Spring Web MVC

MVC : Model, View, Controller의 약자입니다.

현재 가장 많이 사용하는 개발 패턴으로 서비스를 위한 각 부분을 분리해 만들고 이를 통해 개발 및 유지 보수의 효율성을 높이는 것이 목적입니다.

- Model : 데이터를 관리하는 부분
- View : 눈에 보이는 부분을 구성하는 부분
- Controller : 요청에 따른 코드 흐름을 제어하는 부분

(1)	SpringConfigClass {
	환경설정클래스 -ServletAppContext
	환경Root 빈 정의 -RootAppContext
	}
		:
		AbstractAnnotationConfigDispatcherServletInitializer 상속

(2)	@Configuration ServletAppContext { @ComponentScan("kr.co.korea.controller") @ComponentScan("kr.co.korea.dao") @ComponentScan("kr.co.korea.service") @PropertySource("/WEB-INF/properties/db.properties") @Bean -db접속,쿼리접속,쿼리실행	: WebMvcConfigurer 인터페이스 상속
(3)	@Configuration RootAppContext { @Bean("loginUserBean") @SessionScope	

(1) @Bean : 메서드를 통해 반환하는 객체를 Bean으로 등록
ex) [kr.co.korea.config]패키지의 ServletAppContext 클래스의 DB접속정보, 쿼리문, mapper

(2) @Component : 개발자가 만든 클래스의 객체를 생성하여 Bean으로 등록
ex) [kr.co.korea.beans]패키지의 DataBean 클래스의 객체를 생성하여 등록할때

(3) @Controller : Component의 일종으로 사용자 요청에 따라 **자동으로 호출되는 메서드**를 가지고 있는 Bean을 등록. jsp로 리턴 ex) [kr.co.korea.controller]패키지의 HomeController클래스, TestController클래스

(4) @RestController : Component의 일종으로 사용자 요청에 따라 **자동으로 호출되는 메서드**를 가지고 있는 Bean을 등록. Restful API 서버 구성 시 사용. 전달 글자 그대로 리턴

(5) @ControllerAdvice : 예외가 발생했을 때 사용할 Global Exception Handler로 사용할 Bean을 등록
ex) [kr.co.korea.exception]패키지의 GlobalException 클래스 만들때...

(6) @Repository : dao에서 @Repository로 클래스를 빈 정의 해두고 service의 @Service로 정의한 Bean에서 @Autowired 로 주입받아 사용. 이 Bean은 데이터베이스와 관련된 작업을 구현. @Component로 정의한 Bean과 차이가 없음.
ex) [kr.co.korea.dao]패키지 TopMenuDao클래스 만들때

(7) @Service : service에서 빈 정의 해두고 Controller에서 @Autowired 이용하여 주입받아 사용. @Component로 정의한 Bean과 차이는 없음. ex) [kr.co.korea.service]패키지의 UserService클래스, TopMenuService클래스 만들때...

(8) @Configuration : 설정클래스임을 지정
ex) [kr.co.korea.config]패키지의 ServletAppContext.java에서 **설정 클래스**임을 지정

(9) @EnableWebMvc : 설정클래스에서 Spring MVC환경을 구성할 수 있음
ex) [kr.co.korea.config]패키지의 ServletAppContext.java

(10) @ComponentScan("패키지") : 스캔할 패키지 지정
ex) [kr.co.korea.config]패키지의 ServletAppContext.java 에서

(11) @PropertySource(), @PropertySources() : properties파일을 주입받기
ex) [kr.co.korea.config]패키지의 ServletAppContext.java

(12) @Value : properties파일의 값을 주입받기
ex) [kr.co.korea.config]패키지의 ServletAppContext.java

1. Java방법으로 스프링 설정

(1) project.sql

```
create sequence user_seq
start with 0
increment by 1
minvalue 0;

create sequence content_seq
start with 0
increment by 1
minvalue 0;

create table board_info_table(
          board_info_idx number constraint BOARD_INFO_PK primary key,
          board_info_name varchar2(500) not null
);

insert into board_info_table(board_info_idx, board_info_name) values (1, '1자유게시판');
insert into board_info_table(board_info_idx, board_info_name) values (2, '2게시판');
insert into board_info_table(board_info_idx, board_info_name) values (3, '3게시판');
insert into board_info_table(board_info_idx, board_info_name) values (4, '4게시판');

commit;

create table user_table(
          user_idx number constraint USER_PK primary key,
          user_name varchar2(50) not null,
          user_id varchar2(100) not null,
          user_pw varchar2(100) not null
);

create table content_table(
          content_idx number constraint CONTENT_PK primary key,
          content_subject varchar2(500) not null,
          content_text long not null,
          content_file varchar2(500),
          content_writer_idx number not null
```

```
                        constraint CONTENT_FK1 references user_table(user_idx),
        content_board_idx number not null
                        constraint CONTENT_FK2 references board_info_table(board_info_idx),
        content_date date not null
);
```

게시판 정보 테이블(board_info_table)

