## ibatorConfig.xml文件说明

### <classPathEntry>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| location | 数据库驱动文件路径 | lib/mysql-connector-java-5.1.6-bin.jar |

### <ibatorContext>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| defaultModelType |  | 默认值**conditional** |
| targetRuntime |  | Ibatis2Java2：默认值  Ibatis2Java5： |
| introspectedColumnImpl |  |  |

#### <ibatorPlugin>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| type="org.apache.ibatis.ibator.plugins.RenameExampleClassPlugin" | 生成类名规则：将Example替换成Criteria  <property name="searchString" value="Example$" />  <property name="replaceString" value="Criteria" /> |  |
| type=”(package).EqualsHashCodePlugin” | 在生成的model类中添加equals和hashCode方法 |  |
| type=”(package).SerializablePlugin” | 生成的model类实现java.io.Serializable接口 |  |
| type=”(package).SqlMapConfigPlugin” | 生成SqlMapConfig.xml文件  <property name="targetPackage" value="com.pra.payment.common.generated" />  <property name="targetProject" value="dist/java" /> |  |

#### < jdbcConnection>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| driverClass | 驱动类 | driverClass="com.mysql.jdbc.Driver" |
| connectionURL | 连接URL | connectionURL="jdbc:mysql://localhost:3306/pra\_payment?useUnicode=true&amp;characterEncoding=UTF-8" |
| userId | 用户名 | userId="root" |
| password | 密码 | password="123456" |

#### < javaModelGenerator>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| targetPackage | 生成类所在包 | targetPackage="com.pra.payment.common.generated.model" |
| targetProject | 生成类所在工程 | targetProject="dist/java" |
| rootClass | 配置生成model类继承的基类 | <property name="rootClass" value="com.root.RootClass" /> |

1. <property name="enableSubPackages" value="true" />
2. <property name="trimStrings" value="true" />

#### < sqlMapGenerator>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| targetPackage | 生成类所在包 | targetPackage="com.pra.payment.common.generated.dao" |
| targetProject | 生成类所在工程 | targetProject="dist/java" |

1. <property name="enableSubPackages" value="true" />

#### < daoGenerator>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| type | 指定生成dao类的模板（参见第二章） | type="common.ibator.template.impl.PraPaymentapiDaoTemplate" |
| targetPackage | 生成类所在包 | targetPackage="com.pra.payment.common.generated.dao" |
| targetProject | 生成类所在工程 | targetProject="dist/java" |
| implementationPackage | 生成dao实现类所在包 | implementationPackage="com.pra.payment.common.generated.dao" |
| methodNameCalculator | 生成dao方法名规则  default(默认值)  extended | <property name="methodNameCalculator" value="default" /> |
| rootInterface | 指定生成dao接口类继承的基类 |  |

1. <property name="enableSubPackages" value="true" />
2. <property name="exampleMethodVisibility" value="public" />

#### <table>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| tableName | 数据库表名 | tableName="payment\_task" |
| schema | 指定数据库的schema |  |
| catalog | 指定数据库的catalog |  |
| domainObjectName | 指定生成类的名称 |  |

1. <property name="useActualColumnNames" value="true"/><!-- 默认值false，为true时生成列名与数据库相同 -->
2. <generatedKey column="id" sqlStatement="SELECT LAST\_INSERT\_ID()+1"/><!-- 指定生成主键的语句 -->
3. <columnRenamingRule searchString="payee" replaceString="pay" /><!-- 指定生成列名的替换规则 -->
4. <ignoreColumn column="flag"/><!-- 忽略掉无需生成的列 -->
5. <columnOverride column="id" property="taskId"/><!-- 指定生成的列名 -->

#### <commentGenerator>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| type | 指定生成注解的模板 | 必须实现  org.apache.ibatis.ibator.api.CommentGenerator |
| suppressDate |  | <property name="suppressDate" value="true" /> |

