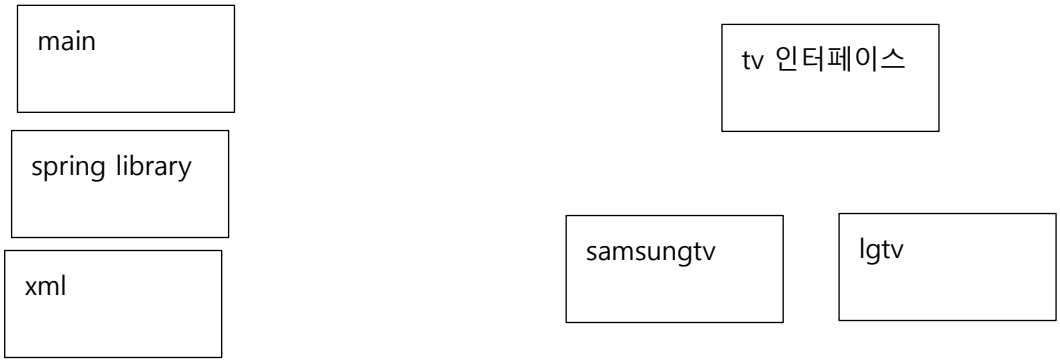
spring framework
"용어" "개념" "여전히 프로그래밍(강제적 규칙 틀)"
- DB연동기능, WEB기능 등의 다양한 프로그래밍 개발 규칙을 포함한 통합 기능 자바 프레임워크

프로젝트 구현 반 구현(스프링 라이브러리) + 나머지 반 개발자 필요사항 구현

framework
0> 어떤 것을 구성하는 뼈대 – 틀 내부 범위 작업
xml 정의 bean들만 객체 생성 제한
main 메소드 객체생성권한 x
xml파일 설정내용 따라서 스프링 제공 생성
1> 자바 언어
2> 자바 클래스 사용 가능 – pojo 클래스 사용 가능
이전 개발 프로젝트 --> 스프링 내부 결합
dao, dto
3> main 메소드 객체생성권한 x
xml파일 설정내용 따라서 스프링 제공 생성
---> inversion of control 개념 용어
---> dependency injection 구현
--> class A{ B 외부에서 전달}
--> setter / 생성자
--> 스프링 + xml (spring bean configuration file)



MAIN

XML

1.MemberDTO객체생성

<bean id="t" class="패키지명.MemberDTO" />

2.

