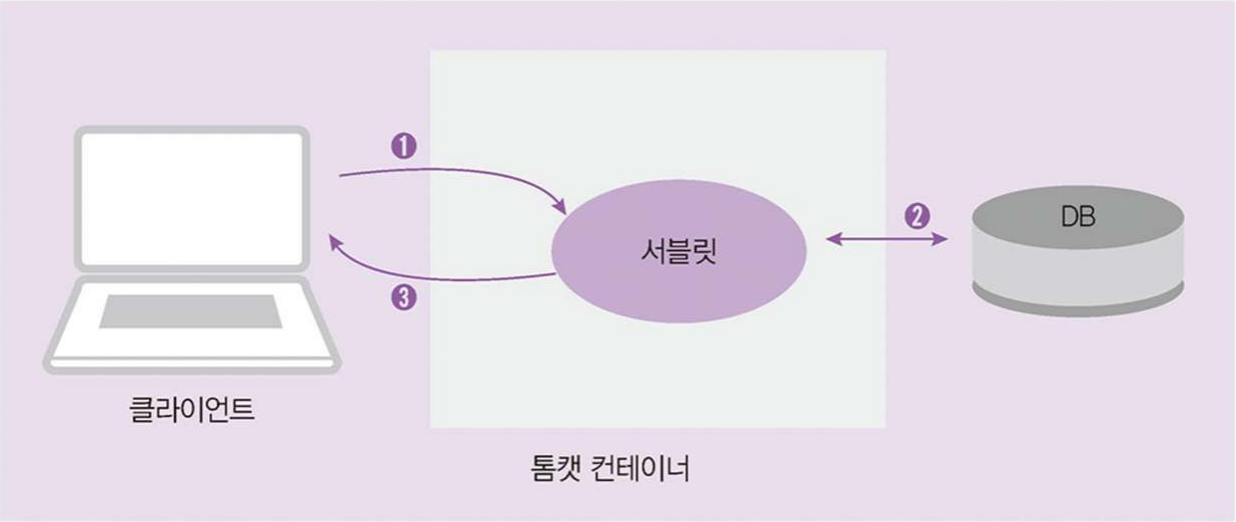
서블릿 기초

서블릿 기본 기능 수행 과정

클라이언트로 부터 요청 받아 비즈니스 로직을 처리하고, 그 결과를 다시 클라이언트에 돌려주는 과정 입니다.



1. 클라이언트의 요청을 받습니다.
2. 데이터베이스 연동과 같은 비즈니스 로직을 처리 합니다.
3. 처리된 결과를 클라이언트에 돌려 줍니다.

서블릿 응답과 요청 수행 API 기능

요청이나 응답과 관련된 API는 모두 **javax.servlet.http**패키지에 있습니다.

- 요청과 관련된 API: `javax.servlet.http.HttpServletRequest` 클래스
- 응답과 관련된 API: `javax.servlet.http.HttpServletResponse` 클래스

HttpServletRequest 메소드

반환형	메서드 이름	기능
boolean	authenticate(HttpServletResponse response)	현재 요청한 사용자가 ServletContext 객체에 대한 인증을 하기 위한 컨테이너 로그인 메커니즘을 사용합니다.
String	changeSessionId()	현재 요청과 연관된 현재 세션의 id를 변경하여 새 세션 id를 반환합니다.
String	getContextPath()	요청한 컨텍스트를 가리키는 URI를 반환합니다.
Cookie[]	getCookies()	클라이어트가 현재의 요청과 함께 보낸 쿠키 객체들에 대한 배열을 반환합니다.
String	getHeader (String name)	특정 요청에 대한 헤더 정보를 문자열로 반환합니다.
Enumeration <String>	getHeaderNames()	현재의 요청에 포함된 헤더의 name 속성을 enumeration으로 반환합니다.
String	getMethod()	현재 요청이 GET, POST 또는 PUT 방식 중 어떤 HTTP 요청인지를 반환합니다.
String	getRequestURI()	요청한 URL의 컨텍스트 이름과 파일 경로까지 반환합니다.
String	getServletPath()	요청한 URL에서 서블릿이나 JSP 이름을 반환합니다.
HttpSession	getSession()	현재의 요청과 연관된 세션을 반환합니다. 만약 세션이 없으면 새로 만들어서 반환합니다.

