5장. jsp 쿠키와 세션

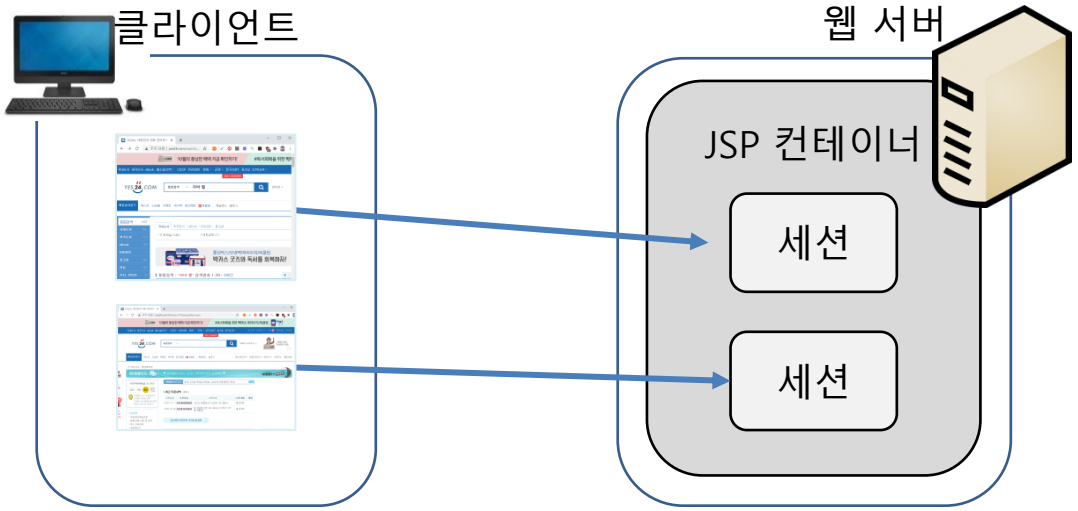
cookie / session

세션(session)

세션(session)

- 클라이언트와 웹 서버 간의 상태를 지속적으로 유지하는 방법을 말한다.
- 사용자 인증을 통해 특정 페이지를 사용할 수 있도록 권한 상태 유지.

  예) 웹 쇼핑몰 – 장바구니나 주문 처리와 같은 회원 전용 페이지 로그인 후 다른 웹 페이지에 갔다가 돌아와도 로그인 상태 유지됨
- 세션은 오직 웹 서버에 존재하는 객체로 웹 브라우저마다 하나씩 존재하므로 브라우저를 닫기 전까지 웹 페이지를 이동하더라도 사용자 정보가 유지된다.



세션(session)

세션(session)이란?

세션은 HTTP 프로토콜을 이용하는 웹 환경에서 상태를 유지하기 위한 기술이다. HTTP는 요청과 응답으로 이루어지며, 특정 URL을 요청할 때마다 **새로운 HTTP 요청이 생성**되기 때문에 이들간에 상태를 유지할 수 있는 방법이 없다.
예를 들어 /login.jsp 라는 URL에서 로그 인을 했다고 하더라도 /boards 페이지로 이동하게 되면 새로운 HTTP 요청이므로 로그 인을 했다는 정보를 확인할 수 없는 것이다.

이러한 HTTP의 무상태(Stateless)한 특성을 극복하고자 나온 기술이 쿠키와 세션이다.
이 둘은 특정 데이터를 저장해두고 페이지를 이동하거나 새로 고침 하는 등 HTTP 요청이 매번 발생해도 특정 상태(데이터)를 유지할 수 있는 기술이다.

세션(session)

세션(session) 생성

createSession.jsp

```jsp
<%@ page session="true" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>세션 생성</title>
</head>
<body>
    <h2>세션 생성 성공</h2>
    <%=session %><br>

    <p>세션 ID : <%=session.getId() %>
</body>
</html>
```

세션 생성 성공

org.apache.catalina.session.StandardSessionFacade@77feec42

세션 ID : 6CE2E8E852EAD924481EF9C40E3FA386

세션(session)

세션의 프로세스(Process)

1. 브라우저가 서버 측에 특정 페이지를 요청합니다. /createSession.jsp 요청

Request URL: http://localhost:8282/session_cookie/sessionCreate.jsp

Request Method: GET

Status Code: 200

Remote Address: [::1]:8282

2. 이후 요청 시마다 해당 세션 ID를 HTTP 요청 헤더에 포함하여 요청한다.

▼ **Request Headers** view source

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,
f,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;
0.9

Accept-Encoding: gzip, deflate, br

Accept-Language: ko,en;q=0.9,ja;q=0.8,en-US;q=0.7,ko-KR;q=0.6

Cache-Control: max-age=0

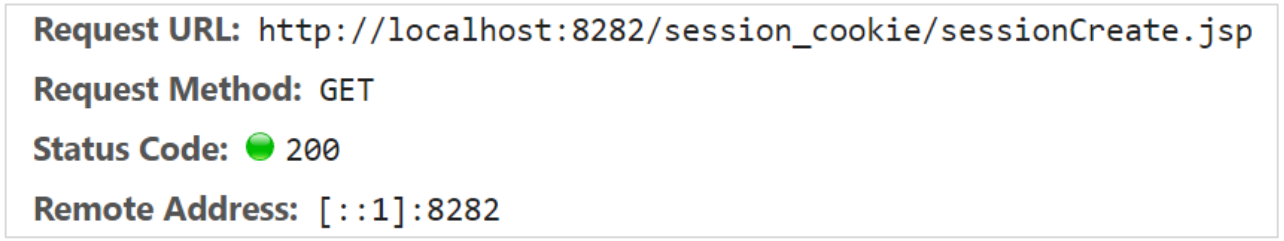
Connection: keep-alive

Cookie: JSESSIONID=6CE2E8E852EAD924481EF9C40E3FA386

세션(session)

세션의 프로세스(Process)

1. 브라우저가 서버 측에 특정 페이지를 요청합니다. /createSession.jsp 요청



2. 이후 요청 시마다 해당 세션 ID를 HTTP 요청 헤더에 포함하여 요청한다.