<bean id="" class=" 패키지명.MemberDAO">

<property name="dto" ref="t" />

</bean>

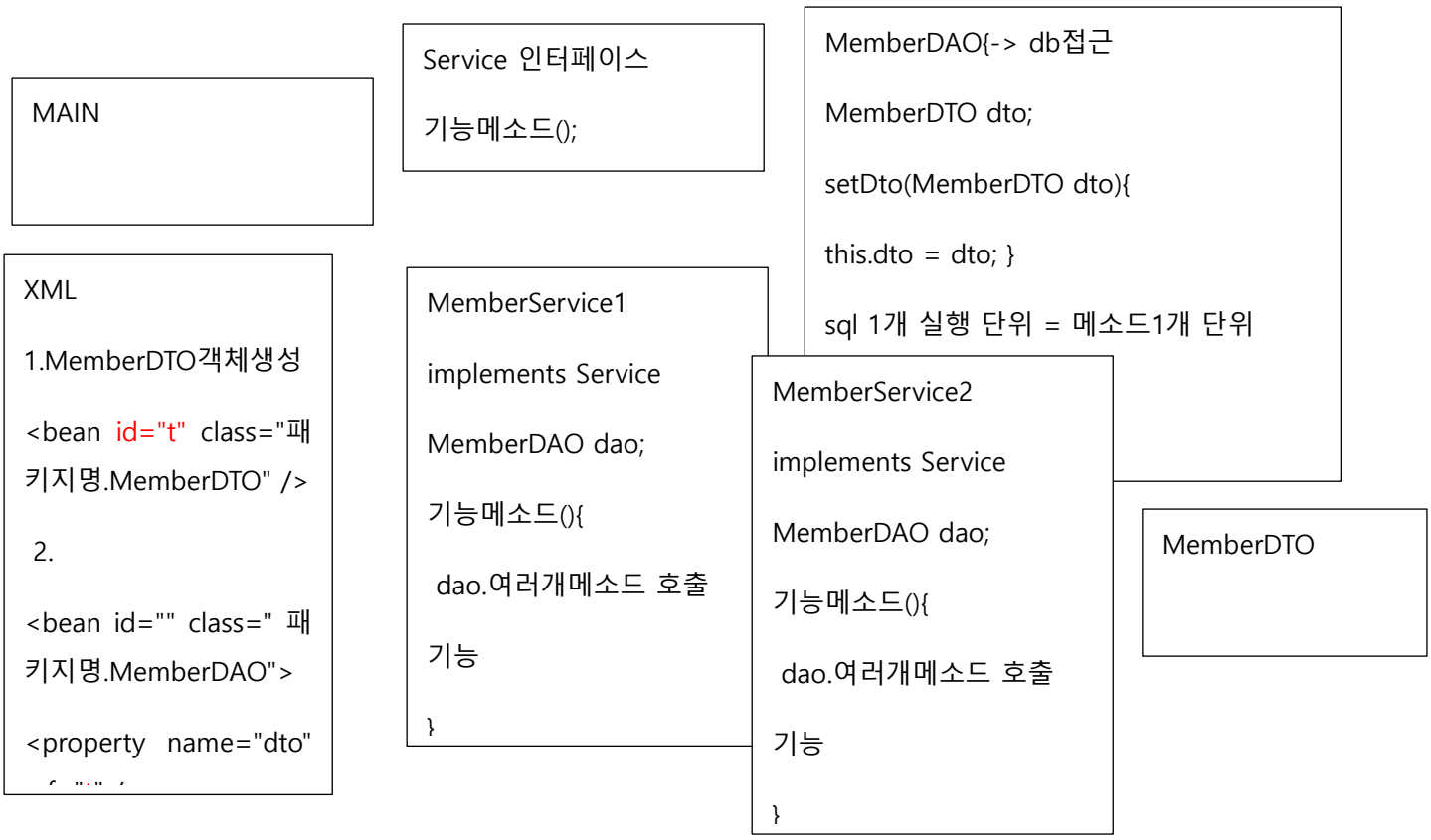
MemberDAO{

MemberDTO dto;

setDto(MemberDTO dto){

this.dto = dto; }

MemberDTO



19장

DI / IOC--> "NEW" 없애자

MAIN->SERVICE->DAO-> DB

----DTO --------------------

18	환경설정
19	SPRING DI , SPRING IOC
26	spring di @
21	SPRING MVC
22,23, 24	SPRING DB이용(자바 JDBC 다르다) MYBATIS SPRING MVC+MYBATIS 연동
26	스프링 애너테이션 기능

	@WebServlet("/test") 자바클래스 url /test 호출 알려줌
27	메이븐과 sts 사용법
28, 29	파일업로드 파일다운로드 ajax , rest mvc 확장
30, 31	샘플 프로젝트

--> sts3

32장 스프링 부트 사용하기

---> sts4

spring legacy project – spring mvc project

spring mvc project 많이 개발중 / 이미 개발완료 사용중

설정 복잡

스프링 부트 - 설정 간소 프로젝트 스프링 제공

---

web.xml
<servlet>
<servlet-name>a </servlet-name>
<servlet-class> test.A </servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>a </servlet-name>
<url-paettern> /aa </url-pattern>
</servlet-mapping>
	package test;
	@WebServlet("/aa")
	class A extends HttpServlet{
	.....
	
	package test;
	@WebServlet("/aa2")
	class A2 extends HttpServlet{
	.....

-spring

spring bean configuration xml 설정 <bean <propeorty <constructor-arg <bean id="list" class="java.util.ArrayList" />	자바소스 내부 annotation 설정 1> xml 간결하다 2> 내가 작성 소스 내부 선언 가능 3> ArrayList 객체생성 힘들다

자바문장	스프링 bean configuration =xml	스프링 annotation 자바소스 내부 설정 사전 필요 <context:component-scan base-package="edu" />

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("a1")
		class A {
		@Autowired
		B b1;
		}
		
		@Component("b1")
		class B{}

class A{ B b1; setB(B b1){this.b1 = b1;}		

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

-annotation

class 위=객체생성	@Component @Service @Repository

변수 위=객체전달주입	**@Qualifier("이름")** **@Autowired** **A a1; ---> A클래스타입객체 (2개이상)자동전달**

