---------------------------------------------------

计划课程安排：

FUNDAMENTAL(7天)     该阶段已结束

OOP(11天)         该阶段已结束

JAVASE01(10天)        该阶段已结束

JAVASE02(12天)       该阶段已结束

WEBBASIC(11天)       该阶段已结束

JQUERY(2天)         该阶段已结束

DATABASE(5天)       该阶段已结束

JDBC(2天)         该阶段已结束

SERVLETJSP(13天) ←当前为该阶段第3天

SPRINGMYBATIS01(8天)

AJAX(2天)

~~SPRINGMYBATIS02(12天)~~ （该阶段已被取消）

~~LINUX(2天)~~ （该阶段已被取消）

~~NGINX(2天)~~ （该阶段已被取消）

~~REDIS(2天)~~ （该阶段已被取消）

~~T-STORE(5天)~~ （该阶段已被取消）

预计合计：100天

预计剩余：25天 （21天正课+4天自习）

注：上述信息随时可能发生更改，仅供参考。

---------------------------------------------------

**2019年5月29日**

**Java就业班-第75天**

**8:30实训室2-早自习**

**今日晨讲主题——数据库工作流程。**

**抱歉，我实在是听不见这位老哥讲的内容，声音太小了……**

**9:10实训室2-正课**

**课程回顾。**

**今天的课程总体偏向练习，新内容不多。**

**# 1.重定向**

**## (1)什么是重定向?**

**服务器通知浏览器立即访问一个新的地址。**

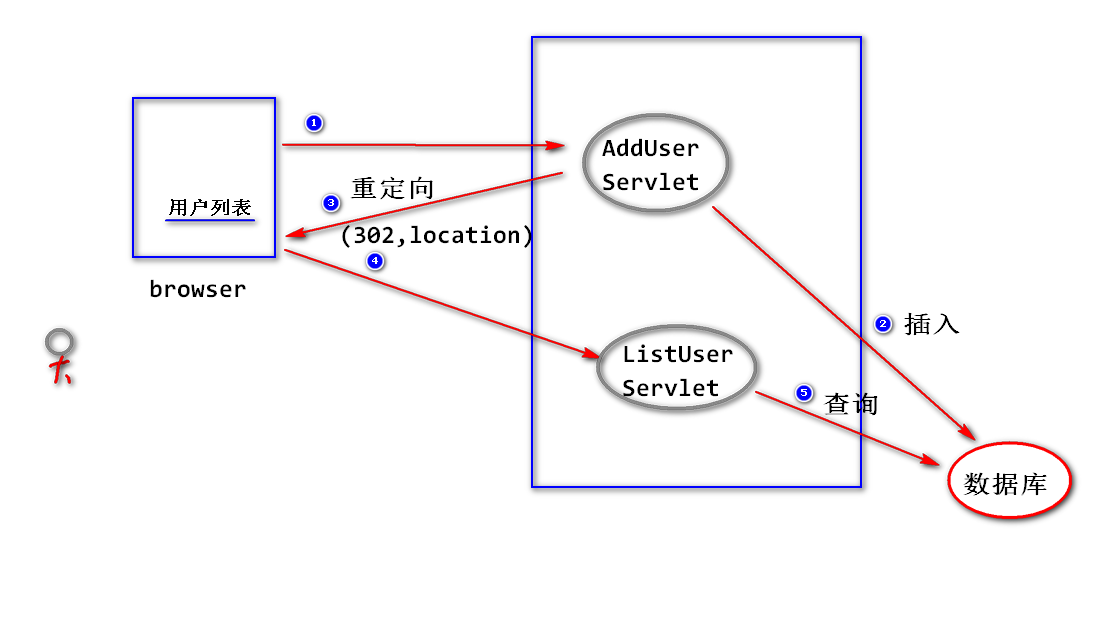
**注：**

**服务器可以通过发送302状态码和location消息头**

**(该消息头的值是一个地址，一般称之为重定向地址),**

**浏览器收到之后，会立即向重定向地址发送请求。**

**![](redirect.png)**



**## (2)如何重定向?**

**response.sendRedirect(String url);**

**注:**

**url就是重定向地址。**

**14:07实训室2-正课**

**## (3)特点**

**a.重定向地址是任意的。**

**b.重定向之后，浏览器地址栏的地址会发生变化。**

**# 2.DAO(Data Access Object 数据访问对象)**

**## (1)什么是DAO?**

**封装了数据访问逻辑的对象。**

**## (2)如何写一个DAO?**

**step1.实体类。**

**该类的属性与表的字段对应，类型要匹配。**

**step2.DAO类。**

**提供一些方法，用于访问数据库。**

**课程代码①：Maven项目servlet03-lab**

**Java代码①：ListUserServlet类**

**package web;**

**import java.io.IOException;**

**import java.io.PrintWriter;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import javax.servlet.ServletException;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import util.DBUtils;**

**/\*\***

**\* 查询所有用户信息数据处理**

**\* @author Administrator**

**\***

**\*/**

**public class ListUserServlet extends HttpServlet {**

**public void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**response.setContentType("text/html;charset=utf-8");**

**PrintWriter out = response.getWriter();**

**// 将所有用户信息查询出来**

**Connection conn = null;**

**PreparedStatement ps = null;**

**ResultSet rs = null;**

**try {**

**conn = DBUtils.getConn();**

**String sql = "select \* from t\_user ;";**

**ps = conn.prepareStatement(sql);**

**rs = ps.executeQuery();**

**// 依据查询到的用户信息，输出表格**

**out.println("<table width='60%' border='1' cellpadding='0'>");**

**out.println("<tr>"**

**+ "<td>ID</td>"**

**+ "<td>用户名</td>"**

**+ "<td>密码</td>"**

**+ "<td>邮箱</td>"**

**+ "<td>操作</td>"**

**+"</tr>");**

**while (rs.next()) {**

**int id = rs.getInt("id");**

**String username = rs.getString("username");**

**String password = rs.getString("password");**

