

Spring Framework

- Mybatis
- 1.Mybatis
- 2.Mybatis 추가하기
- 3.Mybatis Spring 연결
- 4.Mybatis XML 속성

1. MyBatis

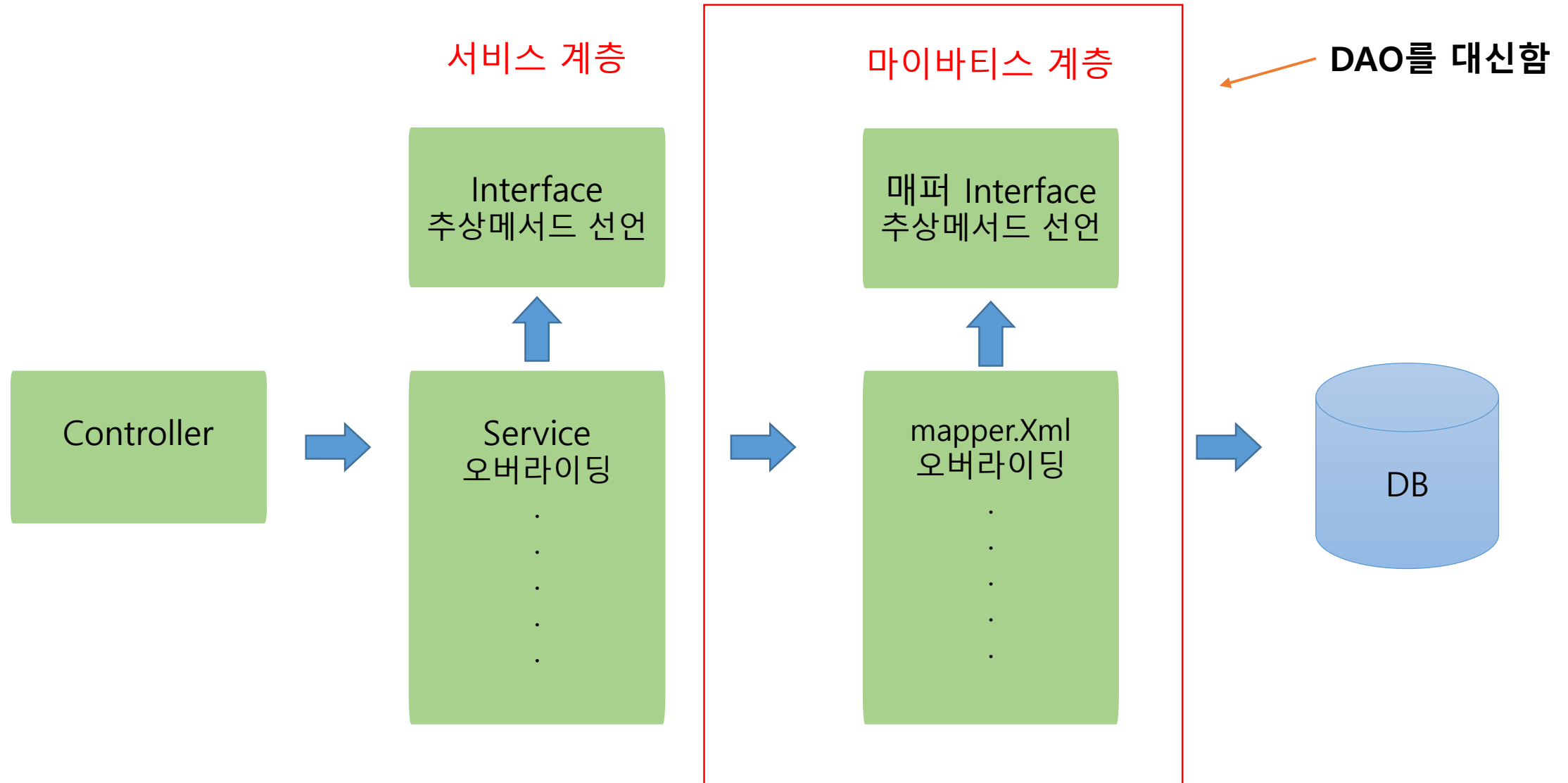
- MyBatis 는 개발자가 지정한 **SQL, 고급 매핑을 지원하는 프레임워크**이다.
- MyBatis 는 JDBC 코드와 수동으로 셋팅하는 파라미터와 결과 매핑을 제거합니다.
- MyBatis 는 복잡한 JDBC코드를 걷어내며 깔끔한 소스코드를 유지합니다.
- MyBatis 는 **DAO계층을 대신**합니다.
- MyBatis 는 기존 **DAO의 Interface의 구현클래스를 xml파일이 대신**합니다.

