KoreaIT Spring03

가로선

# web.xml 수정

| <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath\*:config/spring/context/context-\*</param-value> </context-param> <servlet>  <servlet-name>appServlet</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:config/spring/mvc/servlet-context.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup> </servlet> |
| --- |

web.xml-> root-context.xml -> servlet-context.xml 로 이동하며 원래 시작한다.

아래는 xml 한글 세팅

| <filter>  <filter-name>encodingFilter</filter-name>  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>  <init-param>  <param-name>encoding</param-name>  <param-value>utf-8</param-value>  </init-param>  </filter>   <filter-mapping>  <filter-name>encodingFilter</filter-name>  <url-pattern>\*.do</url-pattern>  </filter-mapping> |
| --- |

## <https://mvnrepository.com/> - maven repository

스프링 라이브러리 다운장소

### dbcp

mvnrepository => pom.xml -text부분에 넣어주면 사용가능.

| *<!-- https://mvnrepository.com/artifact/commons-dbcp/commons-dbcp -->* <dependency>  <groupId>commons-dbcp</groupId>  <artifactId>commons-dbcp</artifactId>  <version>1.4</version> </dependency> |
| --- |

### mybatis

| *<!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->* <dependency>  <groupId>org.mybatis</groupId>  <artifactId>mybatis</artifactId>  <version>3.4.5</version> </dependency> |
| --- |

| *<!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->* <dependency>  <groupId>org.mybatis</groupId>  <artifactId>mybatis-spring</artifactId>  <version>1.3.1</version> </dependency> |
| --- |

### spring-jbcd

| *<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->* <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-jdbc</artifactId>  <version>3.2.2.RELEASE</version> </dependency> |
| --- |

## 

## Session 관련 객체 생성

| *<!-- DB연결을 위한 라이브러리 객체 생성 -->* <bean id="ds" class="org.apache.commons.dbcp.BasicDataSource">  <property name="driverClassName" value="${driver}"/>  <property name="url" value="${url}"/>  <property name="username" value="${user}"/>  <property name="password" value="${pwd}"/>  <property name="maxActive" value="10"/> </bean> |
| --- |

| <bean id="factoryBean" class="org.mybatis.spring.SqlSessionFactoryBean">  <property name="dataSource" ref="ds"></property>  *<!-- 어떤 DB로 접근해야 하는지 알게 된다. -->*  <property name="configLocation" value="classpath:config/mybatis/mybatis-config.xml"/>  *<!-- 어떤 매퍼로 접근을 해야하는지 알고 있다. -->* </bean>  *<!-- 매퍼로 접근하는 경로를 아는 factory로 실제로 접근하는 sqlSessino 객체(bean) 의존주입 생성 -->* <bean id="sqlSessionBean" class="org.mybatis.spring.SqlSessionTemplate" >  <constructor-arg ref="factoryBean"/> </bean> |
| --- |

## 문제가 생길시 해결방법(극단적)

1. 프로젝트 클리어
2. .m2 repository 삭제
3. 프로젝트 삭제후 다시생성

## Mybatis를 활용해 사용과정

DAO VO 객체 생성후 context => 에 빈을 생성해주고

| <bean id="dept\_dao" class="dao.DeptDAO">  <property name="sqlSession" ref="sqlSessionBean"/> </bean> |
| --- |

바로 사용가능.

## 프로젝트 생성시 가져가야 할것.

* web.xml
* pom.xml Overview로 접근해서 artifact id, projectname 수정하고 사용
* resources 안에 들어있는 4개의 패키지
* config.mybatis.mapper-> dept.xml 본인이 사용하려고하는 테이블에 맞게 수정
* mybatis -config.xml에 수정된 내용 추가 혹은 삭제
* context-3-dao.xml 내용을 내가 생성할 dao 코드로 변경
* servlet-context.xml에 컨트롤러 수동생성 부분 수정

## DB에서 내용 조회시 순서

1. VO 생성
2. DAO클래스 생성하고 context-3-dao.xml에서 dao 객체(bean)을 생성
3. 2번에서 만든 bean 객체에 context-2-mybatis.xml에서 생성한 sqlSessinoBean을 참조 시켜놔야 한다.
4. Controller를 생성하고 DAO 객체를 setter또는 생성자 인젝션으로 받기위해 DAO클래스 객체를 생성해 둔다.
5. servlet-context.xml에서 4번에서 준비한 DAO클래스를 참조해서 Controller Bean을 생성해 준다
6. DAO에서 Mapper접근 - > 쿼리문 작성
7. 컨트롤러에서 @RequestMapping()해서 url매핑하고 DAO 매서드를 호출한다.
8. 컨트롤러에서 DB를 거쳐서 포워딩된 정보를 jsp에서 호출