- spring framework

- sts4 환경설정= SPRING 가능하는 이클립스
1> SPRING MVC PROJECT 개발 추가 설치 + 환경 세팅
2> SPRING BOOT PROJECT 개발 집중 툴

- 프레임워크
1>일정 규격 맞추어 프로그래밍하기를 강제화
2> 뼈대 공통 설계 + 몸통 (사용자) 구현
3> 반 이미 구현 제공(스프링) + 나머지 반 구현 + XML -> 완성품

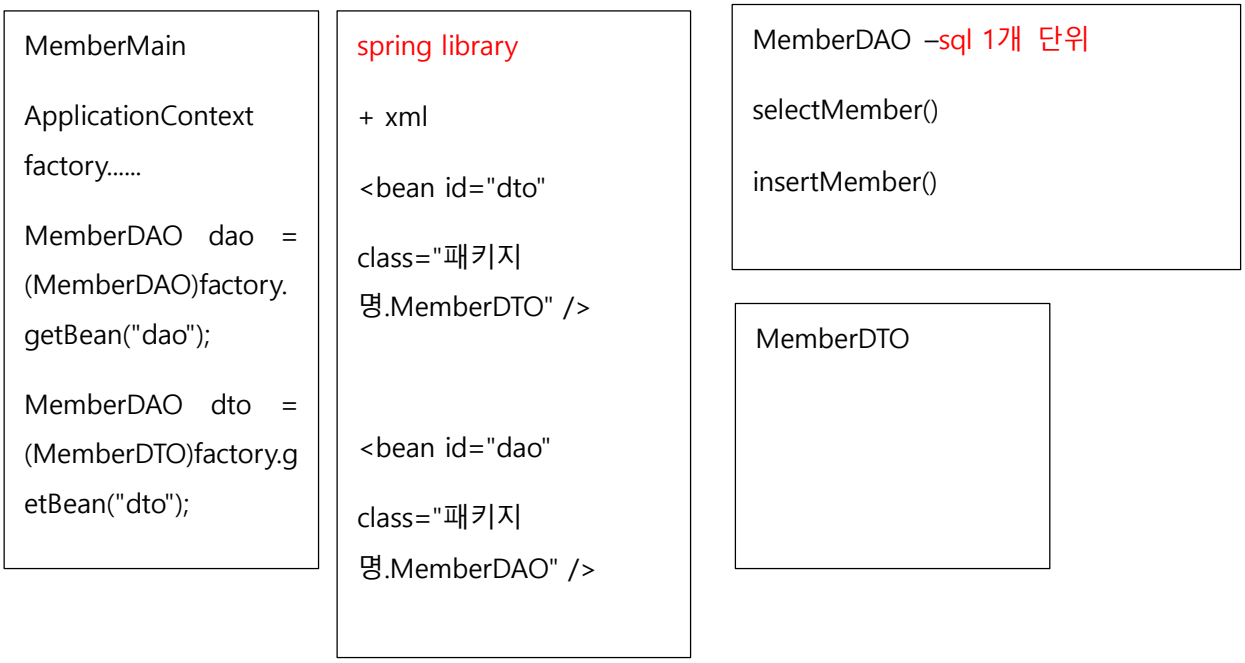
- 스프링 특징
- 자바 프레임워크
- pojo 클래스 사용 – dto, dao 형태 클래스(MAIN + SERVLET+JSP) 그대로 사용
-IOC (개념)==> INVERSION OF CONTROL
1>제어의 역전
2>
원하는 객체 생성하고자 한다면 클래스 내부 new XXXX()
===>MAIN 객체 생성 제어권 기본 있다(제어 역전 아니다)
3>
===>TVFactory 가 객체 생성 제어권 가지고 main 은 제어 역전
===> 객체 타입 한정하자 – 인터페이스
4>
DI ===> dependency injection
1> setter injection
2> constructor injection
====> IOC 개념 구현 방법
main 전달받아야 한다 = 주입받아야 한다
tvfactory 의존하여 생성한 객체만 주입받는다

- spring bean configuration file – xml 파일 형태
스프링 프레임워크 내에서 사용할 객체 설정 파일
<bean id="a1" class="p.A" />
===> p.A a1= new o.A(); 스프링 설정
<bean id="a2" class="p.A" />

```
<bean id="b1" class="p.B" >
<property name="a" ref="a1" />
</bean>
p.B b1 = new p.B();
b1.setA(a1);

<bean id="c1" class="p.C" />
==> p.C c1 = new p.C();

<bean id="c1" class="p.C" >
<constructor-arg name="id" value="jsp"   />
<constructor-arg name="a" ref="a1"   />

</bean>
==> p.C c1 = new p.C("jsp", a1);
```

