JSP 프로그래밍

(재)대덕인재개발원

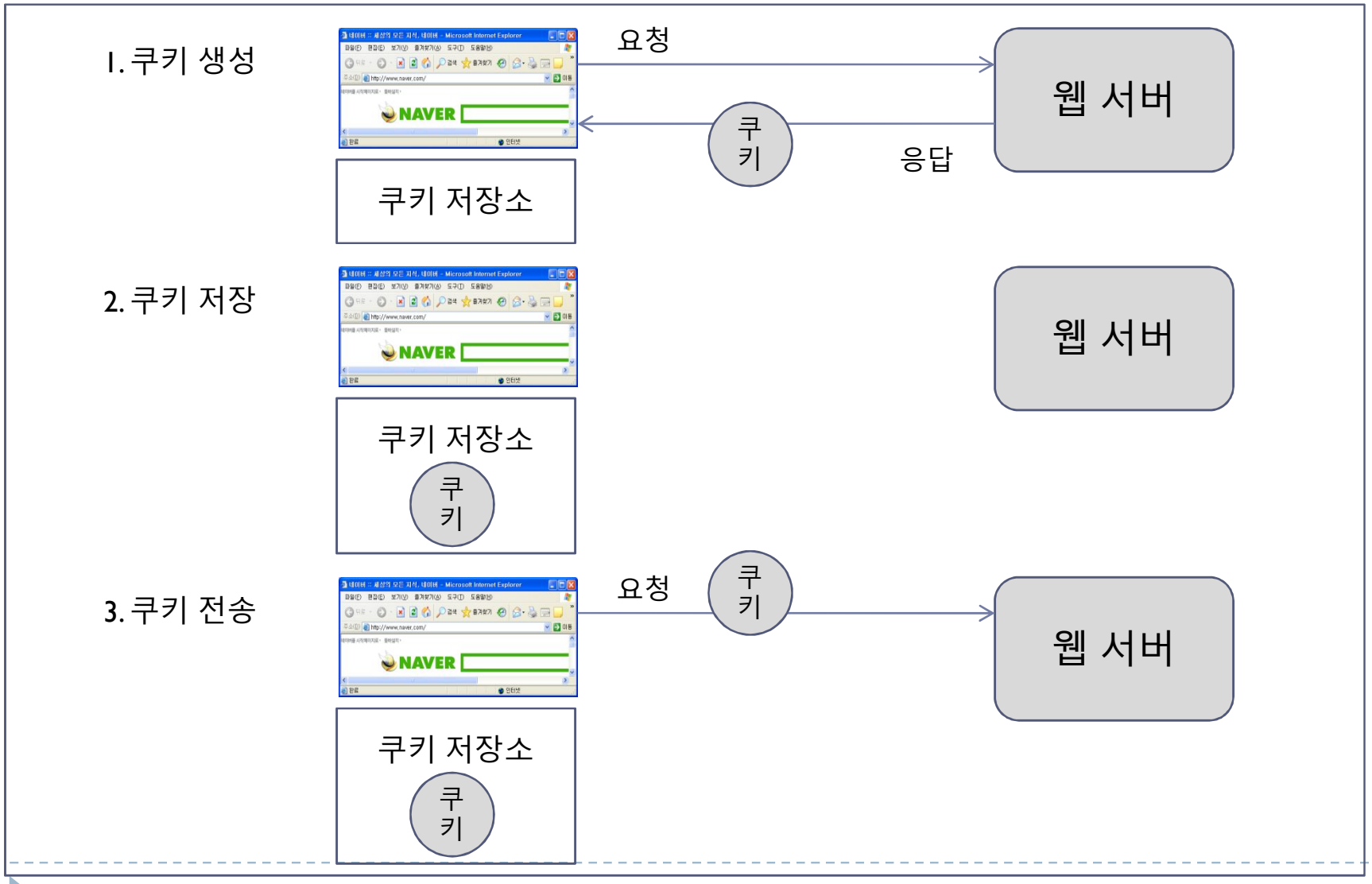
09 클라이언트와의 대화 1 - 쿠키

- 1. 쿠키 사용하기
- 2. 쿠키 처리를 위한 유틸리티 클래스
- 3. 쿠키를 사용한 로그인 유지

9.1 쿠키 사용하기

- 쿠키(cookie)는 웹 브라우저가 보관하고 있는 데이터로서 웹 서버에 요청을 보낼 때 함께 전송된다.
- 쿠키는 웹 서버와 웹 브라우저 양쪽에서 생성할 수 있으며, 웹 서버는 웹 브라우저가 전송한 쿠키를 사용하여 필요한 데이터를 읽어올 수 있다.
- 웹 서버와 웹 브라우저는 쿠키를 사용해서 서로 필요한 값을 공유하게 되며 상태를 유지할 수 있다.

