3. Servlet의 핵심 사항들

목차

- 클라이언트에서 서블릿으로 요청하는 방식
- 서블릿에서 한글 처리하기
- 하나의 파라미터 이름으로 여러 개의 파라미터 값 전송될 때 처리
- 서블릿에서 세션 살펴보기

1. 클라이언트에서 서블릿으로 요청하는 방식

- GET 방식
  - 서버에 존재하는 간단한 페이지 요청, 게시판 글 목록 페이지에서 해당 페이지에 대한 목록 출력 요청 시 간단한 파라미터를 전송하는 경우 사용
  - 사용방식
    - < a href="list.jsp?pageNo=2">[2]
    - Get 방식 요청이 전송되는 경우
      - 브라우저 주소 표시줄에 주소를 직접 입력해서 요청을 전송하는 경우
      - Html 의 a 태그를 사용해서 링크를 걸어 전송하는 경우
        - 목록보기
      - Html 폼 태그에서 method 속성을 get로 지정하는 경우
        - <form action=" " name=" " method="get">

1. 클라이언트에서 서블릿으로 요청하는 방식

- POST 방식
  - 특정 페이지로 많은 양의 파라미터를 전송하여 파라미터에 관한 처리를 할 때 사용되는 방식
  - 사용방식
    - <form name=" " action=" " method="post">
  - 회원 가입 요청, 게시판 글쓰기 요청, 자료실 업로드 등을 처리할 때 사용하는 방식

1. 클라이언트에서 서블릿으로 요청하는 방식

- 서블릿 생성(GET 방식)
  - Dynamic Web Project “ch3” 생성
  - 프로젝트명 ch3에서 마우스 오른쪽 버튼을 클릭하고 [New]-[Servlet]을 선택함
  - Create Servlet 창에서 클래스 이름을 “LoginServlet”으로 지정하고 “Next” 버튼을 클릭함
  - URL mappings의 “/LoginServlet”을 선택하고 “Edit” 버튼을 사용하여 “/login”으로 수정함
  - “Next” 버튼을 다시 클릭하고 “doGet” 메소드만 체크한 후 “Finish” 버튼을 클릭함

1. 클라이언트에서 서블릿으로 요청하는 방식

- 서블릿 생성(GET 방식)
  - <Finish> 버튼을 눌러 생성된 서블릿 코드를 확인한다. 빨간색의 x는 오류를 의미하며 서블릿 API가 자바 빌드 경로 내에 설정되어 있지 않기 때문에 오류가 발생한다.
    - ch3프로젝트를 <마우스 오른쪽> 버튼을 클릭한 다음 Properties를 선택함.
    - Add Library를 선택하고 Server Runtime을 선택함.
    - <Next> 버튼을 누르면 나오는 Apache Tomcat v8.0 항목을 선택하고 “Finish” 버튼을 클릭한 후 <OK> 버튼을 클릭함

1. 클라이언트에서 서블릿으로 요청하는 방식

- 서블릿 생성(GET 방식)
  - Ch3 프로젝트의 WebContent디렉토리에 “login.html” 파일 추가
  - 아래 8행부터 13행까지의 내용 추가

```
7  <body>
8  <h1>로그인</h1>
9  <form action="login" method="get">
10 아이디 : <input type="text" name="id" id="id"><br>
11 비밀번호 : <input type="text" name="passwd" id="passwd">
12 <input type="submit" value="로그인">
13 </form>
14 </body>
```