서비스클래스인식표현X		

@Component("as")
class AService implements Service
	@Component("ad")
	class ADAO {
	
	}
		@Component("at")
		class ADTO {
		
		}

서비스역할클래스표현O	db접근역할-지속	애매한 성격 클래스

@Service("as")
class AService implements Service
	@Repository("ad")
	class ADAO {
	
	}
		@Component("at")
		class ADTO {
		
		}

@Service --> aService
class AService implements Service
	@Repository ->aDAO
	class ADAO {
	
	}
		@Component ->aDTO
		class ADTO {
		
		}

20장 aop

자바 - OOP – OBJECT ORIENTED PROGRAMMING = 객체지향프로그래밍

자바 - AOP – ASPECT ORIENTED PROGRAMMING = 관심지향프로그래밍

핵심관심과 공통관심 분리 --> 필요시 동적 연결 실행

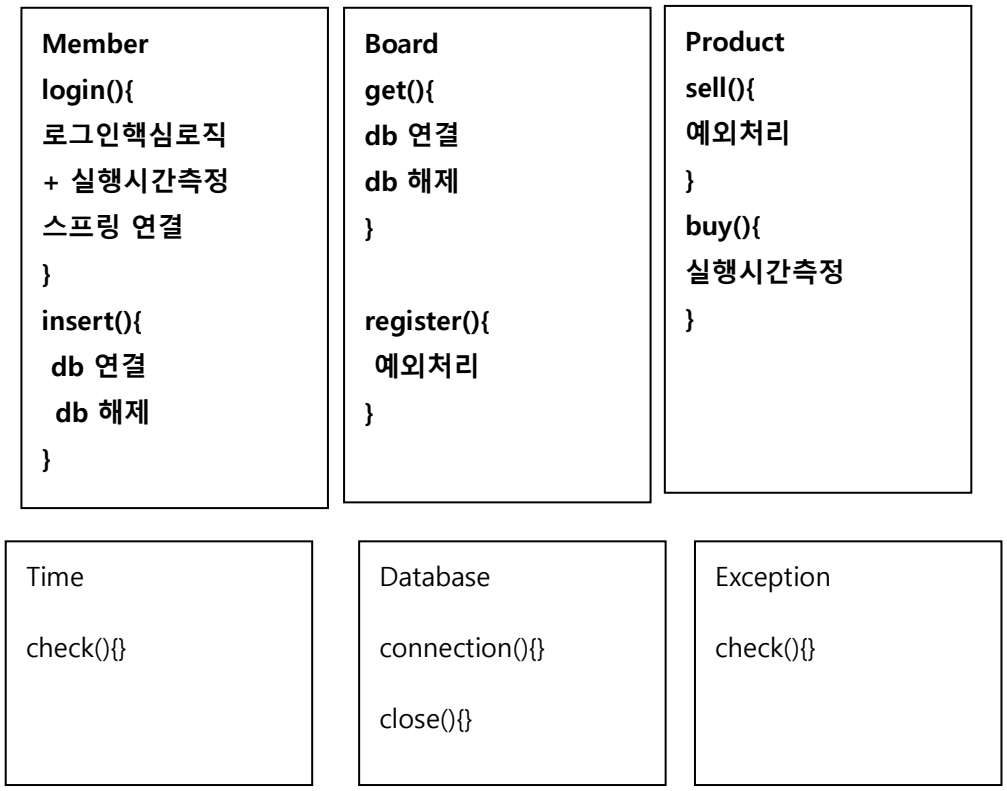
ASPECT 클래스 = 공통관심 = 횡단관심

TARGET 클래스 = 핵심관심 = 종단관심

pointcut=TAGET클래스에 ASPECT을 적용하는 대상과 시점

execution (public int test..*.*(..))

