|  |
| --- |
| JSP/Servlet 공부 내용 정리  2015년 5월 8일 |
| 통합ICTO팀 이영주 사원 |

**JSP** 웹 애플리케이션을 지원하는 Java 지원 기술. JSP = HTML + JSP 태그(스크립트릿)

**웹 서버** 클라이언트의 요청을 받으면 해당 처리를 하여 클라이언트에 결과 페이지를 응답해줄 수 있도록 구성된 서버 또는 그 서버가 설치된 컴퓨터

**웹 애플리케이션 서버** 웹 서버만으로는 데이터베이스와 연동되어 처리되어야 하는 동적인 데이터들을 처리해낼 수 없기 때문에 DBMS와 웹 서버 사이에서 제어를 해주는 프로그램

**doGet 호출 사례**

1. 주소 입력란에서 직접 요청을 하는 경우
2. form 태그의 method 속성 값을 get으로 주거나 생략할 경우
3. <a> 태그에 의해서 하이퍼링크가 걸린 경우

**서블릿과 JSP의 차이점** 서블릿은 자바 클래스 내부에서 자바 코드가 기본이 되고 출력 스트림을 통해 HTML 코드를 문자열 형태로 기술하여 클라이언트에 전송하고 JSP는 HTML 코드가 기본이 되고 스크립트릿 안에 자바 코드를 기술

**인코딩** 인코딩은 문자셋을 컴퓨터가 이해할 수 있는 바이트와의 매핑 규칙. 문자셋을 어떻게 매핑하느냐에 따라 하나의 문자셋이 다양한 인코딩을 가질 수 있다. 따라서 한글을 표현하기 위해서는 그에 맞는 인코딩을 적어줘야 한다.

**localhost** 자신의 컴퓨터를 지칭하는 도메인 네임. IP 주소 : 127.0.0.1

**Tomcat** Apache Software Foundation에서 개발한 서블릿/JSP 컨테이너

**한글 깨짐 방지**

1. 클라이언트에게 전송할 메시지 중 한글 깨짐을 방지
   * response.setContentType(“text/html; charset=UTF-8”); 또는
   * response.setCharacterEncoding("UTF-8");
2. Get 방식일 때(입력 폼에서 입력된 한글을 깨지지 않도록)
   * servel.xml 파일 첫번째 <Connector> 태그에 속성 추가
   * URIEncoding=”UTF-8” 추가
3. Post 방식일 때
   * 서블릿 파일에서 doPost 메서드 수정
   * request.setCharacterEncoding(“UTF-8”);

**서블릿 라이프사이클**

1. init()
   * 메서드에 의한 초기화, web.xml 파일이 실행되면서 최초 접속 클라이언트에 의해 실행된다. 단 한번 호출
2. doGet() 혹은 doPost()
   * 클라이언트의 요청 시 호출. 여러 클라이언트가 동시에 요청을 하더라도 스레드가 생성되므로 동시에 doGet() 혹은 doPost() 메소드가 실행되기 때문에 수행 속도가 빠르다.
3. destroy()
   * 서블릿 컨테이너가 종료(톰캣을 재가동)되거나 서블릿 내용이 변경되어 다시 컴파일해서 클래스 파일이 바뀌는 경우 실행. 메모리를 해제한다.