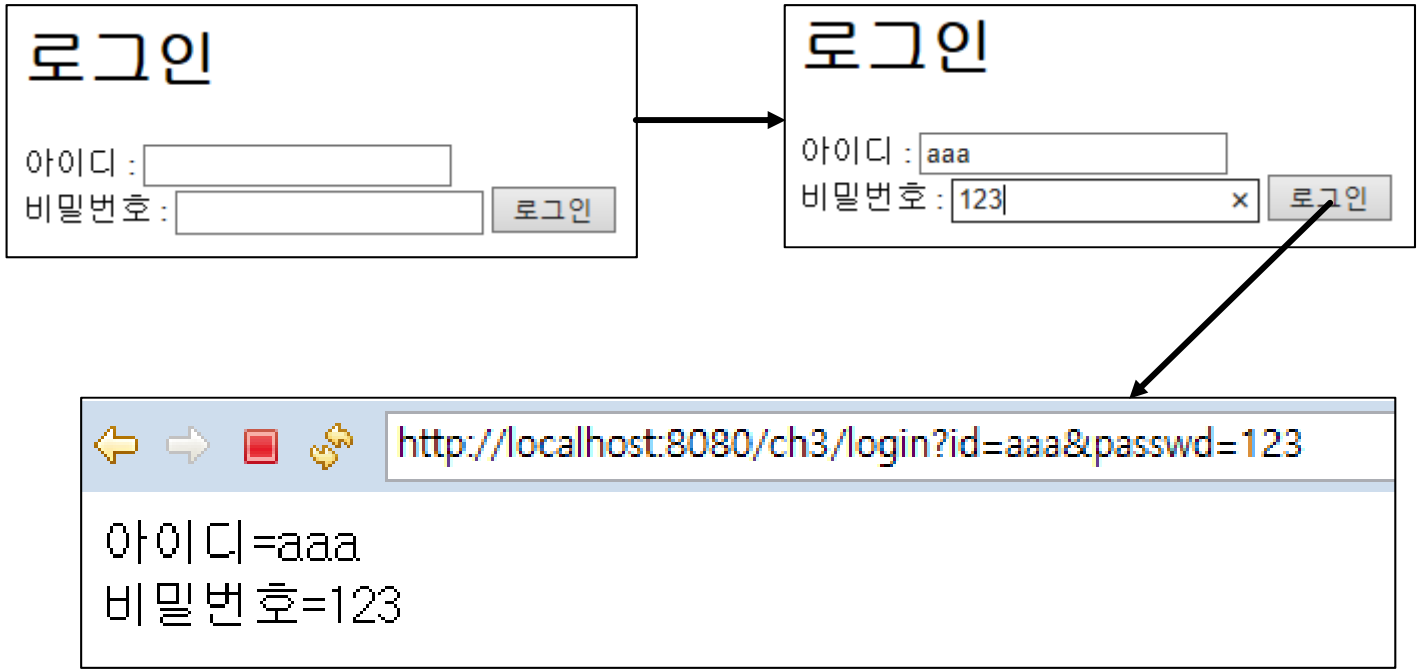
1. 클라이언트에서 서블릿으로 요청하는 방식

- LoginServlet.java 파일 내용 추가
  - "import java.io.PrintWriter;" 추가
  - 코드 하단의 doGet 부분에 아래 내용 추가

```
protected void doGet(HttpServletRequest request, HttpServletResponse
    // TODO Auto-generated method stub
    String id = request.getParameter("id");
    String passwd = request.getParameter("passwd");
    response.setContentType("text/html;charset=euc-kr");
    PrintWriter out = response.getWriter();
    out.println("아이디="+id + "<br>");
    out.println("비밀번호="+passwd + "<br>");
}
```

1. 클라이언트에서 서블릿으로 요청하는 방식

- WebContent의 login.html 파일에서 마우스 오른쪽 버튼을 클릭하고 "Run As->Run on Server"를 선택함
- Run On Server 창이 뜨면 "Finish"를 클릭함



1. 클라이언트에서 서블릿으로 요청하는 방식

- 서블릿 생성(POST 방식)
  - Ch3 프로젝트의 WebContent 디렉토리에 "memReg.html" 파일 추가
  - 아래 8행부터 15행까지의 내용 추가

```
7  <body>
8  <h1>회원 가입</h1>
9  <form action="memReg" method="post">
10 회원명 : <input type="text" name="name"><br>
11 주소 : <input type="text" name="addr"><br>
12 전화번호 : <input type="text" name="tel"><br>
13 취미 : <input type="text" name="hobby"><br>
14 <input type="submit" value="회원 가입">
15 </form>
16 </body>
```

1. 클라이언트에서 서블릿으로 요청하는 방식

- 서블릿 생성(POST 방식)
  - 프로젝트명 ch3에서 마우스 오른쪽 버튼을 클릭하고 [New]-[Servlet]을 선택함
  - Create Servlet 창에서 클래스 이름을 “MemRegServlet”으로 지정하고 “Next” 버튼을 클릭함
  - URL mappings의 “/MemRegServlet”을 선택하고 “Edit” 버튼을 사용하여 “/memReg”으로 수정함
  - “Next” 버튼을 다시 클릭하고 “doPost” 메소드만 체크한 후 “Finish” 버튼을 클릭함

