JDBC

一、JDBC核心类

1、DriverManager用于管理驱动、获得连接

2、Connection用于连接数据库DBMS

3、Statement用于执行SQL语句

4、ResultSet用户获取执行结果

二、获取连接代码 Connection

1、反射获取字节码对象，加载JDBC的驱动

Class.forName("com.mysql.jdbc.Driver");

2、通过驱动设置连接

static Connection getConnection(String url, String user, String password) // 试图建立到给定数据库 URL 的连接。

//获取的到连接的路径，用户及密码

String url = "jdbc: mysql ://localhost ：3306/数据库名"; //数据库的资源路径

Strign user = "root"; //用户对象

String password = "root" ; //密码

//获取到连接的对象

Connection conn = DriverManager.getConnection(url , user , password) ;

注意：导包导的是import java.sql.Connection

三、获取执行SQL语句的对象

Statement

(Connection类的方法)Public Statement createStatement() // 创建一个 Statement 对象来将 SQL 语句发送到数据库。

成员方法：

boolean execute(String sql) // 执行给定的 SQL 语句，并且判断是否成功执行

int executeUpdate(String sql) // 执行给定 SQL 语句，该语句可能为 INSERT、UPDATE 或 DELETE 语句，或者不返回任何内容的 SQL 语句。

返回的值代表SQL语句影响了数据库表中多少数据，如果大于等于1，代表执行成功，否则代表失败

void close( ) ; 释放资源

Connection getConnection() //获取生成此 Statement 对象的 Connection 对象。

PreparedStatement：一个预处理的执行SQL语句的对象

通过Connection获得对象

（Collection类的成员方法）public PreparedStatement prepareStatement(String sql)

成员方法：

boolean execute(String sql) // 执行给定的 SQL 语句，并且判断是否成功执行

int executeUpdate(String sql) // 执行给定 SQL 语句，该语句可能为 INSERT、UPDATE 或 DELETE 语句，或者不返回任何内容的 SQL 语句。

返回的值代表SQL语句影响了数据库表中多少数据，如果大于等于1，代表执行成功，否则代表失败

ResultSet executeQuery(String sql) // 在此 PreparedStatement 对象中执行 SQL 查询，并返回该查询生成的 ResultSet 结果集对象。

注意：此处的executeQuery执行的是查询语句select

四、结果集对象ResultSet

获取结果集对象：

ResultSet res = stmt.executeQuery(String sql) ;

成员方法：

判断功能：

boolean next( ) ; // 将光标向下移，得到每一个执行结果对象 ，也是判断是否还有数据可以取

取出数据：

int getInt(String columnLabel) //以 int 的形式获取此 ResultSet 对象的当前行名中指定列的值。

int getString(String columnLabel) //以String类型获取结果集对象的当前列的值

int getDouble(....)

int getDate(...)

注意：

int getString(int columnIndex) ; //以String类型返回列对应的列的顺序的元素值

五、关闭资源的顺序

1、先关结果集，如果结果集不为空，就关闭 ，进行异常捕获

2、再关执行语句对象 ，如果不为空，就关闭

3、关闭连接对象

注意：

执行顺序必须从上往下

六、加载配置文件 （Properties） （写在静态代码块中，随着类的加载直接就进行了加载）

特性：属性集 ，继承了哈希表（HashTable），用于存储键和值

成员方法：

void load(InputStream inStream) // 从输入流中读取属性列表（键和元素对）。

String getProperty(String key) //用指定的键在此属性列表中搜索属性。

InputStream getResourceAsStream(String name) // 查找具有给定名称的资源。

URL getResource(String name) //查找带有给定名称的资源。

ClassLoader getClassLoader() // 返回该类的类加载器。

注意：加载配置文件的输入流，固定的写法差不多是：

InputStream input = new JDBCUtils( ) .getClass( ).getClassLoder( ) . getResourceStream( "配置文件的名字") ；

意思 ；创建当前类的字节码对象，来获取类加载器，再通过类加载器来访问指定名字的资源文件，得到一个输入流对象

Properties prop = new Properties( ) ;

创建配置属性集对象

Prop. load(input ) ;

意思代表从输入流中读取属性的键和值

七、DataSource : 创建数据源对象

//BasicDataSourceFactory 基础资源工厂

DataSource data = BasicDataSourceFactory . createDataSource(prop) ;

//通过数据源对象获取连接

Connection getConnection( ) ; //得到连接对象 尝试建立与此 DataSource 对象所表示的数据源的连接。

Connection getConnection(String user , String password) ;

八、事务（Connection类的方法 ）

1、提交事务

void commit()

2、获取和设置事务是否自动提交

void setAutoCommit(boolean autoCommit) //将此连接的自动提交模式设置为给定状态。

boolean getAutoCommit() // 获取此 Connection 对象的当前自动提交模式。

3、回滚

void rollback() ；取消在当前事务中进行的所有更改，并释放此 Connection 对象当前持有的所有数据库锁。

一旦事务出错，就回滚之前的操作，取消一系列操作

九、批量提交

1、要求：要结合事务进行使用

(Connection类) setAutoCommit(boolean flag) ; 设置事务是否自动提交

commit( ) ; //提交事务

rollback( ) ; //当事务执行失败时，回滚事务

2、每次使用时，最好将之前设置的参数进行清空

void clearParameters() // 立即清除当前参数值。

3、将一组对象加入到批量处理中

void addBatch() //将一组参数添加到此 PreparedStatement 对象的批处理命令中。

4、在最后使用executeBatch( ) ； //执行批处理 （写在循环语句外面）

//方法：clearBatch()方法：清除批处理命令

十、取出最后一次插入的id , 获取最后一次在连接中插入的id的值

1、必须在同一个连接中

2、使用 select last\_insert\_id ;

