一、C/S与B/S的区别

C/S：

c(client)客户端 s（Server）服务器

特征：需要单独下载客户端 。如 QQ，微信，和平精英，LOL（英雄联盟），钉钉

优点：功能稳定，体验较好

缺点：需要下载，需要更新，占用资源多

B/S：

b（browser）浏览器 s（Server）服务器

特征：不需单独下载客户端，使用浏览器访问

如：京东，淘宝，微博

优点：使用简单，不需下载，开发简单，使用成本低；开发成本低，更新用户无感知

缺点：无法体验过于复杂的功能，受限于网速

原因分析：

1、C/S架构的软件功能较为复杂，软件较大，占用空间较多，需要较多的电脑计算资源

2、B/S架构的软件功能较为单一，多是文字图片类，更容易加载，占用电脑资源较少

得出结果：

需要使用较多的用户端资源，如CPU，内存占用较多，网络传输较为缓慢，用户体验不好。选择放在用户端计算能够提升使用体验

淘宝等BS架构的网站，功能越来越强，网速越来愈快，可以在浏览器上做更多复杂的功能。网络基础设置逐渐完善，开发BS架构的网站是趋势。

二、概念解析

URL：Uniform Resource Locator的缩写 ，统一资源定位符。根据URL地址，可以唯一确定网络中的一台电脑

URI：(Uniform Resource Identifier， URI)，统一资源标志符

URI通常由三部分组成：

①资源的命名机制；

②存放资源的主机名；

③资源自身的名称。

URL的格式由三部分组成：

①第一部分是协议(或称为服务方式)。

②第二部分是存有该资源的主机IP地址(有时也包括端口号)。

③第三部分是主机资源的具体地址，如目录和文件名等。

总结：

URI	URL
URI和URL都定义了资源是什么 较为的抽象的描述了资源	URI和URL都定义了资源是什么 URL还定义了该如何访问资源
可以提供具体资源位置信息 也可以不提供	URL是一种具体的URI，它是URI的一个子集 不仅唯一标识资源，而且还提供了定位该资源的信息
URI 是一种语义上的抽象概念， 可以是绝对的，也可以是相对	URL则必须提供足够的信息来定位，是绝对的

三、Tomcat介绍

Apache Jakarta的开源项目
轻量级应用服务器
开源、稳定、资源占用小

/bin	**存放各种平台下用于启动和停止Tomcat的脚本文件**
/conf	存放Tomcat服务器的各种配置文件
/lib	存放Tomcat服务器所需的各种JAR文件
/logs	存放Tomcat的日志文件
/temp	Tomcat运行时用于存放临时文件
/webapps	当发布Web应用时，默认情况下会将Web应用的文件存放于此目录中
/work	Tomcat把由JSP生成的Servlet放于此目录下

创建一个web项目，建立首页代码

```
<%@ page language="java" import="java.util.*" pageEncoding="ISO-8859-1"%>
<%
String path = request.getContextPath();
String basePath =
request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+pa
th+"/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <base href="<%=basePath%>">

    <title>My JSP 'index.jsp' starting page</title>
    <meta http-equiv="pragma" content="no-cache">
    <meta http-equiv="cache-control" content="no-cache">
    <meta http-equiv="expires" content="0">
    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
    <meta http-equiv="description" content="This is my page">
    <!--
```

```
    <link rel="stylesheet" type="text/css" href="styles.css">
    -->
  </head>

  <body>
    hello world!
  </body>
</html>
```

3使用jsp实现输出

```
<%@page import="java.text.SimpleDateFormat"%>
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%out.print("<h1>课工场Java Web高能充电</h1>"); %>
<p><%out.print("各位小伙伴们~");%> </P>
```

