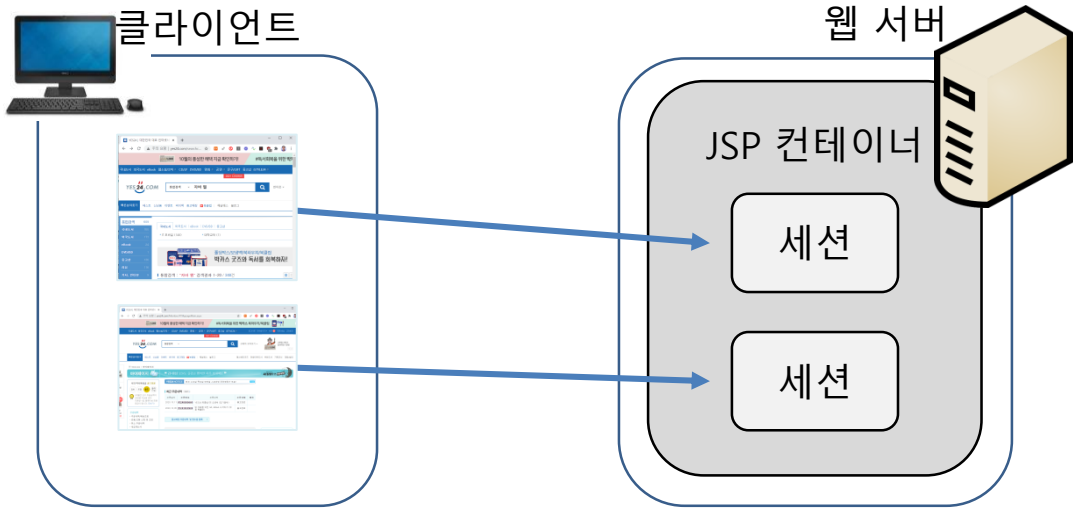
3장. 세션과 쿠키

session / cookie

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.
  예) /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

```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>
```

/session/createSession.jsp

세션 생성 성공

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

세션 ID : 6CE2E8E852EAD924481EF9C40E3FA386

세션(session)

세션의 프로세스(Process) – 브라우저 > 개발자도구 > Network 탭

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

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

세션(session)

JSESSIONID란?

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

Request Cookies ☐ show filtered out request cookies

Name	Value
userId	today
userPw	1234
JSESSIONID	D970B66B26B7EEBB8DB05BF984824364

Response Cookies

Name	Value
userId	today
userPw	1234

세션(session)

세션(session) 내장 객체 메서드

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

세션(session)

세션(session) 생성

- 세션 생성은 session 객체의 setAttribute() 메소드를 사용한다.
- 세션 속성 이름 – userId, 속성 값 - admin

```
void setAttribute("userId", userId);
```

세션(session)

세션(session) 생성

session/session01.jsp

```
<h2>로그인</h2>
<form action="session01_process.jsp" method="post">
    <p>아이디: <input type="text" name="id"></p>
    <p>패스워드: <input type="password" name="passwd"></p>
    <p><input type="submit" value="로그인"></p>
</form>
```

세션(session)

세션(session) 생성(부여)

- 세션이 생성(부여)되는 시기는 로그인에 성공한 때이다.

session/session01_process.jsp

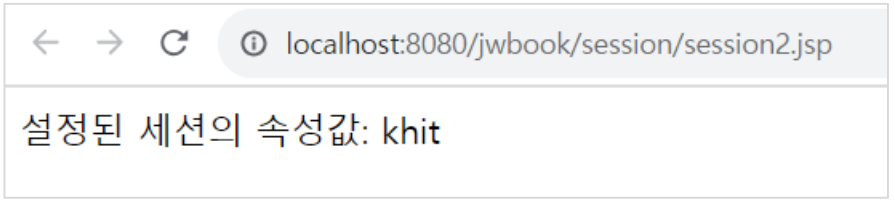
```
<%
    String userId = request.getParameter("uid");
    String userPw = request.getParameter("passwd");

    if(userId.equals("khit") && userPw.equals("1234")){
        //로그인이 성공하면 세션을 발급함(이름:"userID")
        session.setAttribute("userID", userId);
        out.println("세션이 발급되었습니다.");
    }else{
        out.println("<script>");
        out.println("alert('아이디나 비밀번호가 일치하지 않습니다.')");
        out.println("history.back()");
        out.println("</script>");
    }
%>
<p><%=session.getAttribute("userID") %>님이 로그인한 상태입니다.</p>
```

