**学习总结**

姓名 董金伟 专业 计算机科学与技术 班级 2班 日期 2022/3/27

1. 当天所学习主要内容（做一总结）

# JDBC异常处理

public class TestJDBC2 {  
 public static void main(String[] args){  
 *//1.删除throws Exception  
 //2.选中除了关闭资源的代码按ctrl + alt + t选择try catch finally  
 //3.把关闭资源的代码放到finally中  
 //4.分别对关闭资源的代码进行非空判断  
 //5.对关闭资源的代码选择try...catch* Connection conn = null;  
 Statement stmt = null;  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 String username = "root";  
 String password = "123456";  
 String url = "jdbc:mysql://localhost:3306/mysql0307?serverTimezone=UTC&characterEncoding=utf8&useUnicode=true&useSSL=false";  
 conn = DriverManager.*getConnection*(url, username, password);  
 *//准备sql语句,字符串必须使用双引号，其中的内容使用单引号，每一句末尾不加分号  
 //可以在sqlyog中测试一下，是否正确* String sql = "insert into emp values (5,'王五',30)";  
 *//准备”车“对象* stmt = conn.createStatement();  
 *//发车* int effect = stmt.executeUpdate(sql);  
 System.*out*.println("影响了" + effect + "行");  
 } catch (ClassNotFoundException e) {  
 e.printStackTrace();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 *//关闭资源,先开的后关闭* try {  
 if(stmt != null) {  
 stmt.close();  
 }  
 if(conn != null) {  
 conn.close();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 }  
}

# 工具类

# 工具类的特征

1.构造方法私有化，不让程序员在外界创建对象

private表示私有的，除了当前类的内部可以使用之外，其他地方都不能使用

如果构造方法私有化，则外界无法创建对象

2.所有方法都定义成静态方法，可以用类名调用

static关键字的用法：

1. 被static修饰的属性或者方法可以用类名直接调用
2. 静态的只能访问静态的，非静态的可以访问一切

# 编写JDBCUtils

1. 创建JDBCUtils
2. 构造方法私有化
3. 定义获取连接对象的方法
4. 定义关闭2个资源的方法
5. 定义关闭3个资源的方法

public class JDBCUtils {

private JDBCUtils(){}

//定义获取连接对象的方法

public static Connection getConnection(){

try {

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

String username = "root";

String password = "123456";

String url = "jdbc:mysql://localhost:3306/mysql0307?serverTimezone=UTC&characterEncoding=utf8&useUnicode=true&useSSL=false";

return DriverManager.getConnection(url, username, password);

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException throwables) {

throwables.printStackTrace();

}

return null;

}

//定义关闭两个资源的方法

public static void close(Statement stmt, Connection conn){

try {

if(stmt != null) {

stmt.close();

}

if(conn != null) {

conn.close();

}

} catch (SQLException e) {

e.printStackTrace();

}

}

//关闭三个资源的方法

public static void close(ResultSet rs, Statement stmt, Connection conn){

try {

if(null != rs) {

rs.close();

}

} catch (SQLException throwables) {

throwables.printStackTrace();

}

close(stmt,conn);

}

}

# 使用工具类进行增删改查

# 升级JDBCUtils

复习代码块

静态代码块

执行顺序：静态代码块，构造代码块，构造方法

不论创建多少个对象，静态代码块只执行一次

升级JDBCUtils

public class JDBCUtils {

static {

try {

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

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

private JDBCUtils(){}

//定义获取连接对象的方法

public static Connection getConnection(){

try {

String username = "root";

String password = "123456";

String url = "jdbc:mysql://localhost:3306/mysql0307?serverTimezone=UTC&characterEncoding=utf8&useUnicode=true&useSSL=false";

return DriverManager.getConnection(url, username, password);

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

//定义关闭两个资源的方法

public static void close(Statement stmt, Connection conn){

try {

if(stmt != null) {

stmt.close();

}

if(conn != null) {

conn.close();

}

} catch (SQLException e) {

e.printStackTrace();

}

}

//关闭三个资源的方法

public static void close(ResultSet rs, Statement stmt, Connection conn){

try {

if(null != rs) {

rs.close();

}

} catch (SQLException throwables) {

throwables.printStackTrace();

}

close(stmt,conn);

}

}