

1.JSP概念

JSP

https://blog.csdn.net/BruceLiu_code

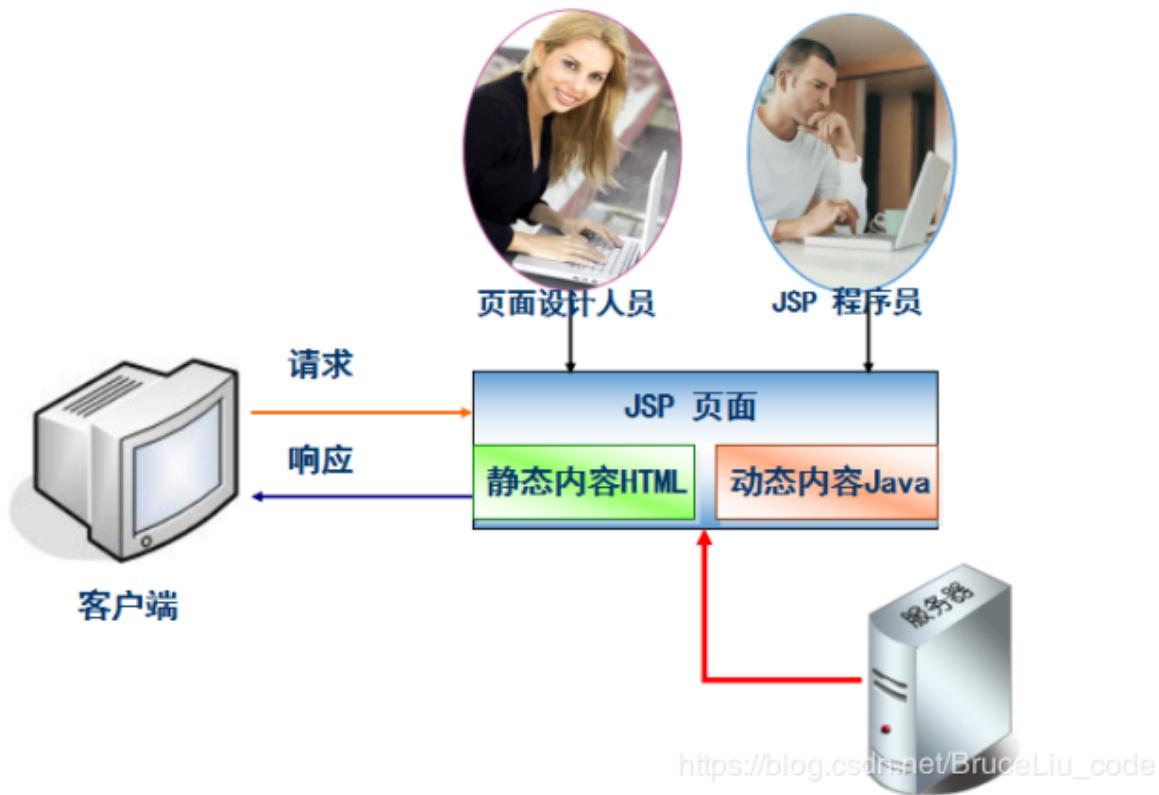
JSP（java server pages），即java服务器页面，它是由Sun公司和其他很多个公司一起建立的一种动态网页技术；主要是用来代替Servlet来完成动态网页的输出。**在JSP中可以书写html代码，还可以书写Java代码！JSP实质上是一个简化的servlet，是一种动态的网页技术的标准！**

