<https://blog.csdn.net/wangwei_cq/article/details/8590120>

**MyBatis扩展连接池**

2013年02月19日 10:36:20

阅读数：6060

下载MyBatis的包，3.06版本

**(1) 下载MyBatis 要使用的JAR  mybatis-3.0.6.jar 到项目文件夹**

同时下载dbcp的jar到项目，3个jar

commons-collections-3.2.1.jar, commons-dbcp-1.4.jar, commons-pool-1.6.jar

**(2) 配置MyBatis的主xml**

新建一个XML文件，保存为dbconfig.xml，下面是示例内容

下面有几个注意的地方

**1. 引用properties文件来配置db的参数**

   引入文件

    <properties resource="application.properties">  
    </properties>

   访问值  "${username}"                这个实际上取的是 application.properties里面的 username = xxx的值

   这样做的好处就是将经常要修改的定义放到另外的独立的简单配置文件，而不需要用户来修改配置这个主xml

**2.. 使用自定义连接池，如DBCP**

MyBatis不能直接使用外部连接池，需要

1. 编写一个类，实现接口类：org.apache.ibatis.datasource.DataSourceFactory;

public class DBCPDataSourceFactory implements DataSourceFactory {  
    private BasicDataSource datasource = null;  
      
    public DBCPDataSourceFactory(){  
        this.datasource = new BasicDataSource();  
    }  
      
    public DataSource getDataSource() {  
        // TODO Auto-generated method stub  
        return datasource;  
    }  
  
    public void setProperties(Properties ps) {  
        datasource.setDriverClassName( ps.getProperty("driverclassname"));  
        datasource.setUsername( ps.getProperty("username"));  
        datasource.setUrl( ps.getProperty("url"));  
        datasource.setPassword( ps.getProperty("password"));  
        datasource.setDefaultAutoCommit( ps.getProperty("defaultautocommit","0").equals("1") );  
        datasource.setInitialSize( Integer.parseInt(ps.getProperty("initialsize","2")) );  
        datasource.setMaxActive( Integer.parseInt(ps.getProperty("maxactive","20")));  
        datasource.setMaxIdle( Integer.parseInt(ps.getProperty("maxidle","0")));  
        datasource.setMaxWait( Long.parseLong(ps.getProperty("maxwait","0")));          
    }  
}

首先是使用了别名定义了我继承实现的类DBCPDataSourceFactory，然后dataSource指向了这个

然然后特别提醒，这个xml配置严格讲究顺序，比如properties 不能配置到typeAliases后面，否则就会出错而不能加载xml

1. 编写一个类，实现接口类：org.apache.ibatis.datasource.DataSourceFactory;

里面定义一个成员DataSource，在getDataSource里面返回他，在setProperties里面给他赋值属性

public class DBCPDataSourceFactory implements DataSourceFactory {  
    private BasicDataSource datasource = null;  
      
    public DBCPDataSourceFactory(){  
        this.datasource = new BasicDataSource();  
    }  
      
    public DataSource getDataSource() {  
        // TODO Auto-generated method stub  
        return datasource;  
    }  
  
    public void setProperties(Properties ps) {  
        datasource.setDriverClassName( ps.getProperty("driverclassname"));  
        datasource.setUsername( ps.getProperty("username"));  
        datasource.setUrl( ps.getProperty("url"));  
        datasource.setPassword( ps.getProperty("password"));  
        datasource.setDefaultAutoCommit( ps.getProperty("defaultautocommit","0").equals("1") );  
        datasource.setInitialSize( Integer.parseInt(ps.getProperty("initialsize","2")) );  
        datasource.setMaxActive( Integer.parseInt(ps.getProperty("maxactive","20")));  
        datasource.setMaxIdle( Integer.parseInt(ps.getProperty("maxidle","0")));  
        datasource.setMaxWait( Long.parseLong(ps.getProperty("maxwait","0")));          
    }  
}

**3. 引入各个模块的配置文件**

    <mappers>  
        <mapper resource="com/talkyun/iaas/util/db/sqlmap/rhea\_kvm.xml" />  
    </mappers>

**4. 配置举例如下：**

<?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>  
    <properties resource="application.properties">  
    </properties>  
    <typeAliases>  
        <typeAlias type="com.talkyun.iaas.util.db.DBCPDataSourceFactory" alias="RHEA\_DBCP" />  
    </typeAliases>  
  
    <environments default="development">  
        <environment id="development">  
            <transactionManager type="JDBC" />  
            <dataSource type="RHEA\_DBCP">  
                <property name="driverclassname" value="${driverclassname}" />  
                <property name="url" value="${url}" />  
                <property name="username" value="${username}" />  
                <property name="password" value="${password}" />  
                <property name="initialsize" value="${initialsize}" />  
                <property name="maxactive" value="${maxactive}" />  
            </dataSource>  
        </environment>  
    </environments>  
      
    <mappers>  
        <mapper resource="com/talkyun/iaas/util/db/sqlmap/rhea\_kvm.xml" />  
        <mapper resource="com/talkyun/iaas/util/db/sqlmap/rhea\_server.xml" />  
        <mapper resource="com/talkyun/iaas/util/db/sqlmap/rhea\_group.xml" />  
        <mapper resource="com/talkyun/iaas/util/db/sqlmap/rhea\_user.xml" />  
        <mapper resource="com/talkyun/iaas/util/db/sqlmap/rhea\_kvm\_template.xml" />  
    </mappers>  
</configuration>

**(3) 上面都配置好后，就测试加载**

import java.io.Reader;  
import java.sql.Connection;  
  
import org.apache.ibatis.io.Resources;  
import org.apache.ibatis.session.SqlSession;  
import org.apache.ibatis.session.SqlSessionFactory;  
import org.apache.ibatis.session.SqlSessionFactoryBuilder;  
  
private static SqlSessionFactory sqlMapper = null;

            String resource = "com/db/config.xml";  
            Reader reader = Resources.getResourceAsReader(resource);  
            sqlMapper = new SqlSessionFactoryBuilder().build(reader);              
            SqlSession ss = sqlMapper.openSession();  
如果没有报异常，说明程序连接正常，可以开始工作