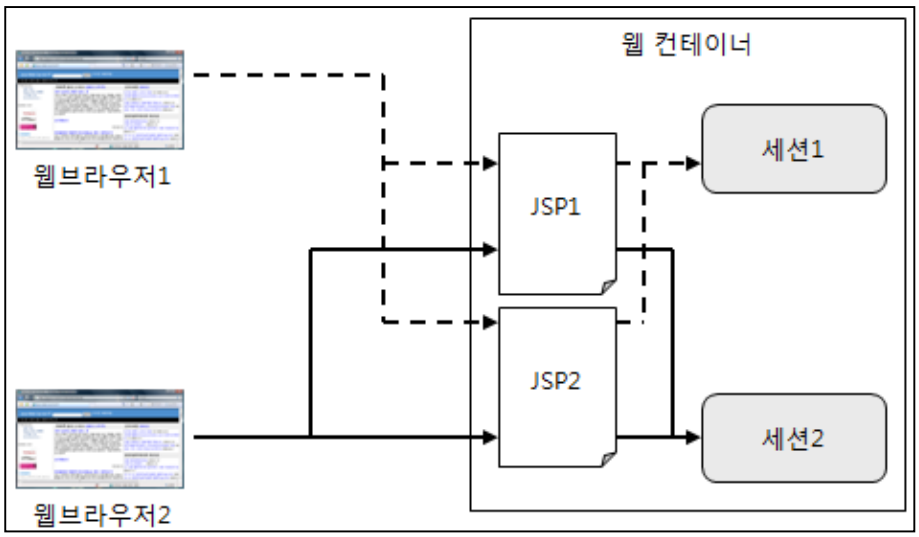
10장 클라이언트와의 대화2: 세션

- 서버 세션을 사용하면 클라이언트의 상태를 저장할 수 있다. 쿠키와의 차이점은 세션은 웹 브라우저가 아니라 서버에 값을 저장한다는 점이다.

10.1 세션 사용하기: session 기본 객체

- 웹 컨테이너에서 클라이언트의 정보를 보관할 때 사용
- 오직 서버에서만 생성
- 클라이언트마다 세션이 생성



10.1.1 세션 생성하기

- JSP에서 세션을 생성하려면 다음과 같이 page 디렉티브의 session 속성을 "true"로 저장하면 된다.

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ page session = "true" %>
<%
   ...
   session.setAttribute("userInfo", userInfo);
   ...
%>
```

10.1.2 session 기본 객체

- 웹 서버는 웹 브라우저에 세션 ID를 전송한다. 웹 브라우저는 웹 서버에 연결할 때마다 매번 세션 ID를 보내서 웹 서버가 어떤 세션을 사용할지 판단할 수 있게 한다.

[chap10/sessionInfo.jsp]

```
<%@ page contentType="text/html; charset=utf-8"%>
<%@ page session="true"%>
<%@ page import="java.util.Date"%>
<%@ page import="java.text.SimpleDateFormat"%>
<%
          Date time = new Date();
          SimpleDateFormat formatter =
            new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
%>
<html>
<head>
<title>세션정보</title>
</head>
<body>
          세션ID:
          <%= session.getId() %>
          <br>
          <%
          time.setTime(session.getCreationTime());
%>
          세션생성시간:
          <%= formatter.format(time) %>
          <br>
          <%
          time.setTime(session.getLastAccessedTime());
%>
          최근접근시간:
          <%= formatter.format(time) %>

</body>
</html>
```

10.1.3 기본 객체의 속성 사용

■ 한 번 생성된 세션은 지정한 유효 시간 동안 유지된다.

[chap10/setMemberInfo.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%
          session.setAttribute("MEMBERID", "madvirus");
          session.setAttribute("NAME", "최범균");
%>
<html>
<head><title>세션에 정보 저장</title></head>
<body>

세션에 정보를 저장하였습니다.

</body>
</html>
```

[chap10/viewMemberInfo.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<html>
<head><title>세션 정보 조회</title></head>
<body>
```

```
MEMBERID = <%= session.getAttribute("MEMBERID") %><br/>
NAME = <%= session.getAttribute("NAME") %>

</body>
</html>
```

10.1.4 세션 종료

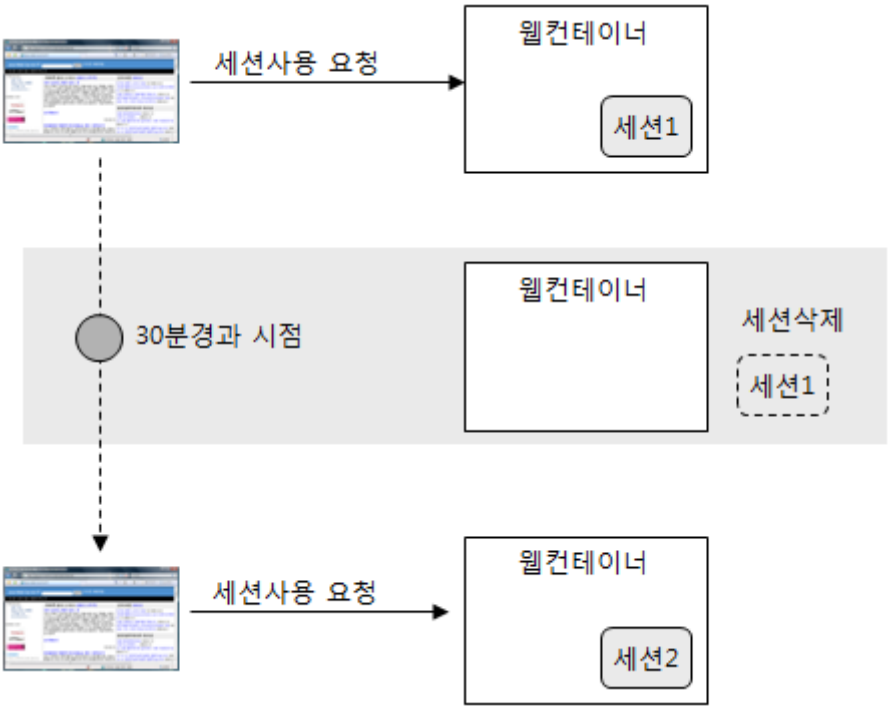
[chap10/closeSession.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%
        session.invalidate();
%>
<html>
<head><title>세션 종료</title></head>
<body>

세션을 종료하였습니다.

</body>
</html>
```