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

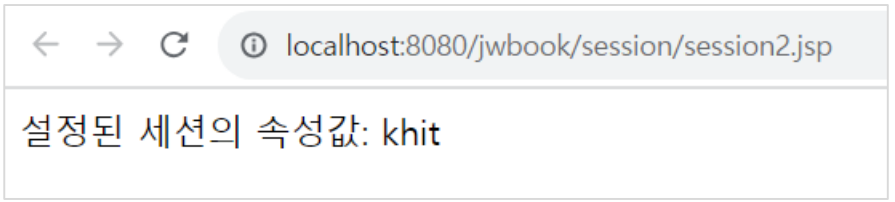
session/session02.jsp

```
<%
    //세션 정보 얻기
    String id = (String)session.getAttribute("userID");

    out.println("설정된 세션의 속성값: " + id + "<br>");
%>
```

세션(session)

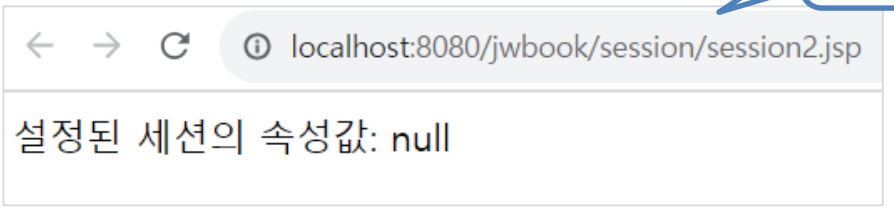
세션(session) 유지 – 웹 브라우저 단위

- 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

Session02.jsp 실행



브라우저를 새로 열면 세션이 유지되지 않음

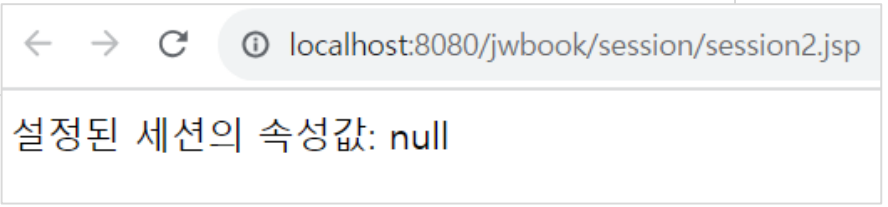
세션(session)

모든 세션 삭제

session.invalidate();

session/session03.jsp

```
<%
    String id = (String)session.getAttribute("userID");

    //모든 세션 삭제
    session.invalidate();
%>
<p>세션을 삭제했습니다.</p>
```



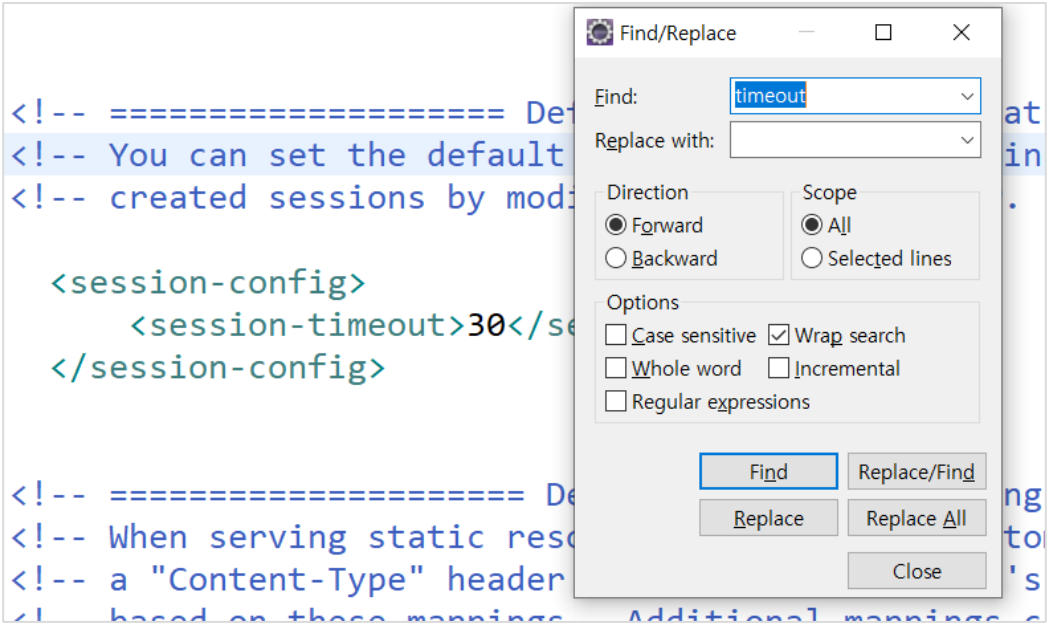
세션(session)

세션(session) 유효시간 설정

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

```
session.setMaxInactiveInterval(10*60);  //세션 유효시간 10분(600초) 설정
```

