【济南中心】JavaEE就业班同步笔记第一阶段：JavaWeb之JDBC

小鲁哥哥 • 2016-12-23 14:36:39

【济南中心】JavaEE就业班同步笔记第一阶段：JavaWeb之数据库部分--JDBC

1 案例一：使用JDBC完成CRUD的操作:

1.1 需求：

对分类管理使用JDBC进行CRUD的操作.

1.2 分析：

1.2.1 技术分析：

【JDBC的概述】

JDBC:Java　DataBase Connectivity Java数据库的连接.

* 是SUN公司统一提供的一套接口规范(JDBC).各个数据库生产商提供实现.

驱动:两个硬件设备之间通信的桥梁.

【JDBC的开发步骤】

注册驱动:

获得连接：

获得执行SQL语句对象:

释放资源:

1.2.2 步骤分析：

【步骤一】：创建一个Java项目.

【步骤二】：引入mysql的驱动包.

【步骤三】：编写代码.

【步骤四】：完成CRUD的操作：

1.3 代码实现：

工具类的抽取：

[mw_shl_code=java,true]public class JDBCUtils {

```
/**
* 注册驱动的方法
*/
public static void loadDriver(){
try {
Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
e.printStackTrace();
}
}
/**
* 获得连接的方法
*/
public static Connection getConnection(){

Connection conn = null;
try {
loadDriver();
conn = DriverManager.getConnection("jdbc:mysql:///web_07","root", "123");
} catch (SQLException e) {
e.printStackTrace();
}
return conn;
}
/**
* 释放资源的方法
*/
public static void release(ResultSet rs,Statement stmt,Connection conn){
if (rs != null) {
try {
rs.close();
} catch (SQLException e) {
```

```
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
rs = null;
}
if (stmt != null) {
try {
stmt.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
stmt = null;
}
if (conn != null) {
try {
conn.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
conn = null;
}
}
public static void release(Statement stmt,Connection conn){
if (stmt != null) {
try {
stmt.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
```

```
stmt = null;
}
if (conn != null) {
try {
conn.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
conn = null;
}
}
}[/mw_shl_code]
```

带有属性文件的工具类的抽取：
定义了一个属性文件:

```
[mw_shl_code=java,true]public class JDBCUtils {
private static final String driverClass;
private static final String url;
private static final String username;
private static final String password;
static {
Properties properties = null;
// 读取属性文件：使用Java中Properties的对象.
try{
InputStream is = new FileInputStream("src/jdbc.properties");
properties = new Properties();
properties.load(is);
}catch(Exception e){
e.printStackTrace();
}
driverClass = properties.getProperty("driverClass");
```

