우저가 끝날 때  
쿠키가 삭제되도록 만드는 값

## ❖ 쿠키 데이터를 삭제하는 방법 – 삭제 기능

### DeleteGookie.jsp 쿠키를 삭제하는 JSP 페이지

```
<% @page contentType= "text/html; charset=euc-kr" %>
<%
    Cookie cookie = new Cookie( "GENDER ", "" );
    cookie.setMaxAge(0);
    response.addCookie(cookie);
%>
<HTML>
    <HEAD><TITLE>쿠키 삭제하기</TITLE></HEAD>
    <BODY>
        GENDER 쿠키가 삭제되었습니다.
    </BODY>
</HTML>
```

# 쿠키의 도메인

## ❖ 도메인

- ❖ 쿠키는 그 쿠키를 생성한 서버에만 전송이 됩니다.
- ❖ 쿠키를 다른 도메인으로 전송하고자 하면 `setDomain()`를 이용하면 됩니다.
- ❖ `.도메인`: 모든 도메인에 쿠키를 전송
- ❖ [www.도메인](#): 특정 도메인에만 전송

## ❖ 경로

- ❖ `setPath(경로)`를 이용해서 특정 경로의 파일 들에게만 쿠키를 전송하는데 웹 애플리케이션 디렉토리를 기준으로 한 URL 경로명을 넘겨주어야 합니다.
- ❖ `/`로 시작해야 하고 `/`로 끝내는 것이 좋습니다.



# 쿠키 한글

```
<%@page import="java.net.URLEncoder"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html> <head> <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>Insert title here</title> </head>
<body>
<%
    Cookie cook2 =
        new Cookie("name",URLEncoder.encode("중앙정보","utf-8"));
    response.addCookie(cook2);
%>
쿠키저장 성공<p>
<a href="cookView2.jsp">쿠키보기</a>
</body>
</html>
```

# *cookView2.jsp*

```
<%@page import="java.net.URLDecoder"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%><!DOCTYPE html PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><head><meta http-equiv="Content-Type" content="text/html;
charset=UTF-8"><title>Insert title here</title></head>
<body>
<%
    Cookie cooks[] = request.getCookies();
    if (cooks != null) {
        for (int i = 0; i < cooks.length; i++) {
            if (cooks[i].getName().equals("name")) {
                out.println("쿠키값 : " +
                    URLDecoder.decode(cooks[i].getValue(),"utf-8"));
            }
        }
    }
%>
</body>
</html>
```

# 쿠키 처리를 위한 클래스 생성(util.CookieBox.java)

```
package util;
import java.io.IOException;
import java.net.URLDecoder;
import java.net.URLEncoder;
import java.util.Map;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServletRequest;

public class CookieBox {
    private Map<String, Cookie> cookieMap =
        new java.util.HashMap<String, Cookie>();
    public CookieBox(HttpServletRequest request) {
        Cookie[] cookies = request.getCookies();
        if (cookies != null) {
            for (int i = 0; i < cookies.length; i++) {
                cookieMap.put(cookies[i].getName(), cookies[i]);
            }
        }
    }
}
```

## 쿠키 처리를 위한 클래스 생성(util.CookieBox.java)

```
public static Cookie createCookie(String name, String value)
    throws IOException {
    return new Cookie(name, URLEncoder.encode(value, "utf-8"));
}

public static Cookie createCookie(String name, String value,
    int maxAge) throws IOException {
    Cookie cookie = new Cookie(name,
        URLEncoder.encode(value, "utf-8"));
    cookie.setMaxAge(maxAge);
    return cookie;
}

public static Cookie createCookie(String name, String value,
    String domain,int maxAge) throws IOException {
    Cookie cookie=new Cookie(name, URLEncoder.encode(value, "utf-8"));
    cookie.setDomain(domain);
    cookie.setMaxAge(maxAge);
    return cookie;
}
```

## 쿠키 처리를 위한 클래스 생성(util.CookieBox.java)

```
public Cookie getCookie(String name) {  
    return cookieMap.get(name);  
}  
  
public String getValue(String name) throws IOException {  
    Cookie cookie = cookieMap.get(name);  
    if (cookie == null) {  
        return null;  
    }  
    return URLDecoder.decode(cookie.getValue(), "utf-8");  
}  
  
public boolean exists(String name) {  
    return cookieMap.get(name) != null;  
}  
}
```

# 쿠키 생성 – CookieBoxMake.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="util.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html> <head> <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>Insert title here</title> </head>
<body>
<%
        response.addCookie(CookieBox.createCookie("name","홍길동"));
%>
쿠키저장 성공<p>
<a href="cookieBoxView.jsp">쿠키보기</a>
</body>
</html>
```

## 쿠키 읽기 – CookieBoxRead.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="util.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html> <head> <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>Insert title here</title> </head>
<body>
<%
    CookieBox cb = new CookieBox(request);
    out.println("쿠키 값 : " + cb.getValue("name"));
%>
</body>
</html>
```



# 쿠키 이용한 회원가입 Member1.java

```
package ch11
import java.util.Date;
public class Member1 {
    private String id;           private String password;
    private String name;        private Date reg_date;
    public String getId() {      return id; }
    public void setId(String id) { this.id = id; }
    public String getPassword() { return password; }
    public void setPassword(String password) { this.password = password; }
    public String getName() {    return name; }
    public void setName(String name) { this.name = name; }
    public Date getReg_date() { return reg_date; }
    public void setReg_date(Date reg_date) { this.reg_date = reg_date; }
}
```

## Table 생성

```
create table member1 (
id varchar2(12) primary key,
password varchar2(12) not null,
name varchar2(12) not null,
reg_date date not null);
```



# MemberDao.java

```
package ch11;
import java.sql.*; import javax.sql.*; import javax.naming.*;
public class MembeDao {
    private Connection getConnection() {
        Connection conn = null; DataSource ds = null;
        try { Context init = new InitialContext();
            ds =(DataSource)init.lookup("java:comp/env/jdbc/OracleDB");
            conn = ds.getConnection();
        } catch(Exception e) { System.out.println(e.getMessage()); }
        return conn;
    }
    public int insert(Member1 member) throws SQLException {
        Connection conn = getConnection(); PreparedStatement pstmt = null;
        String sql = "insert into member1 values (?, ?, ?, sysdate)"; int a = 0;
        try { pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, member.getId());
            pstmt.setString(2, member.getPassword());
            pstmt.setString(3, member.getName());
            a = pstmt.executeUpdate();
        } catch(Exception e) { System.out.println(e.getMessage()); }
        finally { if (pstmt != null) pstmt.close();
            if (conn != null) conn.close(); }
        return a;
    }
}
```

```
public int check(String id, String password) throws SQLException {
    Connection conn = getConnection();
    PreparedStatement pstmt = null;
    String sql = "select password from member1 where id=?";
    int a = -1;
    try { pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, id);
        ResultSet rs = pstmt.executeQuery();
        if (rs.next()) {
            String dbPassword = rs.getString(1);
            if (password.equals(dbPassword)) a = 1;
            else a = 0;
        } else a = -1;
    } catch (Exception e) { System.out.println(e.getMessage());
    } finally {
        if (pstmt != null) pstmt.close();
        if (conn != null) conn.close();
    }
    return a;
}
```