3. 서버는 요청 헤더의 JSESSIONID 쿠키를 참조하여 자신이 가지고 있는 세션 객체를 구분한다. 예를 들어 해당 세션에 로그 인 된 사용자 정보 등이 담겨 있으면 로그 인이 되어 있는 것으로 판단하게 된다.

세션(session)

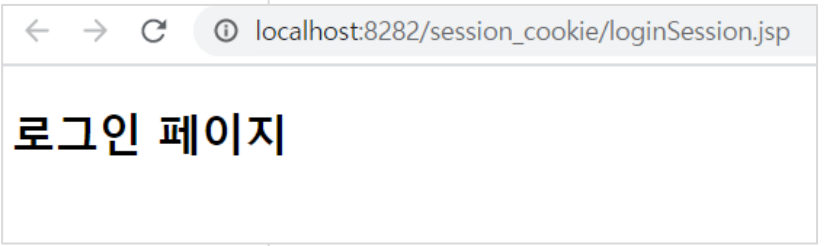
세션(session) 내장 객체 메서드

메소드	반환유형	설 명
setAttribute(String name, Object value)	void	세션 속성 이름이 name 속성에 value를 할당
getAttribute(String name)	java.lang.Object	세션 속성 이름이 name인 값을 Object형으로 반환한다. **형 변환**이 필요함(Object->String)
getId()	Java.lang.String	세션에 할당된 고유 아이디를 String형으로 반환
setMaxInactiveInterval(int interval)	void	해당 세션을 유지하기 위해 세션 유지 시간을 초 단위로 설정
getMaxInactiveInterval(int interval)	int	해당 세션을 유지하기 위해 세션 유지 시간을 반환. 기본값은 1800초(30분)임
removeAttribute(String name)	void	세션 속성 이름이 name인 속성을 제거
invalidate()		현재 세션에 저장된 모든 세션 속성을 제거

세션(session)

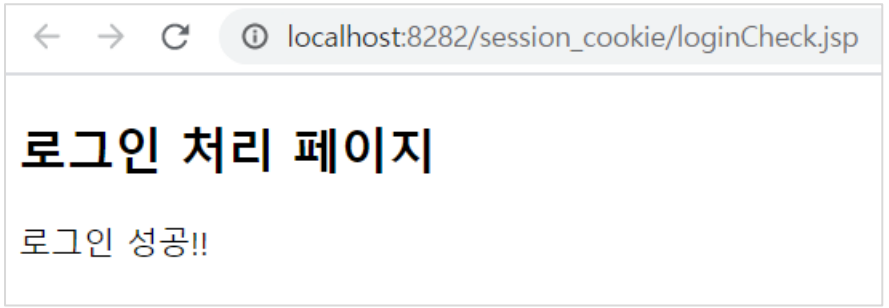
세션 저장(발급) 및 가져오기

```jsp
<%@ page session="true" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>세션 생성</title>
</head>
<body>
    <h2>로그인 페이지</h2>
    <%
        session.setAttribute("login", true);
    %>
</body>
</html>
```



세션(session)

세션 저장(발급) 및 가져오기




```jsp
<body>
    <h2>로그인 처리 페이지</h2>
    <%
        boolean isLogin = (boolean)session.getAttribute("login");

        if(isLogin == true)
            out.print("로그인 성공!!");
        else
            out.print("로그인 실패!!");
    %>
</body>
```

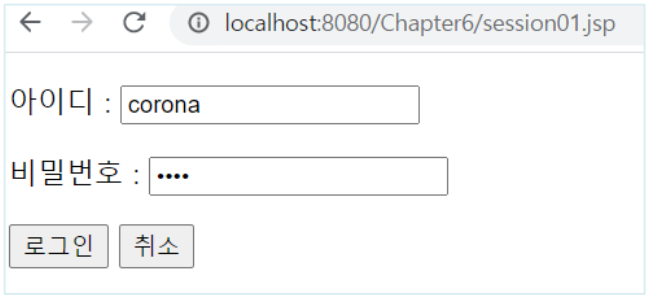
세션(session)

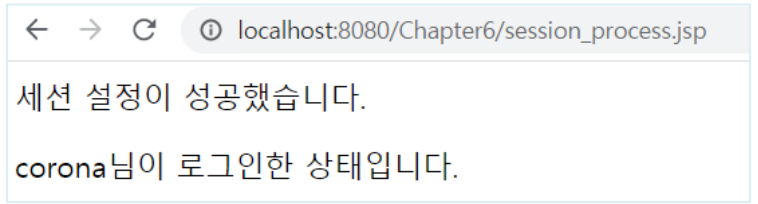
세션(session) 생성 예제2.

- 세션 생성은 session 객체의 setAttribute() 메소드를 사용한다.

  세션 속성 이름 – userId, 속성 값 - admin

void setAttribute("userId", userId);






```
<form action="session01_process.jsp" method="post">
    <p>아 이 디: <input type="text" name="id">
    <p>패스워드: <input type="password" name="passwd">
    <p><input type="submit" value="로그인"> </p>
</form>
```

세션(session)

세션(session) 생성(부여)

- 세션이 생성(부여)되는 시기는 로그인에 성공한 때이다.

```
<%
    String userId = request.getParameter("id");
    String userPw = request.getParameter("passwd");

    if(userId.equals("corona") && userPw.equals("2019")){
        session.setAttribute("userId", userId);
        session.setAttribute("userPw", userPw);
        out.println("세션 설정이 성공했습니다.");
    }
    else{
        out.println("<script>");
        out.println("alert('아이디나 비밀번호가 일치하지 않습니다.')");
        out.println("history.go(-1)");
        out.println("</script>");
    }
%>
<p><%=session.getAttribute("userId") %>님이 로그인한 상태입니다.
```

세션(session)

세션(session) 정보 얻기 – 단일 세션 정보

- 세션 정보는 session 객체의 getAttribute() 메소드를 사용한다.
- 반환 유형이 Object형이므로 반드시 형 변환을 사용해야 한다.