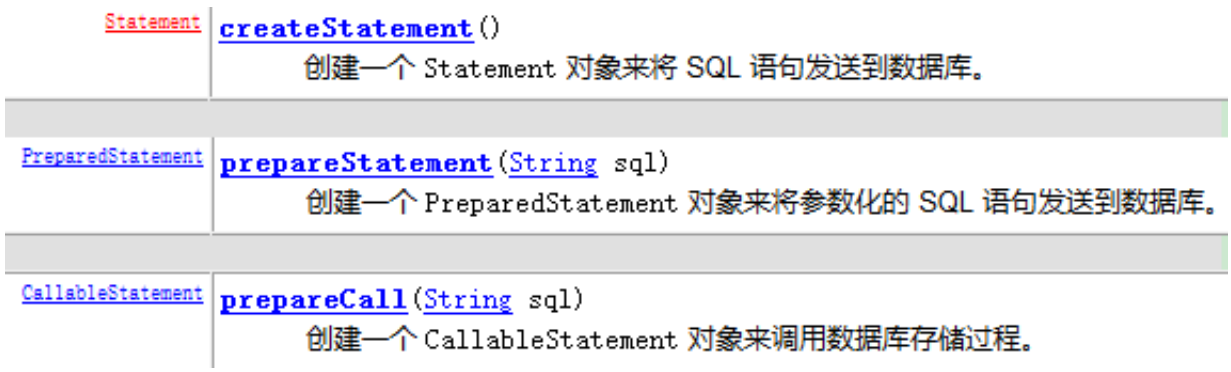
```
url = properties.getProperty("url");
username = properties.getProperty("username");
password = properties.getProperty("password");
}
/**
* 注册驱动的方法
*/
public static void loadDriver(){
try {
Class.forName(driverClass);
} catch (ClassNotFoundException e) {
e.printStackTrace();
}
}
/**
* 获得连接的方法
*/
public static Connection getConnection(){
Connection conn = null;
try {
loadDriver();
conn = DriverManager.getConnection(url, username, password);
} catch (SQLException e) {
e.printStackTrace();
}
return conn;
}

/**
* 释放资源的方法
*/
public static void release(ResultSet rs,Statement stmt,Connection conn){
```

```
if (rs != null) {
try {
rs.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
rs = null;
}
if (stmt != null) {
try {
stmt.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
stmt = null;
}
if (conn != null) {
try {
conn.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
conn = null;
}
}

public static void release(Statement stmt,Connection conn){
if (stmt != null) {
try {
```

```
stmt.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
stmt = null;
}
if (conn != null) {
try {
conn.close();
} catch (SQLException e) {
e.printStackTrace();
}
// 垃圾回收尽快回收对象.
conn = null;
}
}
```
}[/mw_shl_code]
参见JDBCDemo2类:
1.4 总结:
1.4.1 JDBC的API:
【Connection】
创建执行SQL的对象:

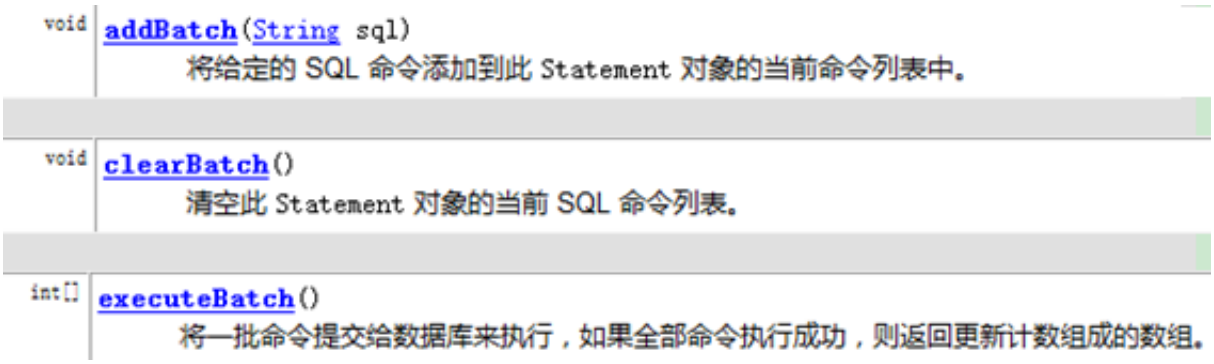


进行事务管理:

【Statement】

执行SQL语句:

执行批处理:



【ResultSet】

获得结果集中的数据:

* getXXX(int idx);

  * select cname,cid from category;

* getXXX(String name);

默认情况下：next();

* 正常的情况下结果集只能向下的.