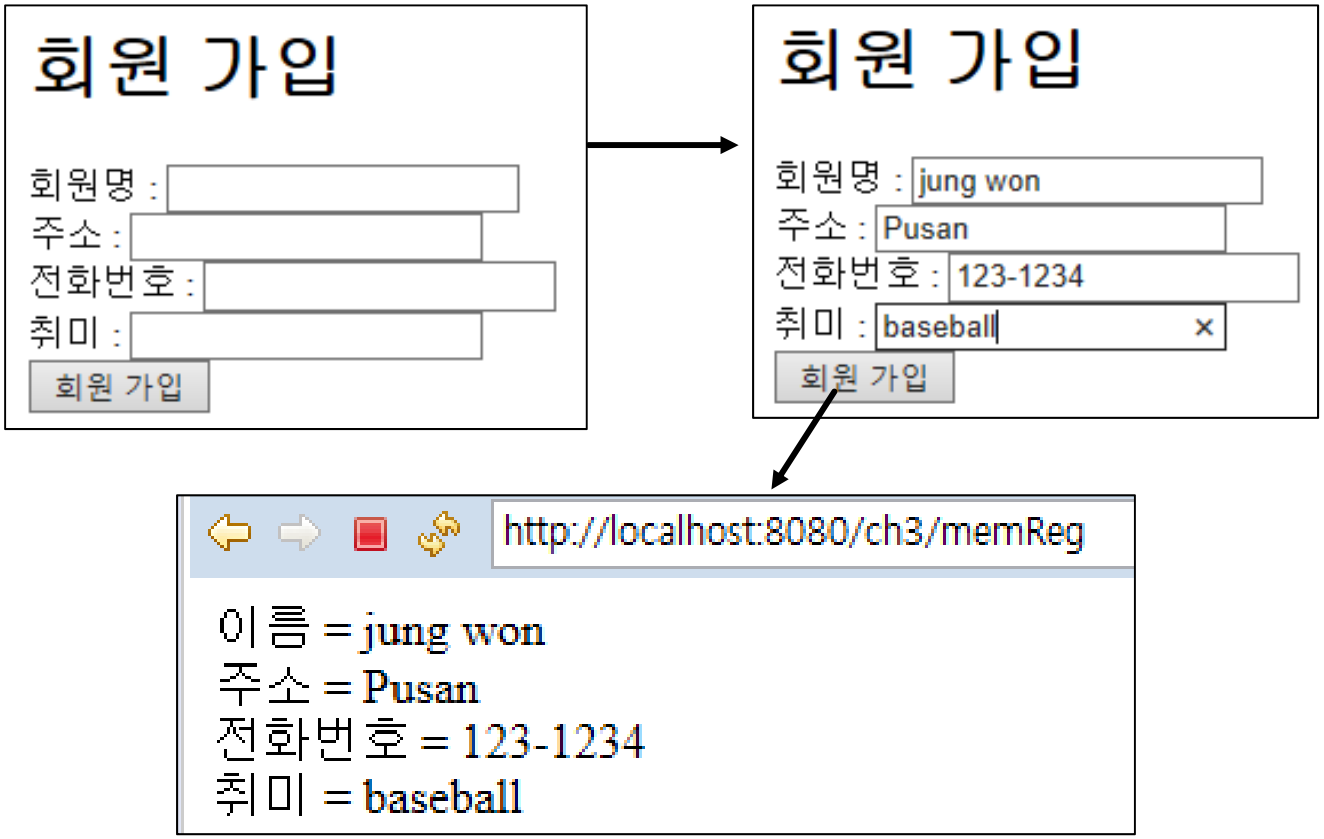
1. 클라이언트에서 서블릿으로 요청하는 방식

- MemRegServlet.java 파일 내용 추가
  - "import java.io.PrintWriter;" 추가
  - 코드 하단의 doPost 부분에 아래 내용 추가

```
protected void doPost(HttpServletRequest request, HttpS
    // TODO Auto-generated method stub
    response.setContentType("text/html;charset=utf-8");
    PrintWriter out = response.getWriter();
    String name=request.getParameter("name");
    String addr = request.getParameter("addr");
    String tel = request.getParameter("tel");
    String hobby = request.getParameter("hobby");
    out.println("이름 = "+name + "<br>");
    out.println("주소 = "+addr + "<br>");
    out.println("전화번호 = "+tel + "<br>");
    out.println("취미 = "+hobby + "<br>");
```

1. 클라이언트에서 서블릿으로 요청하는 방식

- WebContent의 memReg.html 파일에서 마우스 오른쪽 버튼을 클릭하고 "Run As->Run on Server"를 선택함
- Run On Server 창이 뜨면 "Finish"를 클릭함



2. 서블릿에서 한글 처리하기

- Get 방식으로 요청이 전송되어 올 경우
  - Eclipse의 [Window]-[Preference]를 클릭함
  - [Web]-[CSS Files]를 클릭하고 Encoding 방식을 "ISO 10646/Unicode(UTF-8)로 선택한 후 "Apply" 버튼을 클릭하고 "OK" 버튼을 클릭함

2. 서블릿에서 한글 처리하기

- Get 방식으로 요청이 전송되어 올 경우
  - hangul.html 파일 생성

```
7  <body>
8  <form action="hangul" method="get">
9  한글이름 : <input type="text" name="name"><br>
10 <input type="submit" value="확인">
11 </form>
12 </body>
```