9.1 쿠키 사용하기-쿠키 동작 방식



9.1 쿠키 사용하기-쿠키 동작 방식

- 쿠키 생성 단계
  - 쿠키를 사용하기 위해서는 먼저 쿠키를 생성해야 한다. JSP 프로그래밍에서 쿠키는 주로 웹 서버 측에서 생성한다. 생성된 쿠키는 응답 데이터에 함께 저장되어 전송된다.
- 쿠키 저장 단계
  - 웹 브라우저는 응답 데이터에 포함된 쿠키를 쿠키 저장소에 보관한다. 쿠키의 종류에 따라 메모리나 파일로 저장된다.
- 쿠키 전송 단계
  - 웹 브라우저는 한번 저장된 쿠키를 매번 요청이 있을 때마다 웹 서버에 전송한다. 웹 서버는 웹 브라우저가 전송한 쿠키를 사용해서 필요한 작업을 수행할 수 있다.
  - * 웹 브라우저에 쿠키가 저장되면, 웹 브라우저는 쿠키가 삭제되기 전까지 웹 서버에 쿠키를 전송한다. 지속적으로 유지해야 하는 정보는 쿠키를 사용할 수 있다.

9.1.1 쿠키의 구성

- 쿠키의 구성

항 목	설 명
이름	각각의 쿠키를 구별하는데 사용되는 이름
값	쿠키의 이름과 관련된 값
유효시간	쿠키의 유지 시간
도메인	쿠키를 전송할 도메인
경로	쿠키를 전송할 요청 경로

- 쿠키의 네임 규약
  - 쿠키의 이름은 아스키 코드의 알파벳과 숫자만을 포함할 수 있다.
  - 콤마(,), 세미콜론(;), 공백(‘ ’) 등의 문자는 포함할 수 없다.
  - ‘$’로 시작할 수 없다.

9.1.2 쿠키 생성하기

```
<%@ page contentType="text/html; charset=utf-8"  %>
<%@ page import="java.net.URLEncoder" %>
<%
                // 쿠키 객체 생성
                Cookie cookie = new Cookie("name", URLEncoder.encode("이산", "utf-8"));
                // 응답 객체에 쿠키추가
                response.addCookie(cookie);
%>
<html>
<head><title> 쿠키 생성 </title></head>
<body>
<%= cookie.getName() %> 쿠키의 값 = "<%= cookie.getValue() %>"
</body>
</html>
```

9.1.2 쿠키 생성하기

- Cookie 클래스의 주요 메서드

메서드	리턴 타입	설 명
getName()	String	쿠키의 이름을 구한다.
getValue()	String	쿠키의 값을 구한다.
setValue(String value)	void	쿠키의 값을 지정한다.
setDomain(String pattern)	void	이 쿠키가 전송될 서버의 도메인을 지정한다.
getDomain()	String	쿠키의 도메인을 구한다.
setPath(String url)	void	쿠키를 전송할 경로를 지정한다.
getPath()	String	쿠키의 전송 경로를 구한다.
setMaxAge(int expire)	void	쿠키의 유효시간을 초 단위로 지정한다. 음수를 입력할 경우 웹 브라우저를 닫을 때 쿠키가 함께 삭제된다.
getMaxAge()	int	쿠키의 유효 시간을 구한다.