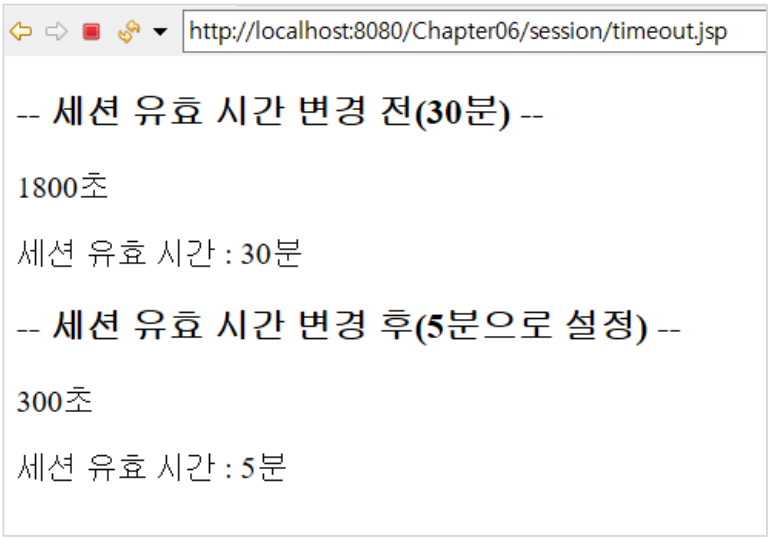
경로 : **servers > web.xml**

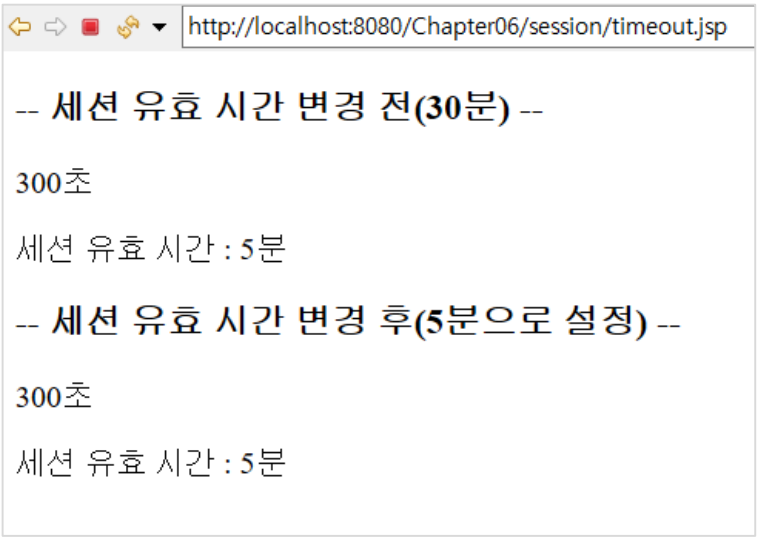


세션(session)

세션(session) 유효시간 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

```jsp
<h3>세션 유효시간 변경 전</h3>
<%
    // 30*60=1800초
    int time = session.getMaxInactiveInterval();
    out.println(time + "초<br>");

    time = time / 60;
    out.println(time + "분<br>");
%>

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

    // 5*60=300초
    time = session.getMaxInactiveInterval();
    out.println(time + "초<br>");

    time = time / 60;
    out.println(time + "분<br>");
%>
```

session/timeout.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

cart/loginform.jsp

```
<title>로그인</title>
<style>
    #container{width: 80%; margin: 0 auto; text-align: center;}
</style>
</head>
<body>
    <section id="container">
        <h2>로그인</h2>
        <hr>
        <form action="selProduct.jsp" method="post">
            <p>
                <input type="text" name="username" placeholder="이름 입력">
                <input type="submit" value="로그인">
            </p>
        </form>
    </section>
</body>
```

세션(session) – 장바구니

cart/selproduct.jsp

```jsp
<%
    //세션 발급
    String username = request.getParameter("username");
    //out.println(username);
    session.setAttribute("sessionName", username);

%>
<section id="container">
    <h2>상품 선택</h2>
    <hr>
    <p><b>[<%=session.getAttribute("sessionName") %>]</b>님 환영합니다.</p>
    <form action="addProduct.jsp" method="post">
        <select name="product">
            <option value="사과">사과</option>
            <option value="바나나">바나나</option>
            <option value="포도">포도</option>
        </select>
        <button type="submit">추가</button>
    </form>
    <p>
        <button onclick="location.href='cart.jsp'">장바구니</button>
    </p>
</section>
```