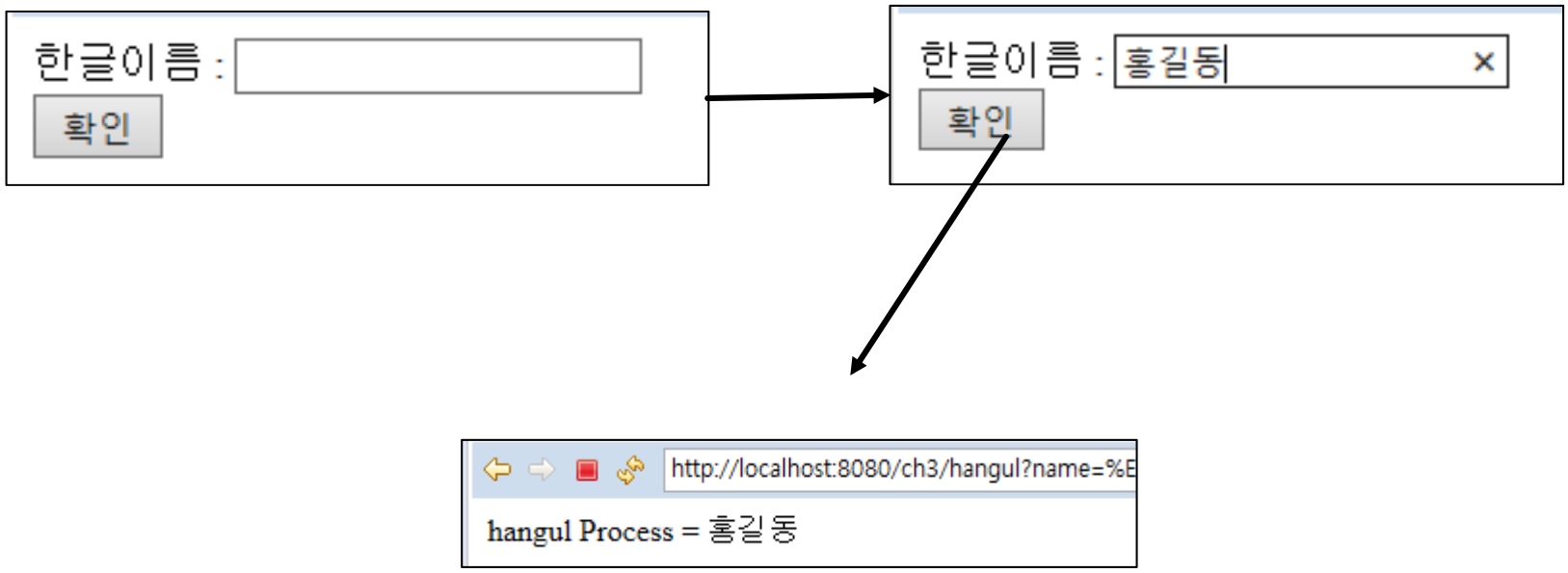
2. 서블릿에서 한글 처리하기

- Get 방식으로 요청이 전송되어 올 경우
  - HangulServlet.java 파일 생성
    - doGet, doPost 모두 선택

```
protected void doGet(HttpServletRequest request, HttpServl
    // TODO Auto-generated method stub
    String name = request.getParameter("name");
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    out.println("hangul Process = " + name);
}
```

2. 서블릿에서 한글 처리하기

- Get 방식으로 요청이 전송되어 올 경우



2. 서블릿에서 한글 처리하기

- Post 방식으로 요청이 전송되어 올 경우
  - hangul.html 파일 수정

```
8  <form action="hangul" method="post">
9  한글이름 : <input type="text" name="name"><br>
0  <input type="submit" value="확인">
```

  - HangulServlet.java 파일 수정

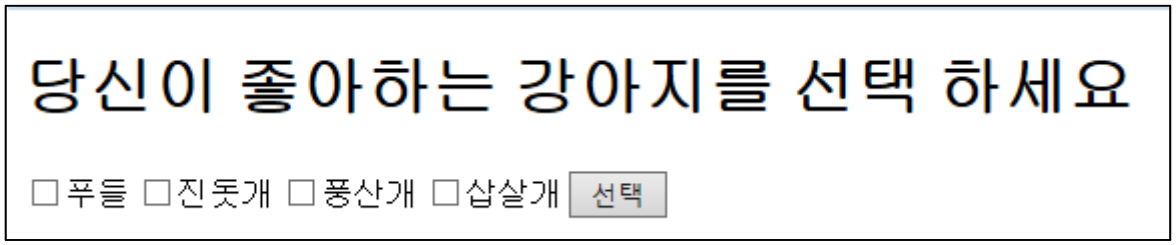
```
protected void doPost(HttpServletRequest request, HttpSe
    // TODO Auto-generated method stub
    request.setCharacterEncoding("UTF-8");
    String name = request.getParameter("name");
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    out.println("hangul Process = " + name);
}
```

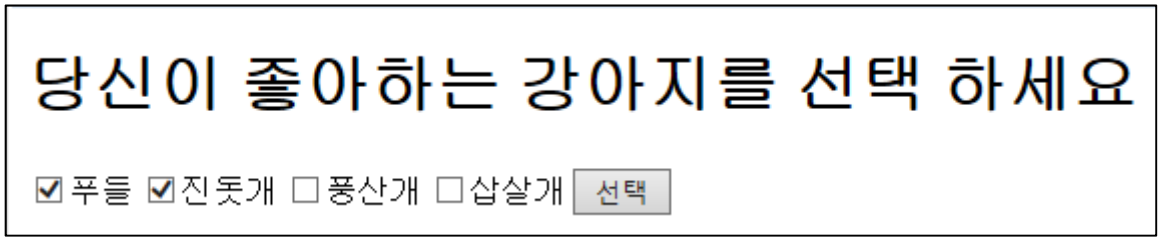
3. 하나의 파라미터 이름으로 여러 개의 파라미터 값이 전송되어 올 경우 처리

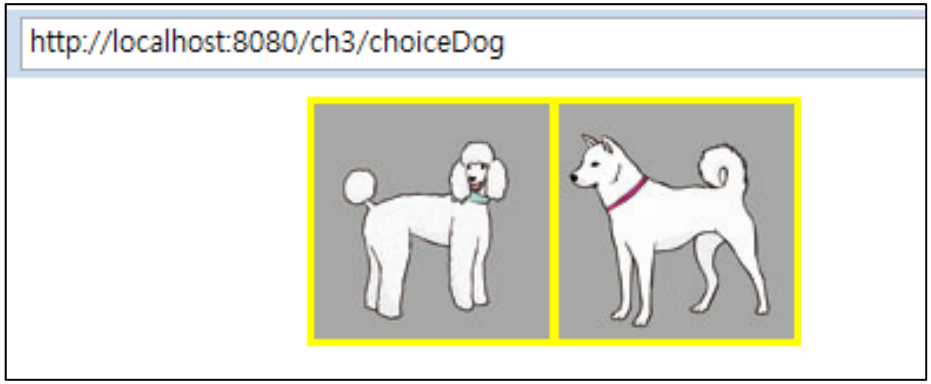
- HttpServletRequest 인터페이스에서 제공되는 String[ ] getParameterValues(String paramName) 메소드 사용