Servlet+JSP Servlet：获取数据、处理业务逻辑、查询数据等 JSP：页面展示

自我总结：servlet主要负责接收请求数据，，然后根据请求参数查询数据，最终通过响应对象向客户端响应动态的网页。Jsp主要用来完成动态网页的输出。

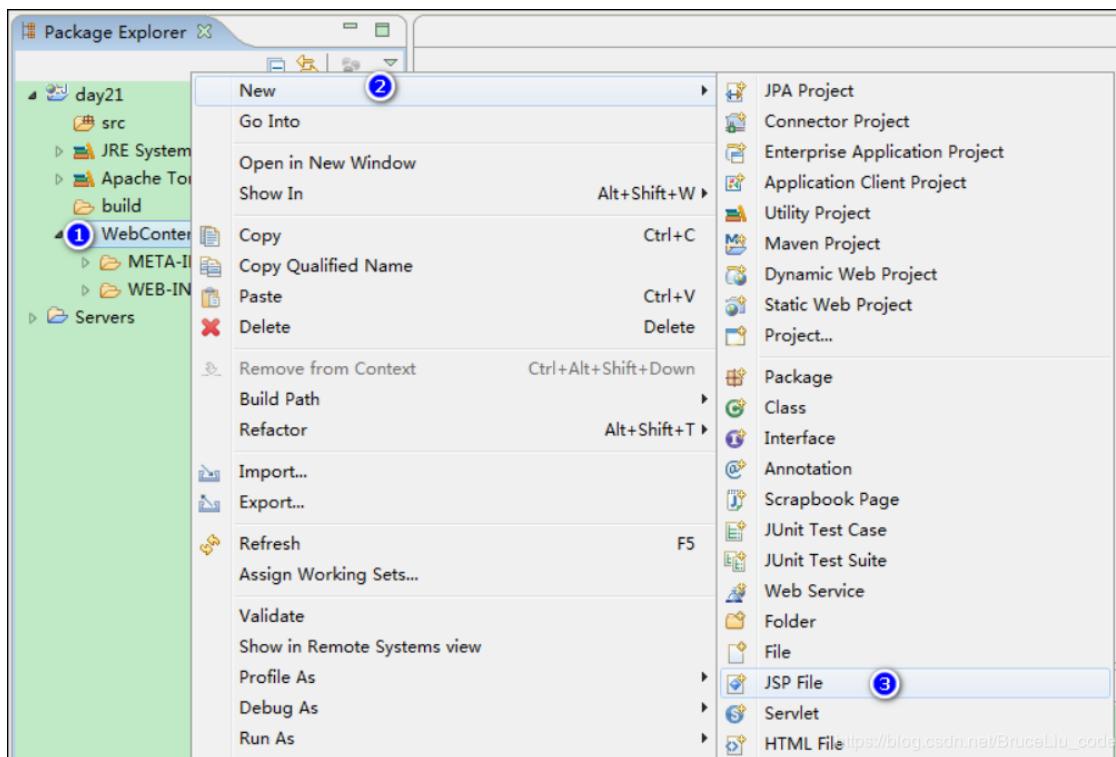
2.JSP引入目的

JSP性能好，可以在html页面中动态嵌入元素 服务器调用的是已经编译好的JSP文件
JSP基于Java Servlet Api,有很多强大企业的支持。 JSP可以与处理业务逻辑的Servlet一起使用，该模式被Java Servlet模版引擎所支持。

