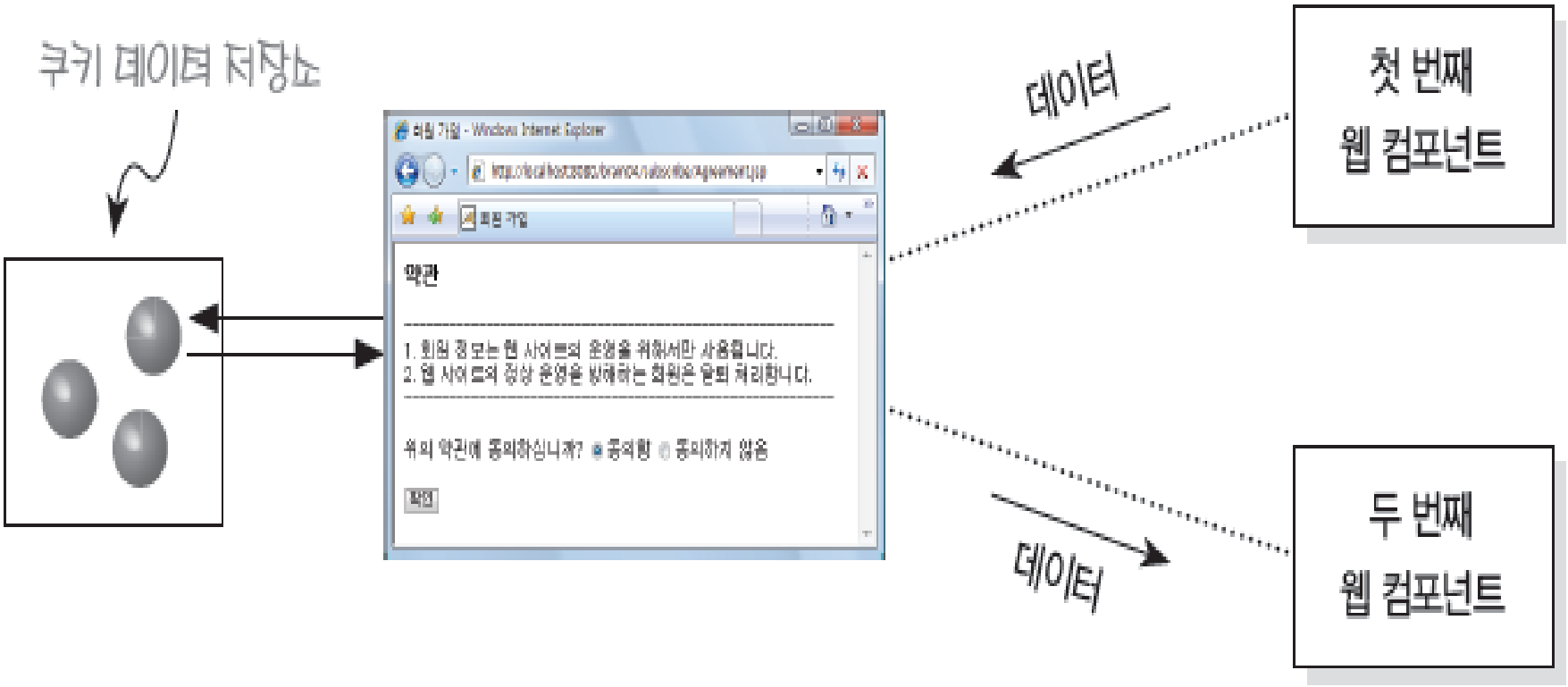
11장. 쿠키 와 세션

강사 강태광

쿠키와 세션 개념 I

1.쿠키 기술은 웹 서버가 웹 브라우저로 데이터를 보냈다가 웹 서버 쪽으로 다시 되돌려 받는 방법을 사용한다.
2. 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**