3. 하나의 파라미터 이름으로 여러 개의 파라미터 값이 전송되어 올 경우 처리

▶ 예







3. 하나의 파라미터 이름으로 여러 개의 파라미터 값이 전송되어 올 경우 처리

- dog.html

```
<body>
<h1>당신이 좋아하는 강아지를 선택 하세요</h1>
<form action="choiceDog" method="post">
<input type="checkbox" name="dog" value="pu.jpg"/>푸들
<input type="checkbox" name="dog" value="jin.jpg"/>진돗개
<input type="checkbox" name="dog" value="pung.jpg"/>풍산개
<input type="checkbox" name="dog" value="sap.jpg"/>삽살개
<input type="submit" value="선택"/>
</form>
</body>
```

3. 하나의 파라미터 이름으로 여러 개의 파라미터 값이 전송되어 올 경우 처리

- ChoiceDogServlet.java
  - "import java.io.PrintWriter;" 추가
  - doPost에 추가

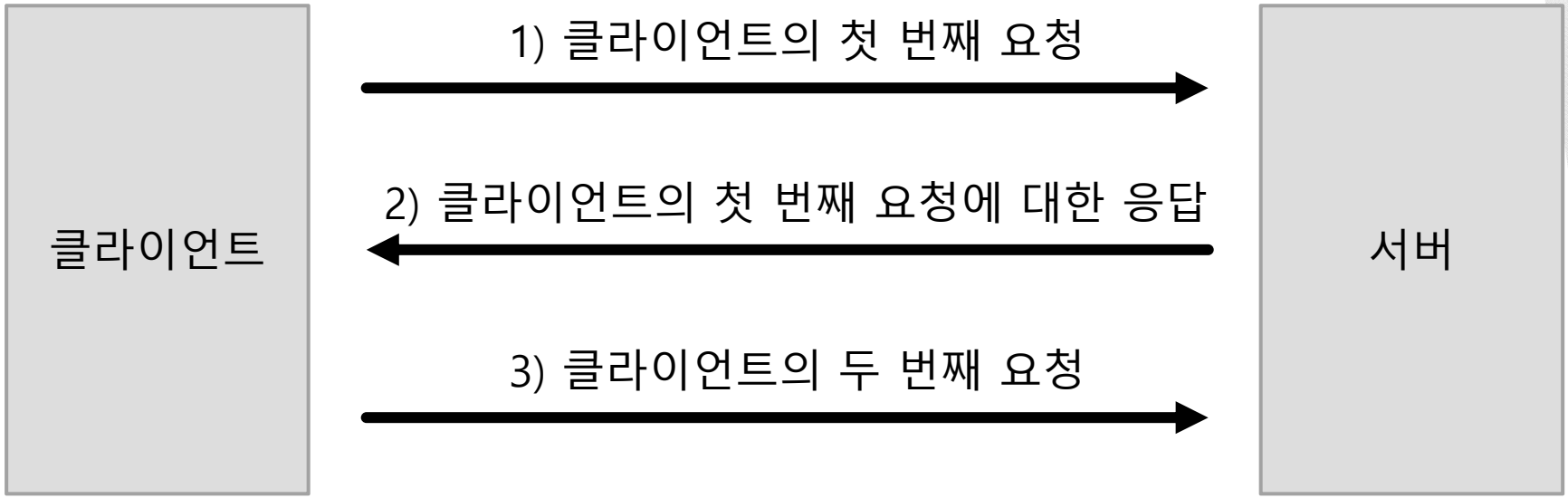
```
response.setContentType("text/html;charset=utf-8");
PrintWriter out = response.getWriter();
String[] dog = request.getParameterValues("dog");
out.println("<html>");
out.println("<head>");
out.println("</head>");
out.println("<body bgcolor='white'>");
out.println("<table align='center' bgcolor='yellow'>");
out.println("<tr>");
for(int i=0;i<dog.length;i++){
out.println("<td>");
out.println("<img src='"+dog[i]+"'/>");
out.println("</td>");
}
out.println("</tr>");
out.println("</table>");
out.println("</body>");
out.println("</html>");
```

4. 서블릿에서 세션 살펴보기

- 세션의 개념
  - HTTP 프로토콜은 상태를 유지하지 않음
    - 클라이언트가 한번 요청을 하고 서버에서 응답을 하면 해당 클라이언트와 서버와의 연결은 유지되지 않음
  - 세션
    - 서블릿에서 클라이언트와 서버의 상태를 유지하기 위해 제공되는 API(Application Programming Interface)