2 案例二：使用连接池改造JDBC的工具类

回复帖子...　回帖

传统JDBC的操作,对连接的对象销毁不是特别好.每次创建和销毁连接都是需要花费时间.可以使用连接池优化的程序.
* 在程序开始的时候,可以创建几个连接,将连接放入到连接池中.用户使用连接的时候,可以从连接池中进行获取.用完之后,可以将连接归还连接池.
2.2 分析:
2.2.1 技术分析:
【自定义连接池】（了解）
* SUN公司提供了一个连接池的接口.(javax.sql.DataSource).
* 定义一个连接池:实现这个接口.
* 使用List集合存放多个连接的对象.
【自定义连接池的代码】
[mw_shl_code=java,true]public class MyDataSource implements DataSource{
// 创建一个List集合用于存放多个连接对象.
private List<Connection> list = new ArrayList<Connection>();
// 在程序开始的时候，初始化几个连接,将连接存放到list中.
public MyDataSource() {
// 初始化3个连接:
for(int i=1;i<=3;i++){
Connection conn = JDBCUtils.getConnection();
list.add(conn);
}
}
@Override
// 获得连接的方法:
public Connection getConnection() throws SQLException {
if(list.size() <= 0){
for(int i=1;i<=3;i++){
Connection conn = JDBCUtils.getConnection();
list.add(conn);
}
}
Connection conn = list.remove(0);

return conn;
}
// 归还连接的方法：
public void addBack(Connection conn){
list.add(conn);
}
...
}[/mw_shl_code]
【自定义连接池中问题及如何解决】
问题?
1.如果使用自定义连接池,那么需要额外记住自定义连接池中的API.
2.能不能使用面向接口的编程方式.
解决：
不额外提供API方法，就可以解决上述两个问题！！！
能不能还调用Connection的close方法.能不能增强Connection的close方法,原有的销毁变为归还!!!
如何增强Connection的close方法:
* 增强一个Java类中的某个方法有几种方式???
  * 一种方式：继承的方式.
    * 能够控制这个类的构造的时候,才可以使用继承.
  * 二种方式：装饰者模式方式.
    * 包装对象和被包装的对象都要实现相同的接口.
    * 包装的对象中需要获得到被包装对象的引用.
    ***** 缺点：如果接口的方法比较多,增强其中的某个方法.其他的功能的方法需要原有调用.
  * 三种方式：动态代理的方式.
    * 被增强的对象实现接口就可以.
【继承和装饰者的案例】
/**
* 继承的方式增强一个类中某个方法:
*/
[mw_shl_code=java,true]class Man{

public void run(){
System.out.println("跑....");
}
}[/mw_shl_code]

[mw_shl_code=java,true]class SuperMan extends Man{
public void run(){
// super.run();
System.out.println("飞....");
}
}[/mw_shl_code]

/**
* 使用装饰者的方式完成类的方法的增强
*/
[mw_shl_code=java,true]interface Waiter{
public void server();
}[/mw_shl_code]

[mw_shl_code=java,true]class Waiteress implements Waiter{
@Override
public void server() {
System.out.println("服务...");
}
}[/mw_shl_code]

[mw_shl_code=java,true]class WaiteressWrapper implements Waiter{
private Waiter waiter;
public WaiteressWrapper(Waiter waiter) {
this.waiter = waiter;
}
@Override

```
public void server() {
System.out.println("微笑...");
// this.waiter.server();
}
}[/mw_shl_code]
```

【使用装饰者模式增强Connection的close方法】

```
[mw_shl_code=java,true]public class MyConnection implements Connection{
private Connection conn;
private List<Connection> list;

public MyConnection(Connection conn,List<Connection> list) {
this.conn = conn;
this.list = list;
}
@Override
public void close() throws SQLException {
list.add(conn);
}
...
}
```

连接池的getConnection方法：

```
@Override
// 获得连接的方法：
public Connection getConnection() throws SQLException {
if(list.size() <= 0){
for(int i=1;i<=3;i++){
Connection conn = JDBCUtils.getConnection();
list.add(conn);
}
}
Connection conn = list.remove(0);
MyConnection myConn = new MyConnection(conn, list);
```