String id = (String)session.getAttribute("userId");

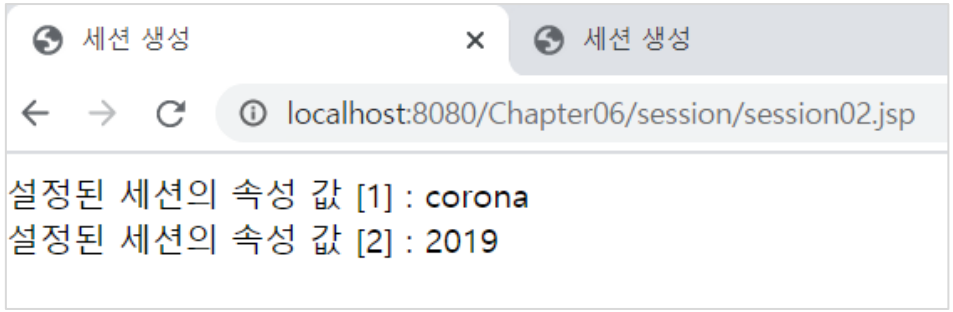
같은 브라우저에서 세션이 부여되면 계속 유지(저장)됨

```
<%
    String id = (String)session.getAttribute("userId");
    String pw = (String)session.getAttribute("userPw");
    //세션을 얻어올때 형변환 해야함

    out.println("설정된 세션의 속성 값 [1] " + id + "<br>");
    out.println("설정된 세션의 속성 값 [2] " + pw + "<br>");
%>
```

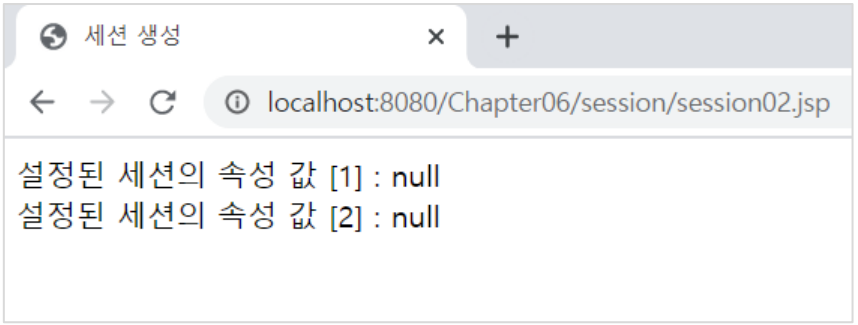
세션(session)

세션(session) 유지 – 웹 브라우저 단위

- 세션은 오직 웹 서버에 존재하는 객체로 웹 브라우저마다 하나씩 존재하므로 브라우저를 닫기 전까지 웹 페이지를 이동하더라도 사용자 정보가 유지된다.



세션이 생성된 브라우저는 새창을 열어도 세션이 계속 유지됨



브라우저를 새로 열면 세션이 유지되지 않음

세션(session)

세션(session) 삭제 – 단일 세션 삭제

- 세션에 저장된 하나의 세션 속성 이름을 삭제하려면 removeAttribute() 사용

session.removeAttribute("userId");

```
----세션을 삭제하기 전----

설정된 세션의 이름 userId : corona
설정된 세션의 이름 userPw : 2019

----세션을 삭제한 후----

설정된 세션의 이름 userId : null
설정된 세션의 이름 userPw : 2019
```

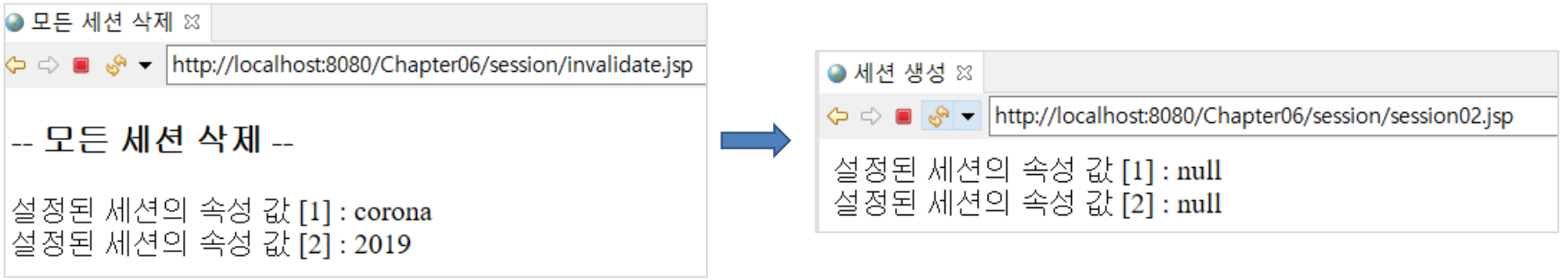
```
<h3>----세션을 삭제하기 전----</h3>
<%
    String id = (String)session.getAttribute("userId");
    String pw = (String)session.getAttribute("userPw");

    out.println("설정된 세션의 이름 userId : " + id + "<br>");
    out.println("설정된 세션의 이름 userPw : " + pw + "<br>");

    session.removeAttribute("userId");
%>
<h3>----세션을 삭제한 후----</h3>
<%
    id = (String)session.getAttribute("userId");
    pw = (String)session.getAttribute("userPw");

    out.println("설정된 세션의 이름 userId : " + id + "<br>");
    out.println("설정된 세션의 이름 userPw : " + pw + "<br>");
%>
```

세션(session)

모든 세션 삭제

- 세션에 저장된 모든 속성 이름을 삭제

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



세션(session)

모든 세션 삭제

```
<h3>-- 모든 세션 삭제 --</h3>
<%
    String id = (String)session.getAttribute("userId");
    String pw = (String)session.getAttribute("userPw");

    out.println("설정된 세션의 속성 값 [1] : " + id + "<br>");
    out.println("설정된 세션의 속성 값 [2] : " + pw + "<br>");

    session.invalidate();   //세션 삭제
%>
```