**String email = rs.getString("email");**

**out.println("<tr>"**

**+ "<td>" + id + "</td>"**

**+ "<td>" + username + "</td>"**

**+ "<td>" + password + "</td>"**

**+ "<td>" + email + "</td>"**

**+ "<td><a href='del?id=" + id + "'>删除"+ "</td>"**

**+ "</tr>");**

**}**

**out.println("</table>");**

**out.println("<p><a href='addUser.html'>添加用户</a></p>");**

**} catch (Exception e) {**

**e.printStackTrace();**

**out.println("系统繁忙，稍后重试。");**

**} finally {**

**DBUtils.close(rs, ps, conn);**

**}**

**}**

**}**

**Java代码②：AddUserServlet类**

**package web;**

**import java.io.IOException;**

**import java.io.PrintWriter;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.SQLException;**

**import javax.servlet.ServletException;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import util.DBUtils;**

**/\*\***

**\* 添加用户数据处理**

**\* @author Administrator**

**\***

**\*/**

**public class AddUserServlet extends HttpServlet {**

**public void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**request.setCharacterEncoding("utf-8");**

**response.setCharacterEncoding("utf-8");**

**response.setContentType("text/html;charset=utf-8");**

**PrintWriter out = response.getWriter();**

**// 读取用户信息**

**String username = request.getParameter("username");**

**String pwd = request.getParameter("pwd");**

**String email = request.getParameter("email");**

**System.out.println(username);**

**// 将用户信息插入到数据库**

**Connection conn = null;**

**PreparedStatement stat = null ;**

**try {**

**conn = DBUtils.getConn();**

**stat = conn.prepareStatement("insert into t\_user values(null,?,?,?)");**

**stat.setString(1, username);**

**stat.setString(2, pwd);**

**stat.setString(3, email);**

**stat.executeUpdate();**

**//重定向到用户列表**

**response.sendRedirect("list");**

**System.out.println("重定向之后的代码！");**

**} catch (Exception e) {**

**/\***

**\* step1.记日志。**

**\* 将异常的所有信息记录下来，一般会记录到文件里面**

**\*/**

**e.printStackTrace();**

**/\***

**\* step2.看异常能否恢复，如果不能恢复**

**\* （比如，数据库服务暂停，网络中断等，**

**\* 一般把这样的异常称之为系统异常），**

**\* 则提示用户稍后重试。如果能够恢复，则立即恢复。**

**\*/**

**out.println("系统繁忙，请稍后再试。");**

**}finally {**

**DBUtils.close(null, stat, conn);**

**}**

**/\***

**\* 如果没有调用out.close方法，则容器会自动调用该方法。**

**\*/**

**//out.close();**

**}**

**}**

**Java代码③：DelUserServlet类**

**package web;**

**import java.io.IOException;**

**import java.io.PrintWriter;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import javax.servlet.ServletException;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import util.DBUtils;**