HttpServletResponse 메소드

반환형	메서드 이름	기능
void	addCookie(Cookie cookie)	응답에 쿠키를 추가합니다.
void	addHeader(String name, String value)	name과 value를 헤더에 추가합니다.
String	encodeURL(String url)	클라이언트가 쿠키를 지원하지 않을 때 세션 id를 포함한 특정 URL을 인코딩합니다.
Collection <String>	getHeaderNames()	현재 응답의 헤더에 포함된 name을 얻어옵니다.
void	sendRedirect(String location)	클라이언트에게 리다이렉트(redirect) 응답을 보낸 후 특정 URL로 다시 요청하게 합니다.
String	getPathInfo()	클라이언트가 요청 시 보낸 URL과 관련된 추가 경로 정보를 반환합니다.

<form> 태그를 이용해 서블릿 요청 하기

<form> 태그로 서블릿에 요청하는 과정

서블릿과 JSP는 HTML, CSS, 자바스크립트에 Java의 기능을 추가하여, 서로 연동하여 동작 합니다. 특히 사용자의 요청은 HTML의 <form>태그나 자바스크립트로 부터 전송 받아서 처리 합니다.

Tip

서블릿/JSP 프로그래밍을 하려면 기본적으로 HTML이나 자바스크립트에 대해 알아두는 것이 좋습니다. 특히 클라이언트에서 서버로 데이터를 전송하는 기능을 담당하는 〈form〉 태그와 〈input〉 태그의 기능은 자주 사용되므로 반드시 익혀 두기 바랍니다.

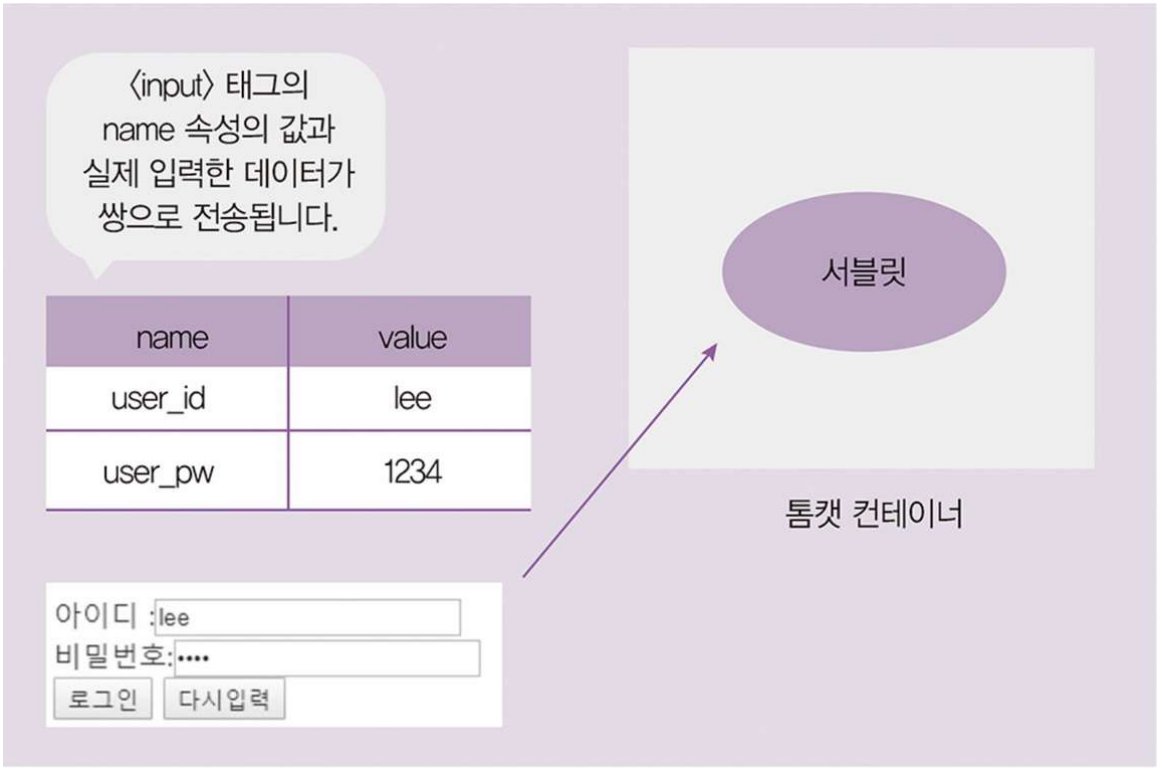
- 웹 프로그램에서 여러 가지 입력 서식을 이용해 전송을 클릭하면 사용자가 입력한 데이터가 서블릿으로 전송 됩니다.