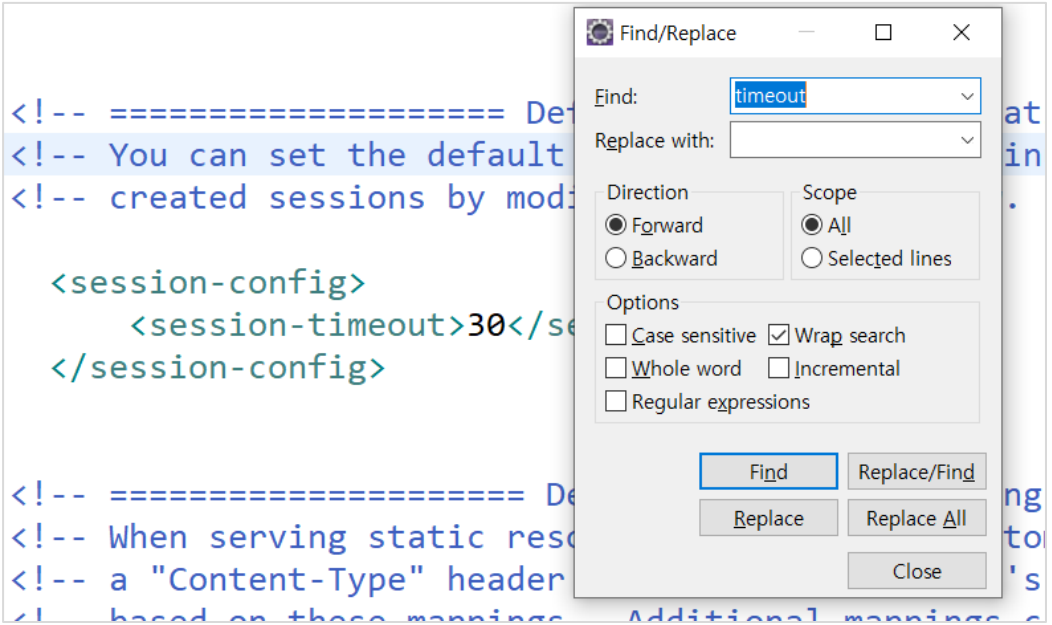
세션(session)

세션(session) 유효시간 설정

- 세션 유효 시간은 세션을 유지하기 위한 세션의 일정 시간을 말한다. 기본값은 30분이다.
- 유효시가을 설정할 때는 session 객체의 **setMaxInactiveInterval()**을 사용한다.

```
session.setMaxInactiveInterval(60*60);  //세션 유효시간 60분(3600초) 설정
```

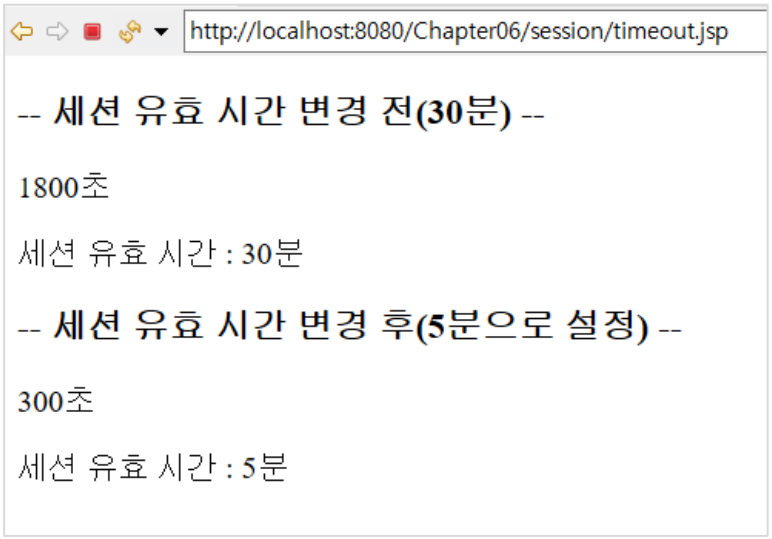
Servers > web.xml

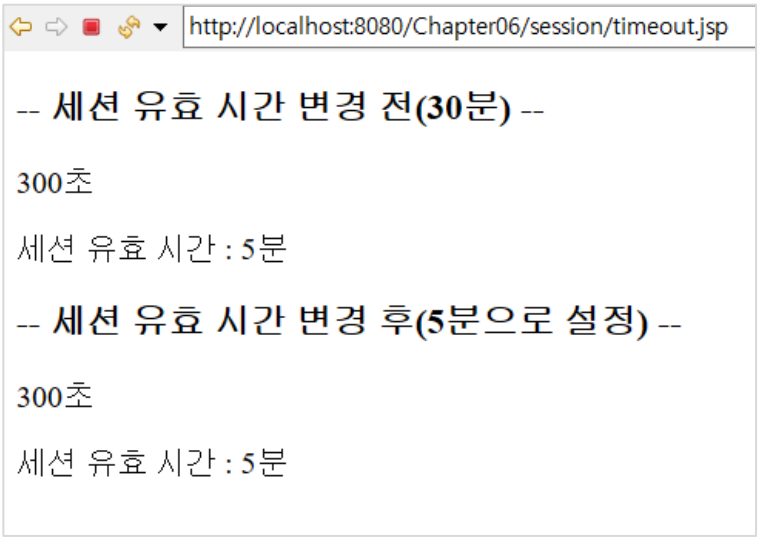


세션(session)

