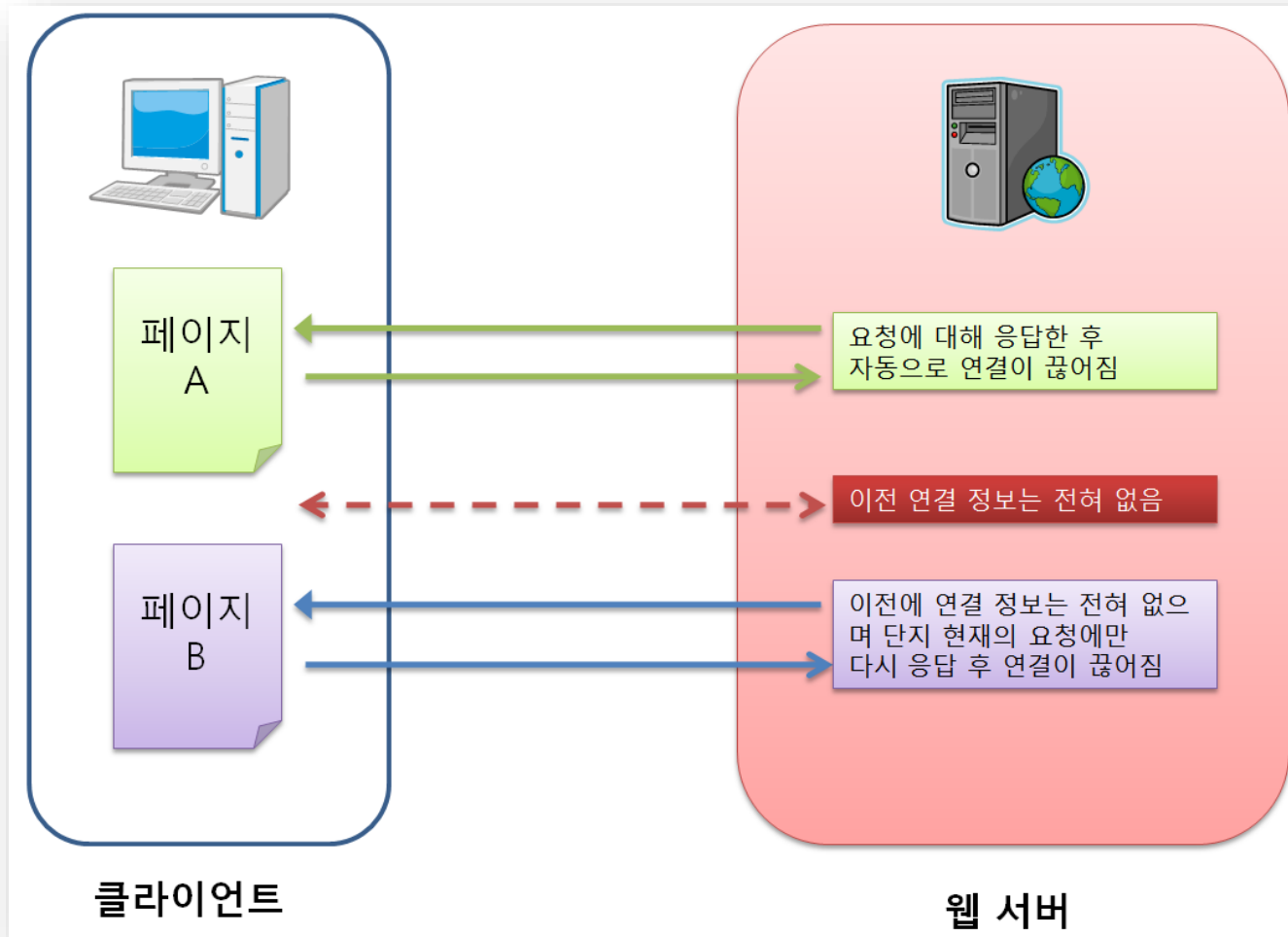


제 09 장 쿠키와 세션

❖ Connectionless





❖ 필요성

- HTTP의 비연결성을 보완
- 장바구니와 같이 여러 페이지로 이동하더라도 사용자 정보와 필요 정보 유지 필요

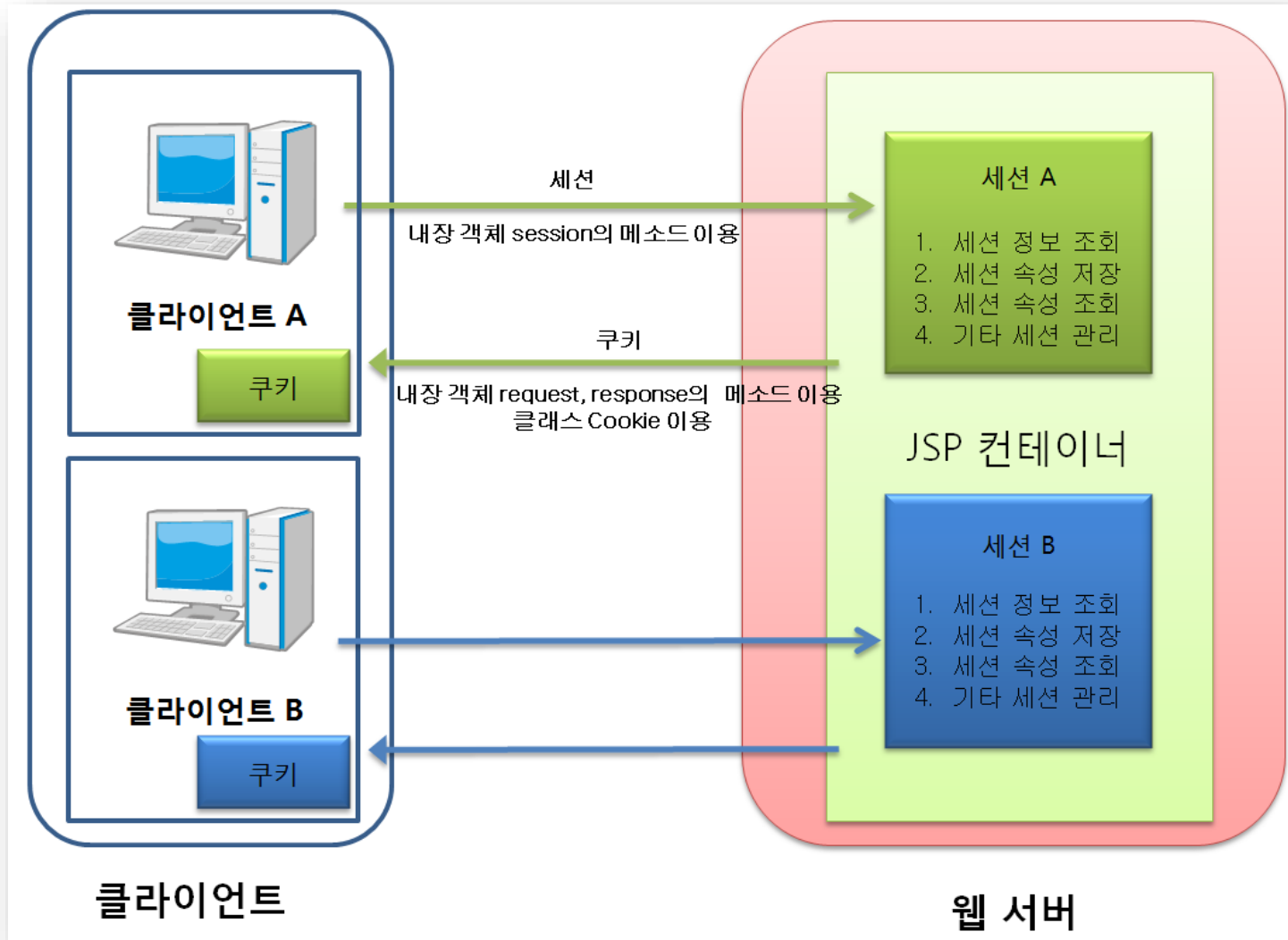
❖ 쿠키

- 사용자 컴퓨터(클라이언트)에 저장
 - 저장된 정보를 다른 사람 또는 시스템이 볼 수 있는 단점
- 유효 시간이 지나면 사라짐

❖ 세션

- 서버에 저장
- 서버가 종료되거나 유효 시간이 지나면 사라짐

쿠키와 세션의 비교





❖ 정의

- 쿠키는 서버에서 만들어진 작은 정보의 단위
- 클라이언트와 웹 서버 간의 상태를 지속적으로 유지하는 방법

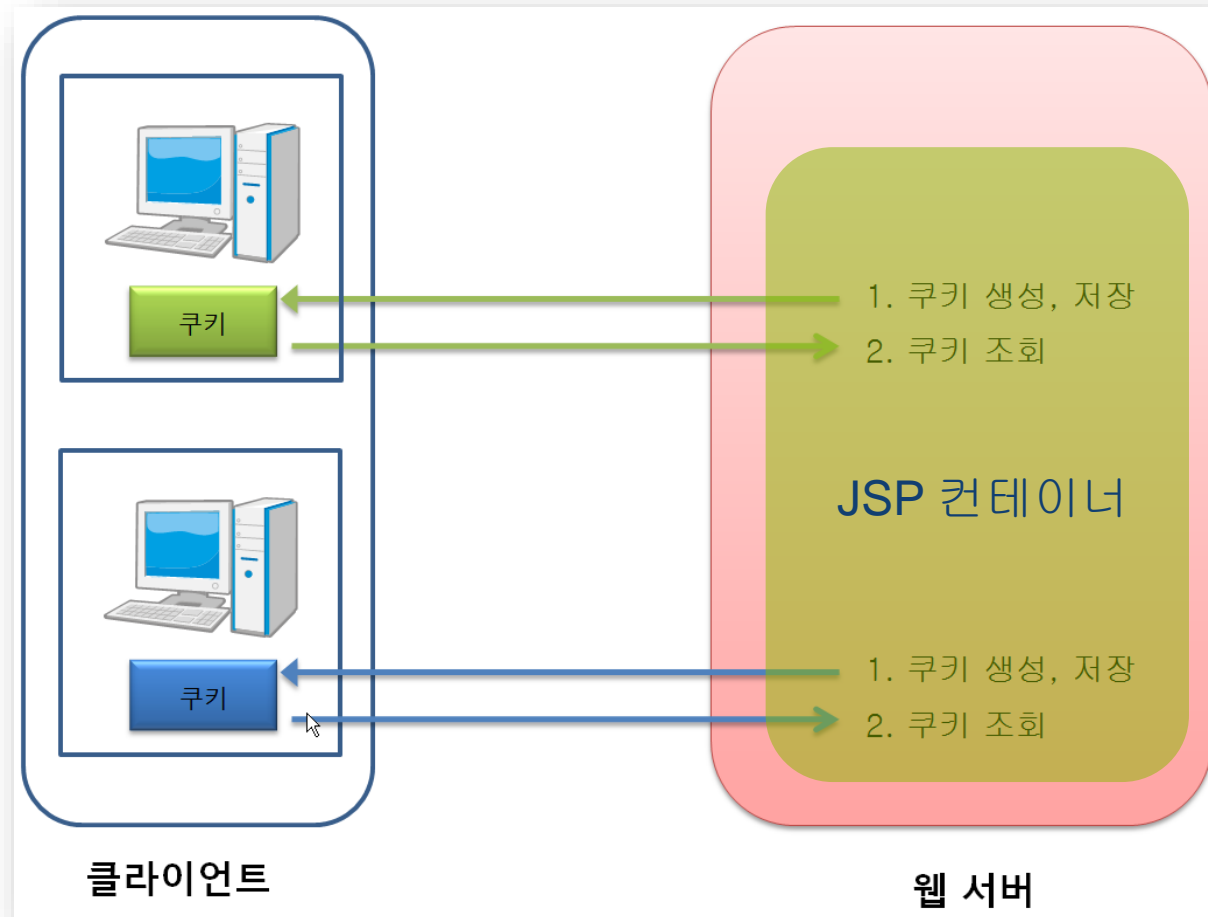
❖ 이용 방법

- 서버에서 클라이언트의 브라우저로 전송되어 클라이언트에 저장
: 웹사이트에 관한 정보와 개인정보가 기록되기 때문에 보안에 문제가 있음 (단점)
- 저장된 쿠키는 다시 해당하는 웹 페이지에 접속 할 때, 브라우저에서 서버로 쿠키를 전송
: 클라이언트의 일정 폴더에 정보를 저장하기 때문에 웹 서버의 부하를 줄일 수 있음(장점)