<form> 태그의 여러 가지 속성

사용자 로그인창의 HTML 태그 구조

```
<!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta http-equiv="X-UA-Compatible" content="IE=edge"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Document</title> </head> <body> <form name="frmLogin" method="get" action="login" enctype="UTF-8"> 아이디 :<input type="text" name="user_id"><br> 비밀번호 : <input type="password" name="user_pw"><br> <input type="submit" value="로그인"> <input type="reset" value="다시 입력"> </form> </body> </html>
```

사용자가 자신의 ID와 비밀번호를 입력한 후 로그인을 클릭하면 <form>캐그의 action속성은 데이터를 전송할 서블릿이나 JSP이름을 지정 합니다. 그러면 지정된 이름이 login인 서블릿으로 ID와 비밀번호가 전송 됩니다.



<form>태그와 관련된 속성

속성	기능
name	• <form> 태그의 이름을 지정합니다. • 여러 개의 form이 존재할 경우 구분하는 역할을 합니다. • 자바스크립트에서 <form> 태그에 접근할 때 자주 사용합니다.
method	• <form> 태그 안에서 데이터를 전송할 때 전송 방법을 지정합니다. • GET 또는 POST로 지정합니다(아무것도 지정하지 않으면 GET입니다).
action	• <form> 태그에서 데이터를 전송할 서블릿이나 JSP를 지정합니다. • 서블릿으로 전송할 때는 매핑 이름을 사용합니다.
encType	• <form> 태그에서 전송할 데이터의 encoding 타입을 지정합니다. • 파일을 업로드할 때는 multipart/form-data로 지정합니다.

서블릿에서 클라이언트의 요청을 얻는 방법

메서드	기능
String getParameter(String name)	name의 값을 알고 있을 때 그리고 name에 대한 전송된 값을 받아오는 데 사용합니다.
String[] getParameterValues(String name)	같은 name에 대해 여러 개의 값을 얻을 때 사용합니다.
Enumeration getParameterNames()	name 값을 모를 때 사용합니다.

HttpServletRequest로 요청 처리

로그인창에서 ID와 비밀번호를 입력 받아 HttpServletRequest로 처리하는 간단하는 프로그램 입니다.

1. pro06 새 프로젝트를 생성 합니다. 그리고 톰캣의 servlet-api.jar를 클래스 패스에 지정 합니다.