세션(session) 유효시간 설정 – 5분으로 설정

은행사이트에 로그인한 경우 10분에서 초단위로 역카운팅 되면서 10분이 지나면 자동 로그아웃[session.invalidate()] 됨





변경후 브라우저를 새로고침하면 변경된 유효시간을 표시함

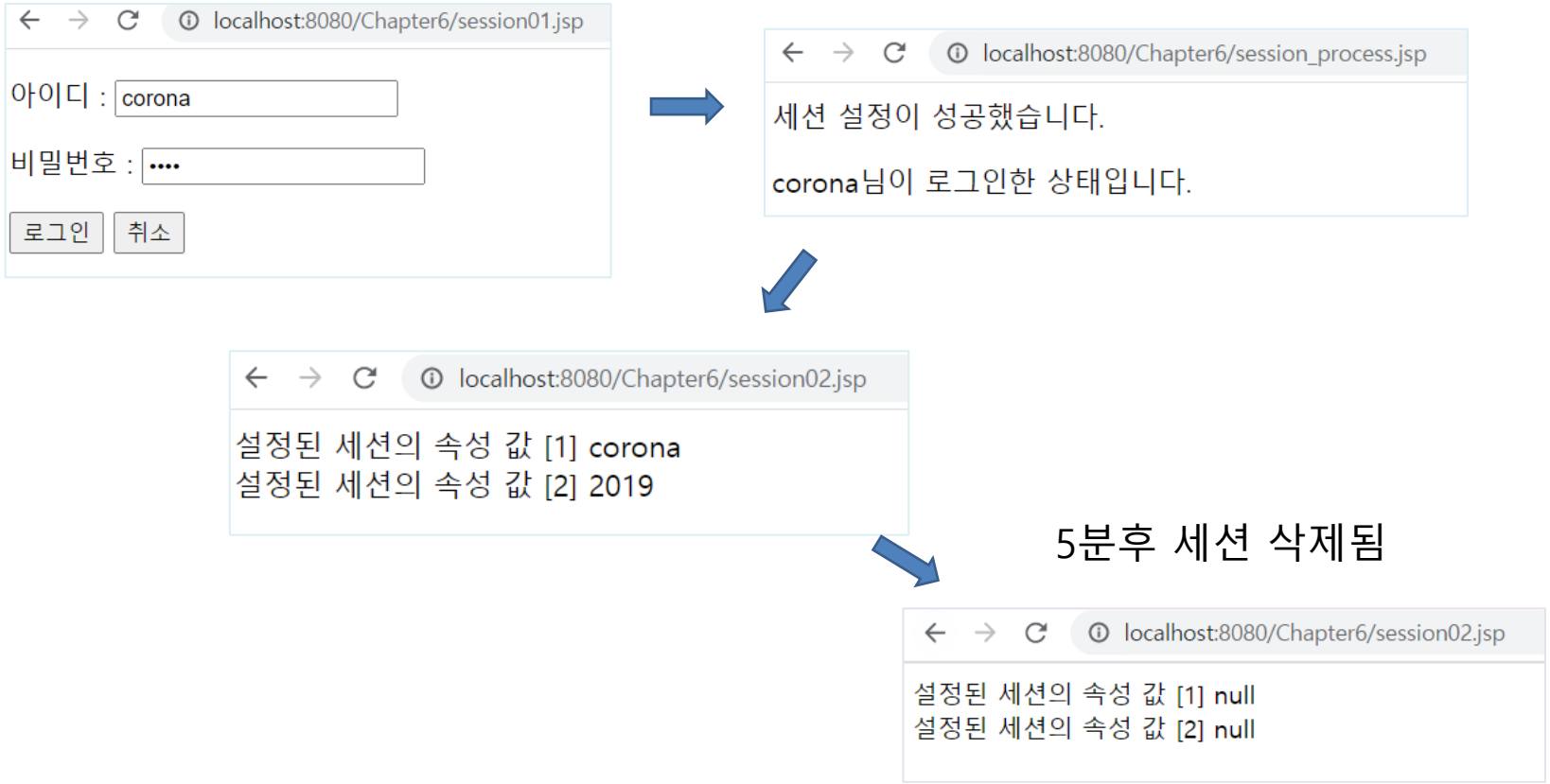
세션(session)

세션(session) 유효시간 설정 – 5분으로 설정

```
<h3>-- 세션 유효 시간 변경 전(30분) -- </h3>
<%
    int time = session.getMaxInactiveInterval() / 60;

    out.println("<p>" + session.getMaxInactiveInterval() + "초</p>");
    out.println("<p>세션 유효 시간 : " + time + "분</p>");
%>

<h3>-- 세션 유효 시간 변경 후(5분으로 설정) --</h3>
<%
    session.setMaxInactiveInterval(5*60);

    time = session.getMaxInactiveInterval() / 60;

    out.println("<p>" + session.getMaxInactiveInterval() + "초</p>");
    out.println("<p>세션 유효 시간 : " + time + "분</p>");
%>
```

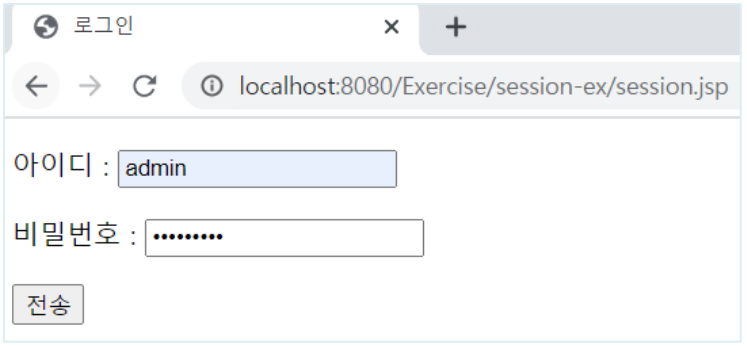
세션(session)

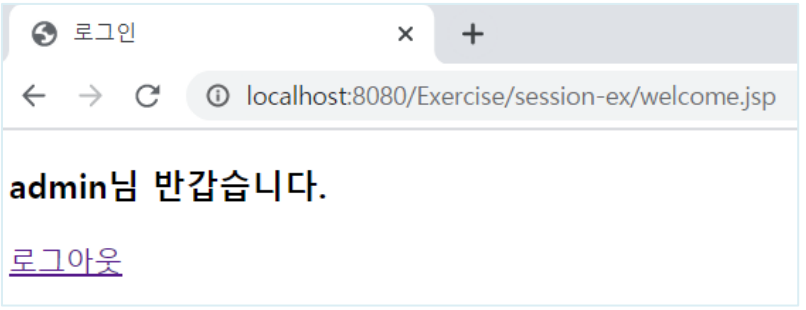
세션(session) 유효시간 실습하기 – 5분으로 설정



Session 연습문제

- 실습 예제





1. session.jsp 파일 생성
2. session_process.jsp 파일 생성
   아이디와 비밀번호가 인증되면 아이디 값을 세션명 userID 값으로 설정
   respons 객체의 sendRedirect() 메서드를 이용하여 welcome.jsp 파일로 이동
3. welcome.jsp 파일을 생성
   설정된 세션명 userID 값이 null이면 sendRedirect()를 이용하여 session_out.jsp 파일로 이동
   <로그아웃>을 클릭하면 설정된 세션을 해제하도록 작성
4. session_out.jsp 파일을 생성
   설정된 모든 세션명을 해제하도록 작성한다.

