뇌를 자극하는 JSP&Servlet

쿠키와 세션

뇌를 자극하는 JSP&Servlet

쿠키와 세션(Cookie & Session)

쿠키와 세션을 사용하는 이유는 HTTP 프로토콜의 특징인 Connetionless, Stateless 특성을 보완하기 위하여 사용한다.
즉, 서버와 클라이언트가 통신을 할 때 클라이언트가 누구이며 상태 인증(로그인과 같은) 을 유지할 수 있도록 해줍니다.

쿠키와 세션의 큰 차이점은 다음과 같다.

1) 기록 정보가 저장되는 위치

쿠키 : 클라이언트 로컬

세션 : 서버

2) 보안성

쿠키는 로컬에 저장이 되므로 변질되거나 Request 단계에서 스나이핑 당할 우려가 있으므로 보안에 취약하지만 세션은 쿠키를 이용해 Session ID 만 저장하고 서버에서 처리하기 때문에 쿠키보다 보안성이 우수하다.

3) 라이프 사이클

쿠키와 세션 모두 만료시간을 정할 수 있으나 쿠키는 클라이언트의 로컬에 저장이 되므로 브라우저를 종료해도 계속해서 정보가 남은 상태를 유지할 수 있으나 세션은 브라우저를 종료하면 만료 시간에 상관없이 삭제된다.

4) 속도

쿠키는 로컬에 정보가 있으므로 속도가 빠르지만 세션은 서버에 정보가 있으므로 처리 과정을 거쳐 비교적 느린 속도를 보인다.

1. 쿠키(Cookie)

쿠키 관련 메소드

- setMAxAge() : 쿠키의 유효기간 설정
- setpath() : 쿠키사용의 유효 디렉토리를 설정
- setValue() : 쿠키의 값을 설정 *
- setVersion() : 쿠키 버전 설정
- getMaxAge() : 쿠키 유효기간 정보를 얻음
- getName() : 쿠키 이름을 얻음
- getPath() : 쿠키사용의 유효 디렉토리 정보를 얻음
- getValue() : 쿠키의 값을 얻음
- getVersion() : 쿠키 버전을 얻음

```
<% Cookie cookie = new Cookie("cookieN", "cookieV"); //쿠키생성, name, value설정
   cookie.setMaxAge(60*60);   //1시간
   response.addCookie(cookie); //reponse객체에 cookie객체를 탑재
%>
```

```
<%
   Cookie[] cookies = request.getCookies();\
   for(int i = 0; i < cookies.length; i++){
      String str = cookies[i].getName();
      if(str.equals("myName")){
         out.println("cookies[" + i + "] name :" + cookies[i].getName() + "<br />");
         out.println("cookies[" + i + "] value :" + cookies[i].getValue() + "<br />");
      }
   }
%>
```

2. 세션(Session)

쿠키와 마찬가지로 서버와의 연결이 끊겼을 때 데이터를 유지하는 수단이다.
단, 쿠키와는 달리 서버상에 객체로 저장하게 된다.
클라이언트의 요청이 발생하면 자동으로 생성되며
세션 내부객체의 메소드를 이용하여 속성을 설정한다.

세션은 서버에서만 접근이 가능하여 보안이 쿠키보다 강하고, 데이터의 용량에 제한이 없다.
최근에는 쿠키보다 세션을 주로 사용하는 추세라고 한다.

세션 관련 메소드

- setAttribute() : 세션에 데이터를 저장
- getAttribute() : 해당하는 세션을얻음 (반환형 : Object)
- getAttributeNames() : 세션에 저장되어 잇는 모든 데이터의 이름(유니크한 키값)을 얻음
- getId() : 자동 생성된 세션의 유니크한 아이디를 얻음
- isNew() : 세션이 최초 생성되었는지, 이전에 생성된 세션인지를 구분
- getMaxInactiveInterval() : 세션의 유효시간을 얻음. 가장 최근 요청시점을 기준으로 카운트(apache-tomcat-\conf\web.xml 참고)
- removeAttribute() : 세션에서 특정 데이터 제거
- invalidate() : 세션의 모든 데이터를 삭제

먼저 다음과 같이 세션을 생성하고 초기화 한다.

```
<%
    //session : 내부객체
    session.setAttribute("SessionName", "SessionData");
    session.setAttribute("Num", 123);
%>
```

세션의 데이터를 얻어오는 방법은 여러가지가 있다.
먼저, getter를 이용해 얻어오는 방법이다.

```
<%
    Object obj1 = session.getAttribute("SessionName");
    String SessionName = (String)obj1;
    Object obj2 = session.getAttribute("Num");
    Integer Num = (Integer)obj2;
%>
```

get메소드의 반환형이 object이기 때문에 캐스팅을 해주어야 한다.

다음으로 Enumeration을 이용한 방법이다.

```
<%
        String sName;
        String sValue;
        Enumeration enumeration = session.getAttributeNames(); //모든이름값 다얻어옴
        while(enumeration.hasMoreElements()){
            sName = enumeration.nextElement().toString();
            sValue = session.getAttribute(sName).toString(); //이름통해서 value얻어옴
            out.println("sName : " + sName + "<br/>");
            out.println("sValue :" + sValue + "<br/>");
        }
%>
```

세션을 삭제할 때는 두 가지 방법이 있는데,

```
<%   session.removeAttribute("SessionName"); //특정세션삭제      %>
```

이렇게 removeArrtibute 메소드를 사용하여 특정 세션을 삭제할 수 있고

```
<%    session.invalidate(); //모든세션삭제   %>
```

invalidate 메소드를 사용해 모든 세션을 삭제해줄 수 있다.