- TV --> 스프링 외부 객체 전달 타입 한정 – 인터페이스
  스프링 API + XML

- 스프링 pojo 사용



1> di

2> pojo 클래스 그대로 사용 + 인터페이스 구조

3> DTO = VO = DO = 값 저장 / 추출 객체

4> DAO = DATA ACCESS OBJECT = 데이터 접근 역할 객체(JDBC , 입출력)

5> SPRING + DTO+DAO

interface MemberService - registerMember();

MemberMain

ApplicationContext factory.....

MemberService s =

factory.getBean("service");

s.regiterMember();

spring library

+ xml

<bean id="dto"

class="패키지명.MemberDTO" />

<bean id="dao"

class="패키지명.MemberDAO" >

<property name="dto"

ref="dto" />

</bean>

<bean id="service"

class="패키지명.MemberServiceImpl">

<peoperty name="dao"

MemberServiceImpl

MemberDAO dao;

setDao(MemberDAO dao){

this.dao = dao;

}

요청 기능 단위

regiterMember(){

boolean f = dao.selectMember();

if(f==true){중복}

else

MemberDAO –

MemberDTO dto;

setDto(MemberDTO dto){

this.dto = dto;

}

sql 1개 단위

selectMember()

insertMember()

MemberDTO

?---service + dao(1. selectMember(), 2.selectMember(String id, String pw)) + dto + db

- ioc , di
- xml 태그
<bean
<property
<constructor-arg

- xml 태그 + JAVA ANNOTATION

자바문장	스프링beanconfiguration =xml	스프링 annotation

edu.A a1 = new edu.A();
a1.setB(new B());
	<bean id="b1" class=" edu.B" >
	<bean id="a1" class=" edu.A" >
	<property name="b" ref="b1"/>
	</bean>
		package edu;
		@Component
		class A {
		@Autowired
		B b1;
		setB(B b1){
		this.b1 = b1;
		}
		}

		package edu;
		@Component
		class B{
		....
		}

edu; @Compoment("b1") class B {}	xml파일 <bean id="b3" class="edu.B" /> <bean id="b2" class="edu.B" /> ==> edu.B 객체 3개 생성

edu;
@Compoment("a1")
class A {
@Autowired(B 타입 객체 여러개 자동 주입)
@Qualifier("b1")
B b1;
setB(B b1){ this.b1 = b1; }
}	

<bean id="" class="패키지명.클래스명" />	@Service("") 또는 @Component("") class xxxServiceImpl .. @Repository 또는 @Component("") class xxxDAO @Component("") class xxxDTO
<property	@Autowired + @Qualifier("")

MemberMain

ApplicationContext factory.....

MemberService s =

factory.getBean("service");

s.regiterMember();

s.login();

spring library

+ xml

<context:component-scan

base-pakage="패키지명"

@Service("service")

MemberServiceImpl

@Autowired

MemberDAO dao;

요청 기능 단위

@Repository

MemberDAO –

@Autowired

MemberDTO dto;

@Compoenent

MemberDTO

xml 태그 + 자바 annotation 설정

19 장 IOC , DI + 스프링관리 객체 + XML 설정+ANNOTATION 설정

xml 태그 + JAVA ANNOTATION

IOC DI

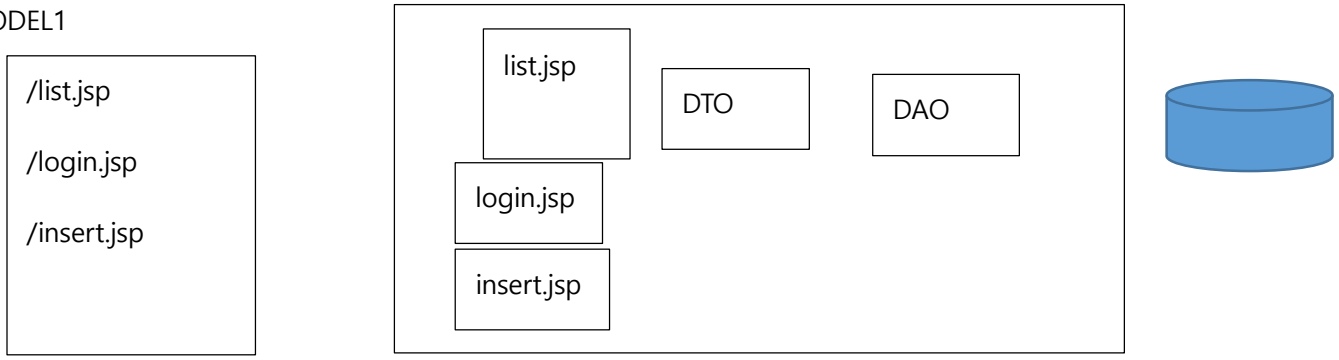
19장	**IOC DI --> XML**
20장	**AOP --> XML** **--> 기능 선택적**
21장 **-25**	**MVC** **MYBATIS연동** **--> XML**
26장	**ANNOTATION**