二、配置文件

1、properties，放在src根目录下

2、后缀为 . properties

3、配置文件的注意事项

A：驱动的名字必须是 driverClassName 否则抛没有加载驱动，并且无法为url路径创建驱动

B；url路径名为url

C3P0连接池

一、通过C3p0对象创建DataSource数据源对象

1、数据源对象用于访问配置文件的资源

//默认访问的是src根目录下的配置文件

DataSource datasource = new ComboPooledDataSource( ) ;

2、通过dataSource来创建连接

Connection conn = datasource . getConnection( ) ;

注意：以上操作都是在静态代码块中执行，随着类的加载自动加载

二、c3p0-config . xml 配置文件

1、必须是放在src根目录下，成为资源路径文件

2、配置文件的名字必须是c3p0-config . xml

3、配置文件内容

<c3p0-config>

<!--默认配置，如果没有指名则使用这个配置-->

<default-config>

<property name=“driverClass”>com.mysql.jdbc.Driver </property>

<property name="jdbcUrl“>

jdbc:mysql://localhost:3306/数据库名

</property>

<property name="user" > root </property>

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

//初始化池的大小

<property name = "initialPoolSize" > 2 </property>

//最大空闲时间

<property name = "maxIdleTime"> 30 </property>

//最多有多少个连接

<property name = "maxPoolSize">10 </property>

//最少有多少 个连接

<property name = "minPoolSize" > 2 </property>

//每次最多可以执行多少个批量语句

<property name = "maxStatements" > 50 </property>

</default-config>

<!--命名的配置-->

<named-config name = "demo" >

<property name=“driverClass”>com.mysql.jdbc.Driver </property>

<property name="jdbcUrl“>

jdbc:mysql://localhost:3306/数据库名

</property>

<property name="user" > root </property>

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

//初始化池的大小

<property name = "initialPoolSize" > 2 </property>

//如果池中数据连接不够时，每次自动增长多少

<property name = "acquireIncrement" > 5 </property>

//最大空闲时间

<property name = "maxIdleTime"> 30 </property>

//最多有多少个连接

<property name = "maxPoolSize">10 </property>

//最少有多少 个连接

<property name = "minPoolSize" > 2 </property>

//每次最多可以执行多少个批量语句

<property name = "maxStatements" > 50 </property>

</named-config>

</c3p0-config>

注意、

//空参，自动到classpath目录下面加载“c3p0-config.xml”配置文件---配置文件的存储位置和名称必须是这样，且使用“默认配置”

ComboPooledDataSource pool = new ComboPooledDataSource();

//加载“c3p0-config.xml”文件中定义的“demo”这个配置元素

ComboPooledDataSource pool = new ComboPooledDataSource("demo");

//加载驱动的键值名

驱动的键名为：driverClass

url路径的键名必须是jdbcUrl

(如果驱动的键名发生改变则会抛异常，抛 “没有合适的驱动”)

(并且url路径的键值名，也不能修改，否则说无法为url路径创建驱动)

JDBC执行SQL语句

一、插入数据INSERT

基本形式

INSERT INTO 表名 （列名1，列名2 ，。。。） VALUES(值1 ，值2，值3 。。。) ；

注意；

1、表名中的对应插的值，必须和列名一 一 对应

2、插入的值的数据类型必须和对应类的数据类型一致

3、若要输入记录时，插入全部数据，可以省略表名后的字段名（列名）

4、括号里的值是输入的值，各项之间是用逗号分隔

A：字符串和日期必须用单引号括起来

B：数值型类型数据不用括起来

C：数据顺序必须和字段顺序一样

D：所有标点符号必须在英文半角的状态下输入

二、删除记录

基本形式

DELETE FROM 表名 WHERE 条件

例如：

从数据库表中删除姓名为 caddo 的记录

delete from 表名 where name = 'caddo'

删除表中所有数据

delete from stu

注意：

如果没有给定条件，那么会删除整个表中的记录

三、更新记录

基本形式

UPDATE 表名 SET 字段1 = 值1 ，字段2 = 值2 where 条件

例如：

将stu表中所有id 大于 900 的记录中socre 的值加上5

update stu set score = score + 5 where id > 900

注意：

如果不加条件，那么将会是针对于表中的所有记录的修改

四、查询记录

基本形式

SELECT 字段名1 ， 字段名2 ，。。。 from 表名 where 条件

例如：

查询stu表中score值大于60 的数据

select \* from stu where score > 60

查询stu表中score 大于60 并且性别为女的 数据

select \* from stu where score >60 and sex ='女'

五、总结注意

1、字符串或者日期类型数据必须用单引号括起来, 并且在Java代码中则要在单引号中加入双引号和+ 号

例如：

insert into stu(name age,sex) values(' " +name+ " ' , ”+21+“ , ' " + sex + " ' )

2、插入数据在java代码中，必须加上加号和双引号

“ + user.getId( ) + ”

Request对象学习