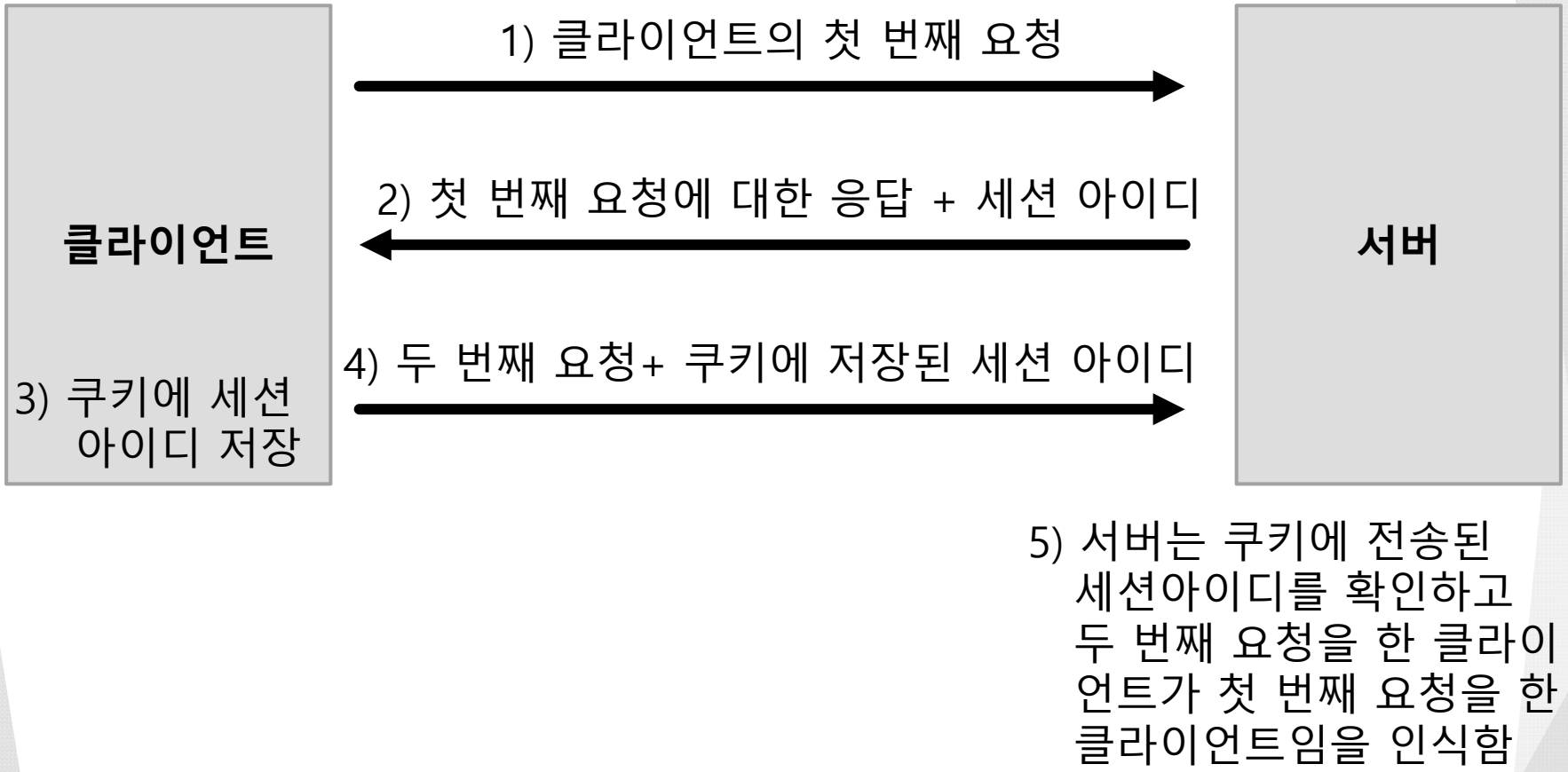
4. 서블릿에서 세션 살펴보기

- HTTP 프로토콜의 상태를 유지하지 않는 특성



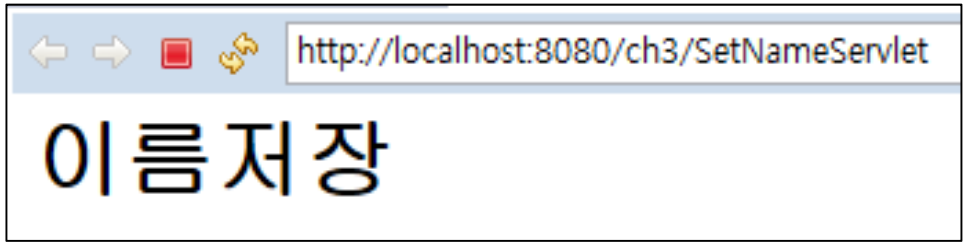
4. 서블릿에서 세션 살펴보기

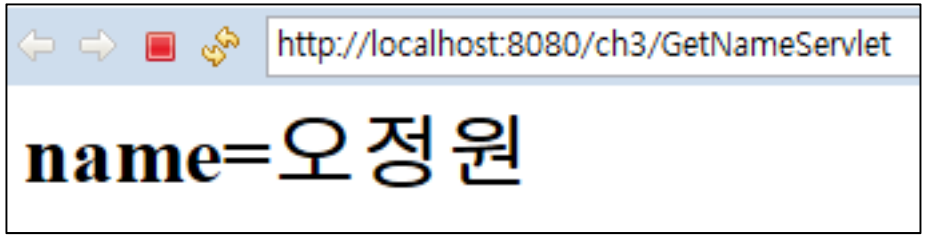
▶ 세션 기능이 적용된 HTTP 프로토콜 요청 처리



4. 서블릿에서 세션 살펴보기

- 예 : SetNameServlet.java / GetNameServlet.java





4. 서블릿에서 세션 살펴보기

- 예 : SetNameServlet.java
  - import java.io.PrintWriter;
  - import javax.servlet.http.HttpSession;
  - doGet 에 추가

