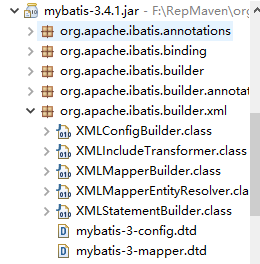
## Xml文件提示问题

通常我们在配置其他xml文件的时候，点alt+/就会出现很多标签的提示供我们选择，而我们刚配好的mybatis全局配置文件，却没有提示。

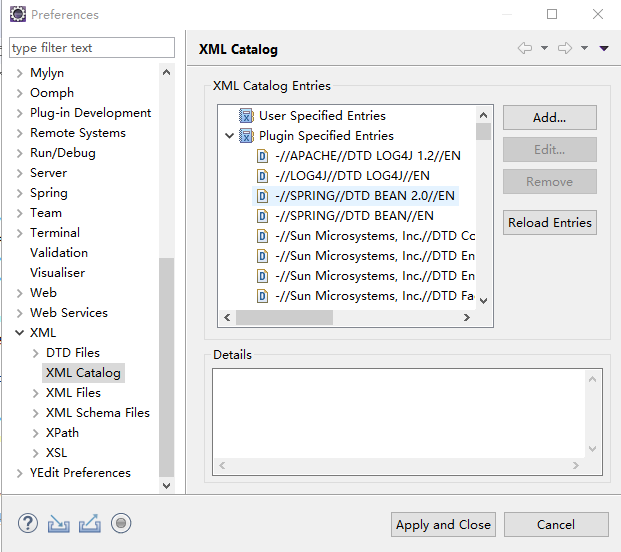
在mybatis全局配置文件的前几行有这样一行代码

"http://mybatis.org/dtd/mybatis-3-config.dtd">

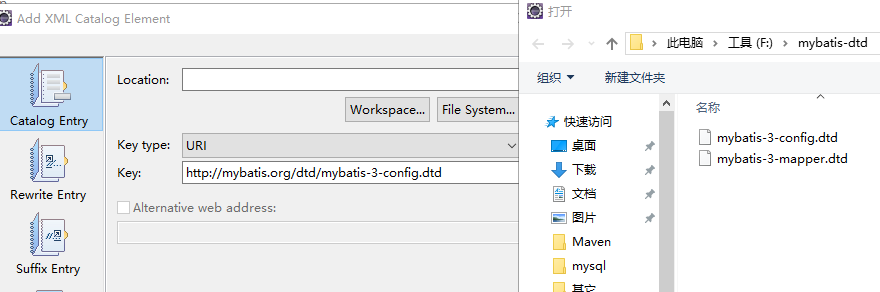
表示dtd约束，只要配置正确的dtd约束，xml文件就会出现正确的提示



在mybatis的jar包中有两个dtd约束文件对应两个配置文件，

1. 找到mybatis的jar包，用压缩文件打开
2. 找到dtd文件，解压
3. 复制<http://mybatis.org/dtd/mybatis-3-config.dtd>
4. 选择Window🡪preferences🡪XML🡪XML Catalog