- 쿠키는 이름(name)과 값(value)으로 구성된 자료를 저장
 - 이름과 값 외에도 주석(comment), 경로(path), 유효기간(maxage, expiry), 버전(version), 도메인(domain)과 같은 추가적인 정보를 저장

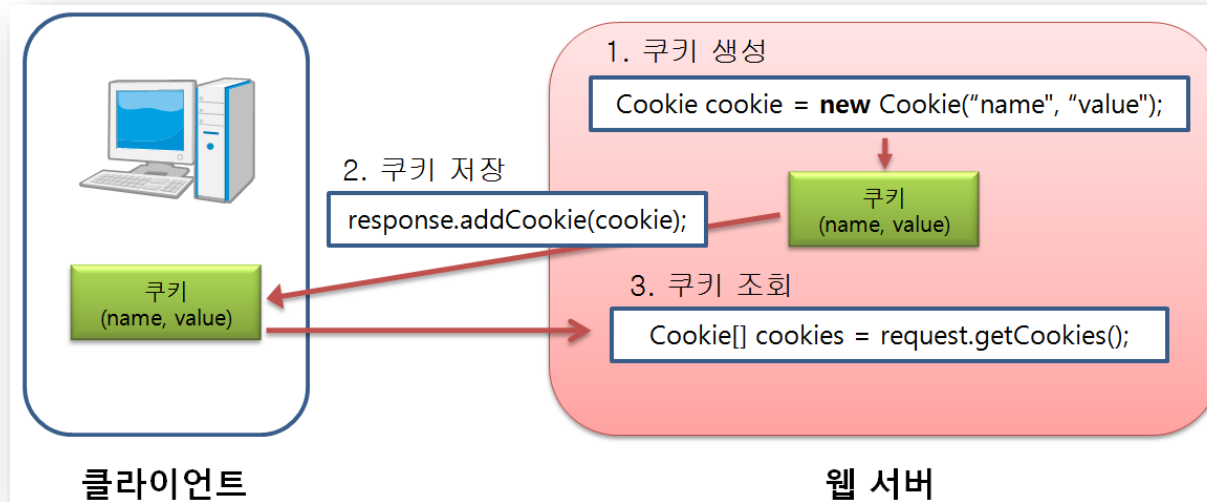
❖ 쿠키는 그 수와 크기에 제한

- 하나의 쿠키는 4K Byte 크기로 제한
- 브라우저는 각각의 웹사이트 당 20개의 쿠키를 허용
- 모든 웹 사이트를 합쳐 최대 300개를 허용
- 그러므로 클라이언트 당 쿠키의 최대 용량은 1.2M Byte

쿠키의 이용



❖ javax.servlet.http.Cookie 클래스



반환형	메소드 이름	기능
int	<code>getMaxAge()</code>	쿠키의 최대지속 시간을 초단위로 지정 -1 일 경우 브라우저가 종료되면 쿠키를 만료
String	<code>getName()</code>	쿠키의 이름을 스트링으로 반환
String	<code>getValue()</code>	쿠키의 값을 스트링으로 반환
void	<code>setMaxAge(int expiry)</code>	쿠키의 만료시간을 초단위로 설정
void	<code>setValue(String newValue)</code>	쿠키에 새로운 값을 설정할 때 사용



❖ javax.servlet.http.Cookie 클래스

메소드	반환 유형	설명
getComment()	String	쿠키에 대한 설명을 반환합니다.
getDomain()	String	쿠키의 유효한 도메인 정보를 반환합니다.
getMaxAge()	int	쿠키의 사용 가능 기간에 대한 정보를 반환합니다.
getName()	String	쿠키의 이름을 반환합니다.
getPath()	String	쿠키의 유효한 디렉터리 정보를 반환합니다.
getSecure()	boolean	쿠키의 보안 설정을 반환합니다.
getValue()	String	쿠키에 설정된 값을 반환합니다.
getVersion()	int	쿠키의 버전을 반환합니다.
setComment(String)	void	쿠키에 대한 설명을 설정합니다.
setDomain(String)	void	쿠키에 유효한 도메인을 설정합니다.
setMaxAge(int)	void	쿠키의 유효 기간을 설정합니다.
setPath(String)	void	쿠키의 유효한 디렉터리를 설정합니다.
setSecure(boolean)	void	쿠키의 보안을 설정합니다.
setValue(String)	void	쿠키의 값을 설정합니다.
setVersion(int)	void	쿠키의 버전을 설정합니다.