세션(session) – 장바구니

cart/addproduct.jsp

```jsp
<%
    request.setCharacterEncoding("utf-8");
    //상품 리스트 생성 및 상품 세션 발급
    ArrayList<String> productList = (ArrayList)session.getAttribute("sessionList");
    if(productList == null){
        productList = new ArrayList<>();
        session.setAttribute("sessionList", productList); //상품 세션 발급
    }

    //선택한 상품 받아서 상품리스트에 저장
    String product = request.getParameter("product");
    productList.add(product);
%>
<script>
    alert("<%=product %>가 추가되었습니다.");
    history.back();
</script>
```

세션(session) – 장바구니

cart/cart.jsp

```jsp
<div id="container">
    <h2>장바구니</h2>
    <h3>선택한 상품 목록</h3>
    <hr>
    <%
        //세션 유지
        ArrayList<String> productList = (ArrayList)session.getAttribute("sessionList");

        //상품 목록
        if(productList != null){
            for(String product : productList){
                out.println(product + "<br>");
            }
        }
    %>
</div>
```

쿠키(cookie)

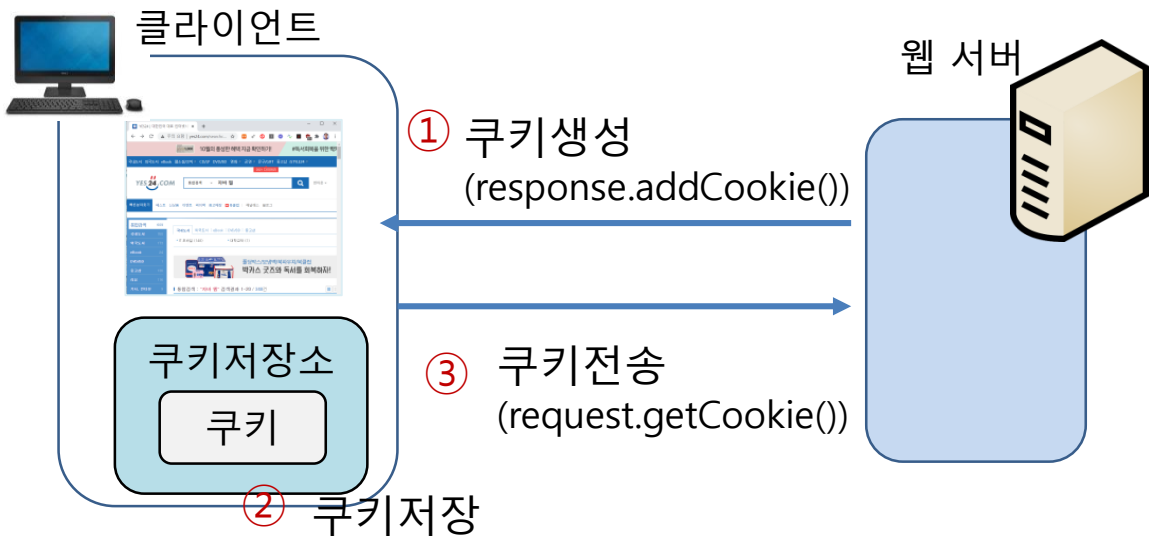
쿠키(cookie)란?

- 클라이언트와 웹 서버간의 상태를 지속적으로 유지하는 방법으로 쿠키와 세션이 있다.
- 쿠키는 서버가 클라이언트에 저장하는 정보로서 클라이언트 쪽에 필요한 정보를 저장해 놓고 필요할 때 추출하는 것을 지원하는 기술이다.
- 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)