**选择add**



**这里选择URI**

点击这个按扭找到dtd文件位置

**这里为刚才复制的内容**

然后保存，关闭xml文件重新打开即可

## 引入外部配置文件

<!-- 引入外部properties文件,resource代表资源，这里是类路径下，所以直接下名字即可 -->

<properties resource=*"dbconfig.properties"*></properties>

该文件内容为：

jdbc.driver=com.mysql.jdbc.Driver

jdbc.url=jdbc:mysql:///mybatis

jdbc.uesrname=root

jdbc.password=123456

在配置中就可以这样写

<property name=*"driver"* value=*"${jdbc.driver}"* />

<property name=*"url"* value=*"${jdbc.url}"* />

<property name=*"username"* value=*"${jdbc.username}"* />

<property name=*"password"* value=*"${jdbc.password}"* />

## ★settings标签

这些是极其重要的调整，它们会修改 MyBatis 在运行时的行为方式。下面这个表格描述了设置信息，它们的含义和默认值。

|  |  |  |  |
| --- | --- | --- | --- |
| 设置参数 | 描述 | 有效值 | 默认值 |
| cacheEnabled | **这个配置使全局的映射器启用或禁用缓存** | **true | false** | **true** |
| lazyLoadingEnabled | **全局启用或禁用延迟加载。当禁用时，所有关联对象都会即时加载。** | **true | false** | **true** |
| aggressiveLazyLoading | **当启用时，有延迟加载属性的对象在被调用时将会完全加载任意属性。否则，每种属性将会按需要加载。** | **true | false** | **true** |
| multipleResultSetsEnabled | **允许或不允许多种结果集从一个单独的语句中返回（需要适合的驱动）。** | **true | false** | **true** |
| useColumnLabel | **使用列标签代替列名。不同的驱动在这方便表现不同。参考驱动文档或充分测试两种方法来决定所使用的驱动。** | **true | false** | **true** |
| mapUnderscoreToCamelCase | **是否开启自动驼峰命名规则(camel | case)映射，即总经典数据库中列名A\_COLUMN到经典Java属性名aColumn的类似映射** | **true | false** | **false** |
| useGeneratedKeys | **允许 JDBC 支持生成的键。需要适合的驱动。如果设置为 true 则这个设置强制生成的键被使用，尽管一些驱动拒绝兼容但仍然有效（比如 Derby）。** | **true | false** | **false** |
| autoMappingBehavior | **指定 MyBatis 如何自动映射列到字段/属性。PARTIAL 只会自动映射简单，没有嵌套的结果。FULL 会自动映射任意复杂的结果（嵌套的或其他情况）。** | **NONE,**  **PARTIAL,**  **FULL** | **PARTIAL** |
| defaultExecutorType | **配置默认的执行器。SIMPLE 执行器没有什么特别之处。REUSE 执行器重用预处理语句。BATCH 执行器重用语句和批量更新** | **SIMPLE,**  **REUSE,**  **BATCH** | **SIMPLE** |
| defaultStatementTimeout | **设置超时时间，它决定驱动等待一个数据库响应的时间。** | **Any**  **positive**  **integer**  **Not Set** | **(null)** |

例：<!-- 所有的设置项均在此标签中设置，name为设置项名，value为值 -->

<settings>

<setting name=*"mapUnderscoreToCamelCaseEnables"* value=*"true"*/>

</settings>

## typeAliases别名处理器

在sql映射文件中需要写很多类的全类名，可以通过别名处理器的方式，为某个类起一个别名，以后在使用这个类的时候，写别名即可。

<!-- 为类起别名，别名不区分大小写 -->

<typeAliases>

<!-- **1**. 直接为一个类其别名

type：为类型的全类名，默认别名为类名小写

alias：指定别名

-->

<typeAlias type=*"entities.Employee"* alias=*"emp"*/>

<!-- **2**. 为一个包及其后代包下的所有类批量起别名，都为默认别名（类名小写） -->

<package name=*"entities"*/>

<!-- **3**. 当批量起别名，个别类型别名冲突时，可以使用@Alias注解的方式指定某个类的别名 -->

</typeAliases>

第三种例子：

@Alias(value = "emp")

**public** **class** Employee {}

基本数据类型、基本包装类和常用对应sql数据类型有默认别名，

基本数据类型为 \_数据类型，如\_int

基本包装类和常用对应sql数据类型为 类名小写，如integer

## typeHandlers类型处理器

无论是 MyBatis 在预处理语句中设置一个参数，还是从结果集中取出一个值时，类型处理器被用来将获取的值以合适的方式转换成 Java 类型。

详细参考技术文档



## plugins插件

## environments配置环境

mybatis可以通过全局配置文件的environments标签来定义许多个开放环境，例如多种数据库的环境，获取开发环境和测试环境

下面是配好的一个mysql的环境

可以通过default属性值来切换环境

<environments default=*"development"*>

配置一个环境，id为环境的唯一标示，这个节点必须要有transactionManager子节点和dataSource子节点，否则会报错

<environment id=*"development"*>

配置一个事务处理器，这里写的是系统默认的JDBC事务处理器

两种取值，type=”[JDBC|MANAGED]