# insertMemberForm.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<html>
<head>
<title>회원가입</title>
</head>
<body>
    <h2>회원가입폼</h2>

    <form method="post" action="insertMemberPro.jsp">
        아이디: <input type="text" name="id" maxlength="12"><br>
        비밀번호: <input type="password" name="passwd"
maxlength="12"><br>
        이름: <input type="text" name="name" maxlength="10"><br>
        <input type="submit" value="회원가입">
        <input type="reset" value="다시입력">
    </form>
</body>
</html>
```

# insertMemberPro.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<%@ page import="java.sql.*"%>
<%@ page import="ch11.MemberDao" %>
<% request.setCharacterEncoding("euc-kr");%>

<jsp:useBean id="member" class="ch11.Member">
    <jsp:setProperty name="member" property="*" />
</jsp:useBean>

<%
    member.setReg_date(new Timestamp(System.currentTimeMillis()));
    MemberDao logon = new MemberDao();
    logon.insertMember(member);
%>

<jsp:getProperty name="member" property="id" />님 회원가입을 축하합
니다.
```

# 쿠키 이용한 사용자 정보유지 cookieMain.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<% String id = "";
Cookie[] cookies = request.getCookies();
if(cookies!=null){
    for(int i=0; i<cookies.length; i++){
        if(cookies[i].getName().equals("id")){id = cookies[i].getValue(); }
    }
    if(id.equals("")){    response.sendRedirect("loginForm.jsp");    }
}else{response.sendRedirect("loginForm.jsp");}
%>
<html>
<head><title>쿠키를 사용한 간단한 회원인증</title></head>
<body>
    <b><%= id %></b>님이 로그인 하셨습니다.
    <form method="post" action="cookieLogout.jsp">
<input type="submit" value="로그아웃">
    </form>
</body>
</html>
```

# loginForm.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"  
    pageEncoding="EUC-KR"%>
```

```
<html>
```

```
<head>
```

```
<title>로그인</title>
```

```
</head>
```

```
<body>
```

```
    <h2>로그인폼</h2>
```

```
    <form method="post" action="cookieLoginPro.jsp">
```

```
        아이디: <input type="text" name="id" maxlength="12"><br>
```

```
        비밀번호: <input type="password" name="passwd"
```

```
maxlength="12"><br>
```

```
        <input type="submit" value="로그인">
```

```
        <input type="button" value="회원가입"
```

```
onclick="location.href='insertMemberForm.jsp'">
```

```
    </form>
```

```
</body>
```

```
</html>
```

## ***cookieLoginPro.jsp***

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<%@ page import="ch11.MemberDao"%>
<% request.setCharacterEncoding("euc-kr");%>
<%    String id = request.getParameter("id");
String passwd = request.getParameter("passwd");
MemberDao logon = new MemberDao();
    int check= logon.userCheck(id,passwd);
    if(check==1){
        Cookie cookie = new Cookie("id", id);
        cookie.setMaxAge(20*60);
        response.addCookie(cookie);
        response.sendRedirect("cookieMain.jsp");
    }else if(check==0){%>
<script>
    alert("비밀번호가 맞지 않습니다.");
    history.go(-1);
</script>
<%}else{ %>
<script>
    alert("아이디가 맞지 않습니다..");
    history.go(-1);
</script>
<%}%>
```



## ***cookieLogout.jsp***

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"  
    pageEncoding="EUC-KR"%>
```

```
<%
```

```
Cookie[] cookies = request.getCookies();
```

```
if(cookies!=null){
```

```
    for(int i=0; i<cookies.length; i++){
```

```
        if(cookies[i].getName().equals("id")){
```

```
            cookies[i].setMaxAge(0);
```

```
            response.addCookie(cookies[i]);
```

```
        }
```

```
    }
```

```
}
```

```
%>
```

```
<script>
```

```
    alert("로그아웃 되었습니다.");
```

```
    location.href="cookieMain.jsp";
```

```
</script>
```



# webStorage

- ❖ 클라이언트의 디스크에 소량의 데이터를 저장하기 위한 스토리지
- ❖ 이전에는 일반적으로 Cookie를 사용
- ❖ Cookie 와 다른 점
  - ❖ 크기에 제한이 없습니다.
  - ❖ 서버로 보내지지 않습니다.
  - ❖ 자바 스크립트 객체의 복사본을 저장할 수 있습니다.
- ❖ 종류는 로컬 스토리지와 세션 스토리지로 나뉩니다.
- ❖ 사용 방법은 동일하며 유효범위가 다름
- ❖ 메서드
  - ❖ length 속성: 스토리지에 저장된 데이터의 수를 반환
  - ❖ key(index): 인덱스의 키를 반환하고 없으면 null을 반환
  - ❖ getItem(key): 키에 대응되는 데이터를 반환
  - ❖ setItem(key, data): key로 스토리지에 데이터를 저장
  - ❖ removeItem(key): key에 대응되는 데이터를 삭제
  - ❖ clear(): 스토리지의 모든 데이터 삭제

- ❖ localStorage

- ❖ 웹 사이트 도메인마다 별도로 생성되는 저장영역
- ❖ 사용자가 명시적으로 지우지 않는 이상 데이터는 영구적으로 저장
- ❖ 사용하고자 하는 경우에는 전역변수 localStorage에 자바 스크립트 객체와 같이 속성과 값을 입력하면 됩니다.