21장 SPRING MVC

- JSP 웹서버 개발 패턴

간단한 웹페이지개발

- MODEL1



1>jsp 내부에 자바 문장과 html 태그를 모두 작성할 수 있으나 복잡하게 뒤섞여있다.

<% 자바 %>

<HTML>

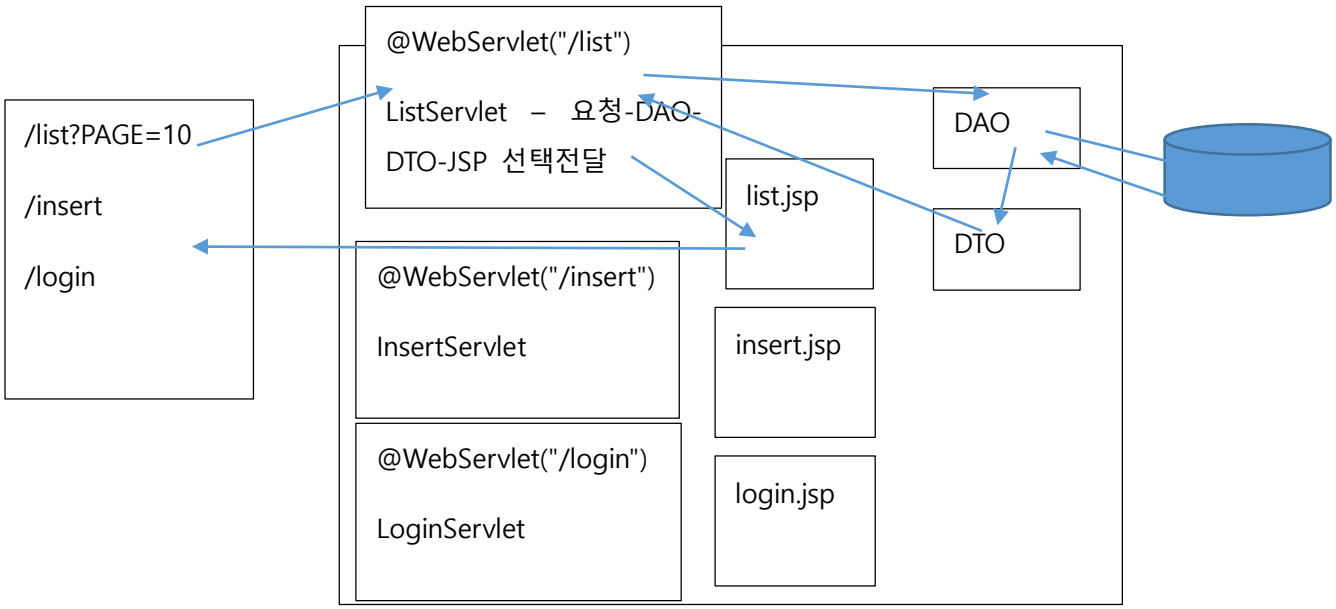
<%= %>

<HTML

<JSP:USEBEAN .... />

2> 자바문장은 보통 로직의 처리, html 태그는 화면 구성에 필요한데 로직 처리만 변경하면 자바문장만, 화면 구성만 변경되면 html태그만 수정할 수 있는 구조가 바람직하다.

