54일차 강의

* 깃허브 사용법
* Only Servelet 방식을 적용한 화면 구현 방식의 완성본 (web04\_227)
* MVC 패턴을 적용한 화면 구현 방식(web05\_280)
* MVC 패턴으로 여태 한 것을 코드를 분리해보자!
* 실행에는 순서가 있다
* Jsp 파일에서 실행시키면 안된다
* 컨트롤러부터 가야되는 것 (자바파일?)
* 모델
  + VO 모델
    - 데이트를 전송하는 트랜스퍼로 기능한다
    - Validation check를 가지고 있음
  + DTO 모델
    - 데이트를 전송하는 트랜스퍼로 기능한다
    - Validatoin check 하지 않음
    - 일체의 가공을 하지 않은 형태

(getter setter 패턴만 있는)

* 변수명 선택이 아주 중요하다
  + - 일관성
* MemberDTO 를 통해 데이터 베이스를 관리한다
* 왜 ArrayList로 만들었나?
* 멤버가 몇 명이나 될지 모르는 것
* RequestDispatcher 를 쓸지 SendRedirect 를 쓸지 어떻게 결정되나?
* RequestDispatcher 를 사용해서도 페이지의 화면전환이 가능하다
* 상대경로를 사용함을 기억하라
* Webcontent 안에 있는 상대경로

map 방식

req.setAttribute("memberList", memberList);

// 리퀘스트를 풀다

RequestDispatcher dispatcher =

req.getRequestDispatcher("/member/memberListView.jsp"); // 다음 경로의 위치 설정

dispatcher.include(req, res); //setAttribute로 담았으니깐 가져가야 페이지의 내용이 소실되지 않는다

* 스크립틀릿 (scriptlet)
  + 표현식: <% %>

<% 자바언어 이안에서 사용가능 %>

* + 사용하려면 import해야한다
    - 지시자를 사용하여 import 한다
    - 페이지 지시자 : <%@ page %>
* <!DOCTYPE html> 보다 페이지 지시자가 위에 있어야 한다

<%@page import=*"java.util.ArrayList"*%>

<%@page import=*"com.tg.member.MemberDto"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<!DOCTYPE html>

* 데이터를 전송하는 것은 모든 것은 오브젝트이다

그래야 문제가 없으므로

* 즉, 오브젝트로 전송받은 것을 사용하기 위해서는

필요하면, 자식들의 형태로 형변환 해준후 사용하라는 것

ArrayList<MemberDto> memberList =

(ArrayList<MemberDto>)request.getAttribute("memberList");

* Iterator 패턴을 for문에 사용할수 있도록 만듬 -> 확장된 for문

**for**(MemberDto memberDto : memberList){

%>

<%=memberDto.getNo()%>,

<a href=*'./update?no=*<%=memberDto.getNo()%>*'*><%=memberDto.getName()%></a>,

<%=memberDto.getEmail()%>,

<%=memberDto.getCreateDate()%>

<a href=*'./delete?no=*<%=memberDto.getNo()%>*'*>[삭제]</a>

<br>

<%

}

%>

* for문을 쓸것인가 forEach(for문의 확장판)을 사용할 것인가?
  + - 제너릭이 생기면서 같이 생긴게 forEach문이다
* forEach 문
  + - i++ 쓰는거 귀찮다

하나씩 꺼내서 비교하자

* + - 이 방식으로 출력하는 것이 훨씬 빠르다
    - 가공처리된 완벽한 데이터를 처리하므로 사용 가능하다
* scriptlet스크립틀릿
* <% 내용 %>
* 쓰면
* <% %> 안에서는 자바언어를 사용할 수 있음
* 이걸 쓰기 위해서는
* 디렉티브(directive?지시자)를 써야함 그것이 바아로 <%@ %> 입니다
* <%@page import="com.tg.member.MemberDto" %>
* <%@ page language="java" contentType="text/html; charset=UTF-8"
* pageEncoding="UTF-8"%>
* 이렇게 해당 자바 클래스파일이나 자바를 쓴다는걸 import시켜야함
* doctype보다 상위에 있어야 함
* 순서는 지시자로 자바와 클래스파일을 import시킨 후 밑에 body태그 안에서
* 자바를 <% %>를 쓸 수 있음
* java.lang은 기본적으로 있어서 쓸 수 있음
* <%= %> expression
* Model (데이터 전속 객체)
* Vo 모델 (value object)
* DTO 모델 (data transport object)
  + **<% %> 스크립틀릿 (**sriptle)
* <% 자바코드 %>
* 스크립트릿 태그안에 자바 코드를 넣어 사용
  + **<%@ %> 디렉티브 (Directive)**
* <%@ %>
* Page:jsp 페이지와 관련된 속성을 정의할 때 사용
* 공통단 (상급지시자들이 다만들어 놓음, 이렇게 해야해)
  + **<%= %> 익스프레션 expression**
* <%= %>
* Jsp 페이지에서 문자열을 출력할 때 사용한다

(문자열에 관한 리턴값이 있어야한다)

<%= %> 안에는 결과를 반환하는 자바 코드가 반드시 와야한다

* Implicit Object(기본객체)

: 별도의 선언 없이 사용하는 자바 객체들

Ex) request 객체 등

* 자바파일에서 ctrl + F11로 실행시키려 하지 말고

주소를 직접 브라우저에 적어서 실행시켜라

* 개발자라면 가장 작은 단위를 수동적으로 할 수 있어야 한다.

[ 파일 관리의 단위들 ]

: 아래와 같이 최소 3개의 파일은 있어야 한다.

* Jsp 파일들은
* view 가된다 (UI)
* Java 파일들
* 자바만 존재하는 DTO (getter setter)
* 상속받아 사용하는 servlet
* CRUD < 개발의 순서 >

1. 조회 한다

* select

1. 추가 한다
2. 수정
3. 삭제

* 접근에는 순서가 있다
  + 자바코드를 안 적은 jsp 파일은, 이 페이지에서 바로 실행시키면 오류가 난다
  + 자바코드를 적은 jsp 파일에서 실행시키면

: 500번대 오류 : NullPointException

* Web.xml 에서 static형식(전역에 걸친)이므로 다 분리한다
* 보안이 털리지 않도록 (이것은 우리의 영역이 아님)
* 일반 클래스에서 관리하면, 나쁜 개발자에의해 다 털림.
* Filter 라는 존재가 interceptor 하여 다 가로채어 관리하는 것.
* 화면단
* 무조건 WebContent 아래에 만든다
* Package 명은 최소 2레벨 방식이어야한다(3레벨 권장)
* Package 나 folder 명은 소문자로 시작
* 클래스명은 대문자로 시작