**/\*\***

**\* 删除用户数据处理**

**\* @author Administrator**

**\***

**\*/**

**public class DelUserServlet extends HttpServlet{**

**public void service(HttpServletRequest request , HttpServletResponse response) throws ServletException,IOException{**

**response.setContentType("text/html;charset=utf-8");**

**PrintWriter out = response.getWriter();**

**//获取要删除的id参数**

**int id = Integer.parseInt(request.getParameter("id"));**

**Connection conn = null ;**

**PreparedStatement ps = null ;**

**try {**

**conn = DBUtils.getConn();**

**String sql = "delete from t\_user where id=?";**

**ps = conn.prepareStatement(sql);**

**ps.setInt(1, id);**

**ps.executeUpdate();**

**//重定向到用户列表**

**response.sendRedirect("list");**

**} catch (Exception e) {**

**e.printStackTrace();**

**out.println("系统繁忙，请稍后再试。");**

**}finally {**

**DBUtils.close(null, ps, conn);**

**}**

**}**

**}**

**Java代码④：DBUtils类**

**package util;**

**import java.io.IOException;**

**import java.io.InputStream;**

**import java.sql.Connection;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.sql.Statement;**

**import java.util.Properties;**

**import org.apache.commons.dbcp.BasicDataSource;**

**import web.AddUserServlet;**

**/\*\***

**\* JDBC工具类**

**\* @author Administrator**

**\***

**\*/**

**public class DBUtils {**

**/\***

**\* getConn()连接数据库，返回Connection对象conn;**

**\*/**

**private static BasicDataSource ds ;**

**static {**

**// 创建数据源对象**

**ds = new BasicDataSource();**

**// 读取配置文件**

**Properties p = new Properties();**

**InputStream ips = AddUserServlet.class.getClassLoader().getResourceAsStream("jdbc.properties");**

**try {**

**p.load(ips);**

**String driver = p.getProperty("driver");**

**String url = p.getProperty("url");**

**String username = p.getProperty("username");**

**String password = p.getProperty("password");**

**// 设置连接信息**

**ds.setDriverClassName(driver);**

**ds.setUrl(url);**

**ds.setUsername(username);**

**ds.setPassword(password);**

**ds.setInitialSize(3);// 设置初始链接数量**

**ds.setMaxActive(5);// 设置最大连接数量**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**public static Connection getConn() throws Exception {**