return myConn;
}[/mw_shl_code]
【常见的开源的数据库连接池】：
DBCP：
DBCP(DataBase connection pool),数据库连接池。是 apache 上的一个 java 连接池项目，也是 tomcat 使用的连接池组件。单独使用dbcp需要2个包：commons-dbcp.jar,commons-pool.jar由于建立数据库连接是一个非常耗时耗资源的行为，所以通过连接池预先同数据库建立一些连接，放在内存中，应用程序需要建立数据库连接时直接到连接池中申请一个就行，用完后再放回去。
C3P0：
C3P0是一个开源的JDBC连接池，它实现了数据源和JNDI绑定，支持JDBC3规范和JDBC2的标准扩展。目前使用它的开源项目有Hibernate，Spring等。
Tomcat内置连接池：
【DBCP连接池的使用】
第一步：引入DBCP连接池的jar包.
第二步：编写DBCP代码：
* 手动设置参数:
* 配置文件设置参数:
【DBCP连接池的使用】
[mw_shl_code=java,true]@Test
/**
* 手动方式:
*/
public void demo1(){
Connection conn = null;
PreparedStatement stmt = null;
ResultSet rs = null;
BasicDataSource dataSource = new BasicDataSource();
dataSource.setDriverClassName("com.mysql.jdbc.Driver");
dataSource.setUrl("jdbc:mysql:///web_07");
dataSource.setUsername("root");
dataSource.setPassword("123");

```
try{
// 获得连接:
conn = dataSource.getConnection();
// 编写SQL:
String sql = "select * from category";
// 预编译SQL:
stmt = conn.prepareStatement(sql);
// 执行SQL:
rs = stmt.executeQuery();
while(rs.next()){
System.out.println(rs.getInt("cid")+""+rs.getString("cname"));
}
}catch(Exception e){
e.printStackTrace();
}finally{
JDBCUtils.release(rs,stmt, conn);
}
}[/mw_shl_code]
  [mw_shl_code=html,true] @Test
/**
* 配置文件方式：
*/
public void demo2(){
Connection conn = null;
PreparedStatement stmt = null;
ResultSet rs = null;
Properties properties = new Properties();
try{
properties.load(new FileInputStream("src/dbcpconfig.properties"));
DataSource dataSource = BasicDataSourceFactory.createDataSource(properties);
// 获得连接:
conn = dataSource.getConnection();
```

// 编写SQL:
String sql = "select * from category";
// 预编译SQL:
stmt = conn.prepareStatement(sql);
// 执行SQL:
rs = stmt.executeQuery();
while(rs.next()){
System.out.println(rs.getInt("cid")+""+rs.getString("cname"));
}
}catch(Exception e){
e.printStackTrace();
}finally{
JDBCUtils.release(rs,stmt, conn);
}
}[/mw_shl_code]
【C3P0连接池的使用】
第一步：引入C3P0连接池的jar包.
第二步：编写代码：
* 手动设置参数:
* 配置文件设置参数：
【C3P0改造工具类】
[mw_shl_code=java,true]public class JDBCUtils2 {
private static final ComboPooledDataSource DATA_SOURCE =newComboPooledDataSour
ce();
/**
* 获得连接的方法
*/
public static Connection getConnection(){
Connection conn = null;
try {
conn = DATA_SOURCE.getConnection();
} catch (SQLException e) {

// TODO Auto-generated catch block
e.printStackTrace();
}
return conn;
}
...
}[/mw_shl_code]

3 案例三：手动抽取一个DBUtils的工具类

3.1 需求：

每次进行JDBC的CURD的操作的时候，有很多的代码都是相似的.可以不可以抽取工具类.完成一些通用性的代码?

3.2 分析：

3.2.1 技术分析：

【JDBC的元数据MataData】（了解）-- 编写通用性较高的代码.

DatabaseMetaData：获得数据库连接的信息,获得数据库的表的信息.

* 获得数据库元数据:Connection中getMetaData();

ParameterMetaData：获得SQL中的参数的个数及类型.

* 获得参数元数据：PreparedStatement中getParameterMetaData()

ResultSetMetaData：获得结果集中的列名及列的类型.