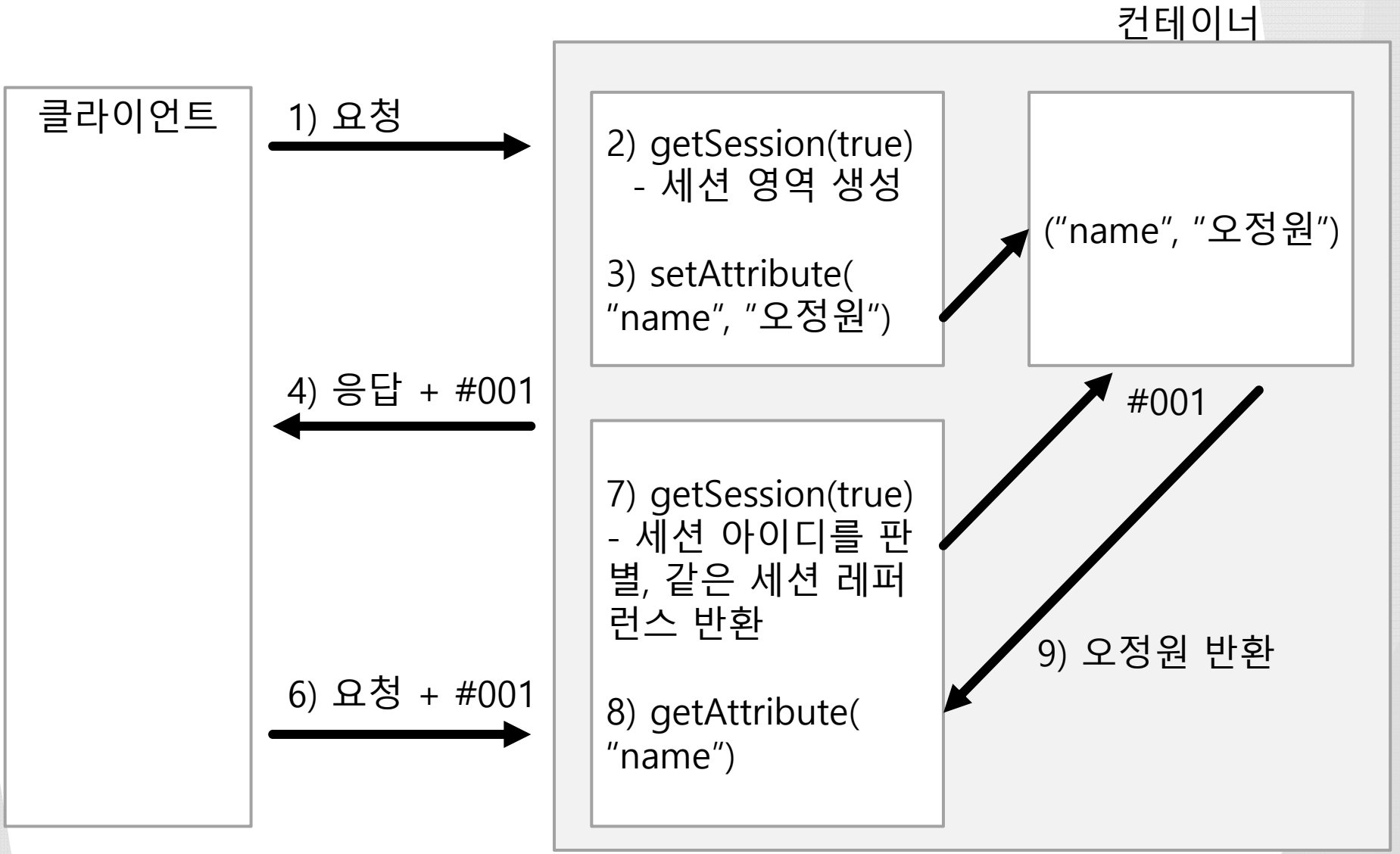
```
HttpSession session = request.getSession();
session.setAttribute("name","오정원");
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
out.println("<h1>이름저장</h1>");
```

4. 서블릿에서 세션 살펴보기

- 예 : GetNameServlet.java
  - import java.io.PrintWriter;
  - import javax.servlet.http.HttpSession;
  - doGet 에 추가

