DbUtils

**什么是DbUtils？**

Commons DbUtils是[Apache](https://baike.baidu.com/item/Apache/8512995)组织提供的一个对[JDBC](https://baike.baidu.com/item/JDBC)进行简单封装的开源工具类库，使用它能够**简化**[**JDBC**](https://baike.baidu.com/item/JDBC)**应用程序的开发**，同时也不会影响程序的性能。

**DbUtils出现的前提是什么？？？**

1. 原生jdbc开发步骤代码冗余、冗长、冗杂
2. 原生jdbc开发步骤代码关联性紧密
3. 原生jdbc在处理查询结果集时特别麻烦！！！支持性不友好

**DbUtils中的核心对象：**

**QueryRunner，替换了原生jdbc中的语句平台(Statement 以及 PreparedStatement),支持？绑定参数，可以有效的防止sql注入。**

**ResultSetHandler**，这个接口是专门用来封装查询结果集的！提供了八大结果集实现类，可以保证不管是单行单列、多行多列、单列多行、单行多列都有对应的结果集。

**DbUtils**，主要用来释放资源、处理事务

DbUtils如何使用

使用DbUtils的第一步，导入相关jar包：

**驱动jar包与dbutils的工具jar包，缺一不可。**

1. 使用DbUtils完成数据库操作

步骤：

**获取数据库连接**

**创建QueryRunner对象**

**使用QueryRunner对象执行sql语句**

**处理结果**

**释放资源**

可变参数的传递形式：

**第一种：直接传数组，sql语句中有几个？，那么数组中就有几个元素，而且？上的数据类型一定要与数组对应元素的类型一致**

**保证类型顺序与数量都需要与？一致！**

**第二种：将需要绑定的？数据，挨着写在sql语句后面，以逗号隔开，此时依旧需要注意**

**类型顺序与数量要与？一致！**

ResultSetHandler此接口是DbUtils专门用于封装查询结果集的对象

javaBean的设计规范

**1、类使用public修饰，公开**

**2、属性私有化，使用private修饰**

**3、为私有属性提供对应的setter、getter方法，方便赋值与取值**

**4、必须保证类中无参构造方法的存在！**

BeanHandler**将结果集中第一条记录封装到一个指定的javaBean中。**

BeanListHandler将结果集中每一条记录封装到指定的javaBean中，将这些javaBean在封装到List集合中

ScalarHandler**它是用于单数据。例如select count(\*) from 表操作。（统计结果）**

ColumnListHandler **将结果集中指定的列的字段值，封装到一个List集合中（单列多行）**

**QueryRunner对象在执行查询语句时，需要使用query()方法**

**此时需要传入连接connection，只需要记住两个重载方法即可**

**query(connection, sql, 结果集) 此方法中的sql语句不能书写？**

**query(connection, sql, 结果集, 可变参数) 此方法中的sql支持？绑定数据**

**结果集使用BeanHandler时，传入了一个类.class，此时用到了反射，反射在此时做了以下操作：**

**1、反射找到类中的无参构造方法，创建了对象！**

**2、通过反射找到了类中的setter方法，为对象的属性完成了赋值(赋值的内容就是表中字段的值)**

**但是我们在使用时，需要注意的点：**

**1、一定要保证无参构造方法的存在**

**2、为了保证属性的赋值成功，需要属性名与表中字段名一致**

DbUtils.close()关闭资源

连接池的概念：

数据库连接池负责**分配、管理和释放**数据库连接，它允许应用程序**重复使用**一个现有的数据库连接，而不是再重新建立一个；释放空闲时间超过最大空闲时间的数据库连接来避免因为没有释放数据库连接而引起的数据库连接遗漏。这项技术能明显提高对数据库操作的性能。

使用数据库连接池技术目的是什么：**得到数据源对象！**

为QueryRunner提供一个数据源对象，从数据源中自动获取连接

常用的连接池

dbcp 2个

**使用dbcp数据源来操作数据库**

**方式1：在创建QueryRunner对象，给定数据源，此时不需要书写连接获取的代码，会自动从给定的数据源中获取连接**

**使用完毕，不需要执行关闭资源的方法，连接对象会自动归还池子**

**方式2：我们可以通过数据源的getConnection()获取到由池子管理的数据库连接对象**

**使用完毕后，执行关闭资源的方法，并不是将连接对象销毁，而是将连接对象归还池子**

druid 1个 QueryRunner qRunner = new QueryRunner(DruidUtils.getDataSource());

c3p0 2个

**如果c3p0的配置文件名称叫c3p0.properties，并且放在了src下，那么就会自动读取配置文件！**

**private** **static** ComboPooledDataSource *dataSource* =**new**ComboPooledDataSource();

QueryRunner qRunner = **new** QueryRunner(*dataSource*);