# jsp简介

jsp全称Java Server Page即java服务页面

## 1.1jsp和html的区别:

html:由html标签组成,输出静态内容

jsp:可以写html标签,还可以写java代码

## 1.2jsp特点

1)既可以写html代码,也可以写java代码

2)jsp页面需要交给tomcat运行

## 1.3jsp运行的目录

tomcat的work目录,这个目录存放了jsp页面运行过程中产生的临时文件

## 1.4Jsp页面如何被tomcat服务器运行

第一次访问jsp页面:

1)tomcat服务器会把jsp页面翻译成java源文件

把hello.jsp ->

C:\Users\lw\.IntelliJIdea2017.2\system\tomcat\Unnamed\_untitled\work\Catalina\localhost\web\org\apache\jsp目录下hello\_jsp.java

注意:这是idea是这样的

2)tomcat服务器会把java源文件编译为class文件,将1)中的源文件编译为:

C:\Users\lw\.IntelliJIdea2017.2\system\tomcat\Unnamed\_untitled\work\Catalina\localhost\web\org\apache\jsp目录下hello\_jsp.class

3)读取class文件,创建类对象

4)执行类中的方法hello\_jsp类的方法

第n次访问jsp:

直接走上述的第4)步骤

注意:修改了jsp页面时,会执行1)2)步骤,即重新走翻译和编译的过程

## 1.6jsp本质

hello\_jsp类是一个继承了HttpServlet的抽象类,即jsp就是一个servlet

## 1.5jsp和servlet生命周期

servlet的生命周期:

构造方法

init方法

service方法

destory方法

jsp的生命周期:

翻译(java文件)

编译(class文件)

构造方法

\_jspinit方法

\_jspservice方法

\_jspDestory方法

# jsp语法

## 2.1Jsp模板

Jsp页面的html代码就是模板

<%

<p>哈哈</p>

%>

## 2.2Jsp表达式

语法:<%=变量或者表达式%>

作用:向浏览器输出变量的值或者表达式计算的结果

范例:

|  |
| --- |
| <%  String name = "eric";  int a=10;  int b=10;  %>  <%=name%>  <%=(a+b)%> |

注意:

1. 原理是使用out.print()方法向浏览器输出内容

## 2.3Jsp脚本

语法:<%java语句%>

作用:执行java语句

范例:

|  |
| --- |
| <%  Random ran = new Random();  float num = ran.nextFloat();  out.write("随机小数:" + num);  %>  <%  for(int i=1;i<=6;i++){  %>  <h<%=i%>>标题<%=i%></h<%=i%>>  <%  }  %> |

原理:脚本就是原封不动的翻译到了java文件的\_jspService方法中执行而已

## 2.4Jsp声明

语法:<%! 变量或者方法 %>

作用:用于声明变量或方法

|  |
| --- |
| <%!  //变量  private String email = "eric";  //方法  public void sayHello(){  System.out.print("xxxx");  }  %> |

原理:声明的变量是成员变量,声明的方法是成员方法

注意:

1)不要声明和翻译后的java文件相同的方法名

## 2.5Jsp注释

语法:<%-- jsp注释--%>

作用:注释jsp页面内容

注意:

html的注释会被翻译和执行,而jsp的注释不会被翻译和执行

# Jsp的三大指令

语法:<%@ 指令 %>

## 3.1taglib指令

## 3.2include指令

作用:用户包含其他页面

|  |
| --- |
| <%@include file="common/head.jsp" %>  页面主要内容 |

原理:直接把包含与被包含页面的内容先合并在一起,然后翻译成一个java源文件,最后编译执行(源码包含,也叫静态包含)