- 스프링에서 사용하려면 **MyBatis-Spring module을 다운로드** 받아야 한다.

관련 API

<http://www.mybatis.org/mybatis-3/>

1. MyBatis



1. MyBatis

전통적인 JDBC 프로그램	MyBatis
<ol style="list-style-type: none">1. 직접 Connection 생성2. 직접 Close() 처리3. 직접 PreparedStatement 생성4. Pstmt의 setxxx() 직접 처리5. Select의 경우 ResultSet 처리	<ol style="list-style-type: none">1. 자동 Connection 생성2. 자동 Close() 처리3. 자동 PreparedStatement 처리4. #{name} 을 통한 ? 처리5. 리턴 타입으로 자동 ResultSet 처리

2. MyBatis 추가하기

SQLSessionFactory

- 마이바티스 **핵심객체**
- 스프링 컨테이너에 생성해 놓으면 된다

-마이바티스 사용시 기본적으로
spring-jdbc 라이브러리가 있어야 함

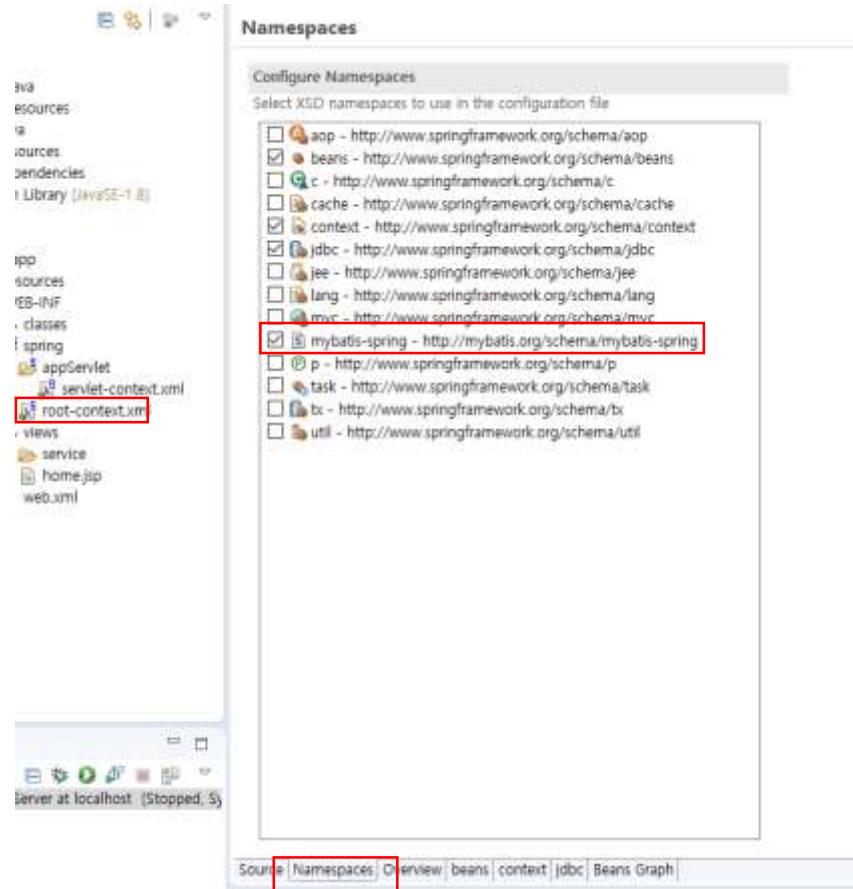
MyBatis를 사용하기 위한
pom.xml 설정 추가

```
<!-- MyBatis 관련 라이브러리 추가 -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.4.6</version>
</dependency>

<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis-spring</artifactId>
    <version>1.3.2</version>
</dependency>
```

2. MyBatis 추가하기

MyBatis를 사용하기 위한
namespace 추가



MyBatis를 사용하기 위한
Root-context.xml 설정 추가

```
<!-- MyBatis의 사용의 핵심 객체 sessionFactory 추가 -->  
<bean id="sqlSessionFactory"  
      class="org.mybatis.spring.SqlSessionFactoryBean">  
    <property name="dataSource" ref="dataSource"></property>  
</bean>
```

```
<mybatis-spring:scan base-package="com.zerock.testmapper"/>
```