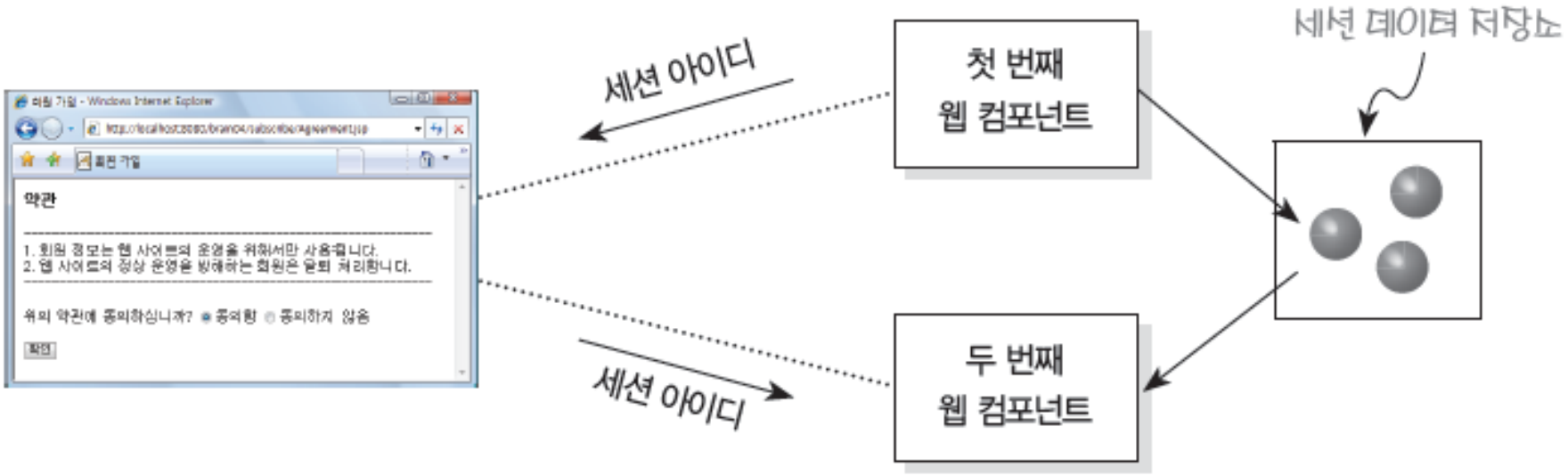
쿠키 기술을 이용한 웹 컴포넌트 간의 데이터 전달



쿠키와 세션 개념 2

1.세션 기술은 웹 브라우저를 거치지 않고 웹 서버에 있는 데이터 영역을 통해 데이터를 전달하는 방법.
2.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.

세션기술을 이용한 웹 컴포넌트 간의 데이터 전달



쿠키

1.웹 서버가 웹 브라우저와 정보를 주고받는 방법 중의 하나
2.쿠키는 웹 브라우저가 보관하고 있는 데이터로 웹 서버에 요청을 보낼 때 함께 전송
3.웹 브라우저가 전송한 쿠키를 사용하여 필요한 데이터를 읽을 수 있습니다.
4.쿠키의 동작 방식
① 쿠키 생성 단계 : 웹 서버 측에서 생성
② 쿠키 저장 단계 : 웹 브라우저는 응답 데이터에 포함된 쿠키를 쿠키 저장소에 보관하는데 종류에 따라 메모리나 파일로 저장
③ 쿠키 전송 단계 : 서버 측에서 요청할 때 마다 웹 서버에 전송
5.쿠키의 구성
①이름 : 쿠키를 구별하는데 사용하는 이름
②값 : 쿠키의 이름과 관련된 값
③유효 시간 : 쿠키의 유지 시간
④도메인 : 쿠키를 전송할 도메인
⑤경로 : 쿠키를 전송할 요청 경로

쿠키 생성 및 읽기

1. 쿠키 생성 및 쿠키 추가
   ① new Cookie("쿠키 이름", "쿠키의 값");
   ② response.addCookie(쿠키 객체)
2.Cookie 클래스가 제공하는 메서드
   ① String getName()
   ② String getValue()
   ③ void setValue(String value)
   ④ void setDomain(String pattern): 쿠키가 전송될 도메인을 설정
   ⑤ String getDomain()
   ⑥ void setPath(String url): 쿠키가 전송할 경로를 지정
   ⑦ String getPath()
   ⑧ void setMaxAge(int expiry): 유효 시간을 초 단위로 설정
   (음수이면 웹 브라우저가 닫힐 때 쿠기가 삭제)
   ⑨ int getMaxAge()
3.쿠키 가져오기
   ① Cookie[] request.getCookies()

쿠키 기술의 사용 방법

1. 새로운 쿠키 데이터를 저장하는 방법 – 입력 기능
   1) 쿠키 데이터를 웹 브라우저 쪽에 저장하기 위해 해야 하는 두 가지 일
      ① 첫째 : Cookie 클래스의 객체를 만든다.
      ② 둘째 : addCookie 메서드를 호출한다.
2. 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

3. 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.
4. 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 내용 확인"
	onclick="javascript:window.location='useCookie.jsp'">
	</P>
</body>
</html>
```

쿠키 데이터를 읽는 방법 – 조회 기능

1.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.

2.쿠키를 받는 일은 getCookies 라는 메서드를 이용해서 해야 한다.

3. getCookies 메서드는 JSP 페이지에서는 request 내장 변수에 대해, 서블릿 클래스에서는 doGet, doPost 메서드 첫 번째 파라미터에 대해 호출 해야 한다.

4. getCookies 메서드는 웹 브라우저가 보낸 모든 쿠키를 Cookie 배열로 만들어서 리턴 하기 때문에 다음과 같은 Cookie 배열 변수에 리턴값을 받아야 한다.