- ❖ 저장

- ❖ localStorage.키이름 = {속성:값, 속성:값...};
- ❖ localStorage["키이름"] = {속성:값, 속성:값...};
- ❖ localStorage.setItem("키이름", {속성:값, 속성:값...});

- ❖ 읽기

- ❖ var 변수명 = localStorage.키이름;
- ❖ var 변수명 = localStorage["키이름"];
- ❖ var 변수명 = localStorage.getItem("키이름");

- ❖ 삭제

- ❖ delete.localStorage.키이름;
- ❖ delete localStorage["키이름"];
- ❖ localStorage.removeItem("키이름");

# 로그인 – login.html

```
<!DOCTYPE html> <html> <head>
<meta charset="UTF-8">
<title>로그인</title>
<script>
window.onload = function() {
    loadStorage();
};
// 아이디와 아이디 저장 여부를 로컬 스토리지에 저장
function saveStorage() {
    var saveId = document.getElementById("saveId").checked;
    var userId = document.getElementById("userId").value;
    var userPassword = document.getElementById("userPassword").value;
    if(userId.length == 0){
        alert("아이디는 비어 있을 수 없습니다.")
        return false;
    }
    if(userPassword.length == 0){
        alert("비밀번호는 비어 있을 수 없습니다.")
        return false;
    }
}
```

## 로그인 – login.html

//아이디 저장에 체크가 되어있으면 저장여부(Y)와 아이디를 저장하고  
// 체크가 되어있지 않으면 아이디를 삭제한다.

```
if (saveId) {  
    window.localStorage.setItem("saveId", "Y");  
    window.localStorage.setItem("userId", userId);  
} else {  
    window.localStorage.removeItem("saveId");  
    window.localStorage.removeItem("userId");  
}
```

```
};
```

// 로컬 스토리지에서 아이디와 아이디 저장 여부를 꺼내서 채운다.

```
function loadStorage() {  
    var saveId = window.localStorage.getItem("saveId");  
    var userId = window.localStorage.getItem("userId");  
    if (saveId == "Y") {  
        document.getElementById("saveId").checked = true;  
        document.getElementById("userId").value = userId;  
    }  
}
```

```
};
```

```
</script>
```

```
</head>
```

```
<body>
<h1>로그인(Web Storage)</h1>
<form method="post" action="login.jsp" onsubmit="return saveStorage()">
  <fieldset>
    아이디:<input type="text" name="id" id="userId" autocomplete="off" />
    <input type="checkbox" id="saveId" />아이디 저장<br />
    비밀번호:<input type="password" name="password"
      id="userPassword" ><br />
    <input type="submit" value="로그인">
  </fieldset>
</form>
</body>
</html>
```

# 로그인처리 – login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>로그인 결과</title>
</head>
<body>
    <h1>메인페이지</h1>
    <%
        String id = request.getParameter("id");
        out.println(id + "님 환영합니다. <br/>");
    %>
    <a href="login.html">로그인 페이지로</a>
</body>
</html>
```

# 세션

- ❖ 세션은 웹 컨테이너에 정보를 저장할 수 있는 객체
- ❖ 웹 컨테이너는 하나의 웹 브라우저에 하나의 세션을 생성
- ❖ 세션 생성
  - ❖ 코드에서 생성: `<%@ page session="true" %>`
  - ❖ request 객체로 부터 생성
    - ❖ HttpSession 변수 = `request.getSession()`
    - ❖ 위의 경우는 세션이 있으면 그 세션을 리턴하고 없으면 새로 생성해서 리턴
    - ❖ 만일 생성된 경우에만 리턴받고자 하는 경우는 `getSession`에 매개변수로 `false`를 전달
- ❖ 세션에서 속성 사용
  - ❖ `session.setAttribute("속성명", "값")`
  - ❖ `session.getAttribute("속성명")`
- ❖ 세션 객체의 메서드
  - ❖ `String getId()`
  - ❖ `long getCreationTime()`
  - ❖ `long getLastAccessedTime()`

## ❖ 서블릿 클래스에서 세션 기술을 사용하는 방법

- 서블릿 클래스에서 세션을 시작하기 위해서는 doGet, doPost 메서드의 HttpServletRequest 파라미터에 대해 getSession이라는 메서드를 호출해야 한다
- getSession 메서드는 세션 정보를 포함하는 javax.servlet.http.HttpSession 타입의 객체를 리턴한다.

```
HttpSession session = request.getSession();
```

세션을 시작하는 메서드

- getSession 메서드가 리턴한 HttpSession 객체에 대해 setAttribute라는 메서드를 호출하면 세션 데이터 영역에 데이터를 저장할 수 있다.

```
session.setAttribute( "ID", "lee77" );
```

데이터 이름

데이터 값



# 객체 메소드알아보기

메소드	설명
setAttribute(String name, Object value)	세션 객체에 속성을 저장한다..
removeAttribute(String name)	저장된 속성을 제거한다.
getAttribute(String name)	저장된 속성을 반환한다.
getId()	클라이언트의 세션 ID 값을 반환한다.
setMaxInactiveInterval(int seconds)	세션의 유지 시간을 설정한다.
getMaxInactiveInterval()	세션의 유지 시간을 반환한다.
invalidate()	현재의 세션을 정보들을 모두 제거한다.

## ❖ 서블릿 클래스에서 세션 기술을 사용하는 방법

### setSession.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<html>
<head>
    <title>세션 사용 예제</title>
</head>
<body>
<%
    String id = "hongkd";
    String passwd = "1234";

    session.setAttribute("id", id);
    session.setAttribute("passwd", passwd);
%>
세션에 id 와 passwd 속성을 설정하였습니다.<br>

<input type="button" value="세션의 설정된 속성확인"
onclick="javascript:window.location='viewSession.jsp'">
</body>
</html>
```

## ❖ 서블릿 클래스에서 세션 기술을 사용하는 방법

[viewSession.jsp

```
i<%@ page contentType="text/html; charset=euc-kr" %>
<%@ page import="java.util.*" %>
<html>
<head> <title>세션 사용 예제 </title> </head>
<body>
<%
    Enumeration attr = session.getAttributeNames();
    while(attr.hasMoreElements()){
        String attrName = (String)attr.nextElement();
        String attrValue = (String)session.getAttribute(attrName);
        out.println("세션의 속성명은 " + attrName + " 이고 ");
        out.println("세션의 속성값은 " + attrValue + "이다.<br>");
    }
%>
</body>
</html>
```

## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

- 서블릿 클래스에서는 새로운 세션을 시작하거나 진행 중인 세션을 계속하기 위해서는 getSession 메서드를 호출해야 하지만, JSP 페이지에서는 JSP 페이지가 서블릿 클래스로 변환되는 과정에서 이 메서드를 호출하는 코드가 자동으로 추가 되기 때문에 getSession 메서드를 호출 할 필요가 없다.
- session 내장 변수를 사용하면 세션 데이터 영역에 데이터를 저장할 수도 있고, 그 영역에 있는 데이터를 읽어오거나 삭제할 수도 있다.

```
session.setAttribute( "ID ", "lee77 ");
```

세션 데이터에 저장하는 메서드

## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

```
String str = (String) session.getAttribute( "ID " );
```

세션 데이터를 가져오는 메서드

```
session.removeAttribute( "ID " );
```

세션 데이터를 삭제하는 메서드

- 세션을 끝내려면 session 내장 변수에 대해 invalidate라는 메서드를 호출하면 된다.

```
session.invalidate();
```

세션을 끝내는 메서드

## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

- 앞 페이지의 네 URL에 해당하는 HTML 문서와 JSP 페이지는 다음과 같이 작성하면 된다.


