제4장
Servlet-Ⅲ

1. 자바스크립트로 서블릿에 요청하기

실무에선 자바스크립트에서 먼저 입력한 값에 대해서 유효성 검사를 한 후 자바스크립트에서 서블릿에 요청함.

다음과 같이 login.html을 작성합니다. 자바스크립트 함수에서 <form> 태그에 접근하여 값 입력 여부를 체크한 후 action 속성에 전송할 서블릿 이름을 지정함.

```
<script type="text/javascript" >
    function fn_validate(){
        var frmLogin=document.frmLogin;
        var user_id=frmLogin.user_id.value;
        var user_pw=frmLogin.user_pw.value;

        if((user_id.length==0 ||user_id=="") ||(user_pw.length==0 ||user_pw=="")){
          alert("아이디와 비밀번호는 필수입니다.");
        }
        else{
          frmLogin.method="post";
          frmLogin.action="login5";
          frmLogin.submit();
        }
    }
</script>
```

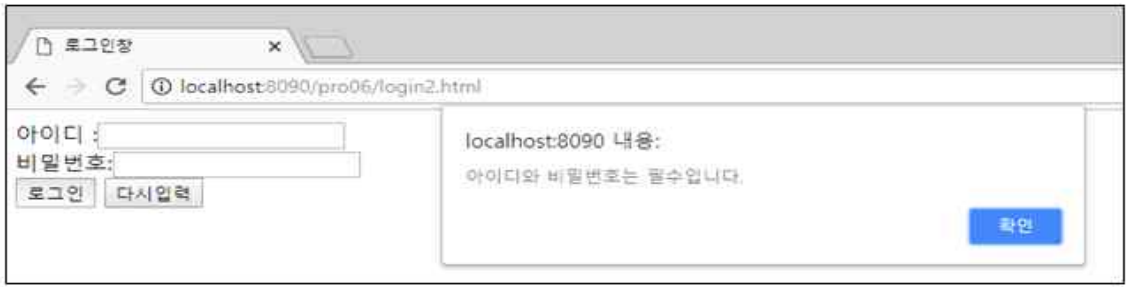
1. 자바스크립트로 서블릿에 요청하기

LoginServlet5 클래스를 다음과 같이 작성합니다.

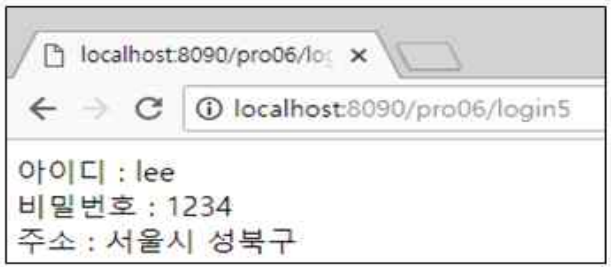
```
String id = request.getParameter("user_id");
String pw = request.getParameter("user_pw");
String address = request.getParameter("user_address");

String data = "<html>";
data += "<body>";
data += "아이디 : " + id;
data += "<br>";
data += "비밀번호 : " + pw;
data += "<br>";
data += "주소 : " + address;
data += "</html>";
data += "</body>";
out.print(data);
```

1. 자바스크립트로 서블릿에 요청하기

http://localhost:8090/login.html로 요청합니다. ID와 비밀번호를 입력하지 않고 로그인을 클릭하면 오류 창이 나타납니다.



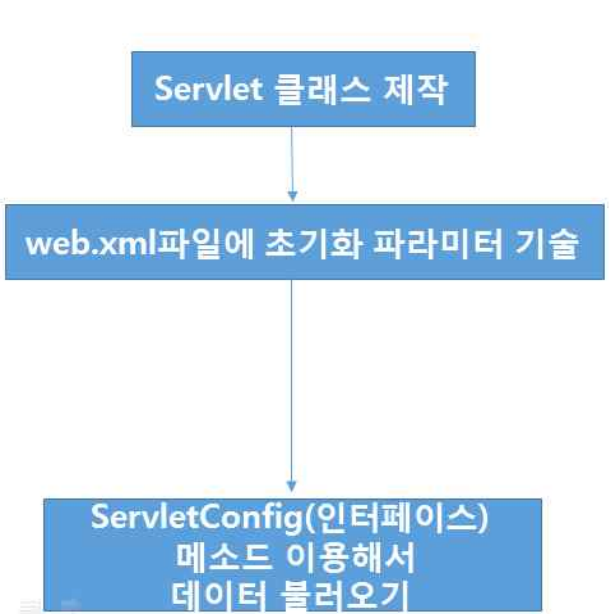
아이디와 비밀번호를 정상적으로 입력한 경우



2. 서블릿 초기화 파라미터 : ServletConfig

특정 Servlet이 생성될 때 초기에 필요한 데이터들이 있다. 예를 들어 특정 경로 및 아이디 정보 등 이다.
이러한 데이터들을 초기화 파라미터라고 하며, web.xml에 기술(첫 번째 방법)하고 servlet파일에서는 servletConfig 클래스를 이용해서 접근(사용)한다.또한 초기화 파라미터를 web.xml이 아닌 servlet파일에 직접 기술(두 번째 방법)하는 방법도 살펴본다.

web.xml파일에 초기화 파라미터(Initialization Parameter) 기술




```
<servlet>
  <servlet-name>ServletInitParam</servlet-name>
  <servlet-class>com.javalec.ex.ServletInitParam</servlet-class>
```

← 서블릿 지정

```
  <init-param>
     <param-name>id</param-name>
     <param-value>abcdef</param-value>
  </init-param>
  <init-param>
     <param-name>pw</param-name>
     <param-value>1234</param-value>
  </init-param>
  <init-param>
     <param-name>path</param-name>
     <param-value>C:₩₩javalec₩₩workspace</param-value>
  </init-param>
```

** 특정 servlet파일을 생성할 때, 초기에 필요한 값을 설정하는데 web.xml파일에 init-param태그를 이용해서 Servlet파일에서 가져와서 쓸 수 있는 방법이다.
다시 말하지만, 특정 Servlet파일에 국한 되어 있는 것이다.

```
</servlet>
<servlet-mapping>
  <servlet-name>ServletInitParam</servlet-name>
  <url-pattern>/InitParam</url-pattern>
</servlet-mapping>
```

← 서블릿 매핑

```
String id = getInitParameter("id");
String pw = getInitParameter("pw");
String path = getInitParameter("path");
```

← web.xml에 지정된 값 읽어오기
ServletConfig인터페이스에 getInitParameter()가 있다.