```
// 웹 브라우저가 보낸 모든 쿠키를 Cookie 배열로 만들어서 리턴하는 메서드
Cookie cookies[] = request.getCookies();
```

5. 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 getName이라는 메서드를 호출해 구할 수 있다.

```
// 쿠키 이름을 가져오는 메서드
String name = cookie.getName();
```

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

```
// 쿠키 값을 가져오는 메서드
String value = cookie.getValue();
```

쿠키 데이터를 읽는 방법 – 조회 예시

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

1. 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

1. 쿠키의 값을 변경하기 위해서는 같은 이름의 쿠키를 새로 생성해서 응답 데이터로 전송 하면 됩니다.
2. 쿠키를 삭제하고자 하는 경우는 직접 삭제하는 것이 아니고 setMaxAge() 메서드를 호출 할 때 인자 값으로 0을 대입해서 쿠키의 수명을 0으로 만드는 방법을 이용합니다
3. 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)

cookie.setMaxAge(0); // 쿠키를 바로 삭제하도록 만드는 값

cookie.setMaxAge(-1); // 웹 브라우저가 끝날 때 쿠키가 삭제되도록 만드는 값

쿠키의 도메인

1. 도메인
   ①쿠키는 그 쿠키를 생성한 서버에만 전송이 됩니다.
   ②쿠키를 다른 도메인으로 전송하고자 하면 setDomain()를 이용하면 됩니다.
   ③ . 도메인: 모든 도메인에 쿠키를 전송
   ④www.도메인: 특정 도메인에만 전송
2.경로
   ① setPath(경로)를 이용해서 특정 경로의 파일 들에게만 쿠키를 전송하는데 웹 애플리케이션 디렉토리를 기준으로 한 URL 경로명을 넘겨주어야 합니다.
   ④ /로 시작해야 하고 /로 끝내는 것이 좋습니다.

```
<%@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;
        }
        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>
```

```
<%@ 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>
```

webStorage

1. 클라이언트의 디스크에 소량의 데이터를 저장하기 위한 스토리지
2. 이전에는 일반적으로 Cookie를 사용
3. Cookie 와 다른 점
   ① 크기에 제한이 없습니다.
   ② 서버로 보내지지 않습니다.
   ③ 자바 스크립트 객체의 복사본을 저장할 수 있습니다.
4. 종류는 로컬 스토리지와 세션 스토리지로 나눕니다.
5. 사용 방법은 동일하며 유효범위가 다름
6. 메서드
   ① length 속성: 스토리지에 저장된 데이터의 수를 반환
   ② key(index): 인덱스의 키를 반환하고, 없으면 null을 반환
   ③ getItem(key): 키에 대응되는 데이터를 반환
   ④ setItem(key, data): key로 스토리지에 데이터를 저장
   ⑤ removeItem(key): key에 대응되는 데이터를 삭제
   ⑥ clear(): 스토리지의 모든 데이터 삭제

localStorage

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

4. 저장
   ① localstorage.키이름 = {속성:값, 속성:값....};
   ② localstorage["키이름"] = {속성:값, 속성:값...};
   ③ localstorage.setItem("키이름", {속성:값, 속성:값...});

5. 읽기
   ① var 변수명 = localstorage.키이름;
   ② var 변수명 = localstorage["키이름"];
   ③ var 변수명 = localstorage.getItem["키이름"];

6. 삭제
   ① delete.localstorage.키이름;
   ② delete localstorage["키이름"];
   ③ localstorage.removeItem("키이름");

로그인 – login.html1

```
<!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;
        }
        //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");
       }
};
```

로그인 – login.html2