#### <javaTypeResolver>

|  |  |  |
| --- | --- | --- |
| 属性名 | 说明 | 示例 |
| type |  | 必须实现  org.apache.ibatis.ibator.api.JavaTypeResolver |
| forceBigDecimals |  | <property name="forceBigDecimals" value="false" /> |

## Template类说明

必须继承org.apache.ibatis.ibator.generator.ibatis2.dao.templates.AbstractDAOTemplate类

### protected void configureSuperClass()方法

用于配置ibator生成的dao类继承的基类

|  |
| --- |
| protected void configureSuperClass() {  setSuperClass(new FullyQualifiedJavaType(  "com.demo.platform.base.api.dao.BaseDAOImpl"));  } |

### protected void configureConstructorTemplate()方法

用于配置生成的dao类的构造方法

|  |
| --- |
| protected void configureConstructorTemplate() {  Method method = new Method();  method.setConstructor(true);  method.setVisibility(JavaVisibility.PUBLIC);  method.addBodyLine("super();"); //$NON-NLS-1$  setConstructorTemplate(method);  } |

## protected void configureDeleteMethodTemplate()方法

用于配置dao类对表进行删除操作时调用的方法

|  |
| --- |
| protected void configureDeleteMethodTemplate() {  setDeleteMethodTemplate("this.deleteObject(\"{0}.{1}\", {2});");  } |

### protected void configureInsertMethodTemplate()方法

用于配置dao类对表进行插入操作时调用的方法

|  |
| --- |
| protected void configureInsertMethodTemplate() {  setInsertMethodTemplate("this.saveObject(\"{0}.{1}\", {2});");  } |

### protected void configureQueryForListMethodTemplate()方法

用于配置dao类对表进行查询操作时调用的方法

|  |
| --- |
| protected void configureQueryForListMethodTemplate() {  setQueryForListMethodTemplate("this.queryForList(\"{0}.{1}\", {2});");  } |

### protected void configureQueryForObjectMethodTemplate()方法

用于配置dao类对表进行查询操作时调用的方法

|  |
| --- |
| protected void configureQueryForObjectMethodTemplate() {  setQueryForObjectMethodTemplate("this.queryForObject(\"{0}.{1}\", {2});");  } |

### protected void configureUpdateMethodTemplate()

用于配置dao类对表进行更新操作时调用的方法

|  |
| --- |
| protected void configureUpdateMethodTemplate() {  setUpdateMethodTemplate("this.updateObject(\"{0}.{1}\", {2});"); //  } |

### protected void configureCheckedExceptions()

用于配置dao类方法抛出的异常

|  |
| --- |
| protected void configureCheckedExceptions() {  FullyQualifiedJavaType type = new FullyQualifiedJavaType("ExceptionClass");  addCheckedException(type);  } |

### protected void configureFields()

用于配置dao类生成时的属性

|  |
| --- |
| protected void configureFields() {  Field field = new Field();  field.setType(FullyQualifiedJavaType.getStringInstance());//设置属性的类型  field.setName("testField");  field.setStatic(true);  field.setVisibility(JavaVisibility.PUBLIC);  addField(field);  } |

### protected void configureImplementationImports()

用于配置dao的实现类需要导入的包

|  |
| --- |
| protected void configureImplementationImports() {  FullyQualifiedJavaType type = new FullyQualifiedJavaType("com.test.TEST");  addImplementationImport(type);  } |

### protected void configureInterfaceImports()

用于配置dao的接口类需要导入的包

|  |
| --- |
| protected void configureInterfaceImports() {  FullyQualifiedJavaType type = new FullyQualifiedJavaType("com.test.TESTInter");  addInterfaceImport(type);  } |

### protected void configureMethods()

用于配置dao实现类中添加方法

|  |
| --- |
| protected void configureMethods() {  Method method = new Method();  method.setConstructor(false);  method.setName("testMethod");  method.setReturnType(FullyQualifiedJavaType.getStringInstance());  method.setVisibility(JavaVisibility.PUBLIC);  method.addBodyLine("System.out.pringln(“testMethod”);");  addMethod(method);  } |