2. webapp > WEB-INF > login.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>로그인 창</title>
</head>
<body>
  <form name="frmLogin" method="get" action="login" encType="UTF-8"> ❶
    아이디 :<input type="text" name="user_id"><br> ❷
    비밀번호:<input type="password" name="user_pw"><br> ❸
    <input type="submit" value="로그인"> <input type="reset" value="다시입력">
  </form>
</body>
</html>
```

---

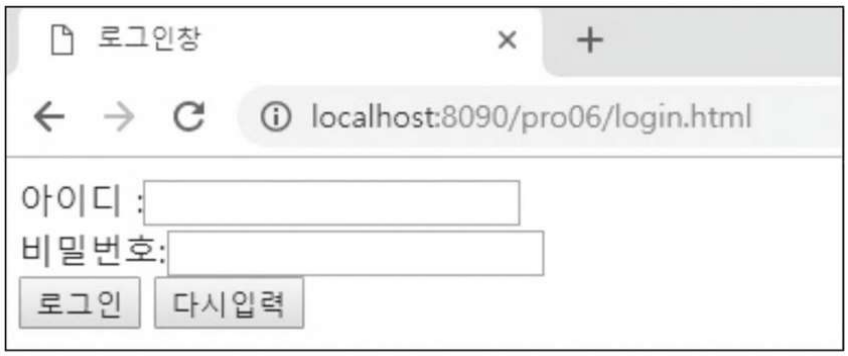
1. 입력된 데이터를 서블릿 매핑 이름이 login인 서블릿으로 전송 합니다.
2. 텍스트 박스에 입력된 ID를 user_id로 전송 합니다.
3. 텍스트 박스에 입력된 비밀번호를 user_pw로 전송 합니다.

---

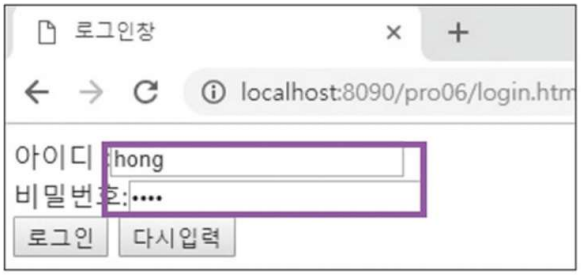
3. sec01.ex01 패키지를 만들고 요청을 받을 서블릿인 LoginServlet 클래스를 생성 합니다.
4. LoginServlet.java 코드 작성 합니다.

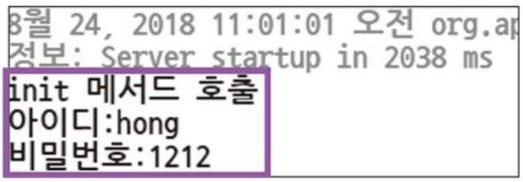
```
package sec01.ex01; import java.io.IOException; import
javax.servlet.ServletConfig; import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet; import
javax.servlet.http.HttpServlet; import
javax.servlet.http.HttpServletRequest; import
javax.servlet.http.HttpServletResponse; @WebServlet("/login") // 서블릿 매
핑 public class LoginServlet extends HttpServlet { public void
init(ServletConfig config) throws ServletException {
System.out.println("init 메소드 호출"); } public void destroy() {
System.out.println("destroy 메소드 호출"); } // 웹브라우저에서 전송한 정보를
톰캣 컨테이너가 HttpServletRequest 객체를 생성 후 goGet()으로 전달 protected
void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException { request.setCharacterEncoding("utf-
8"); // 전송된 데이터를 UTF-8로 인코딩 // getparameter()를 이용해 <input>
태그의 name 속성 값으로 전송된 value를 받아 옴 String user_id =
request.getParameter("user_id"); String user_pw =
request.getParameter("user_pw"); System.out.println("아이디 : " +
user_id); System.out.println("비밀번호 : " + user_pw); } protected void
doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException { doGet(request, response); } }
```

5. http://localhost:8090/pro06/login.html 요청



6. ID와 비밀번호를 입력한 후 로그인을 클릭하면 서블릭이 ID와 비밀번호를 이클립스 콘솔에 출력 합니다.





단, 서블릿이 처리한 후의 응답 기능은 아직 구현하지 않았으므로 웹 브라우저에는 아무것도 출력되지 않습니다.

여러 개의 값을 전송할 때의 요청 처리

한번에 여러개의 입력을 받아 처리하는 예제 입니다.

1. 아래와 같이 input.html을 추가하고 InputServlet클래스를 새로 만듭니다.

2. input.html 작성

```
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>여러 가지
input 타입 표시 창</title> </head> <body> <form name="frmInput"
method="get" action="input"> 아이디 : <input type="text" name="user_id">
<br> 비밀번호 : <input type="text" name="user_pw"><br> <input
type="checkbox" name="subject" value="java" checked>자바 <input
type="checkbox" name="subject" value="C++언어">C++언어 <input
type="checkbox" name="subject" value="JSP">JSP <input type="checkbox"
name="subject" value="안드로이드">안드로이드 <br><br> <input type="submit"
value="전송"> <input type="reset" value="초기화"> </form> </body> </html>
```

❶ name 속성이 모두 subject로 같습니다.
❷ 전송을 클릭하면 매핑 이름이 action에 설정한 input 서블릿으로 전송됩니다.

3. InputServlet 클래스 작성, **getParameterValues()**를 이용해 input.html에서 체크박스의 name인 subject로 전송된 값들을 받아 와서 **문자열 배열에 저장** 합니다.