❖ 쿠키 생성

- Cookie()메소드 사용
- 쿠키는 (이름, 값)의 쌍으로 정보를 입력하여 생성
- 쿠키의 이름은 알파벳과 숫자로만 구성되고, 쿠키 값은 공백, 괄호, 등호, 콤마, 콜론, 세미콜론 등은 포함 불가능
 - Cookie Cookie(String name, String value)
 - **name** : 쿠키를 식별하기 위한 이름
 - **value** : 쿠키 값

❖ 클라이언트의 컴퓨터에 파일 형태로 저장

- 내장객체 response의 addCookie 메소드를 이용
 - response.addCookie(cookie);

ex) Cookie cookie = **new** Cookie("user", "kang");
response.addCookie(cookie);

쿠키 생성 예제 [실습1]



```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Cookie</title>
5 </head>
6 <body>
7 <form action="cookie01_process.jsp" method="POST">
8 <p> 아이디 : <input type="text" name="id">
9 <p> 비밀번호 : <input type="text" name="passwd">
10 <p> <input type="submit" value="전송">
11 </form>
12 </body>
13 </html>
```

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Cookie</title>
5 </head>
6 <body>
7 <%
8 String user_id = request.getParameter("id");
9 String user_pw = request.getParameter("passwd");
10
11 if (user_id.equals("admin") && user_pw.equals("1234")) {
12 Cookie cookie_id = new Cookie("userID", user_id);
13 Cookie cookie_pw = new Cookie("userPW", user_pw);
14 response.addCookie(cookie_id);
15 response.addCookie(cookie_pw);
16 out.println("쿠키 생성이 성공했습니다<br>");
17 out.println(user_id + "님 환영합니다");
18 } else {
19 out.println("쿠키 생성이 실패했습니다");
20 }
21 %>
22 </body>
23 </html>
```

← → ⏏ 🔄 http://localhost:8080/eu

아이디 :

비밀번호 :

전송

← → ⏏ 🔄 http://localhost:8080/eu

쿠키 생성이 성공했습니다
admin님 환영합니다

← → ⏏ 🔄 http://localhost:8080/eu

아이디 :

비밀번호 :

전송

← → ⏏ 🔄 http://localhost:8080/eu

쿠키 생성이 실패했습니다



❖ 클라이언트에 저장된 쿠키를 조회

- 내장객체 request의 get_cookies() 메소드를 이용
 - `Cookie[] request.getCookies()`
- 쿠키 객체가 여러 개일 때는 배열 형태로 가져옴
- 메소드 get_cookies()의 반환 값은 저장된 모든 쿠키의 배열
 - 쿠키가 없으면 `null` 값이 반환

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

❖ 각각의 쿠키를 얻는 방법

- 쿠키 배열 변수 cookies는 다음과 같이 for each 문 이용
 - `Cookie`의 `String getName()`, `String getValue()` 메소드를 이용

```
for (Cookie c : cookies) {  
    out.println("쿠키 이름(name) : " + c.getName() + ", " );  
    out.println("쿠키 값(value) : " + c.getValue() + "<br>" );  
}
```



```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Cookie</title>
5 </head>
6 <body>
7 <%
8     Cookie[] cookies = request.getCookies();
9     out.println("현재 설정된 쿠키의 개수 => " + cookies.length + "<br>");
10    out.println("=====<br>");
11    for (int i = 0; i < cookies.length; i++) {
12        out.println("설정된 쿠키의 속성 이름 [ " + i + " ] : " + cookies[i].getName() + "<br>");
13        out.println("설정된 쿠키의 속성 값 [ " + i + " ] : " + cookies[i].getValue() + "<br>");
14        out.println("-----<br>");
15    }
16    %>
17 </body>
18 </html>
```

현재 설정된 쿠키의 개수 => 3

=====

설정된 쿠키의 속성 이름 [0] : userID

설정된 쿠키의 속성 값 [0] : admin

설정된 쿠키의 속성 이름 [1] : userPW

설정된 쿠키의 속성 값 [1] : 1234

설정된 쿠키의 속성 이름 [2] : JSESSIONID

설정된 쿠키의 속성 값 [2] : 8CD894B7A053C91987D07DEC9466F794



❖ 쿠키의 유효기간 설정

- 메소드 `setMaxAge()`
 - 인자는 유효기간을 나타내는 초 단위의 정수형
 - **`void setMaxAge(int age)`**
 - 만일 유효기간을 0으로 지정하면 쿠키의 삭제
 - 음수를 지정하면 브라우저가 종료될 때 쿠키가 삭제
- 유효기간을 2분으로 지정하려면
 - `cookie.setMaxAge(2 * 60);` //초 단위 : 2분
 - 1주일로 지정하려면 `(7*24*60*60)`로

ex) `Cookie cookie = new Cookie("id", "admin");`
`cookie.setMaxAge(0);`
`response.addCookie(cookie);`