### PersonalInfo.html

```
<HTML>
  <HEAD>
    <META http-equiv= "Content-Type" content= "text/html; charset=euc-kr" >
    <TITLE>회원 가입</TITLE>
  </HEAD>  <BODY>
    <form action= "agree.jsp">
      <table border= "1">
        <tr> <td>아이디</td> <td><input type= "text" name= "id"
          required= "required"> </td> </tr>
        <tr> <td>패스워드</td> <td><input type= "password" name= "password"
          required= "required"> </td> </tr>
        <tr> <td>이름</td> <td><input type= "text" name= "name"
          required= "required"> </td> </tr>
        <tr> <td><input type= "submit" value= "확인"> </td>
          <td><input type= "reset" value= "취소"> </td> </tr>
      </table>
    </form>
  </BODY>
</HTML>
```

## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

Agreement.jsp

```
<html><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title></head><body>
  <% String id = request.getParameter("id");
      String password = request.getParameter("password");
      String name = request.getParameter("name");
      session.setAttribute("id", id);
      session.setAttribute("password", password);
      session.setAttribute("name", name);
  %>
  <h2>약관 동의</h2>
  ----- <BR>
  1. 회원 정보는 웹 사이트의 운영을 위해서만 사용됩니다. <BR>
  2. 웹 사이트의 정상 운영을 방해하는 회원은 탈퇴 처리합니다. <BR>
  ----- <BR>
  <form action="subscribe.jsp">
    동의<input type="radio" name="agree" value="y"> <p>
    거부<input type="radio" name="agree" value="n"> <p>
    <input type="submit" value="확인">
  </form>
</BODY>
</HTML>
```



세션 데이터를 저장합니다

## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

### Subscribe.jsp

```
<% @page contentType= "text/html; charset=euc-kr" %>
<%@page import="java.io.*"%>
<%
    String agree = request.getParameter( "AGREE " );
    String result = null;
    if (agree.equals( "y ")) {
        String id = (String) session.getAttribute( "id " );
        String password =
            (String) session.getAttribute( "password" );
        String name = (String) session.getAttribute( "name" );
        PrintWriter writer = null;
        try {
            String filePath = application.getRealPath( "/WEB-INF/" + id + ".txt " );
            writer = new PrintWriter(filePath);
            writer.println( "아이디: " + id );
            writer.println( "패스워드: " + password );
            writer.println( "이름: " + name );
            result = "success ";
        }
        catch (IOException ioe) {
            result = "fail ";
        }
    }
}
```

```
        finally {
            try {
                writer.close();
            }
            catch (Exception e) {
            }
        }
    }
    else {
        result = "fail ";
    }
    session.invalidate();
    response.sendRedirect( "Result.jsp?result=" +
result );
%>
```



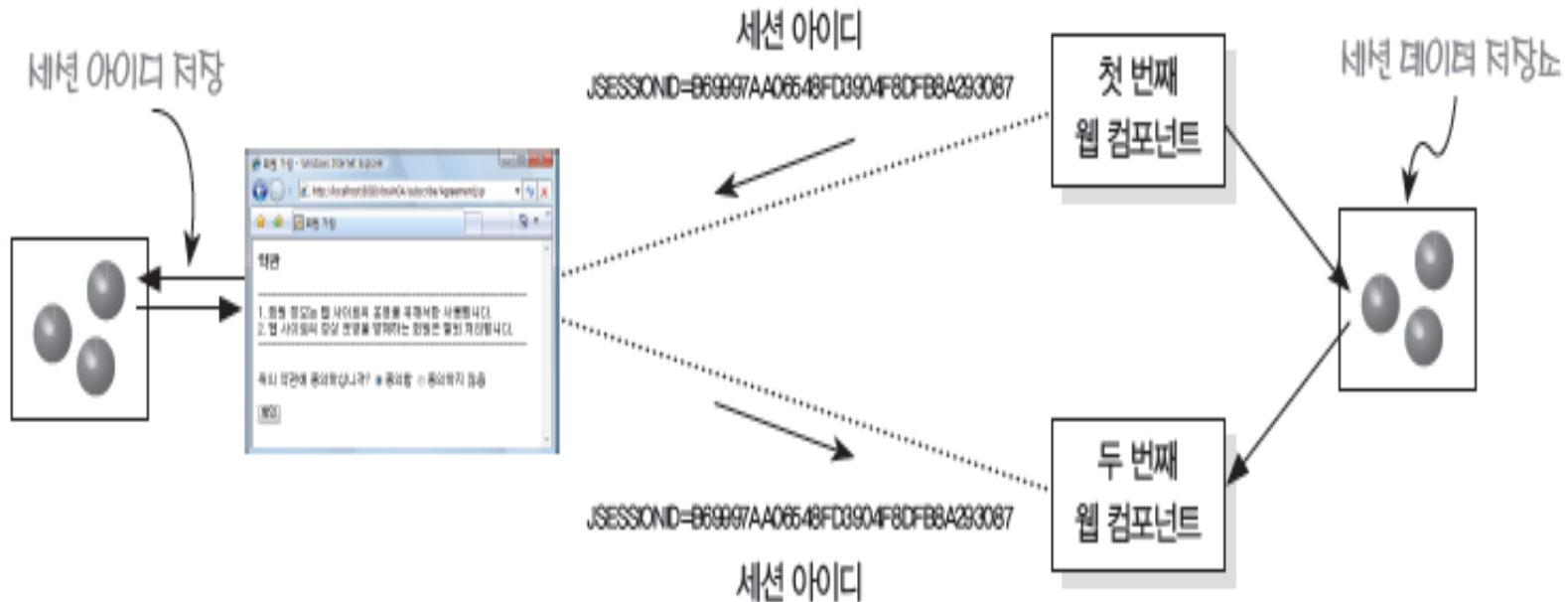
## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

### Result.jsp

```
<% @page contentType= "text/html; charset=euc-kr "%>
<% String result = request.getParameter( "result " ); %>
<HTML>
  <HEAD><TITLE>회원 가입</TITLE></HEAD>
  <BODY>
    <H3>회원 가입 결과</H3>
    <%
      if (result.equals( "success "))
        out.println( "가입되었습니다. " );
      else
        out.println( "가입되지 않았습니다. " );
    %>
  </BODY>
</HTML>
```