- MODEL2- 1개 요청 – 응답 ==> SERVLET + JSP + DAO, DTO
(CONTROLLER) (VIEW) + (MODEL)

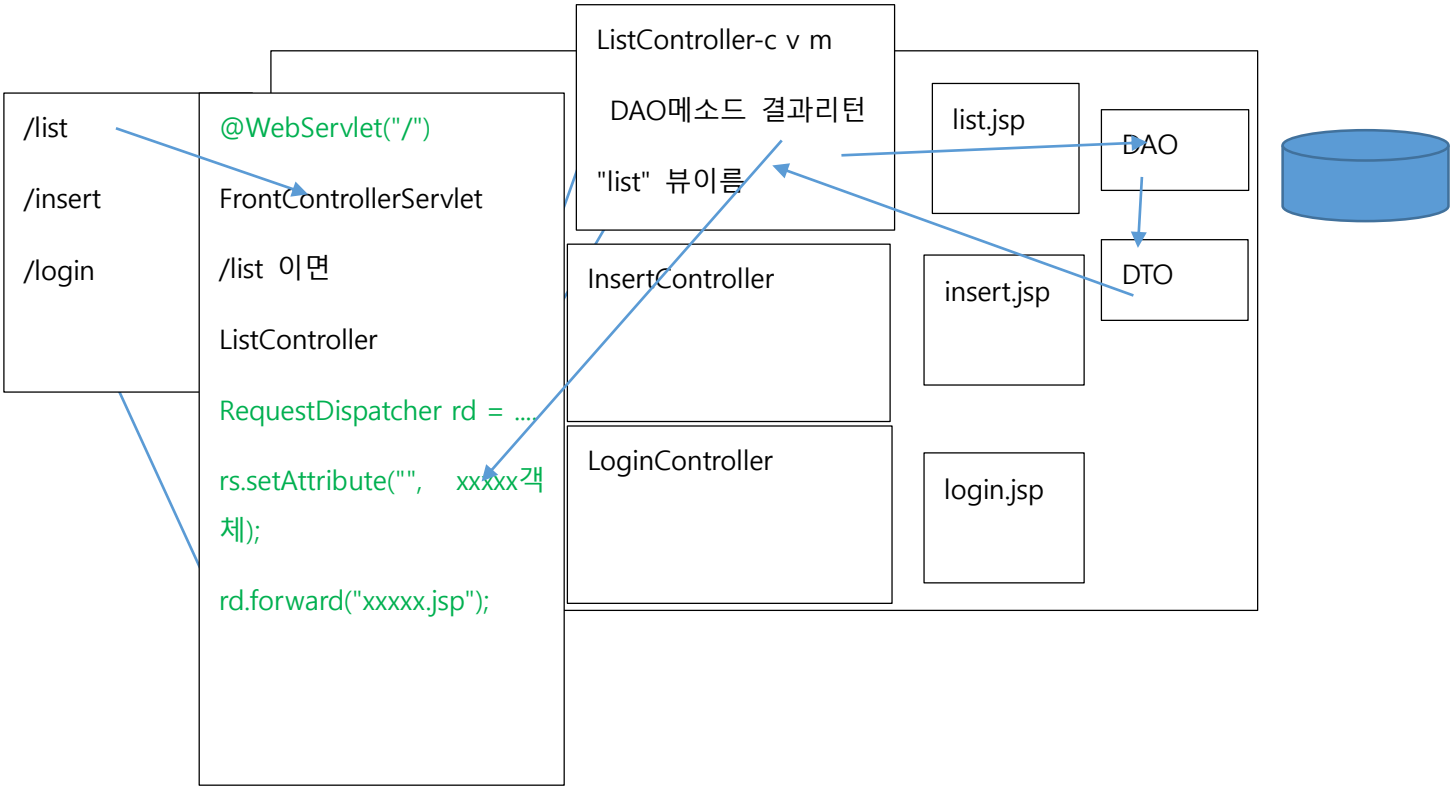



```
@WebServlet("/xxxx")

xxxxServlet

 1.요청받자

2.처리하는 dao 메소드 호출하여 결과를 dto 형태로 리턴받자

3. jsp로 forward하여 처리결과 setAttribute로 전달하자

RequestDispatcher rd = ....

rs.setAttribute("", xxxxx객체);

rd.forward("xxxxx.jsp");
```