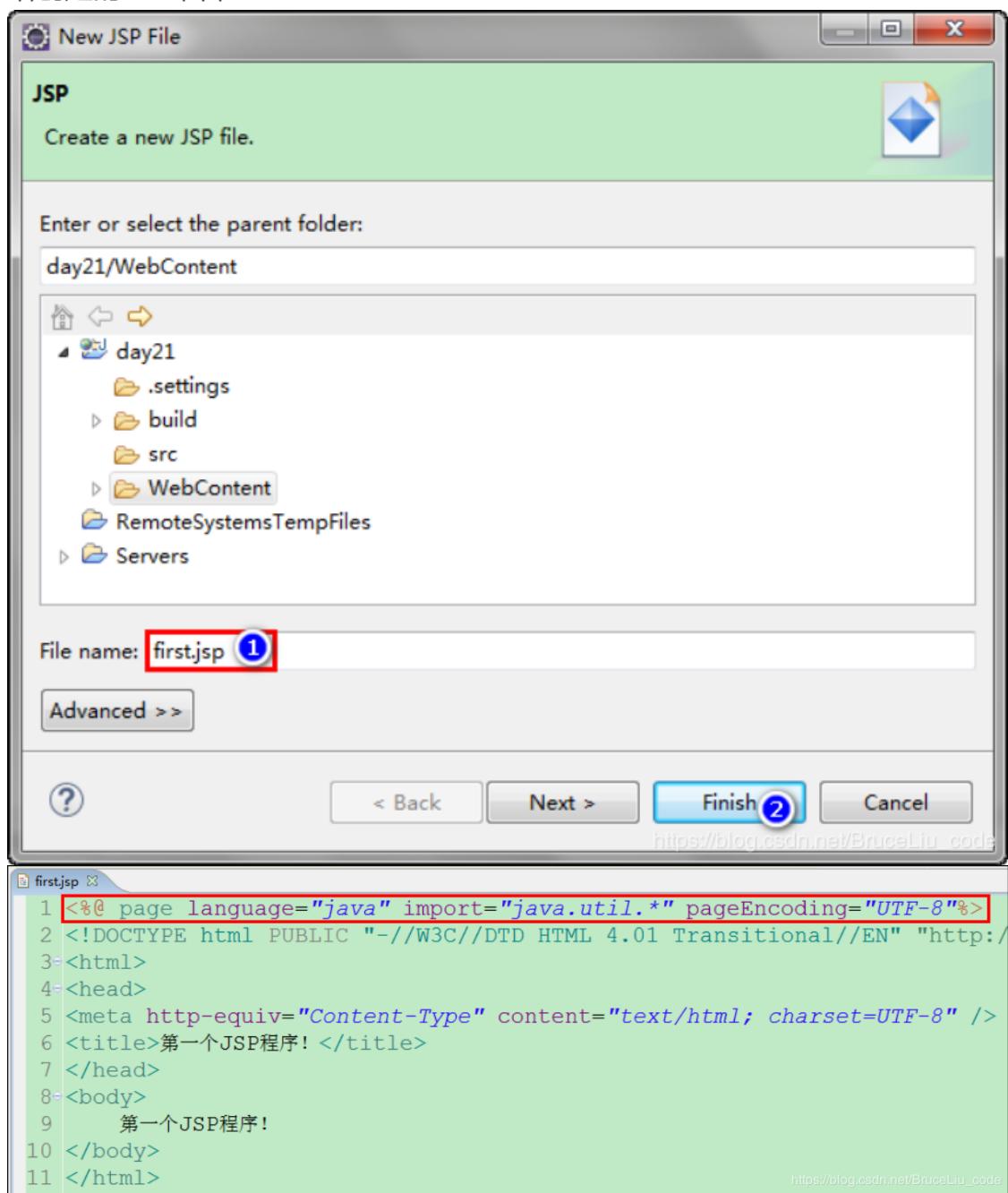


3.JSP快速入门

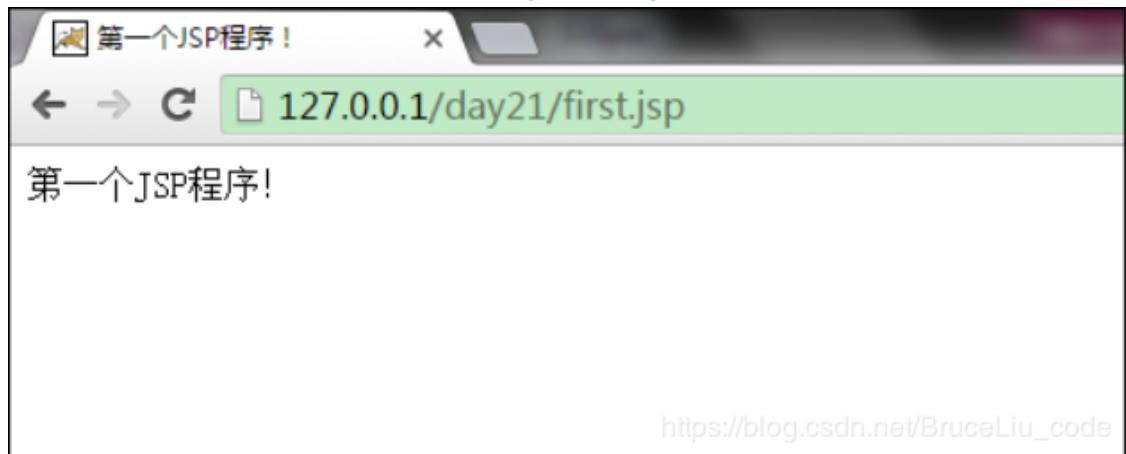
- 点击右键创建JSP页面



- 给创建的JSP命名



- 使用浏览器访问：<http://127.0.0.1/day21/first.jsp>



- 页面空白处右键，查看网页源代码，实际上就是一段html代码