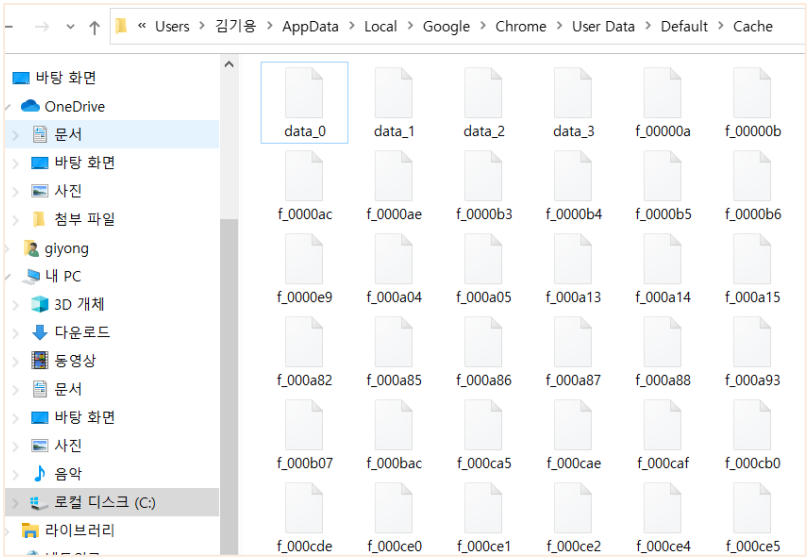
쿠키(cookie)의 작동 원리



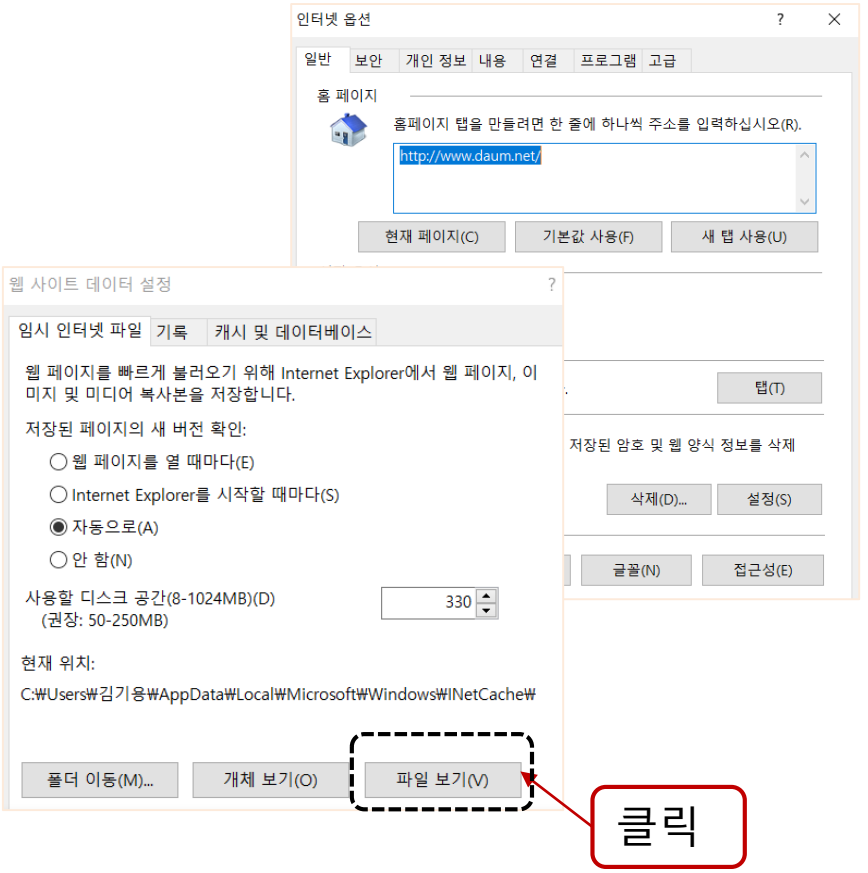
쿠키(cookie)

쿠키(cookie)

Chrome 폴더안의 쿠키 파일



Internet Exploer 쿠키 파일



쿠키(cookie)

쿠키(cookie) 내장 객체 메서드

메서드	설 명
getName()	쿠키의 이름을 반환한다.
getValue()	쿠키에 설정된 값을 반환한다.
setMaxAge(int)	쿠키의 유효 기간을 설정한다.

쿠키(cookie)

