# 14.response

response.setContentType(**this**.getServletContext().getMimeType(filename));//识别文件名的mimie类型

response.setHeader("Content-Disposition", "attachment;filename="+filename); //客户端以下载形式打开

OutputStream out = response.getOutputStream(); //文件输入流和输出流

InputStream in = **new** FileInputStream(realPath);

**byte**[] buffer = **new** **byte**[1024];

**int** len = 0;

**while**((len=in.read(buffer))>0)

{

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

out.write(buffer,0,len);

}

in.close();

out.close();

obj.src="/WEB14/CheckImgServlet?time="+**new** Date().getTime();//点击验证码图片更换图片，修改地址

验证码生成servlet

# 15 Request

String method = request.getMethod();//请求行

String requestURI = request.getRequestURI();

StringBuffer requestURL = request.getRequestURL();

String contextPath = request.getContextPath();

String queryString = request.getQueryString();

String remoteAddr = request.getRemoteAddr();

String header = request.getHeader("referer");//通过识别访问来源里有没有本地ip地址来防止盗链

Map<String, String[]> parameterMap = request.getParameterMap();

**for**(Map.Entry<String, String[]> map: parameterMap.entrySet())

{

System.***out***.println(map.getKey());

**for**(String name: map.getValue())

{

System.out.println(name);

}

System.***out***.println("---------------------");

}//请求体图的遍历

request.getRequestDispatcher("/Servlet1").forward(request, response);//请求转发代码

user.setUid(UUID.*randomUUID*().toString());//UUID字符串

request.setCharacterEncoding("UTF-8");//只适合post方式

username = new String( username.getBytes(“ISO-8859-1”),”UTF-8” );//兼容POST和get方法，但太麻烦，不常用

# 16 Cookie和Session

Cookie cookie = **new** Cookie("usernam","cym");

response.addCookie(cookie);//设置Cookie和添加Cookie

cookie.setMaxAge(20);//设置Cookie保留时间

cookie.setPath("/WEB16");//设置接收该Cookie的路径

# 17 JSP

Page的一些属性

<%@ include= “” %>

标签库

九大内置对象

PageContext

Request

Response

Out(out缓冲区 response缓冲区)

Session

Application = ServletContext

Exception

Config = ServletConfig

Page

第一次访问jsp才翻译成servlet，未变动直接取，修改后再访问再翻译

# 18 el表达式和jstl标签

${requestScope.name }

${sessionScope.user.username }

${applicationScope.userlist[0].username }

${name}

${user.username }

${userlist[0].username}

${pageContext.request.ContextPath}

<c:if test=*"*${**empty** user }*"*>

<li><a href=*"login.jsp"*>登录</a></li>

<li><a href=*"register.jsp"*>注册</a></li>

</c:if>

<c:if test=*"*${!**empty** user }*"*>

<li><a href=*"login.jsp"*>${user.username }</a></li>

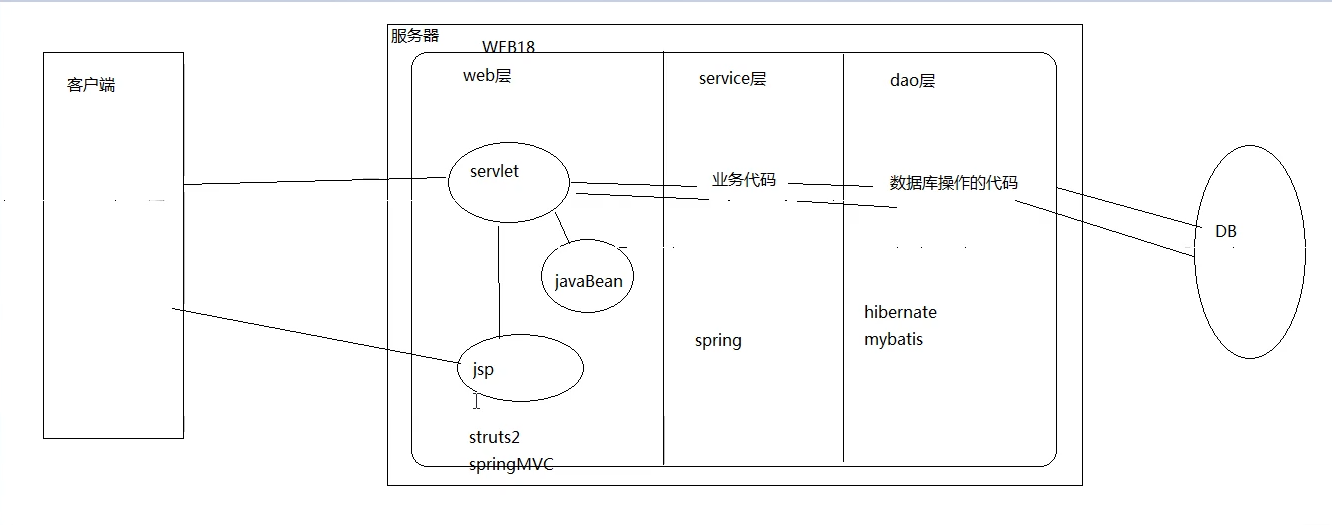
<li><a href=*"register.jsp"*>退出</a></li>

</c:if>

<c:forEach items=*"*${userlist }*"* var=*"user"*>

${user.username }----------${user.password }<br>

</c:forEach>



# 19 事务transation

JDBC控制事务：conn.setAutoCommit(false);

Conn.commit();

Conn.rollback();

DBUtils: ThreadLocal 绑定Connection

# 22 Ajax

**function** fun1() {

$.post("${pageContext.request.contextPath }/AjaxServlet",

{"name":"陈永铭","wife":"林文果"},

**function** (data) {

alert(data.name+" "+data.wife);

},

"json");

}

**function** fun2() {

$.get("${pageContext.request.contextPath }/AjaxServlet",

{"name":"陈永铭","wife":"林文果"},

**function** (data) {

alert(data.name+" "+data.wife);

},

"json");

}

**function** fun3() {

$.ajax({

url: "${pageContext.request.contextPath }/AjaxServlet",

async:**true**,

type:"get",

data:{"name":"陈永铭","wife":"林文果"},

dataType:"json",

success: **function** (data) {

alert(data.name+" "+data.wife);

},

error: **function** (data) {

alert(data.name+" "+data.wife);

}

});

}

String sql = "select \* from product where pname like ? limit 0,8";

List<Object> query = runner.query(sql, **new** ColumnListHandler("pname"), "%"+word+"%");

**return** query;

**for**(**var** i=0; i<data.length; i++){

content+="<div style='padding:5px;cursor:pointer' onclick='fun3(this)' onmouseout='fun2(this)' onmouseover='fun1(this)'>"+data[i]+"</div>";

}

**function** fun3(obj) {

$("#search").val($(obj).html());

$("#showDiv").css("display","none");

}

# 23 Listener

域对象接口

ServletContextListener

ServletContextAttributeListener

实体对象接口

HttpSessionActivationListener

HttpSessionBindingListener

Serializable（一个可以保存在硬盘的接口标志）

实现钝化和活化需要在META-INF文件夹下放context.xml文件

# 24 Filter

# 25 WEB加强

Class clazz = ClassLoader.**class**;

java.lang.ClassLoader classLoader = clazz.getClassLoader();

URL resource = classLoader.getResource("A.B");

System.***out***.println(resource.getPath());