# Mybatis数据源与连接池

  https://blog.csdn.net/luanlouis/article/details/37671851

对于ORM框架而言，数据源的组织是一个非常重要的一部分，这直接影响到框架的性能问题。本文将通过对MyBatis框架的数据源结构进行详尽的分析，并且深入解析MyBatis的连接池。

    本文首先会讲述MyBatis的数据源的分类，然后会介绍数据源是如何加载和使用的。紧接着将分类介绍UNPOOLED、POOLED和JNDI类型的数据源组织；期间我们会重点讲解POOLED类型的数据源和其实现的连接池原理。

以下是本章的组织结构：

一、MyBatis数据源DataSource分类

二、数据源DataSource的创建过程

三、 DataSource什么时候创建Connection对象

四、不使用连接池的UnpooledDataSource

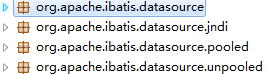
五、为什么要使用连接池？

六、使用了连接池的PooledDataSource

————————————————

## 一、MyBatis数据源DataSource分类

MyBatis数据源实现是在以下四个包中：



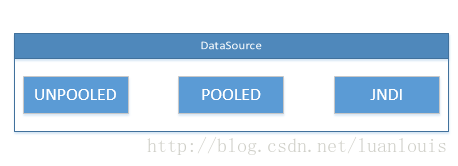
MyBatis把数据源DataSource分为三种：

         UNPOOLED    不使用连接池的数据源

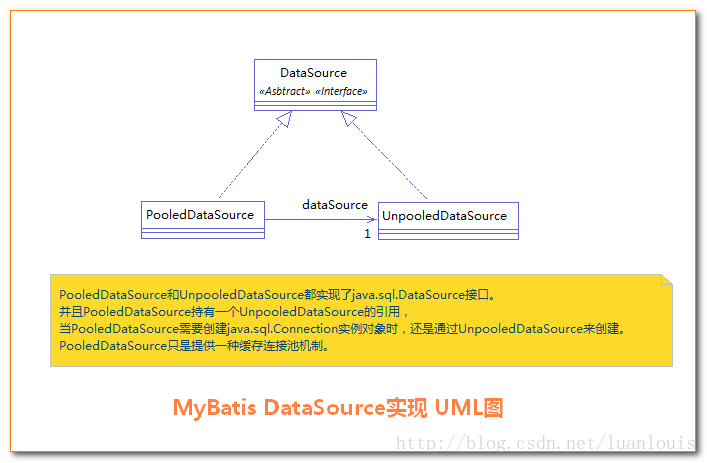
         POOLED      使用连接池的数据源

         JNDI            使用JNDI实现的数据源

即：



相应地，MyBatis内部分别定义了实现了java.sql.DataSource接口的UnpooledDataSource，PooledDataSource类来表示UNPOOLED、POOLED类型的数据源。 如下图所示：



对于JNDI类型的数据源DataSource，则是通过JNDI上下文中取值。

————————————————

## 二、数据源DataSource的创建过程

# 1.说明演示1

## 说明

MyBatis内置了数据源的支持，如：

1. **<environments** default="development"**>**
2. **<environment** id="development"**>**
3. **<transactionManager** type="JDBC" **/>**
4. **<dataSource** type="POOLED"**>**
5. **<property** name="driver" value="${jdbc.driverClassName}" **/>**
6. **<property** name="url" value="${jdbc.url}" **/>**
7. **<property** name="username" value="${jdbc.username}" **/>**
8. **<property** name="password" value="${jdbc.password}" **/>**
9. **</dataSource>**
10. **</environment>**

type为POOLED的数据源是MyBatis内置的数据源类型(它是一个类型别名)。POOLED对应的数据库连接池工厂是org.apache.ibatis.datasource.pooled.PooledDataSourceFactory,它必须实现接口 org.apache.ibatis.datasource.DataSourceFactory

1. **package** org.apache.ibatis.datasource.pooled;
3. **import** org.apache.ibatis.datasource.unpooled.UnpooledDataSourceFactory;
5. **public** **class** PooledDataSourceFactory **extends** UnpooledDataSourceFactory {
7. **public** PooledDataSourceFactory() {
8. **this**.dataSource = **new** PooledDataSource();
9. }
11. }

目前有各种各样的数据源实现，如果在项目中要使用第三方的数据源，如何将它配置到MyBatis中呢？通过实现 org.apache.ibatis.datasource.DataSourceFactory将数据源集成到MyBatis中。本文以Druid数据源为例。

## 实现DataSourceFactory接口

package officialcn.datasource.DruidDataSource;  
  
*/\*\*  
 \** ***@Author*** *了个  
 \** ***@date*** *2020/1/10 9:31  
 \*/*import com.alibaba.druid.pool.DruidDataSource;  
import org.apache.ibatis.datasource.DataSourceFactory;  
  
import javax.sql.DataSource;  
import java.sql.SQLException;  
import java.util.Properties;  
  
public class DruidDataSourceFactory implements DataSourceFactory {  
 private Properties props;  
 @Override  
 public void setProperties(Properties props) {  
 this.props = props;  
 }  
  