```
package sec01.ex01; import java.io.IOException; import
javax.servlet.ServletConfig; import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet; import
javax.servlet.http.HttpServlet; import
javax.servlet.http.HttpServletRequest; import
javax.servlet.http.HttpServletResponse; @WebServlet("/input") public
class InputServlet extends HttpServlet { public void init(ServletConfig
config) throws ServletException { System.out.println("init 메소드 호출");
} public void destroy() { System.out.println("destroy 메소드 호출"); }
protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
request.setCharacterEncoding("utf-8"); String user_id =
request.getParameter("user_id"); String user_pw =
request.getParameter("user_pw"); System.out.println("아이디 : " +
user_id); System.out.println("비밀번호 : " + user_pw); String[] subject =
request.getParameterValues("subject"); for(String str : subject) {
System.out.println("선택한 과목 : " + str); } } protected void
doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException { doGet(request, response); } }
```

4. 브라우저에서 http://localhost:8090/pro06/input.html로 요청 합니다.

5. 체크박스에서 어러 개의 값에 체크한 후 전송을 클릭하면 이클립스 콘솔에 해당 과목명이 출력 됩니다.

getParameterNames() 메소드를 이용한 요청 처리

전송된 데이터가 많아 일일이 name의 값을 기억하기 힘들 때는 getParameterNames() 메소드를 이용하면 편리 합니다.

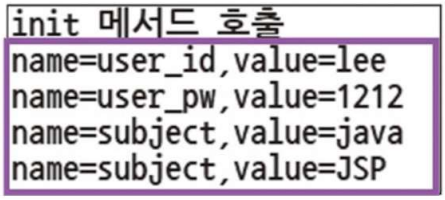
1. InputServlet2 클래스 생성
2. input.html 수정 (매핑 이름을 input2로 수정)

```
<body> <form name="frmInput" method="get" action="input2"> 아이디 : <input type="text" name="user_id"><br> 비밀번호 : <input type="text" name="user_pw"><br> <input type="checkbox" name="subject" value="java" checked>자바 <input type="checkbox" name="subject" value="C++언어">C++언어 <input type="checkbox" name="subject" value="JSP">JSP <input type="checkbox" name="subject" value="안드로이드">안드로이드 <br><br> <input type="submit" value="전송"> <input type="reset" value="초기화"> </form> </body>
```

3. InputServlet2 클래스 작성 (전송되는 데이터가 많은 경우에는 getParameterNames()를 이용해 name 속성만 따로 구할 수 있습니다.)

