Atitit mybatis dyna load datasource and mapper.xml 动态加载数据源和映射文件

String driver = "com.mysql.jdbc.Driver";

String url="jdbc:mysql://10.0.1.13:3306/cp?allowMultiQueries=true"; //创建一个表示数据库路径的字符串

String username = "root"; //创建一个表示数据库用户名的字符串

String password = "kdf2016"; //创建一个表示数据库密码的字符串

DriverManagerDataSource driverManagerDataSource =

new DriverManagerDataSource();

driverManagerDataSource.setJdbcUrl(url);driverManagerDataSource.setDriverClass(driver);

driverManagerDataSource.setUser(username);driverManagerDataSource.setPassword(password);

DataSource dataSource = driverManagerDataSource;

TransactionFactory transactionFactory = new JdbcTransactionFactory();

Environment environment = new Environment("development", transactionFactory, dataSource);

Configuration configuration = new Configuration(environment);

// String packageName = "application/dao/mapper";

// packageName=packageName.replaceAll("\\\\", "/");

// List<String> children = VFS.getInstance().list(packageName);

//configuration.addMappers(packageName);//only class pkg path ,,not xml

// configuration.addma

String resource="application/dao/mapper/db.xml";

InputStream inputStream = Resources.getResourceAsStream(resource);

XMLMapperBuilder mapperParser = new XMLMapperBuilder(inputStream, configuration, resource, configuration.getSqlFragments());

mapperParser.parse();

// configuration.addMapper(BlogMapper.class);

SqlSessionFactory sqlSessionFactory = new SqlSessionFactoryBuilder().build(configuration) ;

List li= sqlSessionFactory.openSession(true).selectList("nmspc.query","select 1");

System.***out***.println(sqlSessionFactory);

System.***out***.println(li);

SqlSessionFactoryBuilder 源码 build依靠xml文件

public class XMLConfigBuilder extends BaseBuilder {

private void mapperElement(XNode **parent**) throws Exception {

if (**parent** != null) {

for (XNode child : **parent**.getChildren()) {

if ("package".equals(child.getName())) {

String mapperPackage = child.getStringAttribute("name");

configuration.addMappers(mapperPackage);

} else {

String resource = child.getStringAttribute("resource");

String url = child.getStringAttribute("url");

String mapperClass = child.getStringAttribute("class");

if (resource != null && url == null && mapperClass == null) {

ErrorContext.*instance*().resource(resource);

InputStream inputStream = Resources.*getResourceAsStream*(resource);

XMLMapperBuilder mapperParser = new XMLMapperBuilder(inputStream, configuration, resource, configuration.getSqlFragments());

mapperParser.parse();

} else if (resource == null && url != null && mapperClass == null) {

ErrorContext.*instance*().resource(url);

InputStream inputStream = Resources.*getUrlAsStream*(url);

XMLMapperBuilder mapperParser = new XMLMapperBuilder(inputStream, configuration, url, configuration.getSqlFragments());

mapperParser.parse();

} else if (resource == null && url == null && mapperClass != null) {

Class<?> mapperInterface = Resources.*classForName*(mapperClass);

configuration.addMapper(mapperInterface);

} else {

throw new BuilderException("A mapper element may only specify a url, resource or class, but not more than one.");

}

}

}

}

Code

package sun.misc;

import java.io.IOException;

import java.io.InputStream;

import java.util.List;

import javax.sql.DataSource;

import org.apache.ibatis.builder.xml.XMLMapperBuilder;

import org.apache.ibatis.io.Resources;

import org.apache.ibatis.io.VFS;

import org.apache.ibatis.mapping.Environment;

import org.apache.ibatis.session.Configuration;

import org.apache.ibatis.session.SqlSessionFactory;

import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import org.apache.ibatis.transaction.TransactionFactory;

import org.apache.ibatis.transaction.jdbc.JdbcTransactionFactory;

import com.mchange.v2.c3p0.DriverManagerDataSource;

public class MybatisDynaSqlfactry {

public static void main(String[] **args**) throws IOException {

//n style="font-size:18px;">

String driver = "com.mysql.jdbc.Driver";

String url="jdbc:mysql://10.0.1.13:3306/cp?allowMultiQueries=true"; //创建一个表示数据库路径的字符串

String username = "root"; //创建一个表示数据库用户名的字符串

String password = "kdf2016"; //创建一个表示数据库密码的字符串

DriverManagerDataSource driverManagerDataSource =

new DriverManagerDataSource();

driverManagerDataSource.setJdbcUrl(url);driverManagerDataSource.setDriverClass(driver);

driverManagerDataSource.setUser(username);driverManagerDataSource.setPassword(password);

DataSource dataSource = driverManagerDataSource;

TransactionFactory transactionFactory = new JdbcTransactionFactory();

Environment environment = new Environment("development", transactionFactory, dataSource);

Configuration configuration = new Configuration(environment);

// String packageName = "application/dao/mapper";

// packageName=packageName.replaceAll("\\\\", "/");

// List<String> children = VFS.getInstance().list(packageName);

//configuration.addMappers(packageName);//only class pkg path ,,not xml

// configuration.addma

String resource="application/dao/mapper/db.xml";

InputStream inputStream = Resources.*getResourceAsStream*(resource);

XMLMapperBuilder mapperParser = new XMLMapperBuilder(inputStream, configuration, resource, configuration.getSqlFragments());

mapperParser.parse();

// configuration.addMapper(BlogMapper.class);

SqlSessionFactory sqlSessionFactory = new SqlSessionFactoryBuilder().build(configuration) ;

List li= sqlSessionFactory.openSession(true).selectList("nmspc.query","select 1");

System.***out***.println(sqlSessionFactory);

System.***out***.println(li);

}

}