```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Cookie</title>
5 </head>
6 <body>
7 <%
8     Cookie[] cookies = request.getCookies();
9
10    for (int i = 0; i < cookies.length; i++) {
11        cookies[i].setMaxAge(0);
12        response.addCookie(cookies[i]);
13    }
14    response.sendRedirect("cookie02.jsp"); //쿠키조회파일
15 %>
16 </body>
17 </html>
```

현재 설정된 쿠키의 개수 => 1

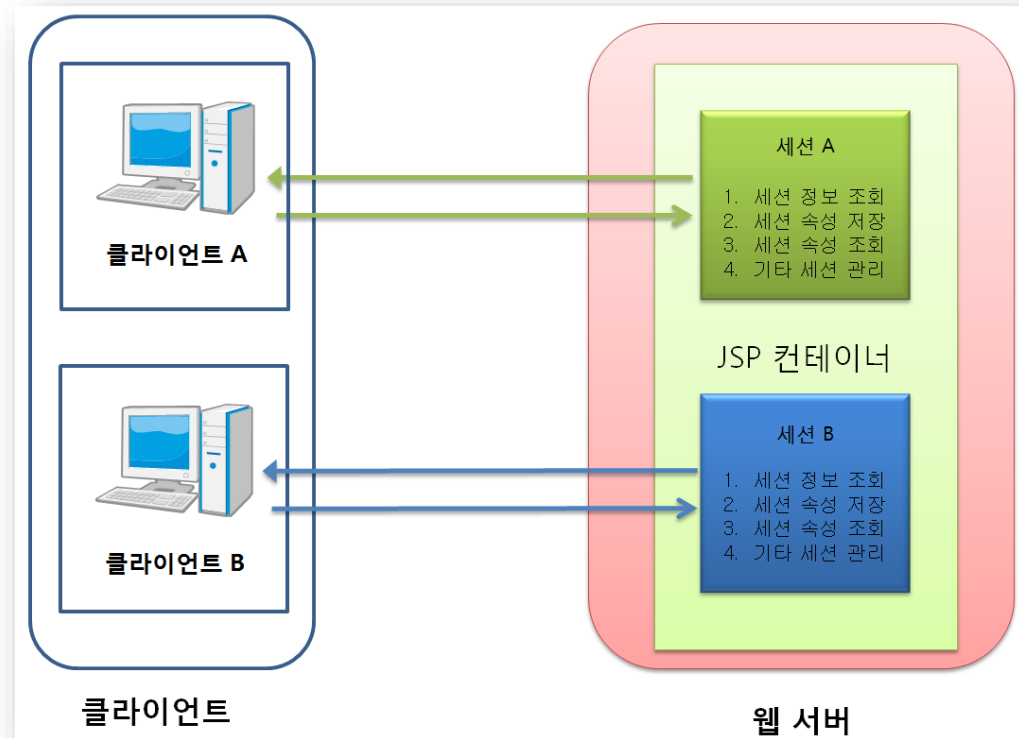
=====

설정된 쿠키의 속성 이름 [0] : JSESSIONID

설정된 쿠키의 속성 값 [0] : 8CD894B7A053C91987D07DEC9466F794

❖ 개념

- 클라이언트의 정보를 클라이언트 PC에 저장하는 것이 **쿠키**
- 클라이언트마다 각기 다른 정보를 서버에 저장하는 것이 **세션**
 - 즉 세션은 클라이언트 사용자 별로 여러 페이지 이동을 인식
 - 필요한 정보를 서버에 저장, 조회 방법과 세션을 관리할 수 있는 방법을 제공
- 웹 서버에서만 접근이 가능하므로 보안 유지에 유리하며 데이터를 저장하는데 한계가 없음



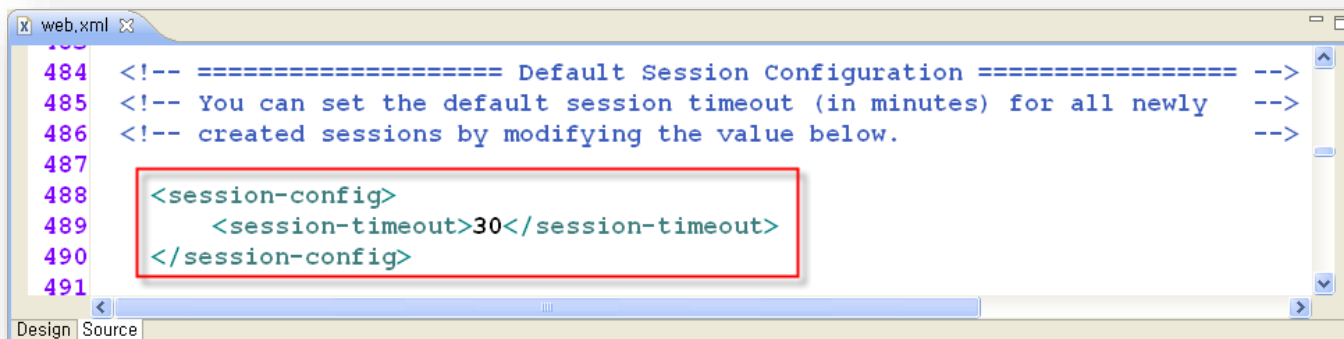


❖ 패키지 javax.servlet.http

■ 인터페이스 HttpSession

메소드	반환 유형	설명
getAttribute(String name)	java.lang.Object	세션 속성 이름이 name인 속성 값을 Object 형으로 반환합니다. 해당되는 속성 이름이 없을 때는 null을 반환합니다. 반환 값이 Object 형이므로 반드시 형 변환을 하여 사용해야 합니다.
getAttributeNames()	java.util.Enumeration	세션 속성 이름을 Enumeration 객체 타입으로 반환합니다.
getCreationTime()	long	세션이 생성된 시간을 반환합니다. 1970년 1월 1일 0시 0초부터 현재 세션이 생성된 시간까지 경과한 시간을 1/1,000초 값으로 반환합니다.
getId()	java.lang.String	세션에 할당된 고유 아이디를 String 형으로 반환합니다.
getLastAccessedTime()	long	해당 세션에 클라이언트가 마지막으로 request를 보낸 시간을 반환합니다.
getMaxInactiveInterval(int interval)	int	해당 세션을 유지하기 위해 세션 유지 시간을 반환합니다. 기본 값은 1,800 초(30분)입니다.
isNew()	boolean	해당 세션의 생성 여부를 반환합니다. 처음 생성된 세션이면 true를 반환하고 이전에 생성된 세션이면 false를 반환합니다.
removeAttribute(String name)	void	세션 속성 이름이 name인 속성을 제거합니다.
setAttribute(String name, Object value)	void	세션 속성 이름이 name인 속성에 value를 할당합니다.
setMaxInactiveInterval(int interval)	void	해당 세션을 유지하기 위한 세션 유지 시간을 초 단위로 설정합니다.
Invalidate()		현재 세션에 저장된 모든 세션 속성을 제거합니다.

- ❖ 세션은 클라이언트가 서버에 접속하는 순간 생성
 - 특별히 지정하지 않으면 세션의 유지 시간은 기본 값으로 30분 설정
 - 세션의 유지 시간이란 서버에 접속한 후 서버에 요청을 하지 않는 최대 시간
 - 30분 이상 서버에 전혀 반응을 보이지 않으면 세션이 자동으로 끊어짐.
 - 이 세션 유지 시간은 서버에서 설정 가능
 - 톰캣에서 설치 폴더 하부 [conf] 폴더에 파일 web.xml
 - session timeout으로 30초가 지정되어 있는 것을 확인



The screenshot shows a text editor window titled 'web.xml'. The code is as follows:

```
484 <!-- ===== Default Session Configuration ===== -->
485 <!-- You can set the default session timeout (in minutes) for all newly -->
486 <!-- created sessions by modifying the value below. -->
487
488 <session-config>
489     <session-timeout>30</session-timeout>
490 </session-config>
491
```

A red rectangle highlights the `<session-timeout>30</session-timeout>` line. The editor has a 'Design' and 'Source' tab at the bottom, with 'Source' selected.



❖ 내장객체 session 메소드 `getCreationTime()`

- 현재 세션이 생성된 시간을 1970년 1월 1일 0시를 기준으로 지난 시간을 계산하여 밀리세컨드로 반환
- 그러므로 이를 년, 월, 일 시의 시간정보로 출력하려면 클래스 `java.util.Date`의 생성자 `Date(long mseconds)`를 이용하여 객체를 만든 후 출력
- `long mseconds = session.getCreationTime();`
- `Date time = new Date(mseconds);`

주요 세션 정보 조회 [실습4]



```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR" pageEncoding="EUC-KR"%>
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
5 <title>JSP 예제 : session.jsp</title>
6 </head>
7 <body>
8     <%@ page import="java.util.Enumeration, java.util.Date" %>
9     <h1>세션 예제</h1>
10    <hr><h2>세션 주요 정보 조회</h2>
11    세션 ID (유일한 식별자) : <%= session.getId() %><br>
12    세션 MaxInactiveInterval (기본 세션 유지 시간) : <%= session.getMaxInactiveInterval() %><br>
13
14    <%
15        long mseconds = session.getCreationTime();
16        Date time = new Date(mseconds);
17    %>
18    세션 CreationTime (1970년 1월 1일 0시 이후의 지난 밀리세컨드): <%=mseconds %><br>
19    세션 CreationTime (시각으로 다시 계산) : <%=time.toLocaleString() %>
20 </body>
21 </html>
```

세션 예제

세션 주요 정보 조회

세션 ID (유일한 식별자) : 8CD894B7A053C91987D07DEC9466F794
세션 MaxInactiveInterval (기본 세션 유지 시간) : 1800
세션 CreationTime (1970년 1월 1일 0시 이후의 지난 밀리세컨드): 1619600673895
세션 CreationTime (시각으로 다시 계산) : 2021. 4. 28 오후 6:04:33



❖ 내장객체 session 메소드 setAttribute(name, value)

- void setAttribute(String name, Object value)
- name과 value의 쌍으로 객체 Object를 저장하는 메소드
 - name : 세션으로 사용한 세션 속성 이름, 혹은 키
 - value : 세션 속성 값, Object타입만 가능하기때문에 int, double, char등의 기본 타입은 사용할 수 없음
- 세션이 유지되는 동안 저장할 자료를 저장

```
ex) session.setAttribute("id", "javajsp");  
    session.setAttribute("pwd", "jdktomcat");
```



```

1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Session</title>
5 </head>
6 <body>
7     <form action="session01_process.jsp" method="POST">
8         <p>아이디 : <input type="text" name="id">
9         <p>비밀번호 : <input type="text" name="passwd">
10        <p><input type="submit" value="전송">
11    </form>
12 </body>
13 </html>

```

```

1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Session</title>
5 </head>
6 <body>
7     <%
8         String user_id = request.getParameter("id");
9         String user_pw = request.getParameter("passwd");
10
11         if (user_id.equals("admin") && user_pw.equals("1234")) {
12             session.setAttribute("userID", user_id);
13             session.setAttribute("userPW", user_pw);
14             out.println("세션 설정이 성공했습니다<br>");
15             out.println(user_id+"님 환영합니다");
16         } else {
17             out.println("세션 설정이 실패했습니다");
18         }
19     %>
20 </body>
21 </html>

```

아이디 :

비밀번호 :

전송

http://localhost:8

세션 설정이 성공했습니다
admin님 환영합니다

아이디 :

비밀번호 :

전송

http://localhost:8080

세션 설정이 실패했습니다



❖ 단일 세션 조회

- `getAttribute(String name)` 메소드
- 세션에 저장된 자료는 다시 `getAttribute(String name)` 메소드를 이용해 조회
 - **name** : 세션에 저장된 세션 속성 이름, 해당 속성 이름이 없을 경우는 **null**을 반환
- 반환 값은 **Object** 유형이므로 저장된 객체로 자료 유형 변환이 필요
- 메소드 `setAttribute()`에 이용한 **name**인 “id”를 알고 있다면 바로 다음과 같이 바로 조회
ex) `String value = (String) session.getAttribute("id");`



❖ 세션에 저장된 속성값 가져와 출력하기

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Session</title>
5 </head>
6 <body>
7 <%
8     String user_id = (String) session.getAttribute("userID");
9     String user_pw = (String) session.getAttribute("userPW");
10
11     out.println("설정된 세션의 속성 값 [1] : " + user_id + "<br>");
12     out.println("설정된 세션의 속성 값 [2] : " + user_pw);
13 %>
14 </body>
15 </html>
```

← → ⏏ 🔄 http://localhost:8080/eur

설정된 세션의 속성 값 [1] : admin
설정된 세션의 속성 값 [2] : 1234



❖ 메소드 `getAttributeNames()`

- 세션의 속성으로 지정한 이름을 모두 알기 위해서
- 반환 값은 인터페이스 `Enumeration`
 - 패키지 `java.util`
 - 페이지 지시자의 `import` 속성을 이용

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

```
...
```

```
Enumeration<String> e = session.getAttributeNames();
```

❖ `Enumeration<String>`

- 자료 유형 `Enumeration`은 여러 개의 내부 원소를 일렬로 저장한 구조를 지원
- 객체 `e`의 자료유형은 `Enumeration<String>`
 - `<String>`의 일반화 유형으로 선언
 - `e.nextElement()`의 반환 값을 저장할 때 `String`으로 자료 유형 변환이 필요 없는 장점

다중 세션 조회 : Enumeration 이용

```
Enumeration<String> e = session.getAttributeNames();

while ( e.hasMoreElements() ) {
    String name = e.nextElement();
    String value = (String) session.getAttribute(name);
    out.println("세션 name : " + name + ", ");
    out.println("세션 value : " + value + "<br>");
}
```