## ❖ JSP 페이지에서 세션 기술을 사용하는 방법

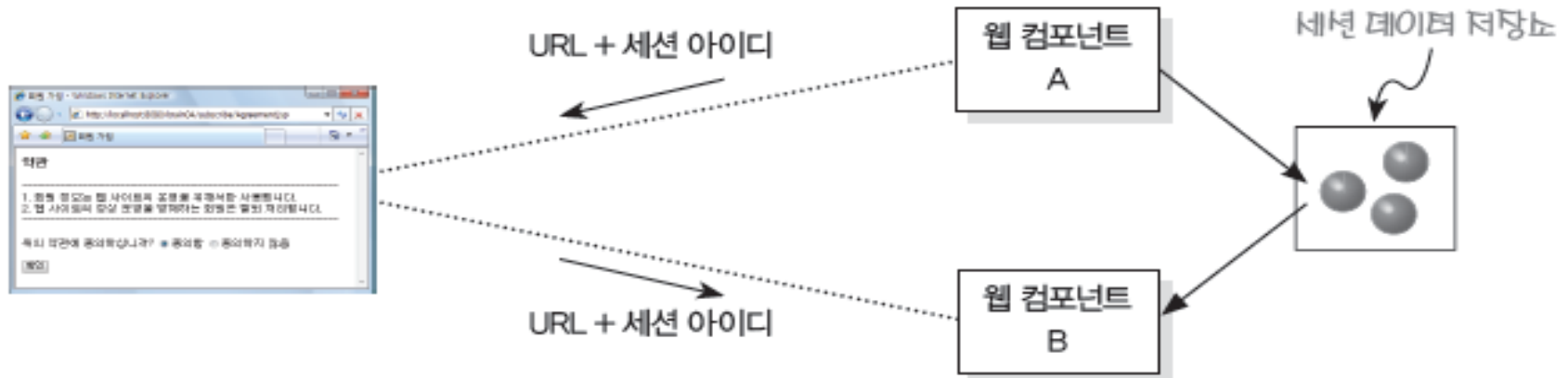
- 세션 기술에서는 웹 브라우저로 세션 아이디를 보낼 때 쿠키 형태로 만들어서 전송하는데, 이 쿠키 이름은 JSESSIONID이다.



쿠키 형태로 전송되는 세션 아이디

## ❖ URL 재작성 메커니즘의 사용 방법

- 쿠키를 사용할 수 없는 웹 환경에서는 URL 뒤에 세션 아이디를 붙여서 전송하는 방법을 사용하면 된다.



URL과 함께 전송되는 세션 아이디

- 이 방법은 본래의 URL을 가지고 새로운 URL을 만드는 방법이기 때문에 URL 재작성 (URL rewriting) 메커니즘이라고 부른다.

본래의  
URL

`http://localhost:8080/brain04/subscribe/Agreement.jsp; jsessionid=8088A1AAA61960F0B113E331A1460089`

URL 재작성으로 추가된 부분

## ❖ URL 재작성 메커니즘의 사용 방법

- URL 재작성을 하기 위해서는 URL 재작성 기능을 제공하는 `encodeURL`이라는 메서드를 사용하면 된다.
- JSP 페이지에서는 `response` 내장 변수에 대해 이 메서드를 호출하면 되고, 서블릿 클래스에서는 `doGet`, `doPost` 메서드의 두 번째 파라미터에 대해 호출하며 된다.

```
String url = response.encodeURL("http://localhost:8181/subscribe/Agreement.jsp");
```

본래의 URL

- `encodeURL` 메서드에는 현재의 웹 컴포넌트를 기준으로 한 상대적인 URL 경로명을 파라미터로 넘겨 줄 수도 있다. 그러면 이 메서드는 URL 경로명 뒤에 세미콜론과 `jessionid=세션_아이디`를 붙여서 리턴할 것이다.

```
String url = response.encodeURL("common/Greetings.jsp");
```

상대적인 URL 경로명

## ❖ URL 재작성 메커니즘의 사용 방법

- encodeURL 메서드에는 슬래시(/)로 시작하는 URL 경로명을 넘겨줄 수도 있는데, 이런 값은 웹 서버 내에서의 URL 경로명을 해석된다.

```
String url = response.encodeURL( “/subscribe/Result.jsp ”);
```

↑  
웹 서버 내에서의 URL 경로명

### WriteSessionData.jsp

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

```
<%
```

```
    session.setAttribute( “NAME ”, “김지영 ” );
```

```
    session.setAttribute( “AGE ”, new Integer(21));
```

```
    session.setAttribute( “GENDER ”, “여 ” );
```

```
%>
```

```
<HTML>
```

```
    <HEAD><TITLE>세션 데이터를 저장하는 JSP 페이지</TITLE></HEAD>
```

```
    <BODY>
```

```
        세션 데이터가 저장되었습니다. <BR><BR>
```

```
        <A href=<%= response.encodeURL(“ReadSessionData.jsp ”) %>>세션 데이터 읽기</A>
```

```
    </BODY>
```

```
</HTML>
```

## ❖ URL 재작성 메커니즘의 사용 방법

### ReadSessionData.jsp

```
<% @page contentType= "text/html; charset=euc-kr" %>
<HTML>
  <HEAD><TITLE>세션 데이터를 읽는 JSP 페이지</TITLE></HEAD>
  <BODY>
    이름: <%= session.getAttribute( "NAME " ) %> <BR>
    나이: <%= session.getAttribute( "AGE " ) %> <BR>
    성별: <%= session.getAttribute( "GENDER " ) %>
  </BODY>
</HTML>
```

- 웹 애플리케이션 디렉터리에 WriteSessionData.jsp와 ReadSessionData.jsp이름으로 저장한다.