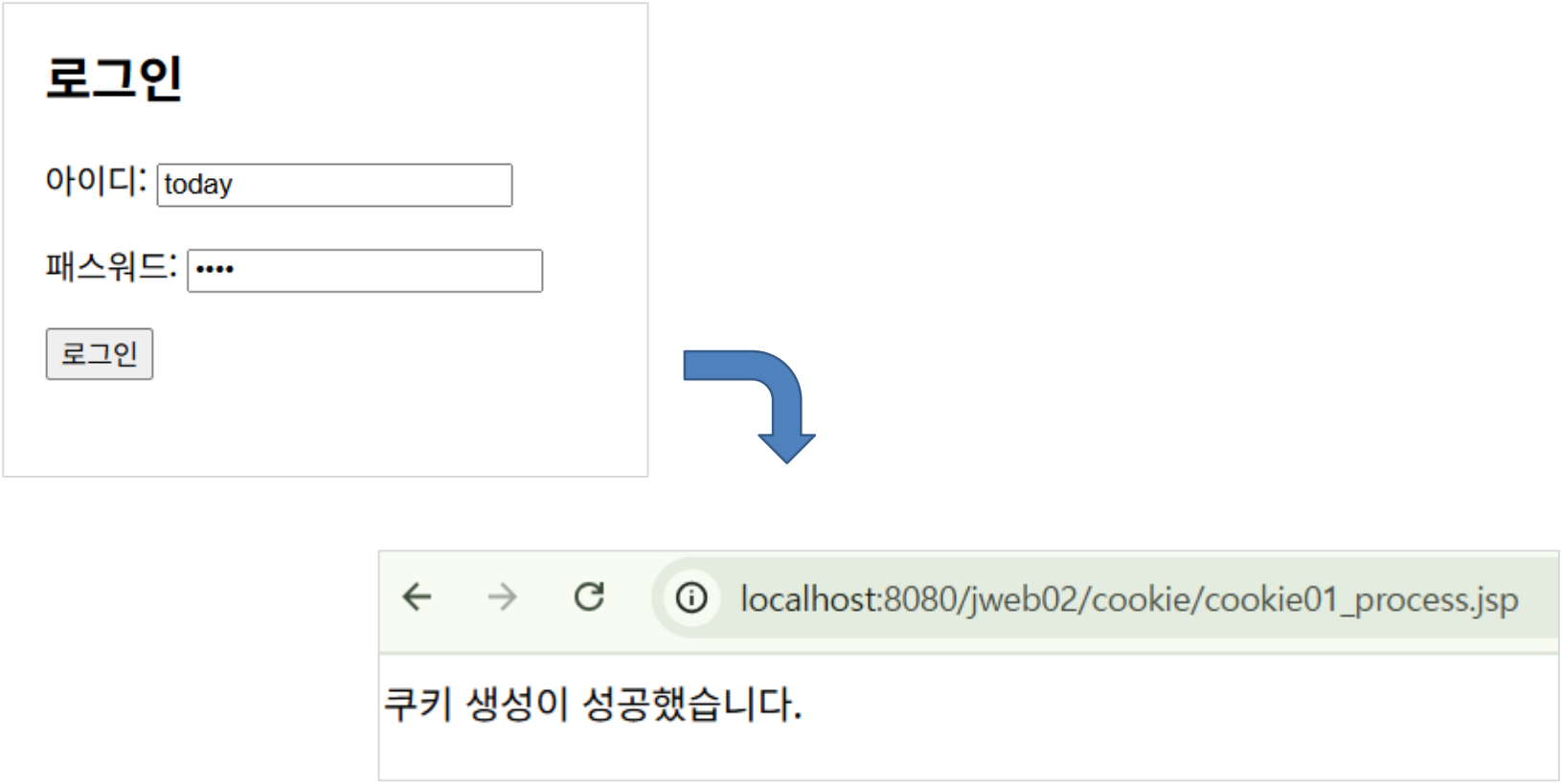
쿠키(cookie) 생성

- 쿠키 생성은 Cookie 클래스로 쿠키를 생성한 후 response 객체의 **addCookie()** 를 사용하여 로컬 컴퓨터(브라우저)에 보낸다.
- 쿠키 이름 – userId, 쿠키 값 - id

```
Cookie cookie = new Cookie("userId", id);
response.addCookie(cookie);
```

쿠키(cookie)

쿠키(cookie) 생성



쿠키(cookie)

쿠키(cookie) 생성

cookie/cookie01.jsp

```
<h2>로그인</h2>
<form action="cookie01_process.jsp" method="post">
    <p>아이디: <input type="text" name="id"></p>
    <p>패스워드: <input type="password" name="passwd"></p>
    <p><input type="submit" value="로그인"></p>
</form>
```

쿠키(cookie)

쿠키(cookie) 생성

cookie/cookieProcess.jsp

```jsp
<%
    String id = request.getParameter("id");
    String pwd = request.getParameter("pwd");

    if(id.equals("today") && pwd.equals("1234")){
        //Cookie(쿠키이름, 쿠키값)
        Cookie cookieId = new Cookie("userId", id);
        Cookie cookiePw = new Cookie("userPw", pwd);

        //브라우저(로컬컴퓨터)로 보내줌
        response.addCookie(cookieId);
        response.addCookie(cookiePw);

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

브라우저 > 개발자도구 > Network 탭

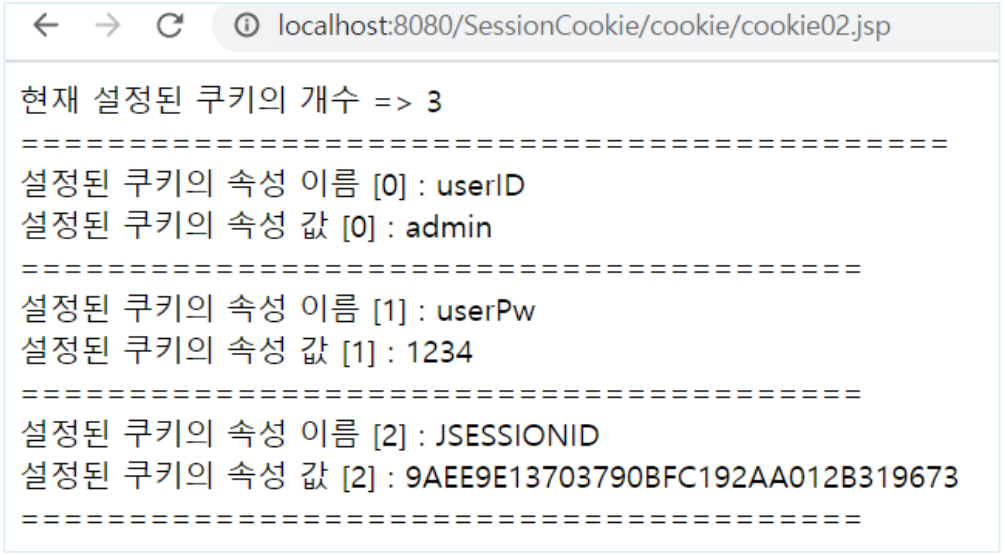
Cookie	userId=today; userPw=1234; JSESSIONID=94F40EB278B5D8DB11B63C263E6C2F88; ajs_anonymous_id=c746b7fd-692b-4c90-b314-c027dc415b91

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/cookie02.jsp