 @Override  
 public DataSource getDataSource() {  
 DruidDataSource dds = new DruidDataSource();  
 dds.setDriverClassName(this.props.getProperty("driver"));  
 dds.setUrl(this.props.getProperty("url"));  
 dds.setPassword(this.props.getProperty("password"));  
 dds.setUsername(this.props.getProperty("username"));  
 //其他配置可以根据MyBatis主配置文件进行配置  
  
 //<!-- 配置初始化大小、最小、最大 -->  
 //<property name="initialSize" value="1" />  
 //<property name="minIdle" value="1" />  
 //<property name="maxActive" value="20" />  
 //  
 //<!-- 配置获取连接等待超时的时间 -->  
 //<property name="maxWait" value="60000" />  
 //<!-- 配置间隔多久才进行一次检测，检测需要关闭的空闲连接，单位是毫秒 -->  
 //<property name="timeBetweenEvictionRunsMillis" value="3000" />  
 //<!-- 配置一个连接在池中最小生存的时间，单位是毫秒 -->  
 //<property name="minEvictableIdleTimeMillis" value="300000" />  
 //  
 //<property name="validationQuery" value="SELECT 'x' FROM DUAL" />  
 //<property name="testWhileIdle" value="true" />  
 //<property name="testOnBorrow" value="false" />  
 //<property name="testOnReturn" value="false" />  
 //<!-- mysql 不支持 poolPreparedStatements-->  
 //<!--<property name="poolPreparedStatements" value="true" />-->  
 //<!--<property name="maxPoolPreparedStatementPerConnectionSize" value="20" />-->  
 //  
 //<!-- 开启Druid的监控统计功能 掉后监控界面sql无法统计-->  
 //<property name="filters" value="stat" />  
  
 dds.setInitialSize(Integer.*valueOf*(this.props.getProperty("initialSize")));  
 dds.setMinIdle(Integer.*valueOf*(this.props.getProperty("minIdle")));  
 dds.setMaxActive(Integer.*valueOf*(this.props.getProperty("maxActive")));  
  
 dds.setMaxWait(Long.*valueOf*(this.props.getProperty("initialSize")));  
 dds.setTimeBetweenEvictionRunsMillis(Long.*valueOf*(this.props.getProperty("timeBetweenEvictionRunsMillis")));  
 dds.setMinEvictableIdleTimeMillis(Long.*valueOf*(this.props.getProperty("minEvictableIdleTimeMillis")));  
  
 try {  
 dds.init();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return dds;  
 }  
}

## Mybatis配置文件

<typeAliases>  
 <typeAlias alias="DRUID" type="officialcn.datasource.DruidDataSource.DruidDataSourceFactory"/>

<!-- 配置环境 -->  
<environments default="development">  
  
  
 <environment id="development">  
 <transactionManager type="JDBC" />  
 <!--<dataSource type="officialcn.datasource.DruidDataSource.DruidDataSourceFactory">-->  
 <dataSource type="DRUID">  
 <property name="driver" value="${driver}"/>  
 <property name="url" value="${url}" />  
 <property name="username" value="${username?:root}" />  
 <property name="password" value="${password?:root}" />  
 <!-- 配置初始化大小、最小、最大 -->  
 <property name="initialSize" value="1" />  
 <property name="minIdle" value="1" />  
 <property name="maxActive" value="20" />  
  
 <!-- 配置获取连接等待超时的时间 -->  
 <property name="maxWait" value="60000" />  
 <!-- 配置间隔多久才进行一次检测，检测需要关闭的空闲连接，单位是毫秒 -->  
 <property name="timeBetweenEvictionRunsMillis" value="3000" />  
 <!-- 配置一个连接在池中最小生存的时间，单位是毫秒 -->  
 <property name="minEvictableIdleTimeMillis" value="300000" />  
  
 <property name="validationQuery" value="SELECT 'x' FROM DUAL" />  
 <property name="testWhileIdle" value="true" />  
 <property name="testOnBorrow" value="false" />  
 <property name="testOnReturn" value="false" />  
 <!-- mysql 不支持 poolPreparedStatements-->  
 <!--<property name="poolPreparedStatements" value="true" />-->  
 <!--<property name="maxPoolPreparedStatementPerConnectionSize" value="20" />-->  
  
 <!-- 开启Druid的监控统计功能 掉后监控界面sql无法统计-->  
 <property name="filters" value="stat" />  
 </dataSource>  
 </environment>  
  
 <!-- 配置mysql的环境-->  
 <environment id="mysql">  
 <!-- 配置事务的类型-->  
 <transactionManager type="JDBC"></transactionManager>  
 <!-- 配置数据源（连接池） -->  
 <dataSource type="POOLED">  
 <!-- 配置连接数据库的4个基本信息 -->  
 <property name="driver" value="${driver}"/>  
 <property name="url" value="${url}"/>  
 <!-- 如果属性 'username' 没有被配置，'username' 属性的值将为 'ut\_user' -->  
 <!--对应上面properties配置中的1：启动默认值特性，2：修改默认值的分隔符-->  
 <property name="username" value="${username?:root}"/>  
 <property name="password" value="${password?:root}"/>  
 </dataSource>  
 </environment>  
</environments>