http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:rte2:psl:dataaccess:mapper_xml_files

Mapper XML File

MyBatis Mapper XML File (SQL Mapping XML) defines the SQL queries to run Mapper.

You can collect the Parameter Object to run SQL query or auto-bind the result of SQL execution to the Result Object.

The properties available for MyBatis Mapper XML File are as follows:

Mapper XML (SQL Mapping XML)

The following properties are available for Mapper XML File:

- <select> : Mapped syntax SELECT
- <insert> : Mapped syntax INSERT
- <update> : Mapped syntax UPDATE
- <delete> : Mapped syntax DELETE
- <sql> : SQL fraction to be reused in other syntaxes
- <resultMap> : Defines how to map the database result to the objects.
- <cache> : To configure caches for the Namespace of your own
- <cache-ref> : Refers to cache configuration of other namespaces

Sample Configuration

```
<?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="egovframework.rte.psl.dataaccess.DeptMapper"> -- You must configure Namespace for MyBatis

	<resultMap id="deptResult" type="egovframework.rte.psl.dataaccess.vo.DeptVO"> -- [Beware] Property Class for iBatis -> Property Type
		<result property="deptNo" column="DEPT_NO" />
		<result property="deptName" column="DEPT_NAME" />
		<result property="loc" column="LOC" />
	</resultMap>

	<select id="selectDept" parameterType="int" resultMap="deptResult"> -- [Beware] Property parameterClass for iBatis -> Property parameterType
		<![CDATA[
			select DEPT_NO,DEPT_NAME,LOC
			from	DEPT
			where	DEPT_NO = #{deptNo}
		]]>
	</select>

	<insert id="insertDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
		<![CDATA[
			insert into DEPT
				(DEPT_NO,DEPT_NAME,LOC)
			values	(#{deptNo},#{deptName},#{loc})
		]]>
	</insert>

	<update id="updateDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
```

```
        <![CDATA[
                update DEPT
                set     DEPT_NAME = #{deptName},
                         LOC = #{loc}
                where   DEPT_NO = #{deptNo}
        ]]>
    </update>

    <delete id="deleteDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
        <![CDATA[
                delete from DEPT
                where           DEPT_NO = #{deptNo}
        ]]>
    </delete>

</mapper>
```

<select>

You can map the referred database to the object by **<select> when you develop SELECT query**.

All you need to do is to refer to the properties available in <select> and take a look at the settings in <select> referred to in the foregoing sample coding.

- **Property**

Property	Description
id	**The sole identifier that tells SQL query. Used to call the corresponding syntax. (Mandatory)**
parameterType	**Name (or alias) of the class that contains the package name to be sent out to the corresponding syntax.**
parameterMap	<Select> You can refer to the external Parameter Object(<parameterMap>). Use of parameterType or Inline Parameter is recommended over parameterMap.
resultType	**Name (or alias) of the class that contains the package name returning the corresponding syntax.**
resultMap	**You can refer to the external Result Object(<resultMap>), outside of <select>.**
flushCache	Set TRUE to flush caches every time a syntax is called (default: false).
useCache	Set TRUE to use caches. (default: true)
timeout	The timeout before the exception is thrown. Represents the standby time for response after database request (some drivers do not support manual setting of timeout).

- **Example:**

```
    <select id="selectDept" parameterType="int" resultMap="deptResult">
        <![CDATA[
                select DEPT_NO,
                         DEPT_NAME,
                         LOC
                from    DEPT
                where   DEPT_NO = #{deptNo} -- Parameter Binding Name#{property}
        ]]>
    </select>
```

You can use ‘selectDept’to call the foregoing <select> syntax. You can load the ‘int’ type parameter to bind to WHERE statement.

You can then map the corresponding object, as per ‘deptResult’ in <resultMap>, returned.

<resultMap id="deptResult" … /> is defined outside <select>.

<insert>, <update>, <delete>

<Insert> / <update> / <delete> are used to refer to the statements INSERT / UPDATE / DELETE. You can receive the parameters required to amend the database.

You'll need to refer to the properties available in <insert> / <update> / <delete> and check how <insert> / <update> / <delete> are configured.