```
<%
    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>");
    }
%>
```

쿠키(cookie)

쿠키(cookie) 유효시간 설정 및 삭제

- 쿠키는 브라우저를 닫으면 유효기간이 종료됨

cookie.setMaxAge(0);

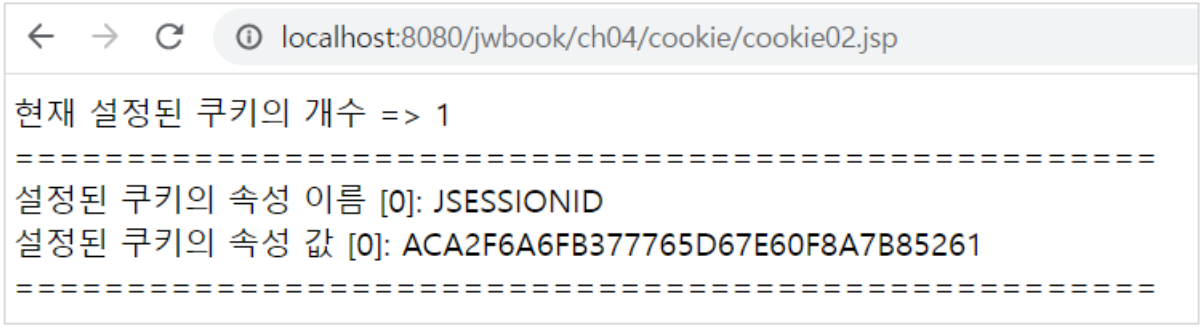
cookie/expiry.jsp

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

    for(int i=0; i<cookies.length; i++){
        //쿠키 유효 시간 설정(2분)
        cookies[i].setMaxAge(2*60);

        //쿠키 삭제
        //cookies[i].setMaxAge(0);
        response.addCookie(cookies[i]);
    }

    response.sendRedirect("cookie02.jsp");
%>
```

쿠키(cookie)

쿠키(cookie) 삭제

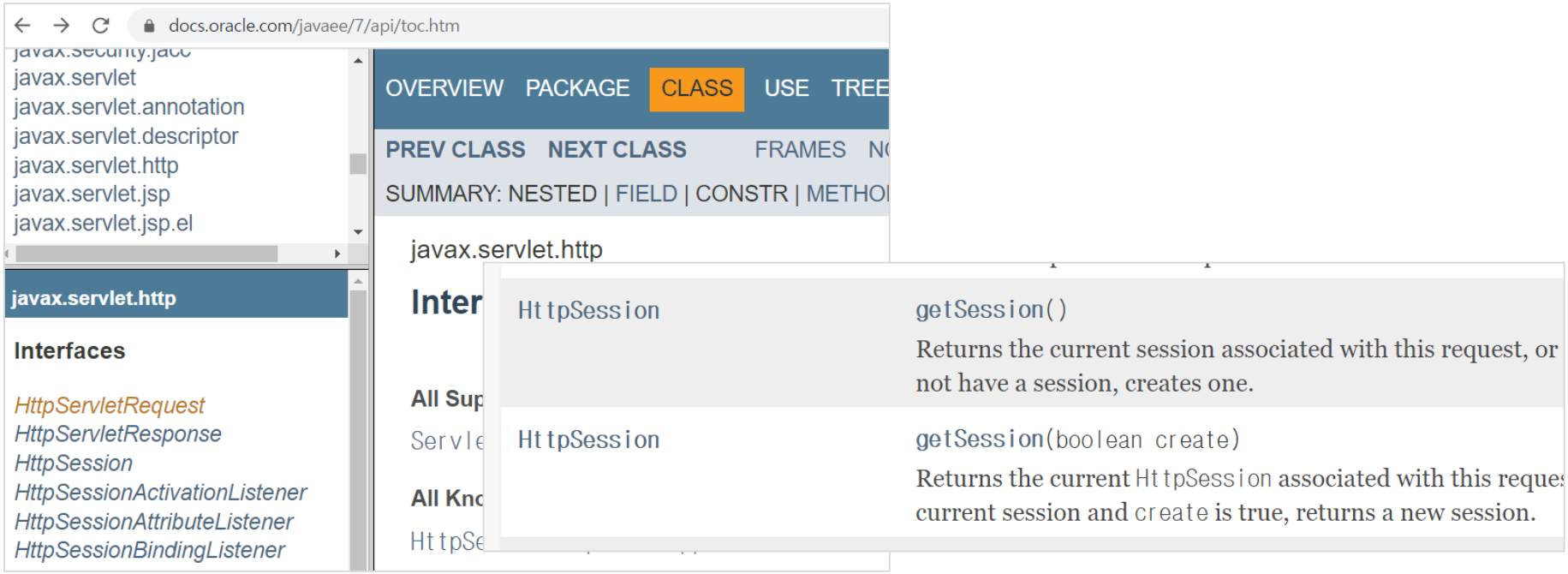
- 쿠키의 유효 기간을 만료한다.
- 2분후에 새로 고침(설정 2분인 경우)