각 3 개의 클래스 공유 기능 – 핵심관심(=종단관심) 구현 중에 핵심관심들 내부 공통 필요 로직=횡단관심=공통관심



-pointcut 문법

<aop:config

<aop:pointcut

```
expression="execution(public * aop1..*.*(..))"
```

```
(modifier 리턴타입 패키지명.클래스명.메소드명(..)
```

* : 모든

(..) : 모든 매개변수

.. : 하위패키지 포함

expression="execution (public int *.*.get*(String))"

1. target클래스 – 핵심관심 구현 클래스들

2. aspect클래스 – 공통관심 구현 클래스들

3. pointcut 에 따라서 1+2 ---> target클래스의 어떤 메소드에 aspect의 어떤 메소드를 끼워넣을지 선정

4-1. target클래스 메소드 실행이전 aspect클래스메소드내용 끼워넣는다

4-2. target클래스 메소드 실행이후 aspect클래스메소드내용 끼워넣는다

4-3. target클래스 메소드 실행이전 /이후 모두 aspect클래스메소드내용 끼워넣는다

실습

1> AOP 스프링 라이브러리 추가

1-1.aspectjweaver.jar-mvnrepository.com

1-2. maven 태그 복사

```
<!-- https://mvnrepository.com/artifact/org.aspectj/aspectjweaver -->
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjweaver</artifactId>
    <version>1.9.9.1</version>
    <scope>runtime</scope>
</dependency>
```

1-3. 프로젝트₩pom.xml ---> MAVEN 기능 사용 설정파일

1-4. MAVEN DEPENDENCIES₩*.jar

2> Aspect 클래스 정의/ target 클래스 정의

target – 핵심관심	aspect
Member	Common
Board	xml 설정 (메소드선정, 실행전/후 시점)

21장 스프링 mvc- WEB

DI, IOC, AOP

dynamic web project ---> non spring mvc

--> 직접 구현 (스프링 mvc 제공 라이브러리 대체)

spring legacy project – spring mvc project

MVC 구조

1> 웹서버

모든 사용 요청 – 서블릿(Controller) - 요청 분석 비지니스 로직 dao, dto, service(MODEL) 객체 일임 –

결과물을 분배 jsp(View) 전달 – 응답

2> SPRING MVC 흉내 구현 FRONTCONTROLLER + MVC

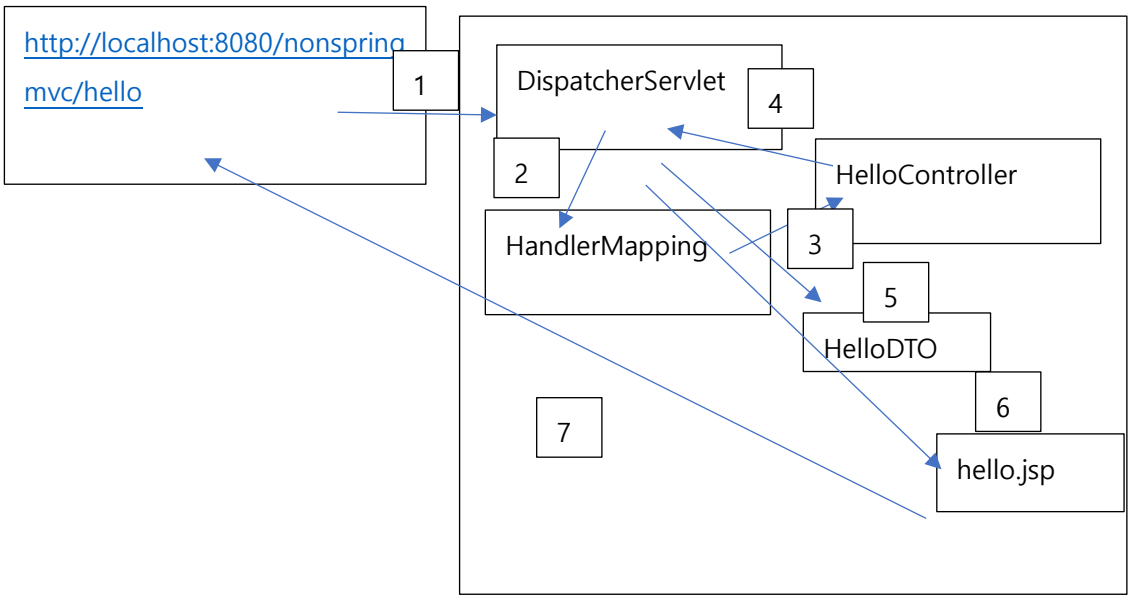
@WebServlet("/") DispatcherServlet	FrontController- 어떤 Controller 호출 handleAndRequest() 호출
HandlerMapping	Controller 찾는 기능 "h"-->HController
Controller 인터페이스	요청 분석 처리 Controller HelloController ListController BoardController
xxxDTO	MODEL 클래스
XXX.jsp	view