```
// 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>
		    아 이 디:<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>메인 페이지</h1>
		<%
			String id = request.getParameter("id");
			out.println(id + "님 환영합니다. <br/>");
		%>
		<a href="login.html">로그인 페이지로</a>
	</body>
</html>
```

세션

1. 세션은 웹 컨테이너에 정보를 저장할 수 있는 객체
2. 웹 컨테이너는 하나의 웹 브라우저에 하나의 세션을 생성
3. 세션 생성
   1) 코드에서 생성: <%@ page session="true" %>
   2) request 객체로 부터 생성
      ① HttpSession 변수 = request.getSession()
      ② 위의 경우는 세션이 있으면 그 세션을 리턴하고 없으면 새로 생성해서 리턴
      ③ 만일 생성된 경우에만 리턴받고자 하는 경우는 getSession에 매개변수로 false를 전달
4.세션에서 속성 사용
   ① session.setAttribute("속성명", "값")
   ② session.getAttribute("속성명")
5.세션 객체의 메서드
   ① String getId()
   ② long getCreationTime()
   ③ long getLastAccessedTime()

서블릿 클래스에서 세션 기술을 사용하는 방법

1.서블릿 클래스에서 세션을 시작하기 위해서는 doGet, doPost 메서드의 HttpServletRequest 파라미터에 대해 getSession이라는 메서드를 호출해야 한다

2. getSession 메서드는 세션 정보를 포함하는 javax.servlet.http.HttpSession 타입의 객체를 리턴한다.

```
HttpSession session = request.getSession();                    // 세션을 시작하는 메서드
```

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

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

데이터이름 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

```
<%@ 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

1. 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.
2. session 내장 변수를 사용하면 세션 데이터 영역에 데이터를 저장할 수도 있고, 그 영역에 있는 데이터를 읽어오거나 삭제할 수도 있다.

```
session.setAttribute( "ID ", "lee77 ");                  //  세션 데이터를 저장하는 메서드
String str = (String) session.getAttribute( "ID ");  //  세션 데이터를 가져오는 메서드
session.removeAttribute( "ID "");                        // 세션 데이터를 삭제하는 메서드
```

3. 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 작성
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>
    // 세션 데이터를 저장
    <%
        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>
```

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);

%>
```