**서블릿을 요청할 때 클래스 이름을 직접 사용하지 않고 URL 매핑하는 이유는** 개발자가 서블릿 이름을 바꾸더라도 사용자는 동일한 URL로 접근할 수 있도록 하기 위함이다.

**웹 어플리케이션의 클라이언트 측 기술** **: 자바스크립트(Javascript)** HTML의 정적인 페이지에 동적인 기능을 부여하는 기술

**쿼리 스트링 전달 방식**

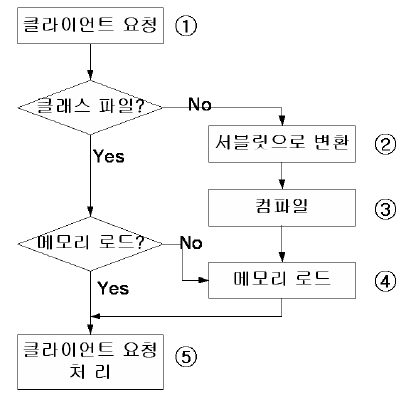
1. 영문자 대소문자와 숫자는 그대로 전달되며 변환되지 않음
2. 공백은 + 문자로 전달
3. 여러 개의 name=value 들은 & 기호로 구분
4. 한글은 % 기호와 함께 16진수로 변환되어 전달

**JSP** Java Server Page의 약어이고, 인터넷을 사용하는 사용자에게 정보를 제공하기 위해서는 HTML 태그를 사용하여 웹 브라우저 내에 정보를 표시하는데 JSP에서는 이러한 HTML 태그를 사용할 수 있어 웹 애플리케이션의 프레젠테이션 역할을 한다. JSP는 JSP에서만 사용 가능한 다양한 태그(예를 들면 스크립트릿과 같은 것)를 이용하여 쉽게 웹 프로그래밍을 할 수 있도록 한다.

**서블릿 컨테이너의 JSP 처리 과정**

1. 브라우저에서 JSP 페이지(“myCount.jsp”)를 웹 서버에게 요청
2. 웹 서버는 JSP에 대한 요청을 서블릿 컨테이너에게 넘긴다. 웹 컨테이너는 해당 JSP를 찾아서 서블릿 파일(myCount\_jsp.java)을 생성
3. 서블릿 파일은 컴파일되어 서블릿 클래스 파일(myCount\_jsp.class)을 생성
4. 컴파일된 서블릿 클래스 파일을 웹 서버로 보낸다.
5. 사용자는 서블릿의 출력 결과인 HTML 형태로 응답 결과를 보게 된다.

**웹 컨테이너가 JSP 페이지를 처리하는 순서도**



**JSP의 지시자(directive) 태그로 처리하는 일들**

* JSP가 Servlet으로 변환될 때 지정되는 부모 클래스 설정 => extends 속성
* HttpSession 객체 생성여부 => session 속성
* Content-type 설정 => contentType 속성

**Action Tag 종류와 기능**

|  |  |
| --- | --- |
| 태그의 종류 | 설명 |
| <jsp:forward> | 다른 사이트로 이동할 때 사용  페이지의 흐름을 제어할 때 사용. |
| <jsp:include> | 정적 혹은 동적인 자원을 현재 페이지의 내용에 포함시킴  페이지를 모듈화할 때 사용 |
| <jsp:param> | 전달인자를 추가할 때 사용 |
| <jsp:useBean> | Bean을 생성하고 사용하기 위한 환경을 정의 |
| <jsp:setProperty> | Bean에서 속성 값을 할당할 때 사용 |
| <jsp:getProperty> | Bean에서 속성 값을 얻어올 때 사용 |

**JSP 내장 객체**

|  |  |
| --- | --- |
| 내장 객체 | 설명 |
| request | 웹 서비스를 위한 클라이언트와 웹 서버 사이의 요청에 관련된 정보는 request 객체에 저장  되어 관리 |
| response | 클라이언트에 대한 응답 처리를 하는 객체인 response는 실행결과를 브라우저로 되돌려 줄  때 사용하는 내장 객체 |
| out | 서버에서 클라이언트로 열려있는 출력 스트림을 의미 |
| page | 하나의 JSP 페이지를 처리할 때 해당 영역을 지시할 경우 사용 |
| config | 서블릿 관련 환경 설정과 관련된 객체 |
| session | 세션과 관련된 속성과 메소드를 재공해주는 객체 |
| application | 하나의 웹 애플리케이션을 관리하고 웹 애플리케이션 안에서의 자원을 공유 |
| pageContext | 외부 환경 정보를 제공하는 객체 |
| exception | 예외 처리를 위한 객체 |