**Servlet** VS **JSP**

**<Servlet>**

* 웹 기반의 요청에 대한 동적인 처리가 가능한 Server Side에서 돌아가는 Java Program

=> data processing(Controller)에 좋음(데이터를 읽고 확인하는 작업)

* Java 코드 안에 HTML 코드 (하나의 클래스)
* 웹 개발을 위해 만든 표준
* Servlet이 수정된 경우 Java 코드를 컴파일(.class 파일 생성)한 후 동적인 페이지를 처리하기 때문에 전체 코드를 업데이트하고 다시 컴파일한 후 재배포하는 작업이 필요 (개발 생산성 저하)

**<ex>**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public ThreeParams extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

response.setContentType("text/html");

printWriter out = response.getWriter();

String title = "Reading Three Request Parameters";

String docType = "<!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">\n";

out.println(docType +

"<HTML>\n" +

"<HEAD><TITLE>" + title + "</TITLE></HEAD>\n" +

"<BODY BGCOLOR=\"#FDF5E6\">\n" +

"<H1 ALIGN=\"CENTER\">" + title + "</H1>\n" +

"<UL>\n" +

"<LI><B>param1</B>: " + request.getParameter("param1") + "\n" +

"<LI><B>param2</B>: " + request.getParameter("param2") + "\n" +

"<LI><B>param3</B>: " + request.getParameter("param3") + "\n" +

"</UL>\n" +

"</BODY></HTML>");

)

}

}

**<JSP>**

* Java 언어를 기반으로 하는 Server Side 스크립트 언어

=> Presentation(View)에 좋음

* HTML 코드 안에 Java 코드
* JSP가 수정된 경우 재배포할 필요가 없이 WAS가 알아서 처리 (쉬운 배포)
* Servlet를 보완하고 기술을 확장한 스크립트 방식 표준

=> Servlet의 모든 기능 + 추가적인 기능

=> all features of servlet + (Implicit objects / predefined tags / expression language / custom tags

<예시>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<HTML>

<HEAD>

<TITLE>Reading Three Request Parameters</TITLE>

<LINK REL=STYLESHEET HREF="JSP-Styles.css" TYPE="text/css">

</HEAD>

<BODY>

<H1>Reading Three Request Parameters</H1>

<UL>

<LI><B>param1</B>: <%= request.getParameter("param1") %>

<LI><B>param2</B>: <%= request.getParameter("param2") %>

<LI><B>param3</B>: <%= request.getParameter("param3") %>

</UL>

</BODY>

</HTML>

**<Servlet과 JSP의 관계>**

* JSP와 Servlet을 모두 이용하는 모델 (MVC Architecture)
  + JSP와 Servlet을 모두 사용하여 프레젠테이션 로직(View)과 비즈니스 로직(Controller)을 분리
  + View(보여지는 부분)는 HTML이 중심이 되는 JSP를 사용
  + Controller(다른 자바 클래스에 데이터를 넘겨주는 부분)는 Java 코드가 중심이 되는 Servlet을 사용
  + Model은 Java Beans로, DTO와 DAO를 통해 Mysql과 같은 Data Storage에 접근
* JSP만 이용하는 모델
* JSP가 사용자의 요청을 받아 Java Bean(DTO, DAO)을 호출하여 적절한 동적인 페이지를 생성
* 동작 과정

1. JSP로 작성된 프로그램은 내부적으로 WAS에서 Servlet 파일로 변환
2. JSP 태그를 분해하고 추출하여 다시 순수한 HTML 웹 페이지로 변환
3. 클라이언트로 응답

* 특징

1. 개발 속도가 빠름
2. 배우기 쉬움
3. 프레젠테이션 로직(View)과 비즈니스 로직(Controller)이 혼재함
4. JSP 코드가 복잡해져 유지 보수가 어려워짐ㄴ