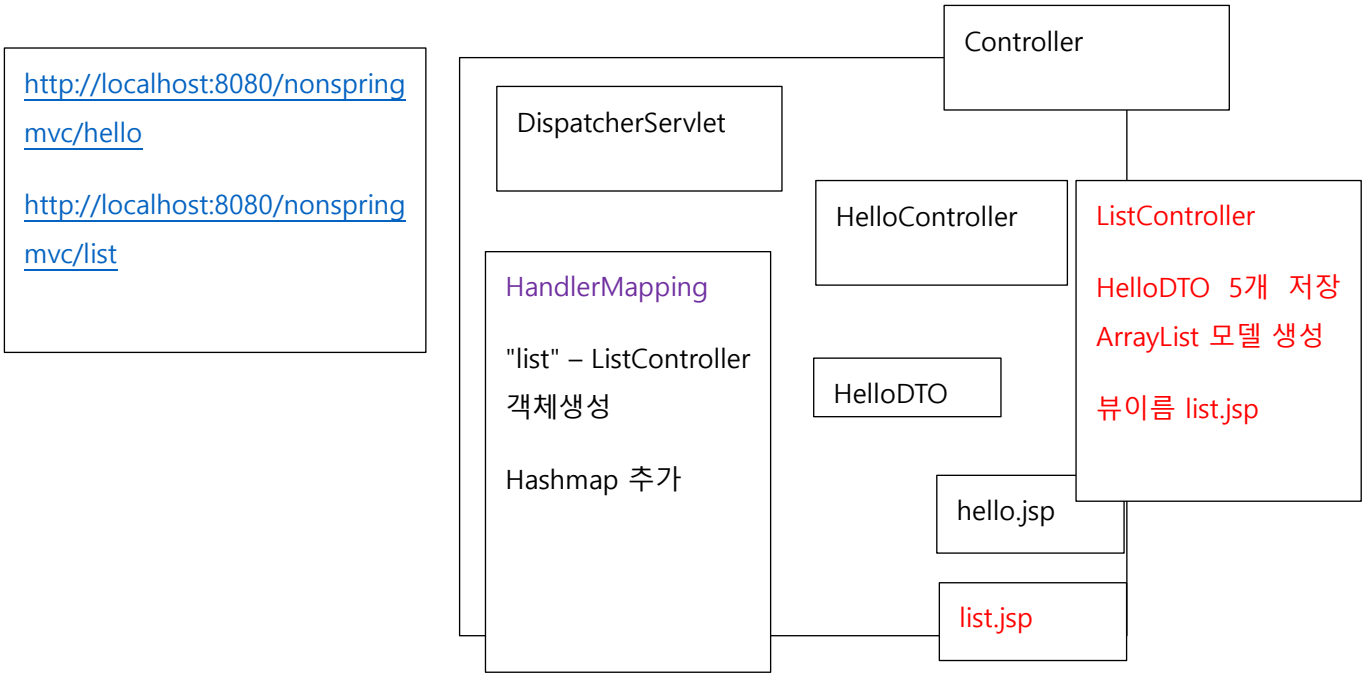
http://127.0.0.1:8080/nonsrpingmvc/list 요청 - x

http://127.0.0.1:8080/nonsrpingmvc/insert

http://127.0.0.1:8080/nonsrpingmvc/login

http://127.0.0.1:8080/nonsrpingmvc/hello - HelloDTO 생성-값 저장 – hello.jsp 전달


http://localhost:8080/nonspring
mvc/hello
1
DispatcherServlet
4
2
HelloController
HandlerMapping
3
5
HelloDTO
6
7
hello.jsp


http://localhost:8080/nonspring
mvc/hello
http://localhost:8080/nonspring
mvc/list
Controller
DispatcherServlet
HelloController
ListController
HelloDTO 5개 저장
ArrayList 모델 생성
뷰이름 list.jsp
HandlerMapping
"list" – ListController
객체생성
Hashmap 추가
HelloDTO
hello.jsp
list.jsp

FrontController + MVC = SPRING MVC

- spring mvc + maven 프로젝트

web.xml	서블릿 매핑, 웰컴파일리스트 web, spring mvc 웹환경설정 파일
pom.xml	maven 라이브러리 다운로드 알려주는 파일
servlet-context.xml	spring mvc 관련설정파일