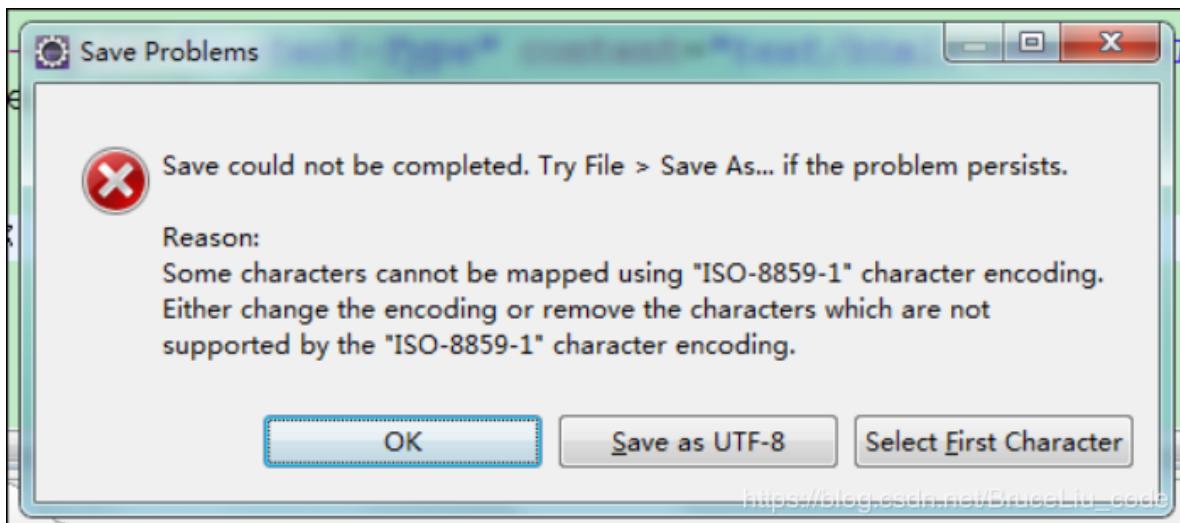
The screenshot shows a browser window with two tabs. The active tab is titled "view-source:127.0.0.1/day21/first.jsp". The content area displays the source code of a JSP file:

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
5 <title>第一个JSP程序！</title>
6 </head>
7 <body>
8     第一个JSP程序！
9 </body>
10 </html>
11
12
13
```

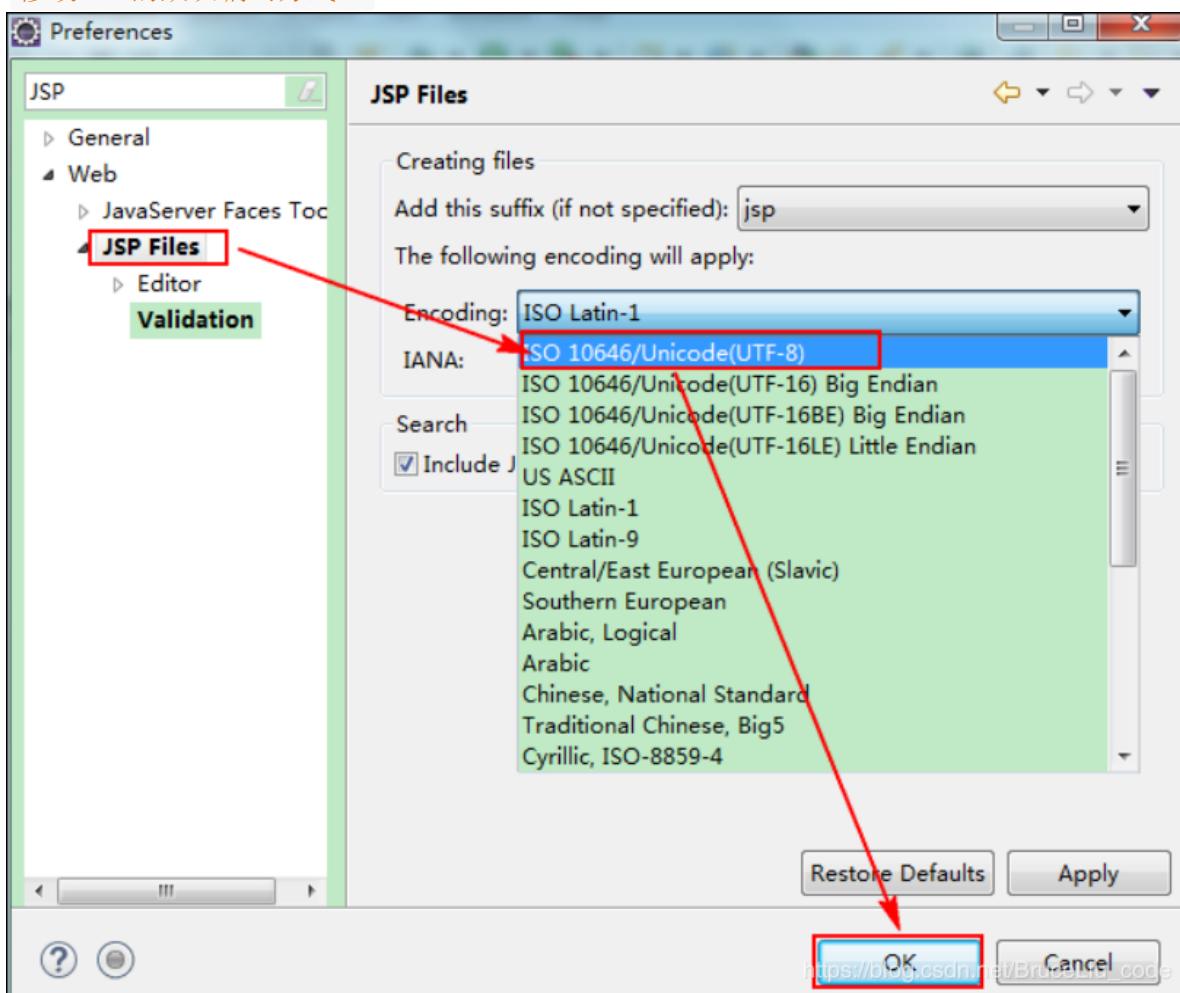
In the bottom right corner of the code area, there is a URL: https://blog.csdn.net/BruceLiu_code.

4.修改默认的JSP编码格式

1 JSP默认使用的是ISO-8859-1编码方式，无法书写中文，保存时报错！



修改JSP的默认编码方式:



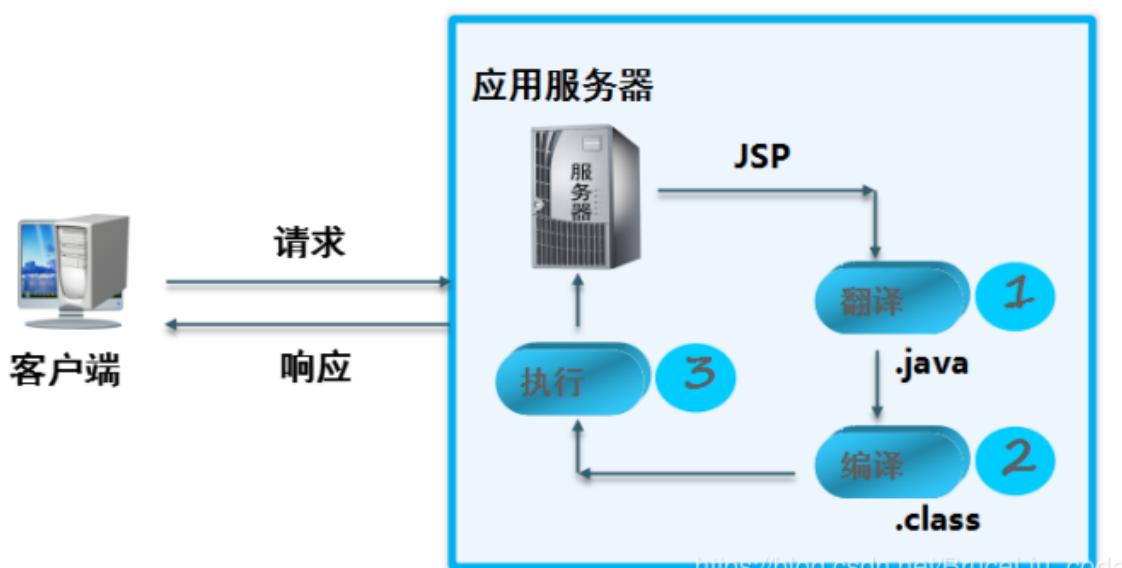
5.JSP中的运行原理（面试题）

Web容器处理JSP文件请求需要经过3个阶段

1. 翻译阶段

2. 编译阶段

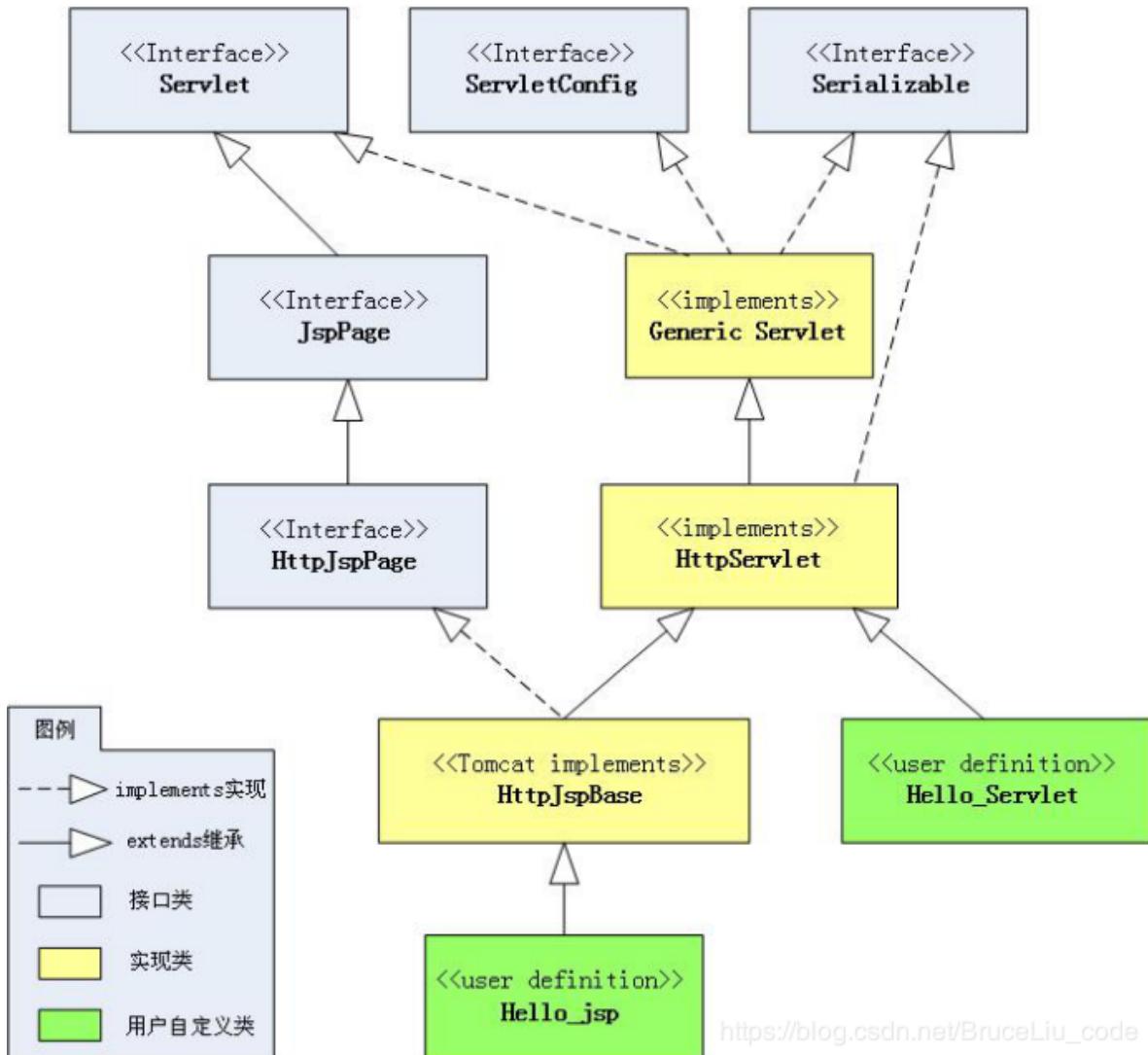
3. 执行阶段