반환형	메소드 이름	기능
boolean	hasMoreElements()	enumeration의 내부에 더 이상의 원소가 있는지 결과를 반환, 있다면 true, 없으면 false를 반환
Object	nextElement()	enumeration의 내부에 더 이상의 원소가 있다면 다음 원소를 반환



```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR" pageEncoding="EUC-KR"%>
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
5 <title>JSP 예제 : sessionattribute.jsp</title>
6 </head>
7 <body>
8     <%@ page import="java.util.Enumeration, java.util.Date" %>
9     <h1>세션 예제</h1>
10    <hr><h2>세션 만들기</h2>
11    <%
12        session.setAttribute("id", "javajsp");
13        session.setAttribute("pwd", "jdktomcat");
14    %>
15    <hr><h2>세션 조회</h2>
16    세션 ID : <%= session.getId() %><br>
17    세션 CreationTime : <%= new Date(session.getCreationTime()) %><br><br>
18    <%
19        Enumeration<String> e = session.getAttributeNames();
20        while ( e.hasMoreElements() ) {
21            String name = e.nextElement();
22            String value = (String) session.getAttribute(name);
23            out.println("세션 name : " + name + ", ");
24            out.println("세션 value : " + value + "<br>");
25        }
26    %>
27    <br>세션 Invalidate : <% session.invalidate(); %>
28 </body>
29 </html>
```

❖ 마지막 부분에서

- 다음과 같이 session.invalidate()를 호출
 -
세션 Invalidate : <% session.invalidate(); %>
 - 이전 세션을 무조건 무효화시킴
- 다시 페이지를 refresh하면 항상 세션 ID와 생성 시간이 바뀜

세션 예제

세션 만들기

세션 조회

세션 ID : 8CD894B7A053C91987D07DEC9466F794
세션 CreationTime : Wed Apr 28 18:04:33 KST 2021

세션 name : userPW, 세션 value : 1234
세션 name : id, 세션 value : javajsp
세션 name : pwd, 세션 value : jdktomcat
세션 name : userID, 세션 value : admin

세션 Invalidate :

세션 예제

세션 만들기

세션 조회

세션 ID : 58E31C8FBFE4758E3FAC2428C69CD7EA
세션 CreationTime : Wed Apr 28 18:43:26 KST 2021

세션 name : id, 세션 value : javajsp
세션 name : pwd, 세션 value : jdktomcat

세션 Invalidate :



❖ 단일 세션 속성 삭제

- 세션에 저장된 하나의 세션 속성 이름 삭제
- `removeAttribute()` 메소드 사용
`void removeAttribute(String name)`
- 메소드 `removeAttribute("id")`와 같이 속성 이름을 인자로 호출
ex) `session.removeAttribute("id");`

❖ 다중 세션 속성 삭제

- 세션에 저장된 모든 세션 속성 이름을 삭제
- `void invalidate()` 메소드 사용
ex) `session.invalidate();`

❖ 메소드 `isNew()`

- 세션이 새로 만들어진 것인지를 **boolean** 유형으로 반환

세션이 생성된 이후 시간, 무반응 시간 계산

❖ 현재 세션이 만들어진 이후 지난 시간을 출력

- 클래스 Date의 메소드 getTime()
 - Date의 시간정보를 1970년 1월 1일 0시 이후의 밀리세컨드 초로 반환

```
long nowtime = new Date().getTime();  
<% long sessiontime = nowtime - session.getCreationTime(); %>  
세션이 만들어진 이후 지난 시간 : <%=sessiontime/1000 %>초
```

❖ 현재 페이지에서 서버에 반응을 보이지 않은 시간 계산

- 현재 시간인 nowtime에서 이전에 참조한 시간이 저장된 beforetime를 뺀
- beforetime은 이전에 참조한 시간을 저장한 변수이므로 소속 변수로 선언
 - 페이지를 종료할 때 다시 nowtime을 beforetime에 저장

```
<%! long beforetime = new Date().getTime();  
//이전 페이지 참조 시간을 저장하는 소속 변수 %>
```

```
<% long inactiveinterval = nowtime - beforetime; %>  
서버에 반응을 보이지 않은 시간 : <%=inactiveinterval/1000 %>초
```

```
<% beforetime = nowtime; %>
```



❖ 세션에 저장된 세션 속성 삭제

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2
3<html>
4<head>
5 <title>Session</title>
6 </head>
7<body>
8     <p><h4>----- 세션을 삭제하기 전 -----</h4>
9     <%
10         String user_id = (String) session.getAttribute("userID");
11         String user_pw = (String) session.getAttribute("userPW");
12         out.println("설정된 세션 이름 userID : " + user_id + "<br>");
13         out.println("설정된 세션 값 userPW : " + user_pw + "<br>");
14
15         session.removeAttribute("userID");
16     %>
17     <p><h4>----- 세션을 삭제한 후 -----</h4>
18     <%
19         user_id = (String) session.getAttribute("userID");
20         user_pw = (String) session.getAttribute("userPW");
21         out.println("설정된 세션 이름 userID : " + user_id + "<br>");
22         out.println("설정된 세션 값 userPW : " + user_pw + "<br>");
23     %>
24 </body>
25 </html>
```

----- 세션을 삭제하기 전 -----

설정된 세션 이름 userID : admin
설정된 세션 값 userPW : 1234

----- 세션을 삭제한 후 -----

설정된 세션 이름 userID : null
설정된 세션 값 userPW : 1234

❖ 세션에 저장된 세션 속성 삭제