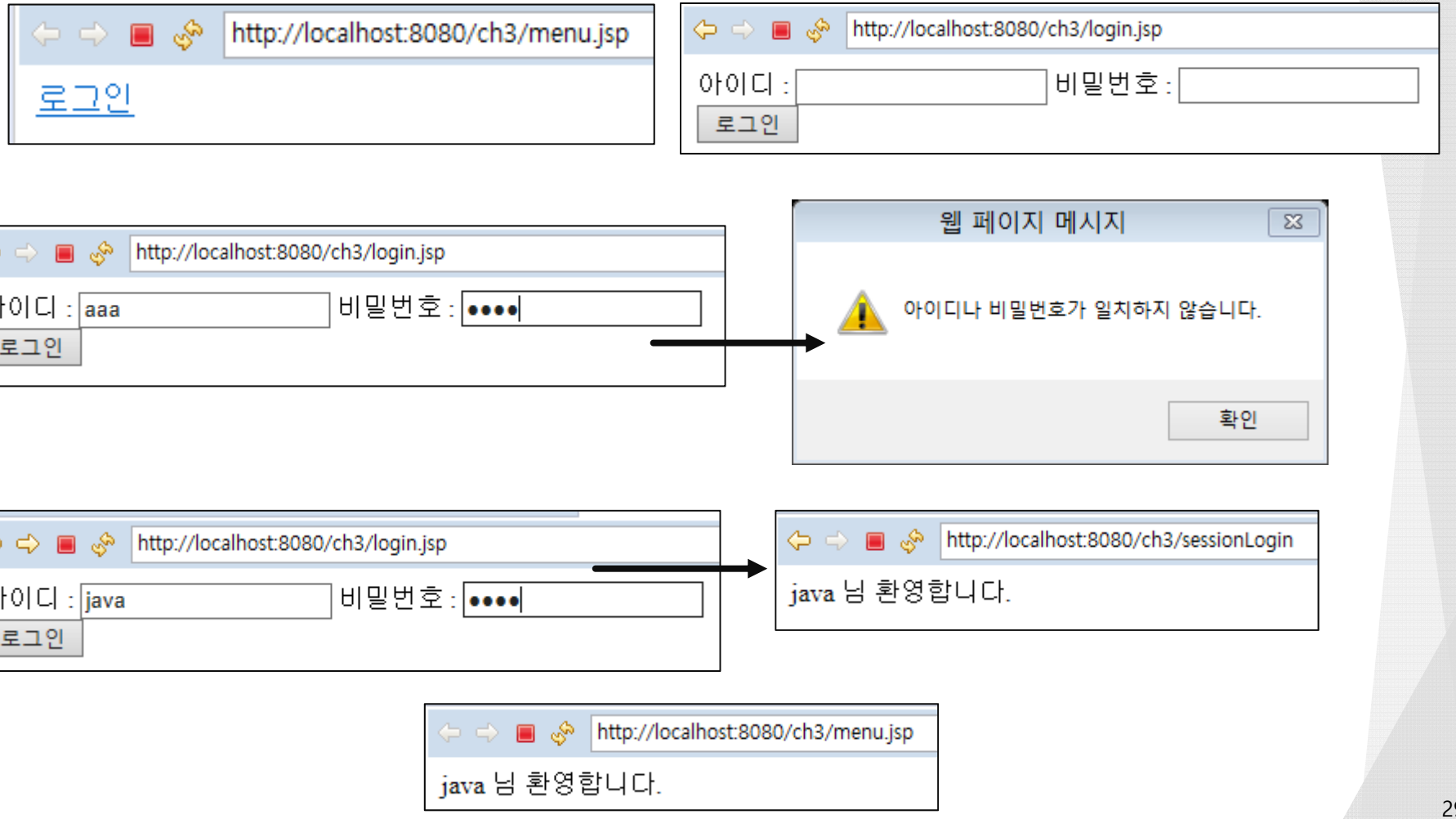
```
HttpSession session = request.getSession();
String name=(String)session.getAttribute("name");
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
out.println("<h1>name="+name+"</h1>");
```

4. 서블릿에서 세션 살펴보기



4. 서블릿에서 세션 살펴보기

- 예 : login.jsp, menu.jsp, SessionLoginServlet.java



4. 서블릿에서 세션 살펴보기

- 예 : login.jsp

```
 9  <body>
10  <form action="sessionLogin" method="post">
11  아이디 : <input type="text" name="id">
12  비밀번호 : <input type="password" name="passwd">
13  <br>
14  <input type="submit" value="로그인">
15  </form>
16  </body>
```

4. 서블릿에서 세션 살펴보기

- 예 : menu.jsp

```
9  <% String id=(String)session.getAttribute("id"); %>
10
11 <body>
12 <%
13     if(id == null) {
14 %>
15 <a href="login.jsp">로그인</a>
16 <%
17     }
18     else{
19 %>
20 <%=id %> 님 환영합니다.
21 <%
22     }
23 %>
24 </body>
```

4. 서블릿에서 세션 살펴보기

- 예 : SessionLoginServlet.java
  - import java.io.PrintWriter;
  - import javax.servlet.http.HttpSession;

4. 서블릿에서 세션 살펴보기

- doPost에 추가

```
request.setCharacterEncoding("utf-8");
response.setContentType("text/html;charset=utf-8");
PrintWriter out = response.getWriter();
String id = request.getParameter("id");
String passwd = request.getParameter("passwd");
if(id.equals("java")&& passwd.equals("1111")){
    HttpSession session = request.getSession();
    session.setAttribute("id", id);
    RequestDispatcher dispatcher =
            request.getRequestDispatcher("menu.jsp");
    dispatcher.forward(request, response);
}
else{
    out.println("<script>");
    out.println("alert('아이디나 비밀번호가 일치하지 않습니다.')");
    out.println("history.back()");
    out.println("</script>");
}
```