## 세션 이용한 정보유지 sessionMain.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><head><meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>Insert title here</title></head>
<body>
    <%
        String id = (String)session.getAttribute("id");
        if (id == null || id == "") {
            response.sendRedirect("loginForm.jsp");
        }
    %>
    <%=id %>님 격하게 환영합니다.<p>
    <input type="button" value="로그아웃"
        onclick="location.href='sessionLogout.jsp'">
</body>
</html>
```

## sessionLoginForm.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>
<html>
<head>
<title>로그인</title>
</head>
<body>
    <h2>로그인 폼</h2>

    <form method="post" action="sessionLoginPro.jsp">
        아이디: <input type="text" name="id" maxlength="12"><br>
        비밀번호: <input type="password" name="passwd"
maxlength="12"><br>
        <input type="submit" value="로그인">
        <input type="button" value="회원가입"
onclick="location.href='insertMemberForm.jsp'">
    </form>
</body>
</html>
```



# sessionLoginPro.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"  
    pageEncoding="EUC-KR"%>  
<%@ page import="ch11.LogonDBBean"%>  
<% request.setCharacterEncoding("euc-kr");%>  
<%  
    String id = request.getParameter("id");  
    String passwd = request.getParameter("passwd");  
    LogonDBBean logon = LogonDBBean.getInstance();  
    int check= logon.userCheck(id,passwd);  
    if(check==1){  
        session.setAttribute("id",id);  
        response.sendRedirect("sessionMain.jsp");  
    }else if(check==0){%>  
        <script>  
            alert("비밀번호가 맞지 않습니다.");  
            history.go(-1);  
        </script>  
    <%}else{ %>  
        <script>  
            alert("아이디가 맞지 않습니다..");  
            history.go(-1);  
        </script>  
    <%}%>
```

# sessionLogout.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"  
    pageEncoding="EUC-KR"%>
```

```
<%session.invalidate(); %>
```

```
<script>  
    alert("로그아웃 되었습니다.");  
    location.href="sessionMain.jsp";  
</script>
```

# JSTL의XML 액션-JSTL xml

xalan.jar 파일을 라이브러리폴더에 등록하기

<http://xml.apache.org/xalan-j/>

Download/Build를 클릭 ->

Xalan-j distribution directory 클릭

The screenshot shows a web browser window with the address bar displaying <http://xml.apache.org/xalan-j/>. The browser tab is titled "Xalan-Java Version 2.7.1". The main content area shows the "Xalan-Java Version 2.7.1" page, which includes a sidebar with a navigation menu and a main content area with links to "What is it?", "How about this release?", "For more information", "Apache License", "Notice file", and "Apache License".

The sidebar menu includes the following items:

- Home
- Xalan-J 2.7.1
- Charter
- What's New
- Release Notes
- Overview
- Download/Build
- Getting Started
- Using XSLTC
- FAQs
- Sample Apps
- Command Line
- Features
- Transform API
- XPath API
- Usage Patterns


The main content area of the "Xalan-Java Version 2.7.1" page includes the following links:

- [What is it?](#)
- [How about this release?](#)
- [For more information](#)
- [Apache License](#)
- [Notice file](#)
- [Apache License](#)

The "Download/Build & Dependencies" page is also visible, showing a sidebar with the same navigation menu and a main content area with links to "Downloading the latest release", "Downloading what else you might need", "Where do I get Xerces-Java?", "How do I view Xalan code in a browser?", "How do I download the latest development code to build myself?", "Using ant", "Rebuilding a sample application", and "Where do I download previous releases?".

The "Download/Build & Dependencies" page also includes a section titled "Downloading the latest release" with the text: "You can download the pre-built binary distributions from one of the mirror sites at [xalan-j distribution directory](#)."

# xalan-j 버전-bin.zip 파일 다운로드



## The Apache Software Foundation

Home » Dyn

Foundation Projects People Get Involved Download **Support**  
Apache

Apache Download Mirrors

We suggest the following mirror site for your download:

<http://mirror.apache-kr.org/xml/xalan-j>

Other mirror sites are suggested below. Please use the backup mirrors only to download PGP and MD5 signatures to verify your downloads or if no other mirrors are working.

### HTTP

<http://mirror.apache-kr.org/xml/xalan-j>

<http://apache.mirror.cdnetworks.com/xml/xalan-j>

<http://apache.tt.co.kr/xml/xalan-j>

http://www.apache.org/dyn/closer.cgi/xml/xalan-j


Apache Download Mirrors

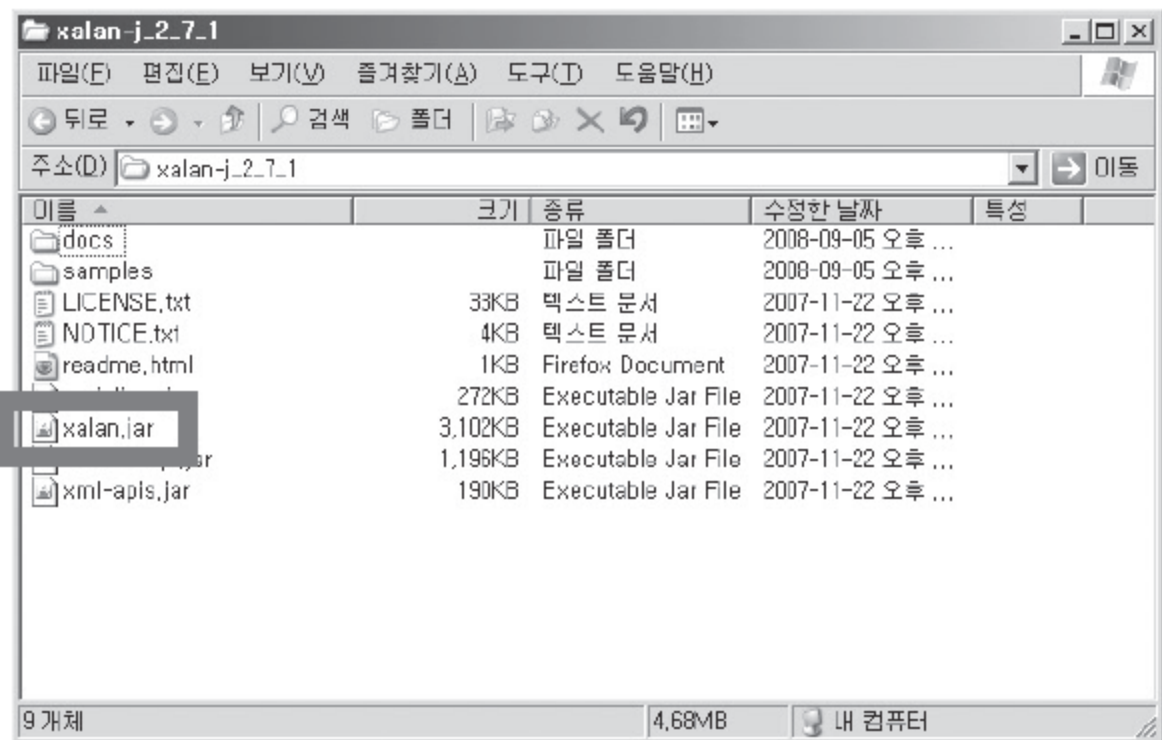
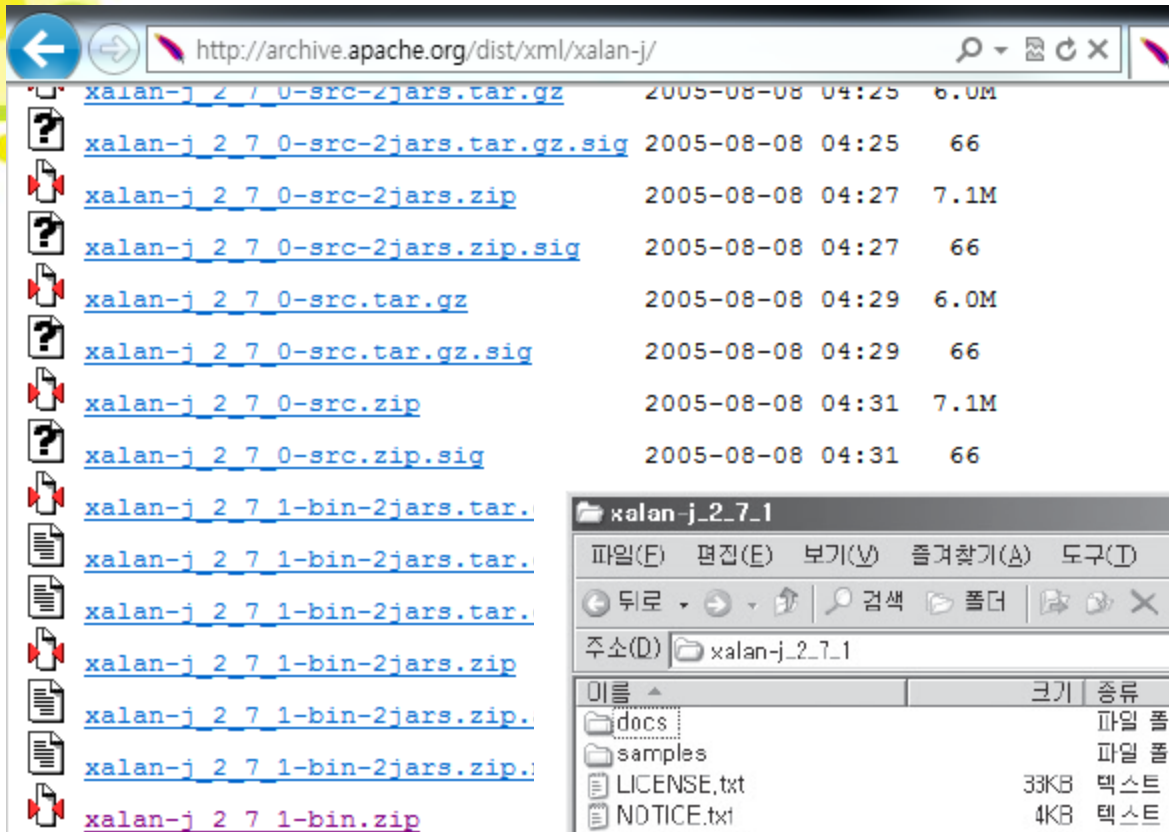
Search