9.1.3 쿠키 값 읽어오기

- 웹 브라우저는 요청 헤더에 쿠키를 저장해서 보내며, request.getCookies() 메서드를 통해 읽어올 수 있다.

```
<body>
쿠키 목록<br>
<%
    Cookie[] cookies = request.getCookies();
    if(cookies != null && cookies.length > 0){
        for(int i = 0; i < cookies.length; i++){
%>
            <%= cookies[i].getName() %> =
            <%= URLDecoder.decode(cookies[i].getValue(), "utf-8") %><br>
<%
        }
    }else{
%>
        쿠키가 존재하지 않습니다.
<%
    }
%>
```

9.1.4 쿠키 값 변경 및 삭제하기

쿠키 값 변경하기

```
Cookie cookie = new Cookie("name", URLEncoder.encode("새로운 값","utf-8"));
response.addCookie(cookie);
```

```
Cookie[] cookies = request.getCookies();
if(cookie != null && cookies.length > 0){
    for(int i = 0; i < cookies.length; i++){
        if(cookies[i].getName().equals("name")){
            Cookie cookie = new Cookie(name, value);
            response.addCookie(cookie);
        }
    }
}
```

쿠키 값 삭제하기

```
Cookie cookie = new Cookie("name", value);
cookie.setMaxAge(0);     // 유효시간을 0으로 지정하면 웹브라우저는 관련쿠키를 삭제한다.
response.addCookie(cookie);
```

9.1.5 쿠키의 도메인

- 기본적으로 쿠키는 그 쿠키를 생성한 서버에만 전송된다.
- 때에 따라 같은 도메인을 사용하는 서버에 대해서 모두 쿠키를 보내고 싶은 경우 setDomain() 메서드를 사용하면 된다.
- .san.net : 점으로 시작하는 경우 관련 도메인에 모두 쿠키 전송.
- www.san.net : 특정 도메인에 대해서만 쿠키 전송.

```
// 관련 도메인
Cookie cookie1 = new Cookie("id", "san");
cookie1.setDomain(".san.net");
response.addCookie(cookie1);

// 기본
Cookie cookie2 = new Cookie("only", "onlycookie");
response.addCookie(cookie2);

// 특정 도메인
Cookie cookie3 = new Cookie("invalid", "invalidcookie");
cookie3.setDomain("java.san.net");
response.addCookie(cookie3);
```

9.1.6 쿠키의 경로

- 쿠키는 도메인뿐만 아니라 경로를 지정할 수도 있다.
- Coolie 클래스의 setPath() 메서드를 사용하여 지정.
- 경로는 URL에서 도메인 이후의 부분을 의미하며, 디렉토리 수준의 경로를 사용한다.
- setPath()메서드를 사용하여 쿠키의 경로를 지정하게 되면, 그 쿠키는 지정한 경로 또는 하위 경로에 대해서만 쿠키를 전송하게 된다.

```
Cookie cookie1 = new Cookie("path1",
    URLEncoder.encode("경로:/chap09/path1", "utf-8"));
cookie1.setPath("/chap09/path1");
response.addCookie(cookie1);

Cookie cookie2 = new Cookie("path2",
    URLEncoder.encode("경로:", "utf-8"));
response.addCookie(cookie2);

Cookie cookie3 = new Cookie("path3",
    URLEncoder.encode("경로:/", "utf-8"));
cookie1.setPath("/");
response.addCookie(cookie3);
```

9.1.7 쿠키의 유효시간

- 쿠키는 유효시간을 갖으며, 그 시간 동안 쿠키가 존재한다.
- 지정하지 않으면 브라우저를 닫으면 쿠키는 자동 삭제됨.

```
<%@ page contentType="text/html; charset=utf-8"  %>
<%
    Cookie cookie = new Cookie("one-hour", "1 Hour");
    cookie.setMaxAge(60 * 60); // 60초(1분) * 60 = 1시간
    response.addCookie(cookie);
%>

<html>
<head><title> 쿠키 유효시간 설정</title></head>
<body>
    유효시간이 1시간인 one-hour 쿠키 설정.
</body>
</html>
```