```
package sec01.ex01; import java.io.IOException; import
java.util.Enumeration; import javax.servlet.ServletException; import
javax.servlet.annotation.WebServlet; import
javax.servlet.http.HttpServlet; import
javax.servlet.http.HttpServletRequest; import
javax.servlet.http.HttpServletResponse; @WebServlet("/input2") public
class InputServlet2 extends HttpServlet { public void init() throws
ServletException { System.out.println("init 메소드 호출"); } public void
destroy() { // TODO Auto-generated method stub } /** * @see
HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response) */ protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
request.setCharacterEncoding("utf-8"); Enumeration enu =
request.getParameterNames(); while(enu.hasMoreElements()) { String name =
(String) enu.nextElement(); String[] values =
request.getParameterValues(name); for(String value : values) {
System.out.println("name=" + name + ", value=" + value); } } } /** * @see
HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response) */ protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException { //
TODO Auto-generated method stub doGet(request, response); } }
```

- 전송되어 온 name 속성들만 Enumeration 타입으로 받아 옵니다.
- 각 name을 하나씩 가져와 대응해서 전송되어 온 값을 출력 합니다.
- 참고 : 매핑으로 동작이 안되는 경우는 html의 이름을 다르게 하면 된다.



서블릿의 응답 처리 방법

서블릿에서 응답을 처리하는 방법은 다음과 같습니다.

- **doGet()**이나 **doPost()**메소드 안에서 처리 합니다.
- **javax.servelet.http.HttpServletResponse** 객체를 이용 합니다.
- **setContentType()**을 이용해 클라이언트에게 전송할 데이터 종류를 지정 합니다.
- 클라이언트와 서블릿의 통신은 자바 I/O 스트림을 이용 합니다.

서블릿의 응답 처리는 doGet()이나 doPosty() 메소드의 두 번째 매개변수인 HttpServletResponse 객체를 이용하여 처리 합니다. 그리고 웹 브라우저와 서블릿의 응답 과정은 자 I/O 기능인 스트림을 이용하여 이루어 집니다.

MIME-TYPE

서버에서 웹 브라우저로 데이터를 전송 할 때 어떤 데이터를 전송하는지 웹 브라우저에 알려줘야 합니다.

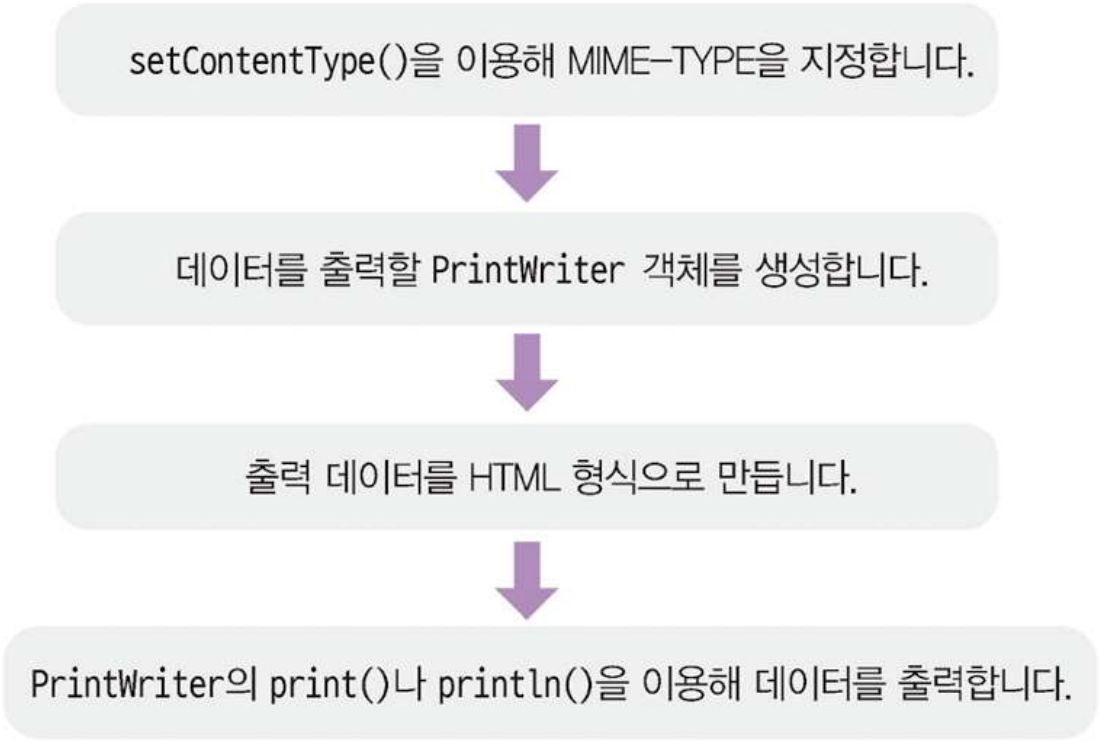
그 이유는 웹 브라우저가 전송 받을 데이터의 종류를 미리 알고 있으면 더 빠르게 처리 할 수 있습니다.

서버에서 웹 브라우저로 데이터를 전송 할 때는 톰켓 컨테이너레서 미리 제공하는 여러 가지 전송 데이터 종류 중 하나를 지정해서 웹 브라우저로 전송 합니다. 이 처럼 톰켓 컨테이너에서 미리 설정해 놓은 데이터 종류들을 **MIME-TYPE**이라고 합니다.

- HTML로 전송 시 : text/html
- 텍스트로 전송 시 " text
- XML 전송 시 : application/xml

HttpServletResponse를 이용한 서블릿 응답 실습

사용자가 입력한 ID와 비밀번호를 전송하면 서블릿이 다시 브라우저에게 응답하는 예제



1. login.html을 수정 합니다.

로그인창세어 ID와 비밀번호를 입력한 후 login2서블릿으로 전송

```
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>로그인 창
</title> </head> <body> <form name="frmLogin" method="post"
action="login2" encType="UTF-8"> 아이디 : <input type="text"
name="user_id"><br> 비밀번호 : <input type="password" name="user_pw"><br>
<input type="submit" value="로그인"> <input type="reset" value="다시입력">
</form> </body> </html>
```