10.1.5 세션 유효 시간

- 세션은 최근 접근 시간을 갖는다. getLastAccessedTime() 메서드는 최근에 session 기본 객체에 접근한 시간을 나타낸다. session.setMaxInactiveInterval() 메서드는 초 단위로 유효 시간을 설정한다.

```
// 세션의 최근 접근 시간
session.getLastAccessedTime();

// 세션의 유효 시간을 30분으로 지정 (방법1)
session.setMaxInactiveInterval(60*30);
```



■ 세션 유형 시간을 web.xml 파일에 다음과 같이 <session-config> 태그를 사용하여 지정할 수도 있다. 이 값의 단위는 분이다. (방법2)

[chap10/WEB-INF/web.xml]

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
             http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         version="3.1">

          <session-config>
                   <session-timeout>30</session-timeout>
          </session-config>
</web-app>
```

10.1.6 request.getSession()을 이용한 세션 생성

■ HttpSession을 생성하는 또 다른 방법은 request 기본 객체의 getSession() 메서드를 사용하는 것이다.

10.2 세션을 사용한 로그인 상태 유지

■ 세션을 사용해서 로그인을 처리하는 방식은 쿠키를 사용한 방식과 비슷하다.

1. 로그인에 성공하면 session 기본 객체의 특정 속성에 데이터를 기록한다.
2. 이후로 session 기본 객체의 특정 속성이 존재하면 로그인한 것으로 간주한다.
3. 로그아웃할 경우 session.invalidate() 메서드를 호출하여 세션을 종료한다.

10.2.1 인증된 사용자 정보 session 기본 객체에 저장하기

[chap10/memeber/sessionLoginForm.jsp]

```
<%@ page contentType="text/html; charset=utf-8"%>
<html>
<head>
<title>로그인폼</title>
</head>
<body>

          <form action="<%= request.getContextPath() %>/member/sessionLogin.jsp"
                    method="post">
                    아이디 <input type="text" name="id" size="10"> 암호 <input
                              type="password" name="password" size="10"> <input
                              type="submit" value="로그인">
          </form>

</body>
</html>
```

[chap10/memeber/sessionLogin.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%
          String id = request.getParameter("id");
          String password = request.getParameter("password");

          if (id.equals(password)) {
                    session.setAttribute("MEMBERID", id);
%>
<html>
<head><title>로그인성공</title></head>
<body>

로그인에 성공했습니다.

</body>
</html>
<%
          } else { // 로그인 실패시
%>
<script>
alert("로그인에 실패하였습니다.");
history.go(-1);
</script>
<%
          }
%>
```

10.2.2 인증 여부 판단

[chap10/memeber/sessionLoginCheck.jsp]

```
<%@ page contentType="text/html; charset=utf-8"%>
<%
          String memberId = (String)session.getAttribute("MEMBERID");
          boolean login = memberId == null ? false : true;
```

```
%>
<html>
<head>
<title>로그인여부 검사</title>
</head>
<body>

          <%
          if (login) {
%>
          아이디 "<%= memberId %>"로 로그인 한 상태
          <%
          } else {
%>
          로그인하지 않은 상태
          <%
          }
%>
</body>
</html>
```

10.2.3 로그아웃 처리

[chap10/memeber/sessionLogout.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%
          session.invalidate();
%>
<html>
<head><title>로그아웃</title></head>
<body>

로그아웃하였습니다.

</body>
</html>
```

10.3 연관된 정보 저장을 위한 클래스 작성

```
<%
   String id = (String)session.getAttribute("id");
   String name = (String)session.getAttribute("name");
   String email = (String)session.getAttribute("email");
   String name = (String)session.getAttribute("male");
   int age = (String)session.getAttribute("age");
%>

public class Memberinfo {
   private String id;
   private String name;
   private String email;
   private String male;
   private int age;
  // get 메서드
}
```

10.4 서블릿 컨텍스트와 세션

- 서로 다른 두 웹 어플리케이션이 다른 세션 ID를 사용하고 다른 JSESSIONID 쿠키를 사용한다. 다시 말하면 서로 다른 웹 어플리케이션이 세션을 공유하지 않음을 의미한다.