**// 获取连接对象 注意导错包 异常抛出**

**Connection conn = ds.getConnection();**

**System.out.println(conn);**

**return conn;**

**}**

**/\*\***

**\* 按照JDBC规范，连接关闭的时候对应的Statment以及ResultSet应该自动关闭。**

**\* 但是，有一些连接池的实现没有完全符合规范，导致关闭连接时，对应的Statment及**

**\* ResultSet并没有关闭。所以建议在关闭连接时，最好手动关闭Statment及**

**\* ResultSet，避免资源的浪费。**

**\*/**

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

**if(rs != null) {**

**try {**

**rs.close();**

**} catch (SQLException e) {**

**e.printStackTrace();**

**}**

**}**

**if(stat != null) {**

**try {**

**stat.close();**

**} catch (SQLException e) {**

**e.printStackTrace();**

**}**

**}**

**if(conn != null) {**

**try {**

**conn.close();**

**} catch (SQLException e) {**

**e.printStackTrace();**

**}**

**}**

**}**

**}**

**16:04实训室2-正课**

**课程代码②：Maven项目servlet03-lab2**

**Java代码①：UserDAO**

**package dao;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import java.util.ArrayList;**

**import java.util.List;**

**import entity.User;**

**import util.DBUtils;**

**/\*\***

**\* 封装了数据访问逻辑**

**\* @author Administrator**

**\***

**\*/**

**public class UserDAO {**

**/\*\***

**\* 将所有用户信息查询出来**

**\* @throws Exception**

**\*/**

**public List<User> findAll() throws Exception{**

**List<User> users = new ArrayList<User>();**

**Connection conn = null;**

**PreparedStatement ps = null ;**

**ResultSet rs = null ;**

**try {**

**conn = DBUtils.getConn();**

**ps = conn.prepareStatement("select \* from t\_user");**

**rs = ps.executeQuery();**

**while(rs.next()) {**

**User user = new User();**

**user.setId(rs.getInt("id"));**

**user.setUname(rs.getString("username"));**

**user.setPwd(rs.getString("password"));**

**user.setEmail(rs.getString("email"));**

**users.add(user);**

**}**

**} catch (Exception e) {**

**e.printStackTrace();**

**throw e ;**

**}finally {**

**DBUtils.close(rs, ps, conn);**

**}**

**return users;**

**}**

**}**

**Java代码②：User**

**package entity;**

**/\*\***

**\* 实体类**

**\* 该类的属性与表的字段对应，类型要匹配。**

**\* @author Administrator**

**\***

**\*/**

**public class User {**

**private int id;**

**private String uname;**

**private String pwd;**

**private String email;**

**public int getId() {**

**return id;**

**}**

**public void setId(int id) {**

**this.id = id;**

**}**

**public String getUname() {**

**return uname;**

**}**

**public void setUname(String uname) {**

**this.uname = uname;**

**}**

**public String getPwd() {**

**return pwd;**

**}**

**public void setPwd(String pwd) {**

**this.pwd = pwd;**

**}**

**public String getEmail() {**

**return email;**

**}**

**public void setEmail(String email) {**

**this.email = email;**

**}**

**@Override**

**public String toString() {**

**return "User [id=" + id + ", uname=" + uname + ", pwd=" + pwd + ", email=" + email + "]";**

**}**

**}**

**17:30实训室2-下课时间**

**今天正课就到这里结束了。明天开始课程的主要内容将围绕JSP展开。**

**18:31实训室2-晚自习**

**今天晚上要进行第三次月考，依旧是不禁用百度和开发工具，这个考试只是走个过场。**

**而明天的中心摸底考试将采用笔试的模式，同时也不能使用百度和开发工具，成绩将记录下来，用于发送给招聘企业。**

**19:37实训室2-晚自习**

**这次月考的题主要是web前端阶段所学的一些内容，正是因为学了前端的内容又没有人监考，这次月考题都是直接打开浏览器的源码直接抄答案，整个过程毫无意义。**

**20:10实训室2-晚自习**

**重新计算了一遍毕业前的剩余时间，若按照原计划则应该还剩余40多天才能上完全部课程，而现在只剩下25天了。能够提前就业也不知道是好事还是坏事，反正顺其自然吧。**

**20:40实训室2-晚自习**

**项目经理过来简单介绍了几种Java的几种设计模式：**

**单例模式：**

**代码：**

**/\*\***

**\* 单例模式(饿汉模式)**

**\* 核心作用：**

**\* – 保证一个类只有一个实例，并且提供一个访问该实例的全局访问点。**

**\* @author Administrator**

**\***

**\*/**

**public class Single {**

**private static Single single = new Single();**

**private Single() {**

**}**

**public Single getInstance() {**

**return single;**

**}**

**}**

**21:02实训室2-晚自习**

**晚自习下课了，由于和人有约就先行回去吧。**

**23:14宿舍-休息**

**今天也就这样了吧，挺普通的，明天晚上有摸底考试，后天晚上似乎是有企业要到机构来开展什么访谈会，反正不知道机构是要整哪一出，不干扰学习就行。**

**今日感想总结：现在就已经达到可以使用Tomcat+JDBC+前端页面的代码实现一个简单的动态页面了，等到JSP和AJAX都掌握了之后，应该可以算是java入门了吧。**

**今日评分：6/10分**

今天就到这里，明天继续投稿。

感谢支持！

================================================================

**Acer程序员群：790482850**

**up主的java学习日记word文件及全部代码（随日记同步更新）：**

Adress：github.com/wt62635