```

1 <%@ page contentType="text/html; charset=utf-8"%>
2 <%@ page import="java.util.Enumeration"%>
3 <html>
4 <head>
5 <title>Session</title>
6 </head>
7 <body><p> <h4>----- 세션을 삭제하기 전 -----</h4>
8 <%
9     String name;
10    String value;
11
12    Enumeration en = session.getAttributeNames();
13    int i = 0;
14
15    while (en.hasMoreElements()) {
16        i++;
17        name = en.nextElement().toString();
18        value = session.getAttribute(name).toString();
19        out.println("설정된 세션 이름 [ " + i + " ] : " + name + "<br>");
20        out.println("설정된 세션 값 [ " + i + " ] : " + value + "<br>");
21    }
22    session.removeAttribute("userID");
23 <%>
24 <p> <h4>----- 세션을 삭제한 후 -----</h4>
25 <%
26    en = session.getAttributeNames();
27    i = 0;
28    while (en.hasMoreElements()) {
29        i++;
30        name = en.nextElement().toString();
31        value = session.getAttribute(name).toString();
32        out.println("설정된 세션 이름 [ " + i + " ] : " + name + "<br>");
33        out.println("설정된 세션 값 [ " + i + " ] : " + value + "<br>");
34    }
35 <%>
36 </body></html>

```

----- 세션을 삭제하기 전 -----

설정된 세션 이름 [1] : userPW
 설정된 세션 값 [1] : 1234
 설정된 세션 이름 [2] : userID
 설정된 세션 값 [2] : admin

----- 세션을 삭제한 후 -----

설정된 세션 이름 [1] : userPW
 설정된 세션 값 [1] : 1234



❖ 세션에 저장된 모든 세션 속성 삭제

```

1 <%@ page contentType="text/html; charset=utf-8"%>
2
3 <html>
4 <head>
5 <title>Session</title>
6 </head>
7 <body> <p> <h4>----- 세션을 삭제하기 전 -----</h4>
8 <%
9     String user_id = (String) session.getAttribute("userID");
10    String user_pw = (String) session.getAttribute("userPW");
11
12    out.println("설정된 세션 이름 userID : " + user_id + "<br>");
13    out.println("설정된 세션 값 userPW : " + user_pw + "<br>");
14
15    if (request.isRequestedSessionIdValid() == true) {
16        out.print("세션이 유효합니다.");
17    }else {
18
19        out.print("세션이 유효하지 않습니다.");
20    }
21
22    session.invalidate();
23 %>
24 <p> <h4>----- 세션을 삭제한 후 -----</h4>
25 <%
26    if (request.isRequestedSessionIdValid() == true) {
27        out.print("세션이 유효합니다.");
28    }else {
29
30        out.print("세션이 유효하지 않습니다.");
31    }
32 %>
33 </body> </html>

```

----- 세션을 삭제하기 전 -----

설정된 세션 이름 userID : admin
 설정된 세션 값 userPW : 1234
 세션이 유효합니다.

----- 세션을 삭제한 후 -----

세션이 유효하지 않습니다.



❖ 세션 유효 시간

- 세션을 유지하기 위한 세션의 일정 시간
- 웹 브라우저에 마지막 접근한 시간부터 일정 시간 이내에 다시 웹 브라우저에 접근하지 않으면 자동으로 세션이 종료
- 세션 유효 시간을 설정메소드

`void setMaxInactiveInterval(int interval)`

- 5초 이상 서버에 반응을 하지 않으면 세션을 무효화 시킴
`session.setMaxInactiveInterval(5);`



❖ 세션 유효 시간 가져와 출력하기

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Session</title>
5 </head>
6 <body>
7     <p> <h4>----- 세션 유효 시간 변경 전 -----</h4>
8     <%
9         int time = session.getMaxInactiveInterval() / 60;
10
11         out.println("세션 유효 시간 : " + time + "분<br>");
12     %>
13     <p> <h4>----- 세션 유효 시간 변경 후 -----</h4>
14     <%
15         session.setMaxInactiveInterval(60 * 60);
16         time = session.getMaxInactiveInterval() / 60;
17
18         out.println("세션 유효 시간 : " + time + "분<br>");
19     %>
20 </body>
21 </html>
```

----- 세션 유효 시간 변경 전 -----

세션 유효 시간 : 30분

----- 세션 유효 시간 변경 후 -----

세션 유효 시간 : 60분



❖ 세션 아이디와 웹 사이트에서 유지한 시간 출력하기

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Session</title>
5 </head>
6 <body>
7 <%
8     String sessin_id = session.getId();
9
10    long last_time = session.getLastAccessedTime();
11
12    long start_time = session.getCreationTime();
13
14    long used_time = (last_time - start_time) / 60000;
15
16    out.println("세션 아이디 : " + sessin_id + "<br>");
17    out.println("요청 시작 시간 : " + start_time + "<br>");
18    out.println("요청 마지막 시간 : " + last_time + "<br>");
19    out.println("웹 사이트에서 경과 시간 : " + used_time + "<br>");
20    %>
21 </body>
22 </html>
```

세션 아이디 : F43D2D609E0C6511A0F023BEE4E127F6
요청 시작 시간 : 1619604029713
요청 마지막 시간 : 1619604029714
웹 사이트에서 경과 시간 : 0

// 사이트에 머문 시간이 오래되면 경과한 시간이 표시됨

쿠키와 세션의 차이



구분	쿠키	세션
사용 클래스	Cookie 클래스	HttpSession 인터페이스
저장 형식	텍스트 형식	Object 형
저장 장소	클라이언트	서버(세션 아이디만 클라이언트에 저장)
종료 시점	쿠키 저장 시 설정(설정하지 않을 경우 웹 브라우저 종료 시 소멸)	정확한 시점을 알 수 없음
리소스	클라이언트의 리소스 사용	서버의 리소스 사용
보안	클라이언트에 저장되므로 사용자의 변경이 가능하여 보안에 취약	서버에 저장되어 있어 상대적으로 안정적



- ❖ 강의 동영상 자료에 있는 실습 문제(총 12개)를 모두 프로그램 한 후 소스와 출력 결과를 편집하여 제출하세요.

-
- ❖ 제출방법 : ppt파일에 소스와 출력 결과 편집 후 제출
 - ❖ 제출마감일 : 2021년 05월10일 월요일 23시 55분까지
 - ❖ 제출장소 : 한림 스마트 LEAD(<https://smartlead.hallym.ac.kr>) 해당 과목 [과제]란에 제출하시면 됩니다. (제출시 제목란에 “여러분의 학번이름”을 쓰시고, 파일을 전송하시면 됩니다.)