Session 연습문제

- 실습 예제

session.jsp

```
<form action="session_process.jsp" method="post">
    <p>
        <label for="id">아이디 : </label>
        <input type="text" id="id" name="id">
    </p>
    <p>
        <label for="passwd">비밀번호 : </label>
        <input type="password" id="passwd" name="passwd">
    </p>
    <p><input type="submit" value="전송"></p>
</form>
```

session_process.jsp

```
<%
    String id = request.getParameter("id");
    String pw = request.getParameter("passwd");

    if(id.equals("admin") && pw.equals("admin1234")){
        session.setAttribute("userID", id);
        response.sendRedirect("welcome.jsp");
    }
    else{
        out.println("아이디와 비밀번호가 일치하지 않습니다.");
    }
%>
```

Session 연습문제

welcome.jsp

```
<%
    String userID = null;
    if(session.getAttribute("userID") != null){
        userID = (String)session.getAttribute("userID");
    }
    else{
        response.sendRedirect("session_out.jsp");
    }
%>
<h3><%=session.getAttribute("userID") %>님 반갑습니다.</h3>
<a href="session_out.jsp">로그아웃</a>
```

session_out.jsp

```
<%
    session.invalidate();

    response.sendRedirect("session.jsp");
%>
```

세션(session) – 장바구니

세션 실습 - 장바구니(Shopping Cart)

프로그램 소스 목록

파일 이름	역 할
loginForm.jsp	로그인 폼 화면, 사용자 이름을 입력하는 양식만 제공
selProcuct.jsp	상품을 선택하는 화면으로 리스트에서 원하는 상품을 선택하고 <추가>버튼을 눌러 상품을 추가한다. 사용자 세션 설정
productAdd.jsp	selProduct에서 선택한 상품을 세션에 넣는다. 선택된 데이터를 모두 선택해야하므로 ArrayList를 이용한다.
checkOut.jsp	세션이 살아 있고, 하나 이상의 상품을 선택한 상태라면 선택한 상품의 목록을 보여준다.

세션(session) – 장바구니

세션을 이용한 장바구니(Shopping Cart) 구현

세션(session) – 장바구니

세션을 이용한 장바구니(Shopping Cart) 구현

loginForm.jsp

```
<head>
    <meta charset="UTF-8">
    <title>Login</title>
    <style type="text/css">
        #container{width: 600px; margin: 0 auto; text-align: center;}
    </style>
</head>
<body>
    <div id="container">
        <h2>로그인</h2>
        <hr>
        <!-- 사용자 이름을 전송 -->
        <form action="selProduct.jsp" method="post">
            <input type="text" name="username">
            <input type="submit" value="로그인">
        </form>
    </div>
</body>
```

세션(session) – 장바구니

selProduct.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    request.setCharacterEncoding("utf-8");

    String username = request.getParameter("username");
    session.setAttribute("userName", username);  //세션 부여
%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Shopping Cart</title>
<style type="text/css">
    #container{ width: 100%; margin: 0 auto; text-align: center}
</style>
</head>
```

세션(session) – 장바구니

selProduct.jsp

```
<body>
    <div id="container">
        <h2>상품 선택</h2>
        <hr>
        <p><%=session.getAttribute("userName") %>님 환영합니다!!
            <form action="productAdd.jsp" method="post">
                <select name="product">
                    <option>사과</option>
                    <option>귤</option>
                    <option>토마토</option>
                    <option>키위</option>
                </select>
                <input type="submit" value="추가">
            </form>
            <p><input type="button" value="계산하기" onclick="location.href='checkOut.jsp'">
    </div>
</body>
</html>
```

세션(session) – 장바구니

productAdd.jsp

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

    //세션 유지
    ArrayList<String> productList = (ArrayList)session.getAttribute("productList");

    //리스트 생성 및 세션 생성
    if(productList == null){  //리스트가 초기화된 상태에서..
        productList = new ArrayList<>();
        session.setAttribute("productList", productList);
    }

    String product = request.getParameter("product");

    //상품 저장
    productList.add(product);
%>
```

세션(session) – 장바구니

productAdd.jsp

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>로그인</title>
</head>
<body>
    <script type="text/javascript">
        alert("<%=product %>가(이) 추가되었습니다.");
        history.go(-1);
    </script>
</body>
</html>
```

세션(session) – 장바구니

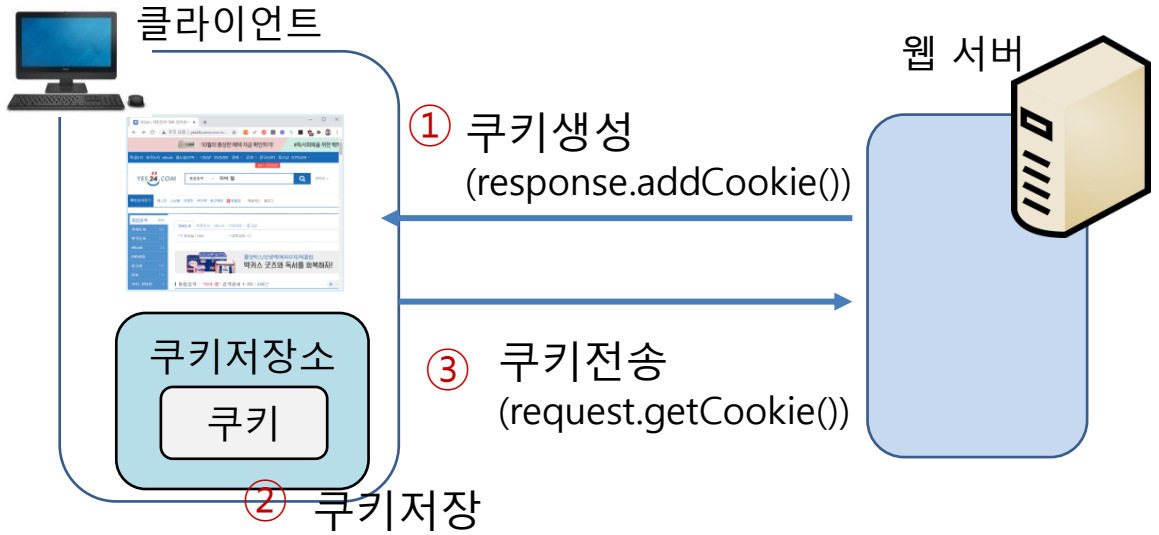
checkOut.jsp