필드명(논리)	필드명	타입	NULL	PK	FK	UQ
게시판 번호	board_info_idx	number	X	O	X	O
게시판 이름	board_info_name	varchar2(500)	X	X	X	X

사용자 정보 테이블(user_table)

필드명(논리)	필드명	타입	NULL	PK	FK	UQ
사용자 번호	user_idx	number	X	O	X	O
사용자 이름	user_name	varchar2(50)	X	X	X	X
사용자 아이디	user_id	varchar2(100)	X	X	X	O
사용자 비밀번호	user_pw	varchar2(100)	X	X	X	X

게시글 테이블(content_table)

필드명(논리)	필드명	타입	NULL	PK	FK	UQ
게시글 인덱스	content_idx	number	X	O	X	O
게시글 제목	content_subject	varchar2(500)	X	X	X	X
게시글 내용	content_text	long	X	X	X	X
첨부파일	content_file	varchar2(500)	O	X	X	X
게시글 작성자 인덱스	content_writer_idx	number	X	X	user_table(user_idx)	X
게시판 인덱스	content_board_idx	number	X	X	board_info_table(board_info_idx)	X
작성날짜	content_date	date	X	X	X	X

(2) pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <groupId>minProjectJava</groupId>
        <artifactId>minProjectJava</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>war</packaging>
        <build>
                <sourceDirectory>src</sourceDirectory>
                <plugins>
                        <plugin>
                                <artifactId>maven-compiler-plugin</artifactId>
                                <version>3.8.1</version>
                                <configuration>
                                        <source>1.8</source>
                                        <target>1.8</target>
                                </configuration>
                        </plugin>
                        <plugin>
                                <artifactId>maven-war-plugin</artifactId>
                                <version>3.2.3</version>
                                <configuration>
```

```
                                <warSourceDirectory>WebContent</warSourceDirectory>
                            </configuration>
                        </plugin>
                </plugins>
        </build>
        <!-- 라이브러리 버전관리 -->
        <properties>
                <javax.servlet-version>4.0.1</javax.servlet-version>
                <javax.servlet.jsp-version>2.3.3</javax.servlet.jsp-version>
                <javax.servlet-jstl-version>1.2</javax.servlet-jstl-version>
                <org.springframework-version>5.2.8.RELEASE</org.springframework-version>
        </properties>

        <!-- 라이브러리 셋팅 -->
        <dependencies>
                <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>javax.servlet-api</artifactId>
                        <version>${javax.servlet-version}</version>
                        <scope>provided</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/javax.servlet.jsp/javax.servlet.jsp-api -->
                <dependency>
                        <groupId>javax.servlet.jsp</groupId>
                        <artifactId>javax.servlet.jsp-api</artifactId>
                        <version>${javax.servlet.jsp-version}</version>
                        <scope>provided</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/javax.servlet/jstl -->
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>jstl</artifactId>
                        <version>${javax.servlet-jstl-version}</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-webmvc</artifactId>
                        <version>${org.springframework-version}</version>
                </dependency>

                <dependency>
                        <groupId>javax.validation</groupId>
                        <artifactId>validation-api</artifactId>
                        <version>2.0.1.Final</version>
                </dependency>

                <dependency>
                        <groupId>org.hibernate.validator</groupId>
                        <artifactId>hibernate-validator</artifactId>
                        <version>6.1.2.Final</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-jdbc</artifactId>
                        <version>5.2.8.RELEASE</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.apache.commons/commons-dbcp2 -->
                <dependency>
                        <groupId>org.apache.commons</groupId>
                        <artifactId>commons-dbcp2</artifactId>
                        <version>2.7.0</version>
                </dependency>

                <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
                <dependency>
                        <groupId>org.mybatis</groupId>
                        <artifactId>mybatis-spring</artifactId>
                        <version>2.0.5</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
                <dependency>
                        <groupId>org.mybatis</groupId>
```

```
                        <artifactId>mybatis</artifactId>
                        <version>3.5.5</version>
                    </dependency>

        </dependencies>

</project>
```

1. mybatis 개념

(1) mapper작성:쿼리문

[kr.co.korea.mapper] UserMapper.java 쿼리문 작성

쿼리문작성	public interface UserMapper {
	@Select("select user_idx, user_name " +
	"from user_table " +
	"where user_id=#{user_id} and user_pw=#{user_pw}")
	UserBean getLoginUserInfo(UserBean tempLoginUserBean);
	}

(2) dao작성:mapper와 직접적으로 연결해주는 메소드정의

[kr.co.korea.dao] UserDao.java

Dao	@Repository
	public class UserDao {
	@Autowired
	private UserMapper userMapper;
	
	public UserBean getLoginUserInfo(UserBean tempLoginUserBean) {
	return userMapper.getLoginUserInfo(tempLoginUserBean);
	}

(3) service작성:dao의 메소드 가져와서 새로운 변형을 위한 메소드정의

[kr.co.korea.service] UserService.java

Service	@Service
	public class UserService {
	
	@Autowired
	private UserDao userDao;
	
	//UserDao의 getLoginUserInfo() 호출하도록 정의
	public void getLoginUserInfo(UserBean tempLoginUserBean) {
	
	UserBean tempLoginUserBean2 = userDao.getLoginUserInfo(tempLoginUserBean);
	
	if(tempLoginUserBean2 != null) { //db에서 select된게 있으면
	loginUserBean.setUser_idx(tempLoginUserBean2.getUser_idx());
	loginUserBean.setUser_name(tempLoginUserBean2.getUser_name());
	loginUserBean.setUserLogin(true);
	}
	}
	}

(4) controller작성:service의 메소드 가져와서 jsp파일(views)로 이동하는 메소드 정의

[kr.co.korea.controller] UserController.java

Controller	@Controller
	@RequestMapping("/user")
	public class UserController {
	//userService 객체로 주입
	@Autowired
	private UserService userService;
	
	@PostMapping("/login_pro")
	public String login_pro(@Valid @ModelAttribute("tempLoginUserBean") UserBean tempLoginUserBean, BindingResult result) {