코드 해석

- > sqlSessionFactory 이름으로 컨테이너에 객체생성
- > dataSource 메서드에 앞서 생성한 dataSource를 주입시킨다.
- > ref는 참조속성 입니다.
- > mybatis-scan 추가
- > 해당 패키지를 스캔하여 xml파일을 객체(bean)로 생성

2. MyBatis 설정 추가하기

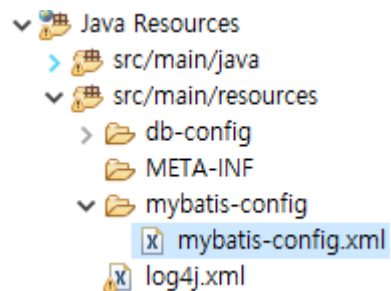
MyBatis를 사용하기 위한
Root-context.xml 설정 추가

```
<!-- MyBatis의 사용의 핵심 객체 sessionFactory 추가 -->
<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="dataSource" ref="dataSource"/>
    <property name="configLocation" value="classpath:/mybatis-config/mybatis-config.xml"/>
</bean>

<mybatis-spring:scan base-package="com.zerock.testmapper"/>
```

코드 해석

`configLocation`은 마이바티스 설정 속성입니다
`classpath:/`는 `src/main/resources`파일 아래의 파일의 경로를 참조하는 방법입니다
`classpath`아래의 파일을 맵핑시켜 마이바티스 설정을 추가할 수 있습니다.



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
    PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-config.dtd">

<!-- 마이바티스 설정관련 xml파일이 됩니다 -->
<configuration>
    <typeAliases>
        <typeAlias type="com.myweb.command.FreeBoardVO" alias="FreeBoardVO"/>
    </typeAliases>
</configuration>
```

3. MyBatis Spring 연결

MyBatis와 Spring의 연결

마이바티스 계층

매퍼 Interface
추상메서드 선언

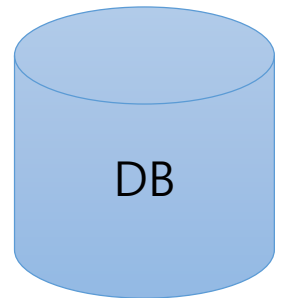


mapper.Xml
오버라이딩

.
. .
. .
. .
. .

```
package com.zerock.testmapper;  
  
public interface TestMapper {  
  
    public String getTime();  
  
}
```

```
<?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">  
  
<!--위 설정을 추가 -->  
  
<mapper namespace="com.zerock.testmapper.TestMapper">  
  
    <select id="getTime" resultType="String">  
        select sysdate() from dual  
    </select>  
  
</mapper>
```



3. MyBatis Mapper XML

```
<?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">

<!--위 설정을 추가 -->

<mapper namespace="com.zerock.testmapper.TestMapper">
  <select id="getTime" resultType="String">
    select sysdate() from dual
  </select>
</mapper>
```



Namespace – 인터페이스의 경로와 동일하게 작성

Id – 인터페이스 추상메서드와 동일하게 작성

resultType – 추상메서드의 리턴타입과 동일하게 작성

3. MyBatis Mapper XML의 주요 속성(중요)

Mapper태그

Namespace – 인터페이스 전체경로 작성(인터페이스 동일한 이름으로 병합해서 처리 함)

Select 속성

Id – 메서드를 찾기위한 구분자(인터페이스 메서드명과 동일)

parameterType – 구문에 전달될 파라미터 타입 (패키지 경로 포함, 전체 클래스명)

resultType – 결과 반환 타입 (패키지 경로 포함, 전체 클래스명)

resultMap – 외부 Map타입을 이용한 반환 타입

Insert, Update, Delete 속성

Id – 메서드를 찾기위한 구분자(인터페이스 메서드명과 동일)

parameterType – 구문에 전달될 파라미터 타입 (패키지 경로 포함, 전체 클래스명)

Sql 구분의 값의 전달

#{name}

부등호 같은 문자를 문자열로 인식시킴

```
<![CDATA[  
    코드  
]]>
```

마이바티스의 두개이상 데이터 맵핑 처리

1. VO클래스 자동맵핑
2. HashMap 자동맵핑
3. @Param 이름지정을 사용해서 맵핑

동적쿼리 지원

1. if
2. chose(when, otherwise)
3. foreach