web知识点小结

1.前端传值，后台收参

- 表单传值，后台是通过name属性收参
- 超链接，标签是通过key接收值 /user?key=value
- 如果表单提交了name属性值，没有传值，后台接收到的数据为" "
- 如果表单提交了,没有对应接收的name,j接收到的数据为null

2.乱码处理和请求路径

- 发送get请求，不会乱码
- 发送post请求，会有乱码出现的现象 我们需要处理请求乱码
- 请求路径如果加上/ 代表服务器路径
- 请求路径如果不加/ 代表项目路径

3.servlet请求方法控制流程

- servlet请求发送过来默认会调用service方法
- get方法只接收get请求信息---service调用
- post方法只接收post请求信息---service调用

4.servlet单例懒汉,生命周期

1. servlet 在第一次发送请求的时候被初始化，而且只初始化一次，所以被成为单例懒汉模式可使用onload的属性解决懒加载
2. servlet的生命周期：servlet有三个函数init,service,destory三个方法，init第一次请求被调用，而且只调用一次，destory销毁的时候是在服务器关闭的时候。
3. httpservlet和Genericservlet都可以接收请求，但没有子类httpservlet的功能强大.

5.服务器与浏览器建立四步

- 建立连接使用tcp/ip协议
- 发送请求 http请求
- 给出响应 response
- 关闭连接

6.get和post的区别

- get不安全，post相对安全
- get可以在url后面拼接数据,post不可以拼接数据
- get速度快，post速度慢
- get请求不会乱码，post在tomcat8.5会乱码

7.request请求或者http请求包括哪写信息

- 请求行 Get /user/pat Http1.1
- 请求头 key----value connection ----keepalive keepalive-----timeout content-length------7
- 请求体 post才有 json串

8.request请求收参的方式

request.getParamter()	只接收name对应value为一个值的
request.getParamterValues()	复选框，数组的
request.getParamterNames()	所有的name属性，也就是key值
request.getParamterMap()	Map<String,String[]> map进行接收

9.转发和重定向的区别

区别	转发	重定向
请求次数	1次	2次以上
数据request	携带数据	不携带数据
url	第一次的路径	最后一次的路径
访问资源	只能访问项目内部,包括web-inf	跨域访问
速度	快	慢
安全性	不好	相对安全

10.响应格式

Java 复制代码

```
● response.setContextType("text/html;image/png;image/jpg;image/gif","")------一般常见的
● 响应下载的时候，有弹出框提示的
// 让浏览器收到这份资源的时候，以下载的方式提醒用户，而不是直接展示
response.setHeader("Content-Disposition",
"attachment;filename="+fileName);
response.setContentType("application/json;charset=utf-8")
new String(fileName.getBytes("utf-8"),""ISO-8859-1"");
//注意：最重要的是我们想要那种格式去web.xml中进行寻找
```

11.验证码功能的实现

1. 生成验证码：使用hutool或者自己写的工具类生成一张验证码图片；
2. 验证码动态刷新：我们可以使用改变图片路径的src来进行动态改变图片；

3. 浏览器缓存原因：我们要让其刷新，需要传递时间戳参数来进行动态刷新

12.文件上传，下载，修改，删除

1. 文件上传
- 第一步：需要将图片名称保存到数据库---crud
- 第二步：需要将上传的文件保存到磁盘--req.part("name");part获取输入流，自己new输出流
2. 文件下载
- 第一步，服务器磁盘路径部署
- 第二步,拼接图片路径 注意/和不带/的区别
3. 文件修改
- 第一步，和上传文件操作一样
- 第二步，删除原来的图片，我们可以使用一个隐藏域将原来的图片名称带过去删除
- 注意，上传文件，用户如果点击修改什么也不做，那么上传文件名为""字符串，所以我们可以使用旧的图片名覆盖新的图片名 req.getServletContext().getRealPath()-----可以获取项目中的路径
4. 文件删除
- 携带图片名称删除即可

13.http协议版本，js的有趣用法，jsp和servlet区别

1. http协议版本：
- http0.9:只支持文本传输
- http1.0:支持文本和二进制传输，短连接
- http1.1:支持文本和二进制传输，长连接传输
- http2.0:大数据推送
2. js有趣玩法:
- href="javascript:checkout()";
- html的表单玩法 <input type="submit" onclick="return checkout()"/>
- 函数传值：js的函数传参随便传，checkout(out,left)
3. jsp和servlet区别:
- jsp和servlet共同点：都要生成java文件，都实现了servlet接口
- jsp和servlet不同点：

区别	jsp	servlet
运行机制	jsp---java---编译---运行	编译----运行
速度	速度慢	速度快
配置	无需配置	web.xml配置or注解配置
支持协议	只支持http	只支持file

14.请求header,路径，注解,idea中的文件路径

1. 请求头我们可以通过setHeader和addHeader方式进行添加　set---可以覆盖,add不可以覆盖
2. 注解我们在配置springmvc的时候经常使用

Java　复制代码

```
1    @WebServlet(urlPatterns = {"/test","/a"},loadOnStartup = 1,
2                initParams = {@WebInitParam(name = "encoding",value = "utf-
    8")})
```

3. 要么E:/　要么E:\\　两种写法

15.jquery小总结

1. 入门知识：$()==jquery()
2. 加载函数：$(function(){});
3. 获取jquery对象：

- $("#id"):只能获取一个jquery对象;
- $(".class/input/*")获取的是数组;
- 过滤选择器：$("tr:odd");$("tr:even")---last---first
- 属性选择器：$("input[id=add]")