```
<div id="container">
    <h2>계산 하기</h2>
    <p>선택한 상품 목록</p>
    <hr>
    <%
        //세션 유지
        ArrayList<String> productList = (ArrayList)session.getAttribute("productList");

        //상품 목록
        for(String product : productList)
            out.println(product + "<br>");
    %>
</div>
```

쿠키(cookie)

쿠키(cookie)

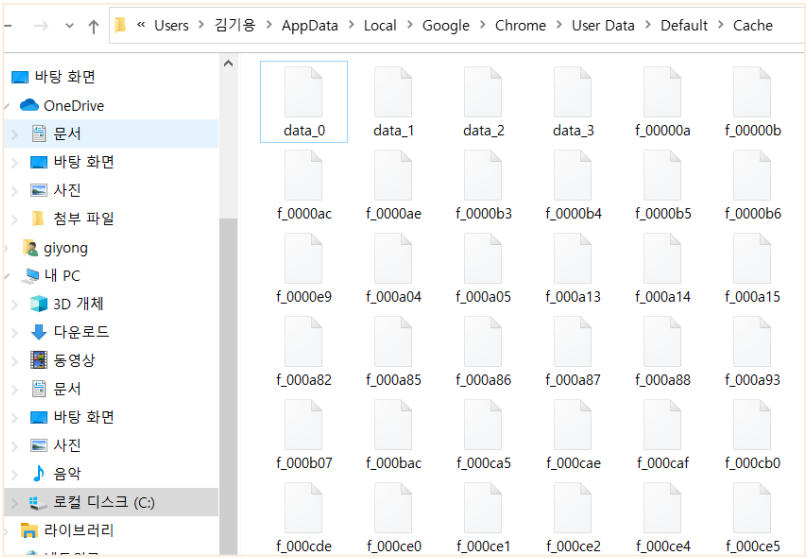
- 클라이언트와 웹 서버간의 상태를 지속적으로 유지하는 방법으로 쿠키와 세션이 있다.
- 쿠키는 세션과 달리 상태 정보를 웹 서버가 아닌 클라이언트에 저장한다.
  예) 어떤 웹 사이트를 처음 방문한 사용자가 로그인 인증을 하고 나면 아이디와 비밀번호를 기록한 쿠키가 만들어지고, 그 다음부터 사용자가 그 사이트에 접속하면 별도의 절차를 거치지 않고 쉽게 접속할 수 있다.
- 쿠키는 클라이언트의 일정 폴더에 정보를 저장하므로 웹 서버의 부하를 줄일 수 있으나 개인 정보 기록이 남기 때문에 보안에 문제가 있다.



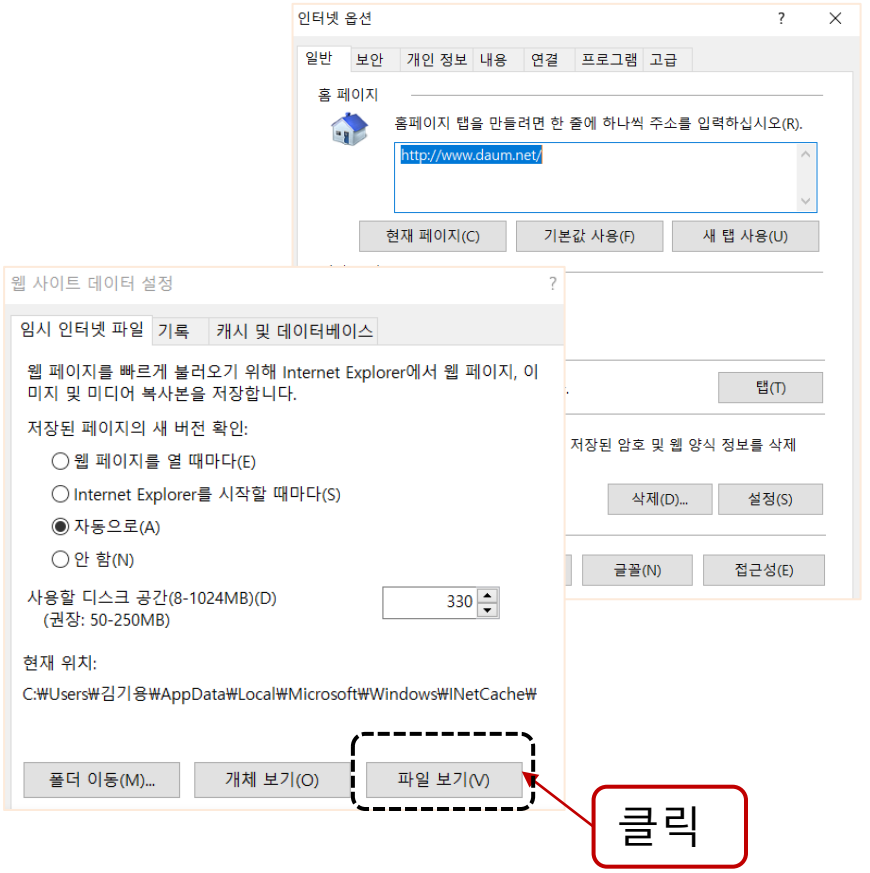
쿠키(cookie)

쿠키(cookie)

Chrome 폴더안의 쿠키 파일



Internet Exploer 쿠키 파일



쿠키(cookie)

쿠키(cookie) 내장 객체 메서드

메소드	반환유형	설 명
getName()	String	쿠키의 이름을 반환한다.
getValue()	String	쿠키에 설정된 값을 반환한다.
setMaxAge(int)	void	쿠키의 유효 기간을 설정한다.

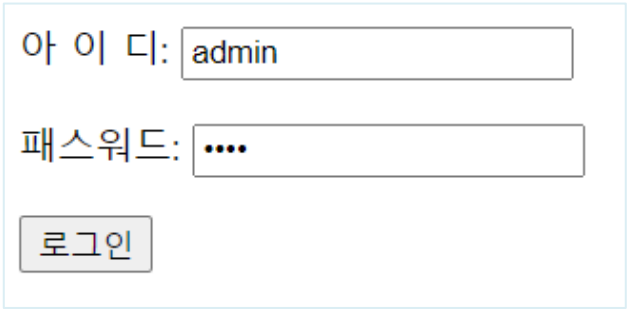
쿠키(cookie)

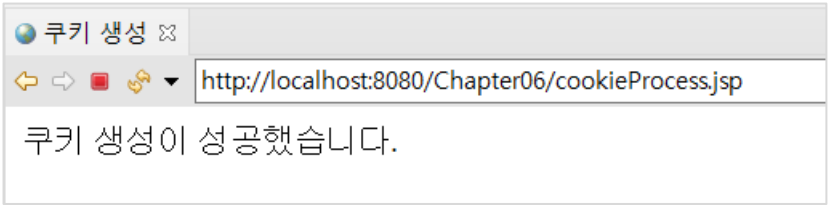
쿠키(cookie) 생성

- 쿠키 생성은 Cookie 클래스로 쿠키를 생성한 후 response 객체의 **addCookie()** 를 사용한다..

  쿠키 이름 – memberId, 쿠키 값 - admin