2. 서블릿 초기화 파라미터 : ServletConfig

Servlet파일에 초기화 파라미터(initialization Parameter) 기술

Servlet 클래스 제작

↓

@WebInitParam에 초기화 파라미터 기술

```
@WebServlet(urlPatterns={"/ServletInitParam"}, initParams={@WebInitParam(name="id", value="abcdef"), @WebIr
```

↓

ServletConfig 메소드 이용해서 데이터 불러오기

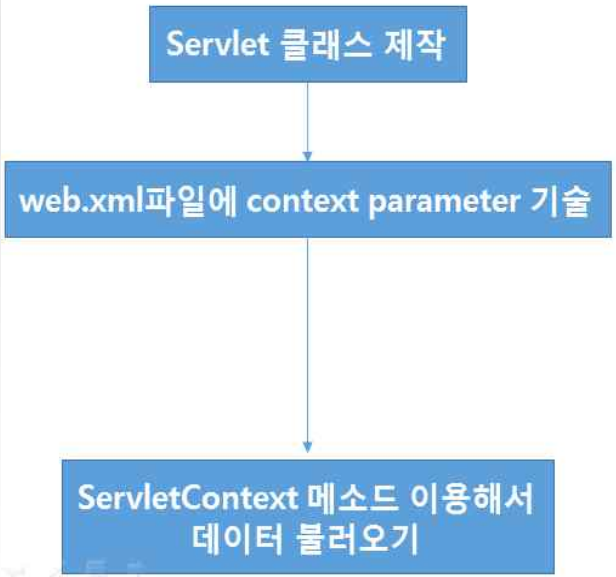
```
String id = getInitParameter("id");
String pw = getInitParameter("pw");
String path = getInitParameter("path");
```

** 앞선 장에서는 web.xml을 이용하여 특정 서블릿 파일에 파라메터를 초기화 하였다. 이번에는 직접 어노테이션을 이용하여 적용하는 부분이다.
이 부분이 적용되기 위해서는 당연히 web.xml이 주석처리가 되어야 할 것이다.

3. 데이터 공유 : ServletContext

여러 Servlet에서 특정 데이터를 공유해야 할 경우 context parameter를 이용해서 web.xml에 데이터를 기술하고, Servlet에서 공유하면서 사용할 수 있다.

web.xml파일에 context parameter 기술




```
<context-param>
  <param-name>id</param-name>
  <param-value>abcdef</param-value>
</context-param>
<context-param>
  <param-name>pw</param-name>
  <param-value>1234</param-value>
</context-param>
<context-param>
  <param-name>path</param-name>
  <param-value>C:₩javalec₩workspace</param-value>
</context-param>
```

모든 서블릿이 쓸 수 있도록 하게 하는 context-param태그는 servlet태그와 별도로 만들어준다.

```
String id = getServletContext().getInitParameter("id");
String pw = getServletContext().getInitParameter("pw");
String path = getServletContext().getInitParameter("path");
```

** 이번 부분은 앞에서는 특정 서블릿에 국한되어진 파라메터 초기화 방법이었지만, 지금은 context-param태그를 이용하여 여러 서블릿에서 함께 공유해서 사용하는 방법이다.

** 중요함.
초기화 파라메터를 가져다 쓰는 방법
1. 특정 서블릿의 경우 : initParam
2. 모든 서블릿의 경우 : contextParam

4. 웹 어플리케이션 감시 : ServletContextListener

웹어플리케이션의 생명주기(LifeCycle)를 감시하는 리스너(Listener)가 있다. 바로 ServletContextListener이다.
리스너의 해당 메소드가 웹 어플리케이션의 시작과 종료시 호출 된다.(**contextInitialized()**, **contextDestroyed()**)

첫 번째 방법 : web.xml파일에 리스너 클래스 기술

리스너 클래스를
따로 제작
(아울러,ServletContextListener를 구현해야 함)

```
public class ContextListenerEx implements ServletContextListener{

    public ContextListenerEx() {
        // TODO Auto-generated constructor stub
    }

    @Override
    public void contextDestroyed(ServletContextEvent arg0) {
        // TODO Auto-generated method stub
        System.out.println("contextDestroyed");
    }

    @Override
    public void contextInitialized(ServletContextEvent arg0) {
        // TODO Auto-generated method stub
        System.out.println("contextInitialized");
    }
```

Listener클래스를 제작할 때는 반드시 ServletContextListener인터페이스를 구현하고 2개의 추상메서드를 오버라이딩하여야 한다는 것을 명심하자.

web.xml파일에 리스너 클래스 기술

```
<listener>
  <listener-class>com.javalec.ex.ContextListenerEx</listener-class>
</listener>
```

4. 웹 어플리케이션 감시 : ServletContextListener

두 번째 방법 : 리스너 클래스에 기술(@WebListener)


```
@WebListener
public class ContextListenerEx implements ServletContextListener{
}
```

감사합니다.