```
                        if(result.hasErrors()) {
                                return "user/login";
                        }

                        userService.getLoginUserInfo(tempLoginUserBean);

                        if(loginUserBean.isUserLogin() == true) {
                                return "user/login_success";
                        } else {
                                return "user/login_fail";
                        }
                }
}
```

(1) java 방식 설정

1) db.properties

[WebContent-WEB-INF-properties폴더]

```
db.classname=oracle.jdbc.OracleDriver
db.url=jdbc:oracle:thin:@localhost:1521:xe
db.username=system
db.password=123456
```

2) BoardMapper.java 인터페이스

[kr.co.korea.mapper]

```
package kr.co.korea.mapper;

public interface BoardMapper {

}
```

3) ServletAppContext.java

```
//1)
@PropertySource("/WEB-INF/properties/db.properties")
public class ServletAppContext implements WebMvcConfigurer {

//2)
        @Value("${db.classname}")
        private String db_classname;

        @Value("${db.url}")
        private String db_url;

        @Value("${db.username}")
        private String db_username;

        @Value("${db.password}")
        private String db_password;


//3) 데이터베이스 접속 정보 관리 (org.apache.commons.dbcp2.BasicDataSource)
        @Bean
        public BasicDataSource dataSource() {
                BasicDataSource source = new BasicDataSource();
                source.setDriverClassName(db_classname);
                source.setUrl(db_url);
                source.setUsername(db_username);
                source.setPassword(db_password);
                // System.out.println("db connect");
                return source;
        }

//4) 쿼리문과 접속 관리하는 객체
        @Bean
        public SqlSessionFactory factory(BasicDataSource source) throws Exception {
                SqlSessionFactoryBean factoryBean = new SqlSessionFactoryBean();
                factoryBean.setDataSource(source);
```

```
                SqlSessionFactory factory = factoryBean.getObject();
                // System.out.println("sql");
                return factory;
        }

//5) 쿼리문 실행을 위한 객체 (BoardMapper생성할것)
        @Bean
        public MapperFactoryBean<BoardMapper> getBoardMapper(SqlSessionFactory factory) throws
Exception {
                MapperFactoryBean<BoardMapper> factoryBean = new
MapperFactoryBean<BoardMapper>(BoardMapper.class);
                factoryBean.setSqlSessionFactory(factory);
                return factoryBean;
        }
```

(2) xml 방식 설정

1) db.properties

[WebContent-WEB-INF-properties폴더]

```
db.classname=oracle.jdbc.OracleDriver
db.url=jdbc:oracle:thin:@localhost:1521:xe
db.username=system
db.password=123456
```

2) board_mapper.xml

[WebContent-WEB-INF-mapper폴더]

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
                                        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="board">

</mapper>
```

3) servlet-context.xml

[WebContent-WEB-INF-config폴더]

```
<!--1) properties파일의 내용을 사용할 수 있도록 Bean를 정의 -->
        <beans:bean class='org.springframework.beans.factory.config.PropertyPlaceholderConfigurer'>
                <beans:property name="location">
                        <beans:value>/WEB-INF/properties/db.properties</beans:value>
                </beans:property>
        </beans:bean>

<!--2) -->
        <beans:bean class='org.apache.commons.dbcp2.BasicDataSource' id='basic_data_source'>
                <beans:property name='driverClassName' value='${db.classname}'/>
                <beans:property name="url" value="${db.url}"/>
                <beans:property name="username" value='${db.username}'/>
                <beans:property name="password" value='${db.password}'/>
        </beans:bean>

<!--3) -->
        <beans:bean class='org.mybatis.spring.SqlSessionFactoryBean' id='sqlSession'>
                <beans:property name="dataSource" ref='basic_data_source'/>
                <beans:property name="mapperLocations" value='/WEB-INF/mapper/*.xml'/>
        </beans:bean>

<!--4) -->
        <beans:bean class='org.mybatis.spring.SqlSessionTemplate' id='sqlSessionTemplate'>
                <beans:constructor-arg index='0' ref='sqlSession'/>
        </beans:bean>
```