```
Cookie cookie = new Cookie("memberId", "admin");
response.addCookie(cookie);
```





쿠키(cookie)

쿠키(cookie) 생성

cookie01.jsp

```
<form action="cookieProcess.jsp" method="post">
    <p>
        <label for="id">아이디 : </label>
        <input type="text" id="id" name="id">
    </p>
    <p>
        <label for="passwd">비밀번호 : </label>
        <input type="password" id="passwd" name="passwd">
    </p>
    <p><input type="submit" value="전송"></p>
</form>
```

쿠키(cookie)

쿠키(cookie) 생성

cookieProcess.jsp

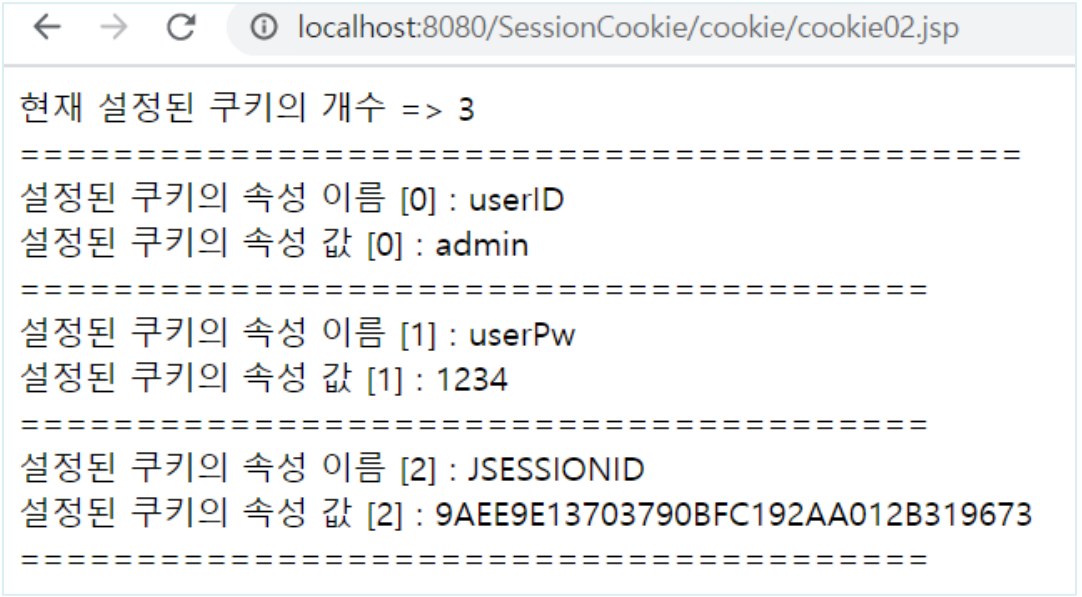
```
<body>
    <%
        String id = request.getParameter("id");
        String passwd = request.getParameter("passwd");

        if(id.equals("admin") && passwd.equals("0000")){
            /* Cookie(쿠키이름, 쿠키값)  */
            Cookie cookieID = new Cookie("userID", id);
            Cookie cookiePW = new Cookie("userePW", passwd);

            //response로 생성
            response.addCookie(cookieID);
            response.addCookie(cookiePW);

            out.println("쿠키 생성이 성공했습니다.");
        }
        else{
            out.println("쿠키 생성이 실패했습니다.");
        }
    %>
</body>
```

쿠키(cookie)

쿠키(cookie) 정보

- 클라이언트에 저장된 모든 쿠키 객체를 가져오려면 request 객체의 **getCookies()** 를 사용한다.

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



쿠키(cookie)

쿠키(cookie) 정보

```
<%
    Cookie[] cookies = request.getCookies();
    out.println("현재 설정된 쿠키의 개수 => " + cookies.length + "<br>");
    out.println("==========================================<br>");
    for(int i=0; i<cookies.length; i++){
        out.println("설정된 쿠키의 속성 이름 [" + i + "] : " + cookies[i].getName() + "<br>");
        out.println("설정된 쿠키의 속성 값 [" + i + "] : " + cookies[i].getValue() + "<br>");
        out.println("==========================================<br>");
    }
%>
```

JSESSIONID란?

- 톰캣 컨테이너에서 세션을 유지하기 위해 발급하는 키
- HTTP 프로토콜은 stateless하다. 요청시마다 새로운 연결이 생성되고 응답후 연결은 끊기게 되므로 상태를 유지할 수 없다.
- 따라서, 상태를 저장하기 위해서 톰캣은 JSESSIONID 쿠키를 클라이언트에게 발급해주고 이 값을 통해 세션을 유지할 수 있도록 한다.

쿠키(cookie)

쿠키(cookie) 삭제

- 쿠키의 유효 기간을 만료한다.

cookie.setMaxAge(0);

```
<%
    //클라이언트 컴에 저장된 쿠키를 가져온다.
    Cookie[] cookies = request.getCookies();

    //쿠키 삭제
    for(int i=0; i<cookies.length; i++){
        cookies[i].setMaxAge(0);
        response.addCookie(cookies[i]);
        //서버쪽에서 클라이언트로 삭제된 쿠키정보를 응답.
    }

    response.sendRedirect("cookie02.jsp");
    //cookie02 페이지로 강제 이동
%>
```