http://mirror.apache-kr.org/xml/xalan-j/

Xml-xalan-j has moved to [Project Xalan](#).

- the [Xalan project page](#)
- the [Xalan java download page](#)
- the [Xalan java download area](#)
- the [XML/Xalan java archive](#)

Name	Last modified	Size	Description
 <a href="#">Parent Directory</a>		-	XML project



## Web-inf의 lib

xalan.jar

Jstl-api-1.2.jar

Jstl-impl-1.2.jar

# JSTL의XML 액션-JSTL xml

JSTL xmlXML 문서에서 자주 사용되는 기능들을 태그라이브러리로 모아 놓은 것  
<%@ taglib prefix="x" uri=<http://java.sun.com/jsp/jstl/xml> %>

## xml 라이브러리 태그 알아보기

- . 출력, 변수 설정태그 : <x:out>, <x:set>
- . 조건 처리태그 : <x:if>, <x:choose>, <x:when>, <x:otherwise>
- . 반복처리태그 : <x:forEach>
- . 기타 XML 관련태그 : <x:parse>, <x:transform>, <x:param>

## <x:out>지정된 Xpath의 내용을 출력하는 태그

- . escapeXml은 기본값으로 false로 지정되어 있으며<, >등의 특수기호의 출력 형태를 설정 할 때 쓰인다. True일 경우< 값은&lt;로 표현되고> 값은&gt;로 표현
- <x:out select="Xpath expression" escapeXml="true 또는 false">

## <x:set>지정된 변수에 지정한 XPath내의 XML 내용을 저장하는 태그

- . var속성이 값을 저장할 변수를 의미
- . select 속성에 Xpath 표현 식을 입력
- . Scope는 변수의 범위를 의미

<x:set var="변수명" select="Xpath expression" scope="범위">

## <x:if> core 라이브러리에 존재하는 <c:if> 태그와 유사한 기능

- . select 속성에 지정된 Xpath 표현 식으로 조건을 판별
- . var 속성의 변수에는 조건처리의 결과를 저장

<x:if select="Xpath expression" var="변수명" scope="범위">

## <x:choose> core 라이브러리의 <c:choose>와 유사한 기능. 조건문의 시작을 알리는 태그로 사용된다.

- . <x:when>은 조건을 지정할 때 사용
- . select 속성에 Xpath 표현식을 입력하여 조건을 지정
- . <x:otherwise> 태그 사이의 내용은 <x:when> 태그의 조건에도 성립하지 않을 경우 처리

<x:choose>

    <x:when select="Xpath expression">

        </c:when>

    <x:otherwise></x:otherwise>

</x:choose>

## <x:forEach> XML에서 반복처리가 필요할 때 사용

- . <x:forEach>에서 자주사용되는 속성은 select 속성이며, 여기에 Xpath 표현식을 입력하여 반복처리를 수행

<x:forEach var="변수명" select="Xpath expression" begin="시작인덱스"  
end="끝 인덱스" step="증감식">



## <x:parse> XML 문서를 파싱할 때 사용

- . 보통은 var속성을 사용하여 파싱할 XML 문서를 var 속성에 입력된 변수에 저장
- . systemId는 파싱되고 있는 문서의 URI를 나타낸다.
- . Filter는 파싱하기 전에 지정된 필터의 내용을 걸러낼 때 사용한다.
- . Doc속성은 직접 XML 문서의 위치를 지정할 때 사용

```
<x:parse var="변수명" varDom="변수명" scope="범위" scopeDom="범위"  
doc="source" systemId="URI" filter="필터">
```

## <x:transform> XML 문서를 XSL 스타일시트를 이용하여 새로운 문서로 변형시키는역할 수행

- . doc 속성에는 XML 문서를 입력
- . xslt 속성에는 XSL 스타일시트를 입력
- . var 속성에는 생성된 결과를 저장
- . result 속성도 생성된 결과를 지정

```
<x:transform var="변수명" scope="범위" result="변수명" doc="source"  
xslt="XSLTStyleSheet">
```