<%@page%>用来设置一个JSP页面的属性
<%out.print();%>或<%out.println(); %>实现页面输出

4 实现jsp输出

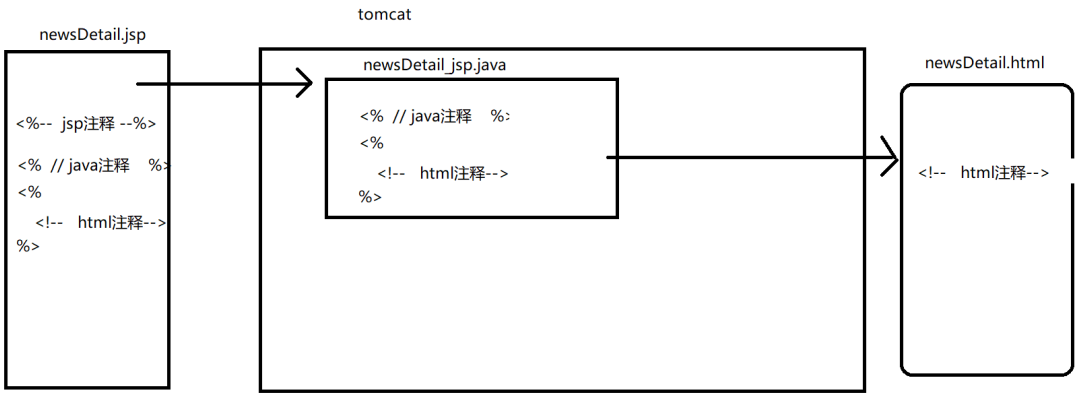
```
<!--新闻的标题-->
<%String title = "课工场\"Java Web\"高能充电";%>
                        <h1><%=title%></h1>
                        <div class="source-bar">
                        <%String author = "小K童鞋";%>
                        发布者：<%=author%> 分类：新闻信息 更新时间：
                        <%
                            Date date = new Date();
                            String time = new SimpleDateFormat("yyyy-MM-
dd").format(date);
                         %>
                         <%=time %>
```

5 JSP声明全局变量及方法

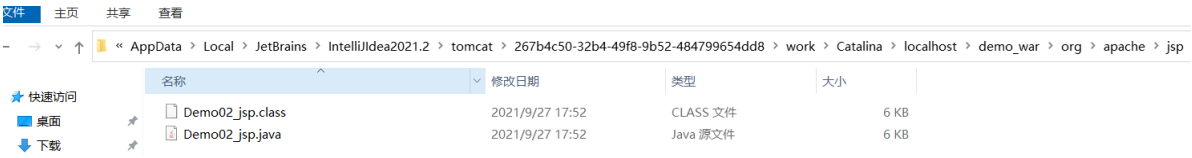
```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<html>
  <head>
  </head>

  <body>
  <%--局部变量和全局变量 --%>
  <%int i = 10; %>
  <%!
    int j = 10;
    public int add(){
        return 5+9;
    }
  %>
  i++; <%= i++ %><br/>
  j++; <%= j++ %><br/>
  <%= add() %>
  </body>
```

```
</html>
```

补充 jsp的工作原理



中间文件的地址：
C:\Users\huyuansong\AppData\Local\JetBrains\IntelliJIdea2021.2\tomcat\267b4c50-32b4-49f8-9b52-484799654dd8\work\Catalina\localhost\demo_war\org\apache\jsp




```
public final class Demo02_jsp extends org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent,
                 org.apache.jasper.runtime.JspSourceImports {


      // 全局变量
      int count = 4 ;

    ..... 省略了一些代码

     try {
      response.setContentType("text/html;charset=UTF-8");
      pageContext = _jspxFactory.getPageContext(this, request, response,
                null, true, 8192, true);
      _jspx_page_context = pageContext;
      application = pageContext.getServletContext();
      config = pageContext.getServletConfig();
      session = pageContext.getSession();
      out = pageContext.getOut();
      _jspx_out = out;

      out.write("\r\n");
      out.write("\r\n");
      out.write("<html>\r\n");
      out.write("<head>\r\n");
      out.write("    <title>Title</title>\r\n");
      out.write("</head>\r\n");
```

```
      out.write("<body>\r\n");
      out.write("\r\n");
      out.write("    ");

        // 局部变量
        //int count = 1 ;
        out.print(count);

      out.write("\r\n");
      out.write("\r\n");
      out.write("\r\n");
      out.write("    ");
      out.write("\r\n");
      out.write("    ");

        out.print(count);

      out.write("\r\n");
      out.write("\r\n");
      out.write("\r\n");
      out.write("</body>\r\n");
      out.write("</html>\r\n");
    } catch (java.lang.Throwable t) {
      if (!(t instanceof javax.servlet.jsp.SkipPageException)){
        out = _jspx_out;
        if (out != null && out.getBufferSize() != 0)
          try {
            if (response.isCommitted()) {
              out.flush();
            } else {
              out.clearBuffer();
            }
          } catch (java.io.IOException e) {}
        if (_jspx_page_context != null)
_jspx_page_context.handlePageException(t);
        else throw new ServletException(t);
      }
    } finally {
      _jspxFactory.releasePageContext(_jspx_page_context);
    }
  }
```