1. <insert>

- **Property**

Property	Description
id	**The sole identifier that tells SQL query. Used to call the corresponding syntax. (Mandatory)**
parameterType	**Name (or alias) of the class that contains the package name to be sent out to the corresponding syntax.**
flushCache	Set TRUE to flush caches every time a syntax is called (default: false).
timeout	The timeout before the exception is thrown. Represents the standby time for response after database request (some drivers do not support manual setting of timeout).
statementType	Options STATEMENT / PREPARED / CALLABLE are available for MyBatis to work Statement, PreparedStatement or CallableStatement. (default: PREPARED)
useGeneratedKeys	You can enable/disable use of JDBC getGeneratedKeys when receiving the key generated inside the DB. (default: false)
keyProperty	You can manually designate the properties to set the keys returned by the Method getGeneratedKeys or sub-element of selectKey in the Syntax Insert.
keyColumn	Set the name of the column of tables featuring generated keys. Required only for the database where the Column 1 does not contain table information (PostgreSQL, etc.).

- **Example:**

```
<insert id="insertDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
        <![CDATA[
                insert into DEPT (DEPT_NO,DEPT_NAME,LOC)
                values (#{deptNo},#{deptName},#{loc}) -- Parameter Binding Name #{property}
        ]]>
</insert>
```

You can use 'insertDept' to call the syntax <insert> by receiving the parameters of 'egovframework.rte.psl.dataaccess.vo.DeptVO' and inserting thereof to the Paragraph INSERT.

2. <update>

- **Property**

Property	Description
id	**The sole identifier that tells SQL query. Used to call the corresponding syntax. (Mandatory)**
parameterType	**Name (or alias) of the class that contains the package name to be sent out to the corresponding syntax.**
flushCache	Set TRUE to flush caches every time a syntax is called (default: false).
timeout	The timeout before the exception is thrown. Represents the standby time for response after database request (some drivers do not support manual setting of timeout).
statementType	Options STATEMENT / PREPARED / CALLABLE are available for MyBatis to work Statement, PreparedStatement or CallableStatement. (default: PREPARED)

- **Example:**

```
<update id="updateDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
        <![CDATA[
                update DEPT
                set     DEPT_NAME = #{deptName}, -- Parameter Binding Name#{property}
                         LOC = #{loc}
                where   DEPT_NO = #{deptNo}
        ]]>
</update>
```

You can use ‘updateDept’ to call the syntax <update> by receiving the parameters of ‘egovframework.rte.psl.dataaccess.vo.DeptVO’ and binding thereof to the Paragraph UPDATE.

3. <delete>

- **Property**

Property	Description
id	**The sole identifier that tells SQL query. Used to call the corresponding syntax. (Mandatory)**
parameterType	**Name (or alias) of the class that contains the package name to be sent out to the corresponding syntax.**
flushCache	Set TRUE to flush caches every time a syntax is called (default: false).
timeout	The timeout before the exception is thrown. Represents the standby time for response after database request (some drivers do not support manual setting of timeout).
statementType	Options STATEMENT / PREPARED / CALLABLE are available for MyBatis to work Statement, PreparedStatement or CallableStatement. (default: PREPARED)

- **Example:**

```
<delete id="deleteDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
        <![CDATA[
                delete from DEPT
                where DEPT_NO = #{deptNo} -- Parameter Binding Name#{property}
        ]]>
</delete>
```

You can use ‘deleteDEPT’ to call the syntax <delete> by receiving the parameters of ‘egovframework.rte.psl.dataaccess.vo.DeptVO’ and binding thereof to the WHERE statement.

[Note]

You can replace the complex class name designated in parameterType into an alias.

```
<!--In MyBatis Configuration XML File -->
<typeAlias type="egovframework.rte.psl.dataaccess.vo.DeptVO" alias="deptVO" />

<!-- Mapper XML File -->
<delete id="deleteDept" parameterType="deptVO">
        <![CDATA[
                delete from DEPT
                where DEPT_NO = #{deptNo} -- Parameter Binding Name#{property}
        ]]>
</delete>
```

<sql>

The element **<sql> is used when you need to define any SQL syntax to be reused in other syntaxes**.

Work <include refid="ID for <sql>" /> to reuse an SQL syntax.

- **Example:**

```
<sql id="userColumns"> id,username,password </sql>

<select id="selectUers" resultType="map">
        <![CDATA[
                select <include refid="userColumns"/>
                from some_table
                where id = #{id}
        ]]>
</select>
```

Parameters

Refer to the following syntax for how to transfer parameter.

```
<insert id="insertDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO">
        <![CDATA[
                insert into DEPT
                        (DEPT_NO,
                         DEPT_NAME,
                         LOC)
                values  (#{deptNo},
                         #{deptName},
                         #{loc})
        ]]>
</insert>
```

While the object with the class-type egovframework.rte.psl.dataaccess.vo.DeptVO is transferred by way of the mapper objective, the object is to find deptNo, deptName and loc for transfer to the parameter PreparedStatement.