4. form表单---我们可以在action后面拼接参数，但发送的请求一定是post请求
5. 单选按钮一定要给value值，默认值为on
6. 遍历数组$.each(arr,function(element,index){　});
7. 获取css $("#id").css("","")　addClass("div1 div2") removeClass()
8. 获取属性值prop $("#id").prop("","")
9. 函数式编程 html()　val()　text()
10. window几个重要对象:history;location;screen;警告框；定时器；
11. 改变图片就是要改变图片路径

16.jsp内置对象，jstl和el表达式

- jsp的九大内置对象：

pageContext request session application page config response out expection

- el表达式:支持四大域对象取值，支持el的表达式运算，支持三目运算符，支持${""eq""}
- jstl表达式:<c:foreach><c:choose><c:if><c:set> test="fn:check()"

17.复杂的模糊查询实现

1. 多个条件的模糊查询：后台参数有一下几种情况

- null------null 指的是没有点击模糊查询的表单
- ""------"" 点击模糊查询的表单，但并没给值
- ""------值 传入一个模糊条件
- 值------"" 传入一个模糊条件
- 都有值 正常传入提交

2. sql拼接实现所有逻辑

SQL 复制代码

```sql
select * from book where 1=1
and name='%"+name+"%'
and pwd='%"+password+"%'
```

18.分页查询实现

1. 工具类属性的理解：

Java 复制代码

```java
        pageSize; //页容量 每页显示的数量    5
        totalCount;  // 总记录数  count(*)
        currentPage; // 当前页
        startIndex;  // 起始的行下标
        totalPages;  // 总页码
        prePage;   // 上一页
        nextPage;  //下一页
```

2. 工具类的使用时机：

- 只要点击查询我们都需要进行分页处理；所以使用时机就是查询所有的方法中
- new 工具类（）需要什么，就给传入什么值,将工具类存入作用域发送到前端，轻松实现上一页，下一页，首页，尾页

3. 模糊查询带分页

- 模糊查询带分页我们必须注意，查询符合记录数的sql时我们也要佩戴模糊条件
- 上一页，下一页，首页，尾页也需要佩戴模糊条件
- 我们第一次点击模糊查询的时候，让模糊条件存到作用域中，发送到前端，方便后面携带

19.过滤器生命周期，执行顺序

1. 过滤器的生命周期：

- 和servlet很是相同：三个方法init,dofilter,destory;可以对比理解

2. 过滤器的执行顺序:

- 当只有一个过滤器的时候，先过滤在放行
- 当有多个过滤器的xml方式时候，web.xml是一种谁在上，谁先执行
- 当有多个过滤器的注解方式时候，过滤器的执行顺序会根据类的首字母进行排序
- springmvc中的配置就是使用过滤器来实现乱码控制的
- 我们配置多个过滤器，过滤器会形成过滤器执行链
- 过滤器默认情况下只过滤重定向路径，不过滤转发路径.

20.服务器域对象的产生时机

- session是第一次访问服务器的时候创建的，解绑时销毁
- request是一次请求创建一次，一次请求结束销毁对象
- servletContext是服务器启动的时候创建的，服务器关闭时销毁的

21.会话技术session,cookie

1. cookie技术

- 我们将浏览器的一次开启和关闭称之为一次会话
- cookie是由服务端创建的，通过增加相应头发送给浏览器，让其cookie可以在浏览器端多保存几天
- 网站的cookie保存功能实现,可以通过点击记住密码传递一个标志到后台

```java
● <input type="checkbox" value="number" name="ck"/>
● if("number".equeals(ck)){
            //创建cookie   设置生存周期
       }else{
//直接响应不需要设置cookie
       }
● cookie代码实现'
//1.创建cookie
Cookie cookie1 = new Cookie("username",username);
//2.设置cookie的生存时间
cookie1.setMaxAge(60*60*24*7);
//3.设置cookie的作用域
cookie1.setPath(req.getServletContext().getContextPath()+"/login.jsp");
//4.响应
resp.addCookie(cookie1)
//5.获取cookie
Cookie[] cookies = req.getCookies();
```

2. session技术：

- 浏览器发送第一次请求，后台服务器会创建个sessionid传给前端浏览器放在cookie中保存，一旦cookie消亡sessionid就会消亡，在次请求服务器，就会重新创建sessionid
- cookie的默认存活时间为一次会话
- 服务器端的session默认存活为30分钟，服务器关闭或者session过期会重新创建session

22.工具类和jsp的动作标记和静态标记

1.工具类

beanUtil-commons:用于map和bean的转换

commons-io文件上传工具类

2.动作标记和静态标记

动作标记生成的是多个java文件，可以定义多个相同名的变量,静态标价刚刚相反.

23.监听器

监听器的职责：

就是帮助我们监听 request,session,servletContext

监听器的分类：

1. 按照域对象分：request,session,servletContext----servletContext
2. 按照监听类型分：监听创建和销毁；监听作用域的存值，修改值，和删除值---servletContextAttribute
3. 特殊监听器：

- session的持久化监听，服务器关闭，重启，时间到出发持久化；
- session的序列化，重新加载，服务器启动

监听器的解绑操作进行监听。

应用：

统计网站的在线人数

统计网站的总人数

统计网站的当前页人数