9.1.8 쿠키와 헤더

- response.addCookie() 로 추가되는 쿠키는 실제로 Set-Cookie 헤더를 통해서 전달된다.
- 한 개의 Set-Cookie 헤더는 한 개의 쿠키 값이 전달되며, Set-Cookie 헤더는 다음과 같이 구성된다.

```
쿠키이름=쿠키값; Domain=도메인값; Path=경로값; Expires=GMT형식의만료일시
```

```
Cookie nameCookie = new Cookie("name", "Lee San");
nameCookie.setDomain("www.san.com");

Cookie idCookie = new Cookie("id", "FreelancerSan");
idCookie.setMaxAge(60*60);

name 쿠키의 경우 : name = Lee San;   Domain=www.san.com
id 쿠키의 경우      : id = FreelancerSan;    Expries=Thu, 29-Jan-2010 16:20:04 GMT
```

- 쿠키는 응답 헤더 형태로 웹 브라우저에 전달되기 때문에, 쿠키 역시 출력 버퍼가 플러시 된 이후에는 새롭게 추가할 수 없다.
- 따라서 쿠키의 추가 및 변경 작업은 반드시 출력 버퍼가 플러시 되기 전에 처래해 주어야 한다.

9.2 쿠키 처리를 위한 유틸리티 클래스

```
public class CookieBox{
    // 쿠키를 저장할 Map 데이터
    private Map<String, Cookie> cookieMap = new HashMap<String, Cookie>();
    // 생성자, 패러미터로 받은 request 객체에서 쿠키 배열을 읽어와 cookieMap 에 저장한다.
    public CookieBox(HttpServletRequest request){
    }
    // 쿠키 생성 메서드
    public static Cookie createCookie(String name, String value) throws IOException{
    }
    // 쿠키 생성 메서드
    public static Cookie createCookie(String name, String value, String path, int maxAge) throws IOException{
    }
    // 쿠키 생성 메서드
    public static Cookie createCookie(String name, String value, String domain, String path, int maxAge) throws IOException{
    }
    // 쿠키를 가져오는 메서드
    public Cookie getCookie(String name){
    }
    // 쿠키의 값을 가져오는 메서드
    public String getValue(String name) throws IOException{
    }
    // 쿠키의 존재 여부 확인 메서드
    public boolean exists(String name){
    }
}
```

9.3 쿠키를 사용한 로그인 유지

- 1. 로그인을 하면 관련 쿠키를 생성.
- 2. 관련 쿠키가 존재하면 로그인한 상태라고 판단한다.
- 3. 로그아웃을 하면 관련 쿠키를 삭제한다.

9.3.1 로그인 처리

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

    if(id.equals(password)){
        // ID와 Password 가 같으면 로그인 성공으로 판단.
        // 로그인 관련 정보 쿠키에 저장,
        // path="/", 유효시간(-1) : 브라우저 닫기전까지
        response.addCookie(CookieBox.createCookie("LOGIN", "SUCCESS", "/", -1));
        response.addCookie(CookieBox.createCookie("ID", id, "/", -1));
%>
<html>
<head><title> 로그인 성공 </title></head>
<body>
    로그인에 성공했습니다.
</body>
</html>
<%
    }else{
        out.println("<script>");
        out.println("alert('로그인 실패하였습니다.')");
        out.println("histoty.go(-1)");
        out.println("</script>");
    }
%>
```

9.3.2 로그인 여부 판단

```
<%@ page contentType="text/html; charset=utf-8" %>
<%@ page import="util.CookieBox" %>
<%
    CookieBox cookieBox = new CookieBox(request);
    boolean login = cookieBox.exists("LOGIN") &&
        cookieBox.getValue("LOGIN").equals("SUCCESS");
%>
<html>
<head><title> 로그인여부 검사 </title></head>
<body>
<%
    if(login){
%>
        아이디 "<%= cookieBox.getValue("ID") %>"로 로그인 한 상태
<%
    }else{
        out.println("로그인하지 않은 상태");
    }
%>
</body>
</html>
```

9.3.3 로그아웃 처리

```
<%@ page contentType="text/html; charset=utf-8" %>
<%@ page import="util.CookieBox" %>
<%
    // 로그아웃을 위해 관련 쿠키 삭제
    response.addCookie(
        CookieBox.createCookie("LOGIN", "", "/", 0));

    response.addCookie(
        CookieBox.createCookie("ID", "", "/", 0));
%>
<html>
<head><title> 로그아웃 </title></head>
<body>

로그아웃하였습니다.

</body>
</html>
```