You may have a parameter conain the data type information as follows:
#{property,javaType=int,jdbcType=NUMERIC}

<resultMap>

The element **<resultMap> is used to map the inquired result of SELECT,** replacing the JDBC code when loading data from ResultSet.

- **Property**

Property	Description
id	The sole identifier in Statement that refers to resultMap.
type	Name (or alias) of the class that contains the package name of the object type where the syntax-returned results are to be mapped.

- **Example:**

```
<!-- In MyBatis Configuration XML File -->
<!-- <typeAlias type="egovframework.rte.psl.dataaccess.vo.DeptVO" alias="deptVO" /> -->

<resultMap id="deptResult" type="egovframework.rte.psl.dataaccess.vo.DeptVO">
```

```
<!-- When the class DeptVO is defined as typeAlias, you can also work <resultMap id="deptResult"
type="deptVO">    -->
        <result property="deptNo" column="DEPT_NO" />
        <result property="deptName" column="DEPT_NAME" />
        <result property="loc" column="LOC" />
</resultMap>

<select id="selectDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO"
resultMap="deptResult">
        <![CDATA[
                select DEPT_NO,DEPT_NAME,LOC
                from     DEPT
                where    DEPT_NO = #{deptNo}
        ]]>
</select>
```

The result of SELECT statement is mapped in, **as per the attribute of resultMap, <resultMap id="deptResult">**and returned.

In <select>, you can also use **resultType, instead of resultMap**, to designate the type of class where the result is to be mapped.

If intending to use resultType, your DB column name should match the name of property:

```
<select id="selectDept" parameterType="egovframework.rte.psl.dataaccess.vo.DeptVO"
resultType="egovframework.rte.psl.dataaccess.vo.DeptVO ">
<!-- When the class DeptVO is defined as typeAlias, you can also work <select id="selectDept"
parameterType="deptVO" resultType="deptVO">    -->
        <![CDATA[
                select deptno,deptname,loc
                from     DEPT
                where    deptno= #{deptno}
        ]]>
</select>
```

Note that the foregoing two coding examples are intended to draw comparison between the name of column for SELECT statement and the attribute name of Result Object(DeptVO), searching the attribute setter accordingly and mapping the result thereof.

- **When the name of DB column differs from the name of property**

```
<!-- 1) When using the attribute resultType -->
<select id="selectDept" resultType="deptVO">
        <![CDATA[
                select
                        DEPT_NO         as "deptNo",
                        DEPT_NAME       as "deptName",
                        LOC             as "loc"
                from DEPT
                where DEPT_NO = #{deptNo}
        ]]>
</select>

<!-- 2) When using <resultMap> -->
<resultMap id="deptResult" type="deptVO">
        <result property="deptNo" column="DEPT_NO" />
        <result property="deptName" column="DEPT_NAME" />
        <result property="loc" column="LOC" />
</resultMap>
```

```
<select id="selectDept" parameterType="deptVO" resultMap="deptResult">
        <![CDATA[
                select DEPT_NO,DEPT_NAME,LOC
                from     DEPT
                where    DEPT_NO = #{deptNo}
        ]]>
</select>
```

For Example 1), you need to match the name of property using the column alias.

For Example 2), you need to ensure the name of property matches the name of column by using <result>, the sub-element of <resultMap>.

[Note] You can also use hashmap, etc. to process ResultSet, by mapping in the form of key, which should not be preferred over Java Bean or POJO as a domain model of resultMap.

<cache>

MyBatis features query caching that is easily configurable, with the cache implementor featuring a variety of amendments for easy configuration.

In MyBatis, cache does not come into play unless the local session needs caching, very much improving the performance and dependency. All you need to do to activate caching is just to add the following paragraph in the SQL mapping file.

- **<cache/>**
- Default configuration
    1. The results from <select> statements in a mapping file are cached.
    2. A cache is flushed when the entirety of <insert> / <update> / <delete> statements are called.

Check out MyBatis User Manual for more information on cache configuration.

<cache-ref>

While MyBatis refers to the previous section, a cache is used for the specific Namespace or you can delete caches within the same Namespace.

When you need to share the cache configuration and instances among plural Namespaces,

You can use the cache-ref element to refer to other caches.

- <cache-ref namespace=”com.someone.application.data.SomeMapper” />