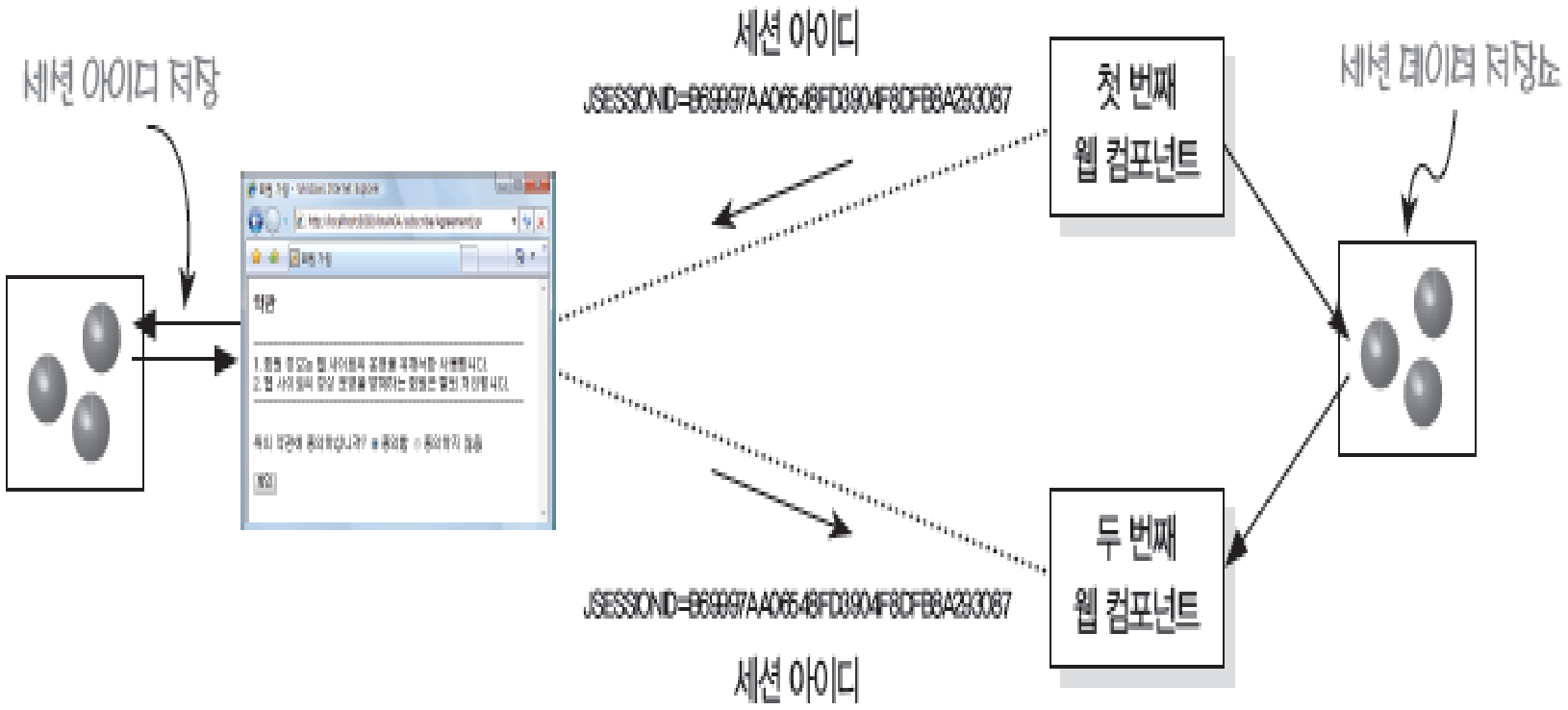
JSP 페이지에서 세션 기술을 사용하는 방법

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>
```

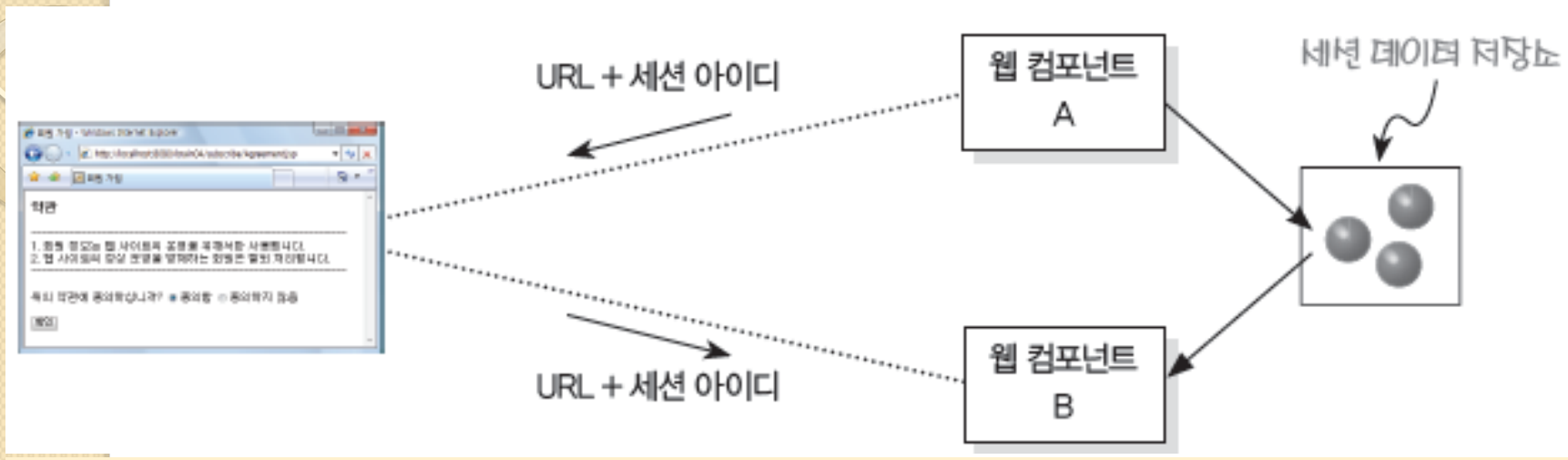
JSP 페이지에서 세션 기술을 사용하는 방법

세션 기술에서는 웹 브라우저로 세션 아이디를 보낼 때 쿠키 형태로 만들어서 전송하는데,
이 쿠키 이름은 **JSESSIONID**



URL 재작성 메커니즘의 사용 방법

쿠키를 사용할 수 없는 웹 환경에서는 **URL 뒤에 세션 아이디를 붙여서 전송하는 방법을 사용**



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

URL 재작성 메커니즘의 사용 방법

1. URL 재작성을 하기 위해서는 URL 재작성 기능을 제공하는 encodeURL이라는 메서드를 사용.
2. JSP 페이지에서는 response 내장 변수에 대해 이 메서드를 호출하면 되고, 서블릿 클래스에서는 doGet, doPost 메서드의 두 번째 파라미터에 대해 호출하면 된다.

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

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

// 상대적인 URL 경로명
String url = response.encodeURL("common/Greetings.jsp ");

4. encodeURL 메서드에는 슬래시(/)로 시작하는 URL 경로명을 넘겨줄 수도 있는데, 이런 값은 웹 서버 내에서의 URL 경로명을 해석된다.
// 웹 서버 내에서의 URL 경로명
String url = response.encodeURL("/subscribe/Result.jsp ");

URL 재작성 메커니즘의 사용 방법 예시

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>
```

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 저장**