<transactionManager type=*"JDBC"* />

配置一个数据源，这里是数据库连接池的数据源，为mybatis自带的数据库连接池

Type取值：UNPOOLED（每次都重新获取一个连接）

POOLED（从数据库连接池中获取连接）

JNDI（在外部配置连接，如用spring）

<dataSource type=*"POOLED"*>

<property name=*"driver"* value=*"${jdbc.driver}"* />

<property name=*"url"* value=*"${jdbc.url}"* />

<property name=*"username"* value=*"${jdbc.username}"* />

<property name=*"password"* value=*"${jdbc.password}"* />

</dataSource>

</environment>

</environments>

## 多种数据库连接

<databaseIdProvider>标签，可以声明多种数据库厂商的别名，加以引用

支持多数据库厂商，增强可移植性，type="DB\_VENDOR"，作用是获取数据库厂商的标识，如MySql，Oracle，Sql Server

<databaseIdProvider type=*"DB\_VENDOR"*>

为浏览器厂商起别名

<property name=*"MySql"* value=*"mysql"*/>

<property name=*"Oracle"* value=*"oracle"*/>

<property name=*"Sql Server"* value=*"sqlservler"*/>

</databaseIdProvider>

在配置sql语句的时候就可以配置各种数据库支持的sql语句了

<通过设置databaseId，在切换数据库厂商时，mybatis会根据厂商标识自动切换sql执行语句

<select id=*"getEmployee"* parameterType=*"int"* resultType=*"emp"* databaseId=*"mysql"*>

select \* from tbl\_employee where id = #{id}

</select>

<select id=*"getEmployee"* parameterType=*"int"* resultType=*"emp"* databaseId=*"oracle"*>

select \* from tbl\_employee where id = #{id}

</select>

**切换数据库厂商，在运行环境<environments>的default属性切换**

## Mappers映射

将我们写好的sql映射文件（EmployeeMapper.xml）一定要注册到全局配置文件（mybatis-config.xml）中

mappers：将sql映射注册到全局配置中

<mappers>

mapper:**注册一个sql映射**

**注册配置文件**

resource：引用类路径下的sql映射文件

mybatis/mapper/EmployeeMapper.xml

url：引用网路路径或者磁盘路径下的sql映射文件

file:///var/mappers/AuthorMapper.xml

**注册接口**

class：引用（注册）接口，需要遵守规则

1、**有sql映射文件，映射文件名必须和接口同名，并且放在与接口同一目录下，可以在配置文件夹中新建相同包名，然后把配置文件放进去，更加整洁**；

2、**没有sql映射文件，所有的sql都是利用注解写在接口上;**

推荐：

比较重要的，复杂的Dao接口我们来写sql映射文件

不重要，简单的Dao接口为了开发快速可以使用注解；

<mapper resource="mybatis/mapper/EmployeeMapper.xml"/>

<mapper class="com.atguigu.mybatis.dao.EmployeeMapperAnno"/>

**批量注册，依然要准守上述规则**

<package name=*"com.atguigu.mybatis.dao"*/>

</mappers>

注解方式例子：

@Select("select \* from tbl\_employee where id = #{id}")

Employee getEmployee(Integer id);

注解有select update delete insert等

## 配置顺序

配置必须严格按照顺序来写，否则会报错

(properties?,settings?,typeAliases?,typeHandlers?,objectFactory?,objectWrapperFactory?,reflectorFactory?,plugins?,environments?,databaseIdProvider?,mappers?)