* 获得结果集元数据：ResultSet中getMeta()

【元数据的使用】

[mw_shl_code=java,true]@Test
/**
* 数据库元数据
*/
public void demo1(){
Connection conn = null;
conn = JDBCUtils2.getConnection();
// 获得数据库元数据:
try {
DatabaseMetaData metaData = conn.getMetaData();

```
System.out.println("获得驱动名称:"+metaData.getDriverName());
System.out.println("获得驱动URL:"+metaData.getURL());
System.out.println("获得用户名:"+metaData.getUserName());
// 获得表中的主键：
ResultSet rs = metaData.getPrimaryKeys(null, null, "category");
if(rs.next()){
String name = rs.getString("COLUMN_NAME");
System.out.println(name);
}
} catch (SQLException e) {
e.printStackTrace();
}
}
@Test
/**
* 参数元数据：
*/
public void demo2(){
Connection conn = null;
PreparedStatement stmt = null;
try{
conn = JDBCUtils2.getConnection();
String sql = "update category set cname = ? where cid = ?";
stmt = conn.prepareStatement(sql);
ParameterMetaData metaData = stmt.getParameterMetaData();
int count = metaData.getParameterCount();
System.out.println(count);
}catch(Exception e){
}
}
@Test
/**
```

```
* 结果集元数据：
*/
public void demo3(){
Connection conn = null;
PreparedStatement stmt = null;
ResultSet rs = null;
try{
conn = JDBCUtils2.getConnection();
String sql = "select * from category";
stmt = conn.prepareStatement(sql);
rs = stmt.executeQuery();
ResultSetMetaData metaData = rs.getMetaData();
int count = metaData.getColumnCount();
for(int i = 1;i<=count ;i++){
String name = metaData.getColumnName(i);
String type = metaData.getColumnTypeName(i);
System.out.println(name+type);
}
}catch(Exception e){
}
}[/mw_shl_code]
```

回帖

43条回帖

jkdong

沙发沙发，我是沙发

沙发 • 2016-12-23 18:24:53

回帖

jkdong

一直在学习，谢谢老师

藤椅 • 2016-12-23 18:25:33

回帖

小虎同学

学习，谢谢老师哦(⊙o⊙)哦

板凳 • 2016-12-23 21:54:55

回帖

来自宇宙超级黑马专属苹果客户端

孤独于我

感谢分享啊啊啊

报纸 • 2016-12-25 22:10:25

回帖

IT小孩

黑马论坛真是一个好地方啊

地板 • 2016-12-26 21:14:47

回帖

coder2468

好好好好好好好好好好好好

7# • 2017-1-1 00:59:11

回帖

@211

学习，一直努力！

8# • 2017-1-1 16:28:36

回帖

潜力蓝色香菇

谢谢老师~！~感谢感谢

9# • 2017-1-1 22:21:04

回帖

事难懂

好东西，收藏了

10# • 2017-1-3 13:28:08 回帖

hittor

感谢分享!

11# • 2017-1-3 19:19:32 回帖

vip825779

黑马有你更加精彩

12# • 2017-1-4 08:52:32 回帖

Yin,,,,Yan

好帖子 收藏下

13# • 2017-1-10 00:04:28 回帖

maogela

可不可以上传附件? 感谢!

14# • 2017-1-13 00:09:37 回帖

zmanx

济南那边工资怎么样啊,来自还没完有毕业的黑马小白的好奇

15# • 2017-2-12 22:09:42 回帖

爱吃橘子的小泽

顶一下,祝黑马越办越好

16# • 2017-2-12 22:10:53 回帖

a690223483

楼主 真的良心

17# • 2017-2-22 21:08:24 回帖

zhangkaitong

谢谢分享 赞赞赞!!!!!

18# • 2017-2-24 13:28:41 回帖

xulinml

谢谢老师~！~感谢感谢

19# • 2017-2-28 08:53:08 回帖

为过去的我买单

谢谢分享！

20# • 2017-3-2 10:11:34 回帖