## 3.3page指令

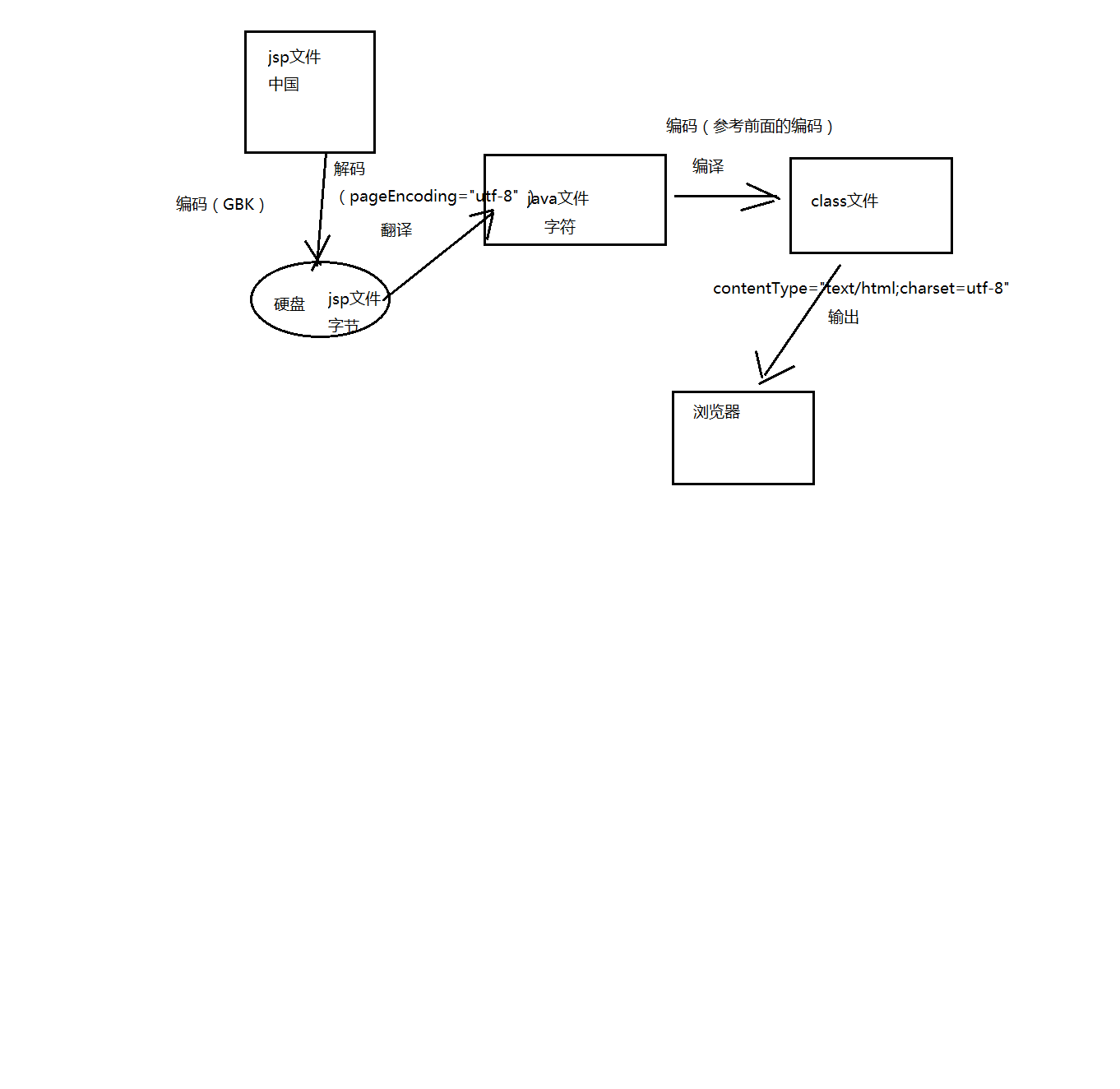
作用:告诉浏览器如何翻译jsp文件

|  |
| --- |
| <%@ page  language="java" 使用什么语言来翻译jsp文件  import="java.util.\*" 导包  pageEncoding=”utf-8” 使用什么编码翻译jsp文件  contentType="text/html; charset=UTF-8" 服务器发送给浏览器数据类型和编码格式  errorPage="/common/error.jsp" 指定jsp错误处理页面  isErrorPage="true" 指定当前页面为错误处理页面,如果是错误处理页面,就可以用exception内置对象,可以用来查看异常信息  session="true" 是否打开session功能,true就是打开,false就是关闭  buffer="8kb" jsp页面缓存区大小  isELIgnored=”false” 是否忽略el表达式,ture是忽略,false是不忽略  %> |

配置全局错误处理页面子啊web.xml里面配置:

|  |
| --- |
| <web-app>  <error-page>  <error-code>404</error-code>  <location>/common/404.html</location>  </error-page>  <error-page>  <error-code>500</error-code>  <location>/common/500.jsp</location>  </error-page>  </web-app> |

编码图:



# jsp最佳实践

servlet技术:使用java语言开发动态网页

功能:写java代码,输出html代码

jsp技术:使用java语言开发动态网页

功能:写java代码,输出html代码

分工问题:

servlet技术:写java代码(写普通java类)

jsp技术:输出html代码（类似于html页面）

项目中设涉及的功能:

1. 接受参数信息(servlet做)
2. 执行业务逻辑(CURD)(servlet做)
3. 返回结果,输出结果(jsp做)
4. 跳转页面(servlet做)

# 5.Jsp内置对象

在jsp开发有些对象使用频率比较高,每次自己手动获取这些对象,太麻烦了,Jsp把这些对象都创建或者获取好了,开发这直接使用这些对象就好了,这些对象叫做内置对象

总共有9个内置对象

**jsp对象名 类型**

request HttpServletRequest 请求对象

response HttpServletResponse 响应对象

config ServletConfig 配置对象

application ServletContext Servlet上下文对象

session HttpSession 会话对象

exception Throwable 异常信息对象

this Object 代表jsp文件翻译后的java类

out JspWriter 代表输出对象,reponse.getWriter() PrintWriter

相当于带缓存功能的PrintWriter

pageContext PageContext 代表jsp的上下文对象

## 5.1pageContext对象

作用:

1)通过pageCotext对象获取其他8个内置对象

场景:使用自定义的标签的时候

pageContext.getRequest等同于request

pageContext.getResponse等同于response

2)作为域对象

总共4个域,从哪个域中保存,就从哪个域中获取

a.默认情况下保存到了page域里,当前页面有用

保存数据:

pageContext.setAttribute(“message”,”hello,world”)

取出数据

pageContext.getAttribute(“message”)

b.保存到其他三个域中(request,session,application)

|  |
| --- |
| pageContext.setAttribute("message","hello,world",PageContext.REQUEST\_SCOPE);  pageContext.getAttribute("message",PageContext.REQUEST\_SCOPE);  等价于  request.setAttribute("message","hello,world");  String str= (String)request.getAttribute("message"); |

4个域对象int值:

|  |
| --- |
| PageContext.APPLICATION\_SCOPE  PageContext.SESSION\_SCOPE  PageContext.REQUEST\_SCOPE  PageContext.PAGE\_SCOPE |

# 6.Jsp的四个域对象

request域:HttpServletRequest

application域:ServletContext

session域:HttpSession

page域:PageContext

## 6.1域对象作用

共享数据,保存数据,获取数据

## 6.2域对象的方法

setAttribute()

getAttribute()

removeAttribute()

findAttribute()//自动搜索四个域的数据,从范围小的page域到application域

## 6.3域的作用范围

从小到大:

page域:在同一个jsp页面中数据有效

request域:在同一个请求中有效数据有效

session域:在同一个会话中数据有效

application域:在同一个网站中数据有效