一，默认适配器模式

装饰模式：

1，编写一个类，实现与被包装类相同的接口（具备相同的行为）

2，定义一个被包装类类型的变量

3，定义构造方法，把被包装类的对象注入，给被包装类变量赋值

4，对于不需要改写的方法，调用原有的方法

5，对于要改写的方法，写自己的代码

适配器模式是装饰模式的一种变体

适配器模式：（让子类更加干净）

（使用适配器模式的好处：该用的时候就用，不用的时候直接继承过来就可以了）

1，编写一个类，实现与被包装类相同的接口（具备相同的行为）

2，定义一个被包装类类型的变量

3，定义构造方法，把被包装类的对象注入，给被包装类变量赋值

4，对于不需要改写的方法，调用原有的方法

二，常用的数据源配置

（日后都使用数据源，一定要配置一下）

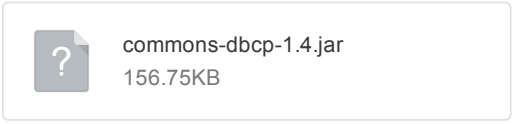
1，DBCP：Apache推出的DataBase Coonection Pool

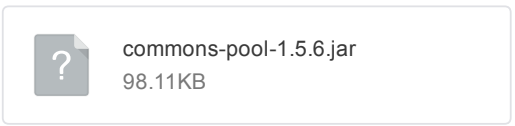
使用步骤：

添加jar包，commons-dbcp-1.4.jar　　commons-pool-1.5.6.jar

添加属性资源文件

编写数据源工具类





配置文件：

DBCPutil：

```
package com.itheima.utils;

import java.io.IOException;
```

```
import java.io.InputStream;
import java.net.URL;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Properties;

import javax.imageio.stream.FileImageInputStream;
import javax.sql.DataSource;

import org.apache.commons.dbcp.BasicDataSourceFactory;

public class DBCPUtil {
	private static DataSource ds = null;
	static{
		Properties prop = new Properties();
		try {

prop.load(DBCPUtil.class.getClassLoader().getResourceAsStream("dbcpconfig.properties"));//
根据DBCPUtil的classes的路径，加载配置文件
		    ds = BasicDataSourceFactory.createDataSource(prop);//得到一个数据源
		} catch (Exception e) {
			throw new ExceptionInInitializerError("初始化错误，请检查配置文
件");
		}
	}

	public static Connection getConnection(){
		try {
			return ds.getConnection();
		} catch (SQLException e) {
			throw new RuntimeException("服务器忙。。。");
		}
	}

	public static void release(Connection conn,Statement stmt,ResultSet rs){
		//关闭资源
				if(rs!=null){
```

```
                try {
                        rs.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
                rs = null;
            }
            if(stmt!=null){
                try {
                        stmt.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
                stmt = null;
            }
            if(conn!=null){
                try {
                        conn.close();//关闭
                } catch (Exception e) {
                        e.printStackTrace();
                }
                conn = null;
            }
        }

}
```

数据库连接池的配置文件：

```
#连接设置
driverClassName=com.mysql.jdbc.Driver
url=jdbc:mysql://localhost:3306/day13
username=root
password=abc

#<!-- 初始化连接 -->
initialSize=10

#最大连接数量
```

```
maxActive=50

#<!-- 最大空闲连接 -->
maxIdle=20

#<!-- 最小空闲连接 -->
minIdle=5

#<!-- 超时等待时间以毫秒为单位 6000毫秒/1000等于60秒 -->
maxWait=60000



#JDBC驱动建立连接时附带的连接属性属性的格式必须为这样：[属性名=property;]
#注意：″user″ 与 ″password″ 两个属性会被明确地传递，因此这里不需要包含他们。
connectionProperties=useUnicode=true;characterEncoding=utf8

#指定由连接池所创建的连接的自动提交（auto-commit）状态。
defaultAutoCommit=true

#driver default 指定由连接池所创建的连接的只读（read-only）状态。
#如果没有设置该值，则“setReadOnly”方法将不被调用。（某些驱动并不支持只读模式，如：
Informix）
defaultReadOnly=

#driver default 指定由连接池所创建的连接的事务级别（TransactionIsolation）。
#可用值为下列之一：（详情可见javadoc。）NONE,READ_UNCOMMITTED, READ_COMMITTED,
REPEATABLE_READ, SERIALIZABLE
defaultTransactionIsolation=REPEATABLE_READ
```