




## Day10; 20220916

날짜	@2022년 9월 16일
유형	@2022년 9월 16일
태그	

spring/src at main · u8yes/spring  
Contribute to u8yes/spring development by creating an account on GitHub.

u8yes/spring


<https://github.com/u8yes/spring/tree/main/src>

1 Contributor  
0 Issues  
1 Star  
0 Forks

tomcat를 WAS라고도 부른다.

웹 어플리케이션의 서비스를 가능하게 해주는 것이 Container.

Servlet Container, JSP Container

Presentation Layer = Controller + View

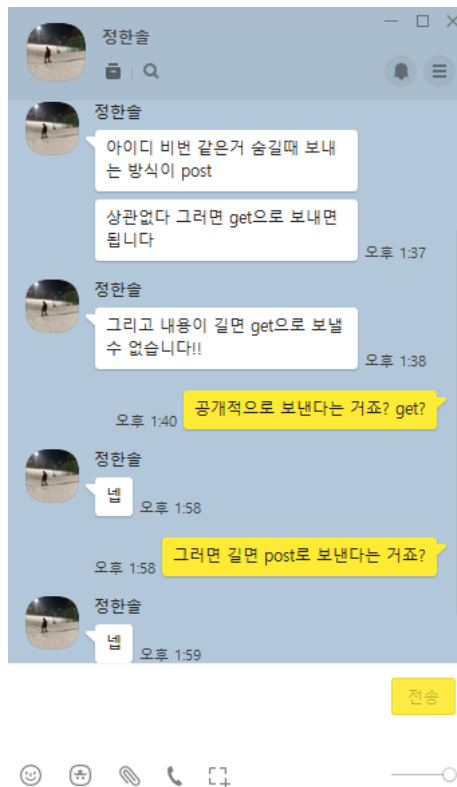
Business component = Service Layer + Persistence Layer

DAO, VO(DTO), DB 작업하는 공간을 Repository layer = Persistence layer

@Component - @Service와 @Repository, @Controller를 제공

boardDAO에서 사용하는 Annotation - @Repository("boardDAO")

get과 post 방식 차이



## BoardController.java

```
package com.springproj.controller;

import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;

import com.springproj.biz.board.vo.BoardVO;
import com.springproj.biz.service.BoardService;

@Controller // implements Controller 삭제하면서 Controller 사용가능해짐.
public class BoardController {

    @Autowired
    BoardService boardService;

    // @RequestMapping(value = "/insertBoard.do", method = RequestMethod.GET) // /insertBoard.do를 부르면 바로 Controller 작업 들어가라
    @GetMapping("/insertBoard.do")
    public String insertBoard() {
        System.out.println("글 등록 화면 처리");

        return "insertBoard";
    }

    // @RequestMapping(value = "/insertBoard.do", method = RequestMethod.POST) // /insertBoard.do를 부르면 바로 Controller 작업 들어가라
    @PostMapping("/insertBoard.do")
    public String insertBoard(BoardVO board){
        System.out.println("글 등록 처리");
    }
}
```

```

        boardService.insertService(board);

        return "redirect:getBoardList.do";
    }

@RequestMapping("/getBoard.do") // /insertBoard.do를 부르면 바로 Controller 작업 들어가라
public String getBoard(BoardVO board, Model model) { // command 객체를 Spring에서 만들어줘서 board, model 구분해서 따로 작업해줄 수가 있다.
    System.out.println("Board 상세페이지 처리."); // 혹시나 모를 문제발생을 대비하기 위해 출력 Check!! Debugging

    BoardVO vo = boardService.getService(board.getSeq());

    model.addAttribute("board", vo);

    return "getBoard";
}

@RequestMapping("/getBoardList.do") // /insertBoard.do를 부르면 바로 Controller 작업 들어가라
public String getBoardList(Model model) {
    System.out.println("글 목록 검색 처리");

    List<BoardVO> list = boardService.getServiceList();

    model.addAttribute("boardList", list);

    return "getBoardList";
}

@RequestMapping("/updateBoard.do") // /insertBoard.do를 부르면 바로 Controller 작업 들어가라
public String updateBoardProc(BoardVO board) {
    //System.out.println("글 수정 처리");

    boardService.updateService(board);

    return "redirect:getBoardList.do";
}

@RequestMapping("/deleteBoard.do") // /insertBoard.do를 부르면 바로 Controller 작업 들어가라
public String deleteBoard(BoardVO board) {
    //System.out.println("글 삭제 처리");

    boardService.deleteService(board.getSeq());

    return "redirect:getBoardList.do";
}
}

```

## servlet-context.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:p="http://www.springframework.org/schema/p"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd
        http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd
        http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.3.xsd
        http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx.xsd">

    <!-- DispatcherServlet Context: defines this servlet's request-processing infrastructure -->
    <context:component-scan base-package="com.springproj.controller" />

    <!-- Enables the Spring MVC @Controller programming model -->
    <annotation-driven />

    <!-- Handles HTTP GET requests for /resources/** by efficiently serving up static resources in the ${webappRoot}/resources directory -->
    <resources mapping="/resources/**" location="/resources/" />

```

```
<!-- Resolves views selected for rendering by @Controllers to .jsp resources in the /WEB-INF/views directory -->
<beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
  <beans:property name="prefix" value="/WEB-INF/views/" /> <!-- property는 set 메서드 넣어줄 value -->
  <beans:property name="suffix" value=".jsp" />
</beans:bean>

<!-- 빈 객체를 생성하려고 자동으로 만들어진 것 -->
<context:component-scan base-package="com.springproj.controller" />

</beans:beans>
```

---

