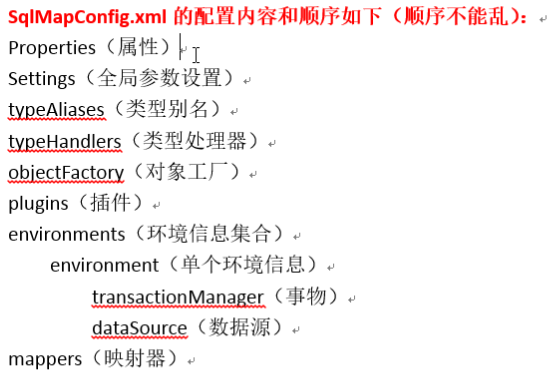
MyBatis的全局配置文件详解

# 官方参考文档

<http://www.mybatis.org/mybatis-3/zh/configuration.html>



官方文档固然全面，不过还是有必要总结一些经常使用到的，毕竟复习的时候好复习。



# MyBatis全局配置文件的命名及位置

**名字**可以随意起，必须是**xml文件**，如名称为MyBatisConfig.xml；

**存放位置**，随意，在创建SessionFactory时加载文件能够找到即可，利用Resource类进行加载。reader = Resources.getResourceAsReader("config/SqlMapConfig.xml");

（与那个**log4j**不一样，log4j的配置文件默认文件名**log4j.properties**或**log4j.xml**，默认情况下放在src下，此时不用手动加载，否则需要显式指出位置）

# MyBatis全局配置文件示例



**<?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">**

<configuration>

<typeAliases>

<typeAlias alias="int" type="java.lang.Integer" />

</typeAliases>

<environments default="development">

<environment id="development">

**<transactionManager** type="JDBC">**</transactionManager>**

**<dataSource** type="POOLED">

<property name="driver" value="com.mysql.jdbc.Driver" />

<property name="url"