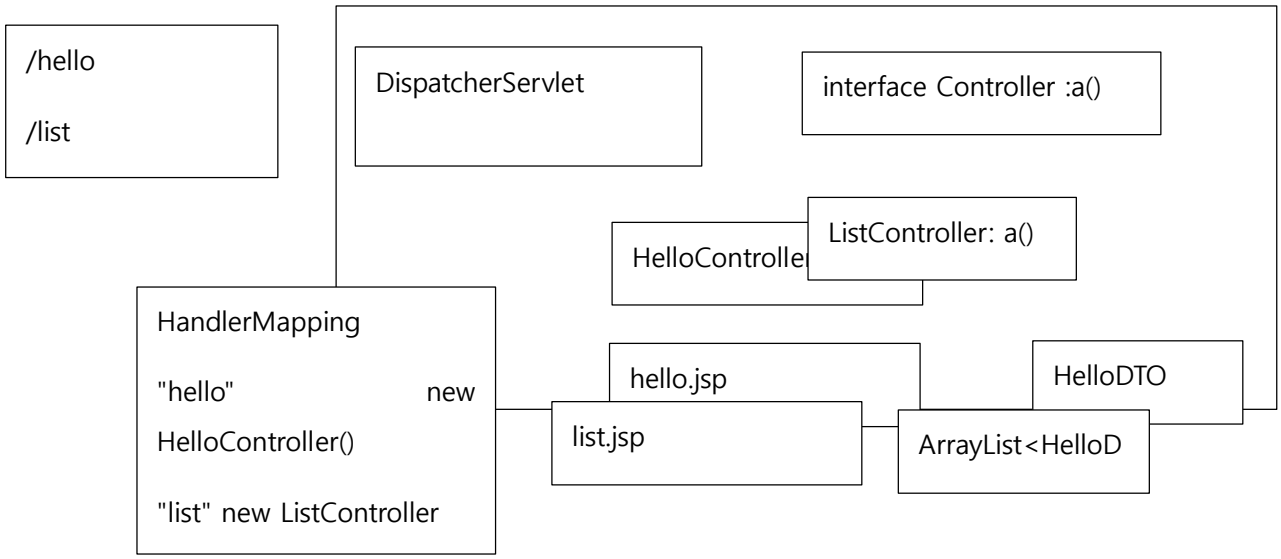
model2 = mvc

spring mvc = frontcontroller 클래스 + model2

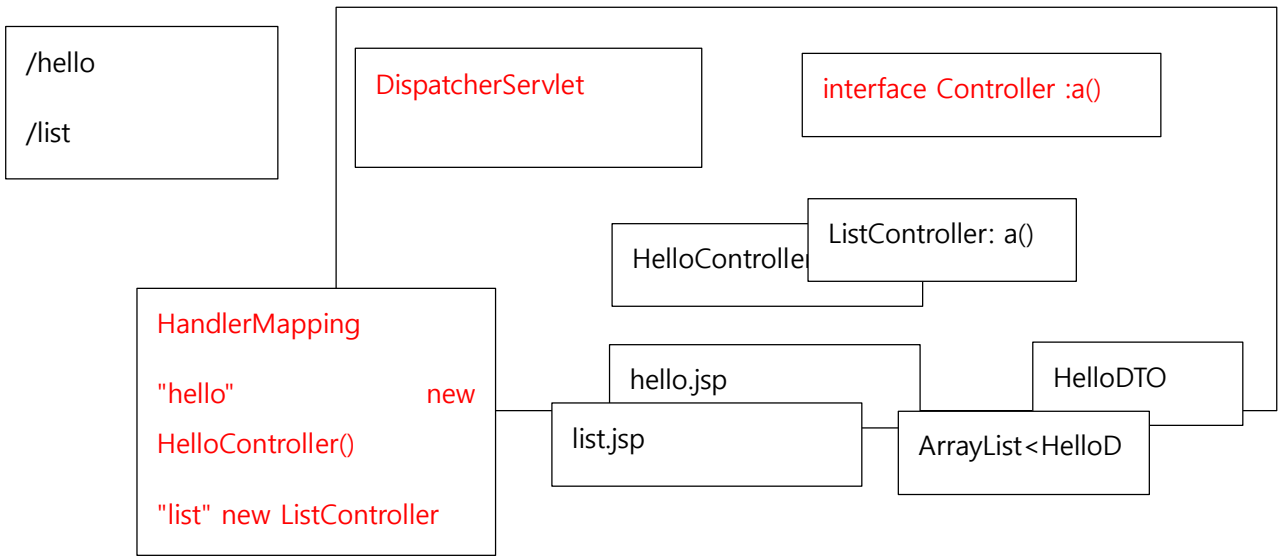
- MVC
- MODEL2=frontcontroller 개념 + mvc



1. non-spring mvc- dynamic web project



2. spring mvc 변경- spring mvc project



-servlet

@WebServlet("/aa") - url패턴 java annotation	web.xml
package edu; class A extends HttpServlet doGet doPost	<servlet> <servlet-name> i </ <servlet-class> edu.A </ </servlet <servlet-mapping <servlet-name> i <url-pattern > /aa