<http://blog.csdn.net/qq_32588349/article/details/51464936>

* Maven配置

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>4.2.1.Final</version>

</dependency>

* 解决Hibernate在Maven中获取不到非resource下配置文件

<build>

<resource>

<directory>src/main/java</directory>

<includes>

<include>xml路径</include>

</includes>

</resource>

</build>

* hibernate.cfg.xml配置（陆续更新）

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<!--数据库连接-->

<property name="connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="connection.url">jdbc:mysql://localhost:3306/hibernate\_data</property>

<property name="connection.username">Kiseki\_BCC</property>

<property name="connection.password">root</property>

<!--辅助参数-->

<!--数据库方言-->

<property name="dialect">org.hibernate.dialect.MySQL5Dialect</property>

<!--在控制台显示生成的SQL语句,便于学习-->

<property name="show\_sql">true</property>

<!--格式化控制台显示的SQL语句,便于查看-->

<property name="format\_sql">true</property>

<!--create-drop:表示在hebarinate初始化时创建表格,程序运行结束的时候会删除相应的表格。

create:在hibernate初始化时会创建表格，在运行结束之后不删除表格，

而是在下一次运行的时候如果有旧的删掉，没有旧的，重新建表格

update:只是根据映射文件去和数据库中的表对应起来，如果不一致，就更新表的结构

validate:校验映射文件和数据库中的表是不是能对应起来，不能对应报错，实际中常用-->

<property name="hbm2ddl.auto">validate</property>

<!--当前会话线程绑定-->

<property name="current\_session\_context\_class">thread</property>

<!--批处理-->

<property name="hibernate.jdbc.batch\_size">10</property>

<!--关闭二级缓存-->

<property name="hibernate.cache.use\_second\_level\_cache">false</property>

<!--映射文件-->

<mapping resource="com/bc/pojo/Emp.hbm.xml"/>

<mapping resource="com/bc/pojo/Dept.hbm.xml"/>

</session-factory>

</hibernate-configuration>

* HibernateSessionFactory类

package com.bc.util;

import org.hibernate.HibernateException;

import org.hibernate.Session;

import org.hibernate.cfg.Configuration;

import org.hibernate.service.ServiceRegistry;

import org.hibernate.service.ServiceRegistryBuilder;

/\*\*

\* Configures and provides access to Hibernate sessions, tied to the

\* current thread of execution. Follows the Thread Local Session

\* pattern, see {@link http://hibernate.org/42.html }.

\*/

public class HibernateSessionFactory {

/\*\*

\* Location of hibernate.cfg.xml file.

\* Location should be on the classpath as Hibernate uses

\* #resourceAsStream style lookup for its configuration file.

\* The default classpath location of the hibernate config file is

\* in the default package. Use #setConfigFile() to update

\* the location of the configuration file for the current session.

\*/

private static final ThreadLocal<Session> threadLocal = new ThreadLocal<Session>();

private static org.hibernate.SessionFactory sessionFactory;

private static Configuration configuration = new Configuration();

private static ServiceRegistry serviceRegistry;

static {

try {

configuration.configure();

serviceRegistry = new ServiceRegistryBuilder().applySettings(configuration.getProperties()).buildServiceRegistry();

sessionFactory = configuration.buildSessionFactory(serviceRegistry);

} catch (Exception e) {

System.err.println("%%%% Error Creating SessionFactory %%%%");

e.printStackTrace();

}

}

private HibernateSessionFactory() {

}

/\*\*

\* Returns the ThreadLocal Session instance. Lazy initialize

\* the <code>SessionFactory</code> if needed.

\*

\* @return Session

\* @throws HibernateException

\*/

public static Session getSession() throws HibernateException {

Session session = (Session) threadLocal.get();

if (session == null || !session.isOpen()) {

if (sessionFactory == null) {

rebuildSessionFactory();

}

session = (sessionFactory != null) ? sessionFactory.openSession()

: null;

threadLocal.set(session);

}

return session;

}

/\*\*

\* Rebuild hibernate session factory

\*

\*/

public static void rebuildSessionFactory() {

try {

configuration.configure();

serviceRegistry = new ServiceRegistryBuilder().applySettings(configuration.getProperties()).buildServiceRegistry();

sessionFactory = configuration.buildSessionFactory(serviceRegistry);

} catch (Exception e) {

System.err.println("%%%% Error Creating SessionFactory %%%%");

e.printStackTrace();

}

}

/\*\*

\* Close the single hibernate session instance.

\*

\* @throws HibernateException

\*/

public static void closeSession() throws HibernateException {

Session session = (Session) threadLocal.get();

threadLocal.set(null);

if (session != null) {

session.close();

}

}

/\*\*

\* return session factory

\*

\*/

public static org.hibernate.SessionFactory getSessionFactory() {

return sessionFactory;

}

/\*\*

\* return hibernate configuration

\*

\*/

public static Configuration getConfiguration() {

return configuration;

}

}

版权声明：本文为博主原创文章，转载请注明出处。