value="jdbc:mysql://127.0.0.1:3306/antifraud?characterEncoding=utf-8" />

<property name="username" value="root" />

<property name="password" value="" />

**</dataSource>**

</environment>

</environments>

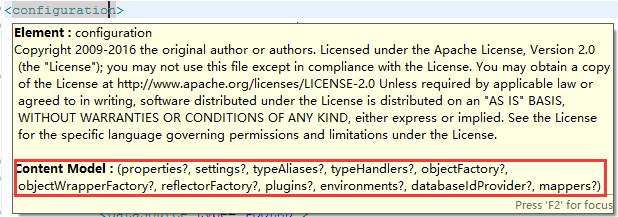
**<mappers>**

**<mapper resource="config/shortmessagelog.xml" />**

**</mappers>**

</configuration>

# 将鼠标指向configuration上：



# 错误

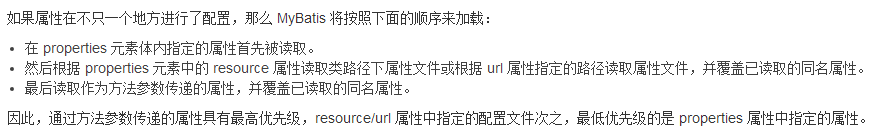
**xml中的标签顺序不能随便调换，否则会提示错误**

# MyBatis全局配置文件解读

## 开头是固定的，拷贝即可，注意中间有个关键字configuration。与映射文件的区别是，映射文件是mapper元素。

## 整体配置信息包含在configuration标签中。内部的标签有一定的顺序，不能随意设置，否则报错。

## properties标签

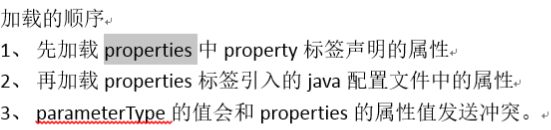


**具有的属性： resource=”db.properties”**

**具有的子标签：property**

**两者中的key重复时，以resource中的为准。为什么？**

**因为加载顺序是：先加载property标签中的name--value，然后再加载resource指定的properties文件。所以以properties文件为准。**





## settings标签：什么时候需要使用查表即可。



## typeAliases标签（可以没有）

类型别名是为 Java 类型设置**一个短的名字**。它只和 XML 配置有关，**存在的意义仅在于用来减少类完全限定名的冗余**。

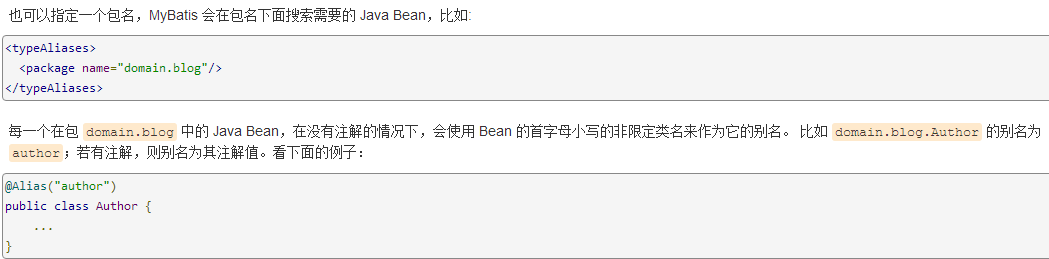
单个加载方式：

<typeAliases>

**<typeAlias alias="int" type="java.lang.Integer" />**

</typeAliases>

**批量加载：package**：默认该包下所有的POJO别名为小写字母名称。（建议）



也可以通过**注解方式**给POJO起别名，利用Spring框架。

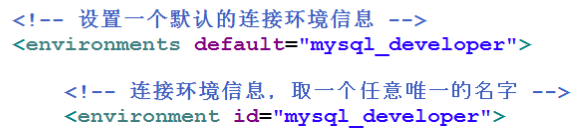
## typeHandlers类型处理器

已经提供了，一般不需要再进行设置了。

## environments标签（重要）配置环境

配置mybatis的环境信息。

一个连接配置就是一个**environment标签**。可以配置多个environment标签，但是需要设置一个默认的。配置文件中可以配置多个，但是同一时间只可以连接一个。

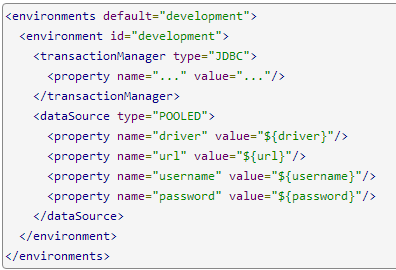


environment标签下，有

**transactionManager事务管理器**

**dataSource数据源：**

**必须至少有四个配置：driver、url、username、password。**



## mappers标签（映射器）



加载mappers映射文件有四种方式：（重要）

### 通过resource加载单个映射文件：mapper-resource

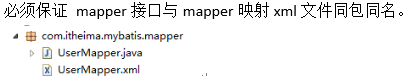
1. **<mappers>**
2. **<mapper** resource="sqlmap/User.xml"**/>**
3. **</mappers>**

### 通过mapper接口加载映射文件class：mapper-class

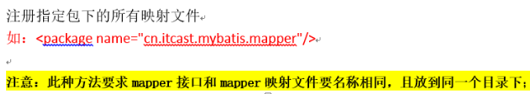
1. <!-- 加载映射文件 -->
2. **<mappers>**
3. <!-- 通过mapper接口加载映射文件 -->
4. <!--遵循规范：**需要将mapper接口类名与xml文件映射名称保持一致，且在一个目录中**。
5. 上边规范的前提是：使用的是**mapper代理的方法** --**>**
6. **<mapper** class="cn.itcast.mybatis.mapper.UserMapper"**/>**
7. **</mappers>**



### 批量加载映射文件mapper-package-name



1. <!-- 加载映射文件 -->
2. **<mappers>**
3. <!-- 批量加载
4. **指定接口的包名称，自动扫描包下的所有mapper接口进行加载** --**>**
5. **<package** name="cn.itcast.mybatis.mapper"**/>**
6. **</mappers>**



### 利用mapper-url方式



# 多个environment

MyBatis可以配置多个环境，比如测试环境和产品环境；但是，一个SqlSessionFactory只能对应一个environment，要想同时使用多个数据库，需要生成多个SqlSessionFactory。

**MyBatis can be configured with multiple environments.** This helps you to apply your SQL Maps to multiple databases for any number of reasons. For example, you might have a different configuration for your Development, **Test** and **Production environments**. Or, you may have multiple production databases that share the same schema, and you’d like to use the same SQL maps for both. There are many use cases.

One important thing to remember though: **While you can configure multiple environments, you can only choose ONE per SqlSessionFactory instance.**

So if you want to connect to two databases, you need to **create two instances of SqlSessionFactory**, one for each. For three databases, you’d need three instances, and so on. It’s really easy to remember:

**One SqlSessionFactory instance per database.**

**如何识别不同的environment呢？**

**每个environment元素都有一个id属性，作为标识该environment。其中environments有个default属性，指定默认的环境。通过指定environments的default属性，就可以设置首选的环境。**

**也可以通过Java代码传入environment对象，指定环境。**

To specify which environment to build, you simply pass it to the SqlSessionFactoryBuilder as an optional parameter. The two signatures that accept the environment are:



**传入environment对象：**

To specify which environment to build, you simply pass it to the SqlSessionFactoryBuilder as an optional parameter. The two signatures that accept the environment are:



**不指定的时候，使用默认的环境：**

If the environment is omitted, then the default environment is loaded, as follows:



**Notice the key sections here:**

### The default Environment ID (e.g. default="development").

### The Environment ID for each environment defined (e.g. id="development").

### The TransactionManager configuration (e.g. type="JDBC")

### The DataSource configuration (e.g. type="POOLED")

The default environment and the environment IDs are self explanatory. Name them whatever you like, just make sure the default matches one of them.