- 第一次访问JSP的时候，JSP会被翻译成.java的源文件，然后再被编译成.class的字节码文件，最后执行字节码文件，呈现运行结果。
- 第二次访问该JSP的时候，**先去检测这个JSP内容有没有发生改变**，如果内容有改变，那么将会执行**翻译→编译→执行**过程。如果JSP没有发生改变，那么将直接运行字节码文件，返回结果。

6.JSP和Servlet的关系

JSP与Servlet的关系



JSP本质就是一个Servlet！只不过JSP侧重于显示数据，Servlet侧重于接收和响应数据！

JSP 迎宾服务员

Servlet 点菜服务员

JSP是Servlet技术的扩展，本质上是Servlet的简易方式，更强调应用的外表表达。JSP编译后是"类servlet"。Servlet和JSP最主要的不同点在于，Servlet的应用逻辑是在Java文件中，并且完全从表示层中的HTML里分离开来。JSP的情况是Java和HTML可以组合成一个扩展名为.jsp的文件。JSP侧重于页面显示，Servlet主要用于控制逻辑（功能）。

7.JSP中的3种脚本（了解）

JSP中可以书写java代码，来完成动态的html的拼接，相对于在servlet中拼接网页要简单很多！在JSP中书写java代码，称为java脚本！

7.1.脚本的声明

语法格式:

```
<%! java代码 %>
```

这个标签可以用来声明变量,方法,类,变量和方法是成员变量和成员方法。类是成员内部类。

示例:

The diagram illustrates the translation of JSP script declarations into Servlet code. On the left, under the heading "jsp", is the following JSP code:

```
<!-- 第一种书写脚本的方式: 脚本声明 -->
<%!
    int a = 1;
    public static int add(int a, int b) {
        return a+b;
    }
%>
```

An arrow labeled "翻译" (Translation) points from the JSP code to the "翻译后的Servlet" (Translated Servlet) section on the right.

翻译后的Servlet

```
public final class script_jsp extends org.apache.jasper.runtime.JspScript
    implements org.apache.jasper.runtime.JspSourceDependent {
    int a = 1;
    public static int add(int a, int b) {
        return a+b;
    }
}
```

A red box highlights the "int a = 1;" declaration and the "add" method definition. Below this, a red note states:

脚本的声明，经过翻译之后，生成的java代码在对应的Servlet中的成员位置，因此，在Servlet的成员位置的代码可以怎么书写，脚本的声明中就可以写

https://blog.csdn.net/BruceLiu_code

7.2.脚本的表达式

语法格式:

```
<%=表达式%>
```

在这个标签中写的java代码将会出现在out()中向页面输出该标签中的代码一定不要加分号! 示例:

The diagram illustrates the translation of JSP script expressions into Servlet code. On the left, under the heading "JSP", is the following JSP code:

```
<!-- 第二种脚本方式: 脚本表达式 -->
body>
    a:<%=a %><br/>
    2+1=<%=add(1, 2) %>
body>
```

An arrow labeled "翻译" (Translation) points from the JSP code to the "翻译后的Servlet" (Translated Servlet) section on the right.

翻译后的Servlet

```
out.write("</head>\r\n");
out.write("<body>\r\n");
out.write("\t<!-- 第二种脚本方式 -->");
out.write("\ta:");
out.print(a);
out.print(add(1, 2));
out.write("</body>\r\n");
out.write("</html>");
} catch (java.lang.Throwable e) {
    if (!e instanceof javax.servlet.ServletException)
        e.printStackTrace();
}
```

A red box highlights the "a:<%=a %>
" and "2+1=<%=add(1, 2) %>" expressions. Below this, a red note states:

在脚本表达式中书写的表达式，翻译之后，对应Servlet的Service方法中out.print()方法中的参数
也就是说，方法中的参数可以怎么写，表达式就可以怎么写

https://blog.csdn.net/BruceLiu_code

脚本表达式，相当于在servlet的doGet或者是doPost方法中使用

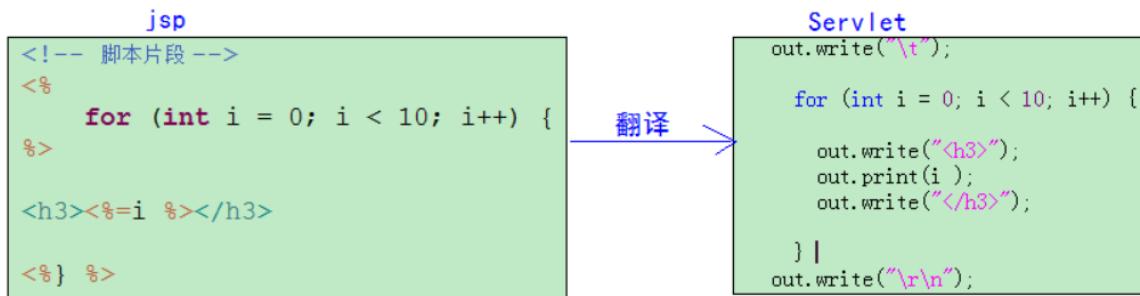
`response.getWriter().print()`方法中参数，所以脚本表达式中，不可以书写分号

7.3.脚本片段

语法格式:

```
<% java代码片段1 %> <% java代码片段2 %> <% java代码片段3 %> ....
```

最终这些片段组合在一起，必须是一段完整的java代码! 在这个标签中编写的java代码 局部的JAVA代码 JSP翻译之后代码是在 service方法中! 示例:



jsp中的脚本片段，可以完成网页的动态的输出效果，可以加一些循环、判断等语句，来控制输出html代码

https://blog.csdn.net/BruceLiu_code

8.JSP中的3大指令

8.1.page指令

语法:

```
1 <%@ 指令名称 属性名称1="属性值1" 属性名称2="属性值2" ..... %>
```

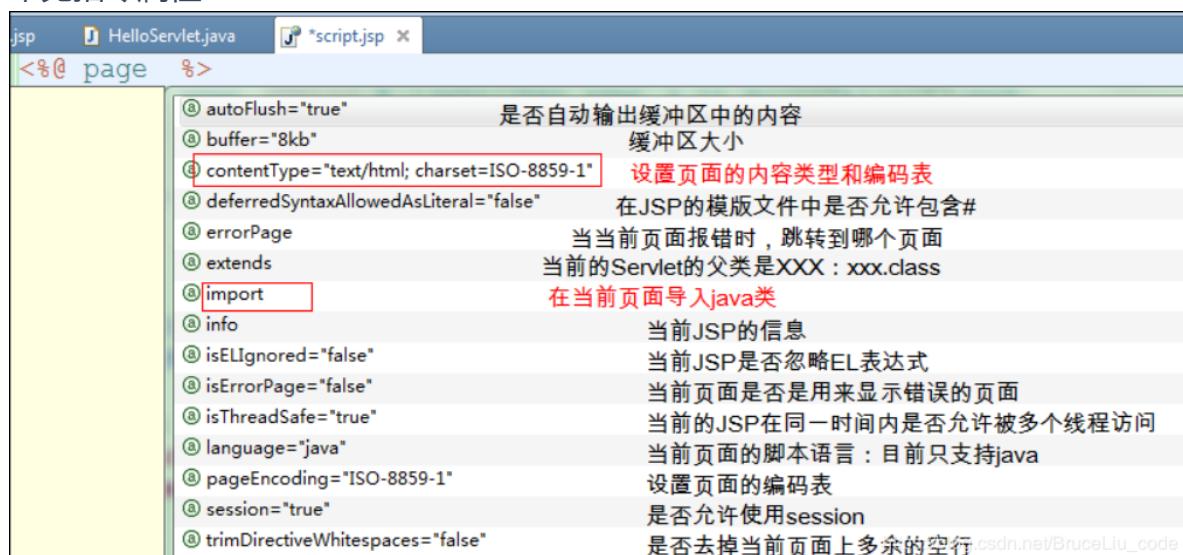
作用:

1 告诉JSP引擎如何jsp文件中的内容

示例:

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
```

常见指令属性:



8.2.includ指令

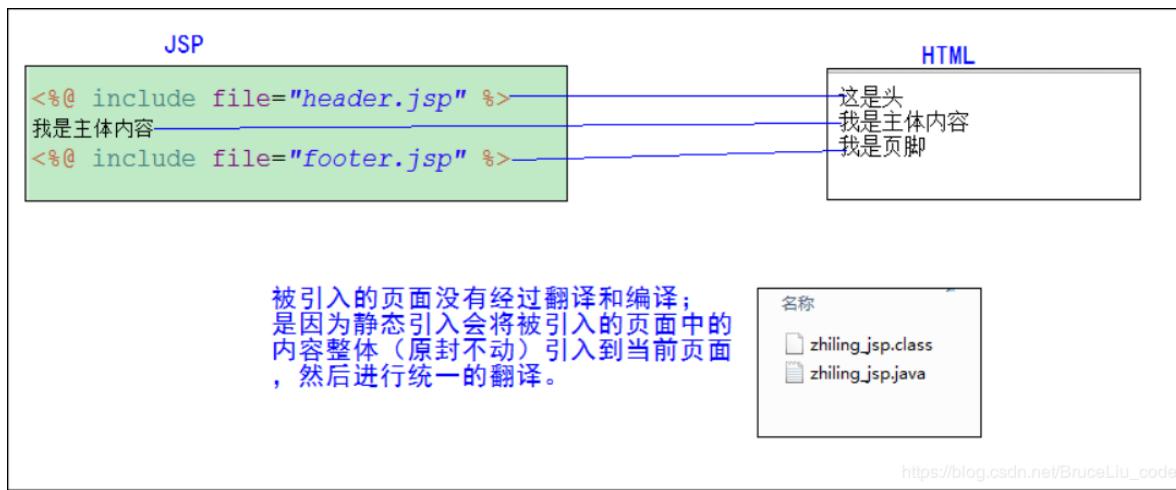
静态包含：

- 1 把其它资源包含到当前页面中

语法格式：

```
1 <%@ include file="header.jsp" %>
```

示例：



注意，这种方式，称为静态引入，这种引入方法，会将引入的页面中的所有内容原封不动的导入到当前的页面，然后对整体进行统一翻译；在引入的页面和被引入的页面中，不能包含相同的变量名，会报变量的重名错误。

特点：

- 1 先合并，后翻译！

8.3.taglib指令(会在JSTL标签库讲解到这指令！)

作用：

- 1 在JSP页面中导入JSTL标签库。替换jsp中的java代码片段。

语法：

```
1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```

JSTL标签库后面详细讲解，JSTL标签库，用来代替脚本片段完成循环和判断等语句