## <x:param>

- . <x:transform> 태그 사이에 파라미터 값을 전달하기 위해 사용
- . name 속성에는 전달할 파라미터의 이름 입력
- . value 속성에는 전달할 파라미터 값을 입력

```
<x:param name="이름" value="값">
```



# Jstl\_xml\_ex.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"%>
<%@ taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml" %>
<html><head><title>JSTL xml 라이브러리 사용 예제</title></head>
<body>
<x:parse var="xml/data">
<students>
<student><name>홍길동</name><age>18</age><gender>남</gender>
    <phone>011-3456-11xx</phone>
</student>
<student><name>김길동</name><age>19</age><gender>남</gender>
    <phone>010-4567-00xx</phone>
</student>
<student><name>홍길순</name><age>18</age><gender>여</gender>
    <phone>없음</phone>
</student>
<student><name>김길순</name><age>18</age><gender>여</gender>
    <phone>없음</phone>
</student>
</students>
</x:parse>
```

```
<x:forEach select="$xml/data//student">
<x:if select="./name!='홍길순'">
<x:out select="./name"/>
<x:set select="./age" var="age"/>
<x:out select="$age"/>
<x:out select="./gender"/>

<x:choose>
<x:when select="./phone!='없음'">
[전화번호 : <x:out select="./phone"/>]
</x:when>
<x:otherwise>
[전화 없음]
</x:otherwise>
</x:choose>
<br>
</x:if>
</x:forEach>
</body>
</html>
```

# JSTL의SQL 액션-JSTL sql

## JSTL sql

- . SQL 관련기능을 제공해 주는 JSTL 라이브러리
- . sql 라이브러리를 이용해서 데이터베이스서버에 접근할 수 있으며, 쿼리를 전송할 수도 있다. 또 트랜잭션 처리도 가능하다.

<%@ taglib prefix="sql" uri=<http://java.sun.com/jsp/jstl/sql> %>

## sql 라이브러리 태그 알아보기

- . 데이터베이스 연결 태그 : <sql:setDataSource>
- . 쿼리 전송 관련 태그 : <sql:query>, <sql:update>, <sql:param>, <sql:dateParam>
- . 트랜잭션 태그 : <sql:transaction>

## <sql:setDataSource> 데이터베이스 서버에 접근하기 위해 존재

- . 접근 방법 두가지 방법이 있다. dataSource 속성을 이용하거나 driver, url, user, password 속성을 이용하는 방법
- . dataSource 속성을 사용할 때는 미리 작성해둔 JNDI 리소스가 존재해야 한다. 리소스를 그대로 가져다 사용할 경우 dataSource 속성만 이용하면 편리하게 데이터베이스 서버에 접근할 수 있다.
- . JNDI 리소스가 존재하지 않을 경우에는 직접 드라이버이름과, URL 및 아이디, 비밀번호를 설정하여 DB 서버에 접속할 수 있다.

<sql:setDataSource var="변수명" scope="범위" dataSource="dataSource"  
driver="driver" uri="uri" user="user" password="password">

## <sql:query> 데이터베이스 서버에 쿼리를 전송할 때 사용

- . dataSource속성에는 JNDI 리소스나 <sql:setDataSource>로 데이터베이스 서버에 접속한 연결정보를 얻어온 변수를 설정
  - . Sql속성에는 실행할 SQL문을 지정
  - . Var속성의 변수에는 SQL 쿼리가 실행된 결과를 저장
  - . startRow 는얻어온 쿼리결과와 시작행 값을 의미. 1번째 레코드 값부터 시작하려면 0으로 설정한다.
  - . maxRows는 얻어온 쿼리결과와 레코드 최대 수를 의미
- <sql:query var="변수명" scope="범위" sql="sql" dataSource="dataSource" startRow="startRow" maxRows="maxRows">

## <sql:update>주로레코드의추가, 수정, 삭제기능을 사용

var속성에입력한변수에는레코드를업데이트한결과가저장

sql속성에는레코드를업데이트할SQL문을지정

<sql:update var="변수명" scope="범위" sql="sql" dataSource="dataSource">

## <sql:param>

- . <sql:query> 태그나 <sql:update> 태그로 SQL 문장을 전송할 때 파라미터를 전달해주는태그
- . 예를 들어 'SELECT \* FROM STUDENT WHERE NAME=?' 쿼리를 전송할 때? 부분의 값을 <sql:param> 태그를 사용하여 지정할 수 있다.

<sql:param value="value">

### <sql:dateParam>

- . <sql:param>과 같이 파라미터를 전달하는 기능을 제공
  - . 차이점은 <sql:dateParam>태그는 날짜 정보의 파라미터를 전달할 때 사용
  - . type 속성에는 date, time, timestamp 중 하나의 값이 올 수 있다
- <sql:param value="value">

### <sql:transaction>

- . 데이터베이스 작업 시 트랜잭션을 사용하도록 해 준다.
  - . dataSource속성에는 JNDI 리소스 이름을 입력하거나 <sql:setDataSource>로 데이터베이스에 연결한 정보를 얻은 변수를 입력한다.
  - . Isolation 속성을 이용하여 격리수준을 지정할 수 있다. 지정할 수 있는 속성값으로는 read\_committed, read\_uncommitted, repeatable\_read, serializable을 입력할 수 있다.
- <sql:transaction dataSource="dataSource" isolation="isolation">

### Table 생성

Create table test( num number, name varchar2(10), primary key(num));

# jstl\_sql\_ex.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql" %>
<html><head><title>JSTL sql 라이브러리 사용 예제</title></head>
<body>
<sql:setDataSource var="conn" driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:orcl" user="scott" password="tiger"/>
<sql:update dataSource="${conn}">
INSERT INTO test (num, name) VALUES (1, '홍길동') </sql:update>
<sql:update dataSource="${conn}">
INSERT INTO test (num, name) VALUES (2, '조준동') </sql:update>
<sql:update dataSource="${conn}">
INSERT INTO test (num, name) VALUES (3, '홍길동') </sql:update>
<sql:update dataSource="${conn}">
INSERT INTO test (num, name) VALUES (4, '홍길순') </sql:update>
<sql:query var="rs" dataSource="${conn}">
SELECT * FROM test WHERE name=?
<sql:param>홍길동</sql:param></sql:query>
<c:forEach var="data" items="${rs.rows}">
<c:out value="${data['num']}" />
<c:out value="${data['name']}" />
<br>
</c:forEach>
</body></html>
```