세션(session)

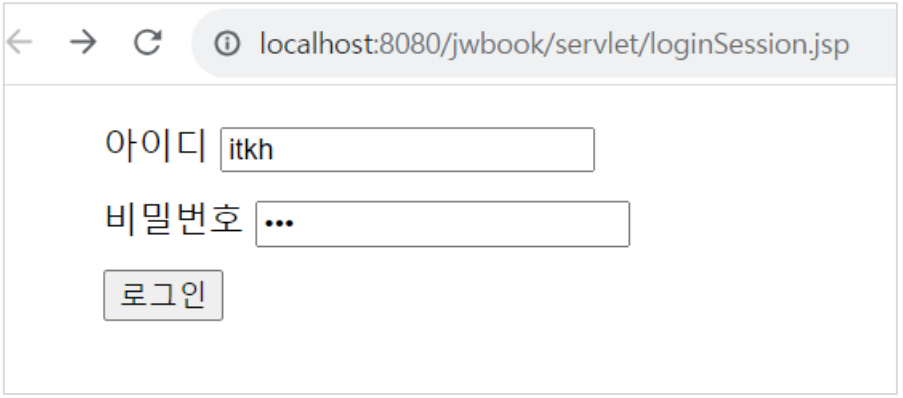
서블릿에서 세션 사용하기

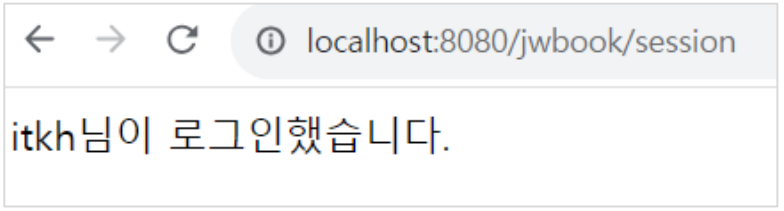
- HttpSession – Java EE API로 검색
- 서블릿에서 세션을 이용하려면 HttpSession 인터페이스의 객체를 생성해야 한다.
  HttpSession객체는 HttpServletRequest의 getSession() 메서드를 호출해서 생성한다.



세션(session)

서블릿에서 세션 사용 실습





세션(session)

서블릿에서 세션 사용 실습

```
<form action="/jwbook/session" method="post">
    <ul>
        <li>
            <label for="uid">아이디</label>
            <input type="text" name="uid" id="uid">
        </li>
        <li>
            <label for="passwd">비밀번호</label>
            <input type="password" name="passwd" id="passwd">
        </li>
        <li>
            <input type="submit" value="로그인">
        </li>
    </ul>
</form>
```

세션(session)

서블릿에서 세션 사용 실습

```
@WebServlet("/session")
public class LoginSession extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse
        //한글 인코딩(폼에서 받을때)
        request.setCharacterEncoding("utf-8");

        //브라우저로 응답(내보내기) - 한글 인코딩
        response.setContentType("text/html; charset=utf-8");

        //폼에 입력된 데이터 받기
        String id = request.getParameter("uid");
        String pw = request.getParameter("passwd");

        PrintWriter out = response.getWriter();
```

세션(session)

서블릿에서 세션 사용하기

```
        //세션 클래스
        HttpSession session = request.getSession();

        if(id.equals("itkh") && pw.equals("b19")) {
            session.setAttribute("userId", id);
            out.append(id + "님이 로그인했습니다.");
        }else {
            out.println("<script>");
            out.println("alert('아이디나 비밀번호가 틀립니다.')");
            out.println("history.go(-1)");
            out.println("</script>");
        }
}
```