Servlet vs JSP(Java Server Pages)
==========

Servlet => meant for processing logic
eg: verify the user, checking account balance, collecting data from database,....

JSP => meant for Presentation logic, to display something to the end user
eg: Display login page, display inbox page, display error page

Note:
1. Inside Servlet(java code) we should not write presentation logic(no usage of println()).
2. Inside JSP we should not write buisness logic meant for presentation logic only (no usage of java code inside jsp's).
3. JSP program consists of only tags.
4. JSP programs are autocompiled and autoloaded.
5. Servlet programs are executed by "CATALINA" container vs JSP programs are executed by "JASPER" container.
6. JSP is a technology which is built on top of Servlet Technology.

Writing the code using jsp technology is so easy compared to writing the code using Servlet?
Ans. JSP is easy because of its abstraction from the programmer.

JSP API
======
1. It contains 2 interfaces
a. JspPage(I)
b. HttpJspPage(I)

JspPage(I)
a. It contains 2 methods jspInit(), jspDestroy()

HttpJspPage(I)
It is an interface which contains only one method _jspService(HSR,HSResp) throws SE,IOE

Note:

```
public abstract class HttpJspBase{
    public final void init(ServletConfig config) throws
javax.servlet.ServletException{
                jspInit();
    }
    public final void destroy(){
                jspDestroy();
    }
    public final void service(HttpServletRequest request,HttpServletResponse
response) throws SE,IOE{
                _jspService(request,response);//inside this method automatically
our logic written in jsp will be placed in translated servlet
    }
}
```

Is it possible to write init(), destroy(),service() in our jsp code?
Answer: No, not possible becoz these methods are final in HttpJspBase class.

Is it possible to write _jspService(requese,response) method explicitly inside JSP?
Answer: No, if we write inside the translated servlet there will be 2 _jspService() method which would lead to
compile time error.

What is the significance of _ symbol in _jspService()?
Answer: It indicates this method is generated automatically by JSP engine/jasper container and we cannot write it explicitly.

Life Cycle of JSP
==========
1. Translation phase(.jsp -> .java)
2. Compilation phase(.java -> .class)
3. Servlet loading
4. Servlet Instantiation
5. Servlet Initialization (jspInit())
6. Request Processing(_jspService(request,response))
7. Servlet DeInstantiation(jspDestroy())

refer: FirstApp

In case of JSP
execution,Translation,Compilation,Loading,Instantiation,Initialization will happen for firstrequest where as
from second request only "RequestProcessing" will happen.
This would increase the response time for the first request and would not bring uniformity in the response time for clients.
To avoid this we need to go "Precompilation" of jsp code as shown below
	http://localhost:9999/FirstApp/index.jsp?jsp_precompile=true
		Translation,Compilation,Servlet loading,Servlet Instantiation,ServletInitialization

After compilation of jsp code we can send the request as shown below
	first request	=> http://localhost:9999/FirstApp/index.jsp-----> only request processing
	second request => http://localhost:9999/FirstApp/index.jsp----> only request processing

JSP Scripting Elements
==========
1. Template Text
2. Scripting Elements
3. Standard and custom actions
4. Express language elements

1. Template Text
	It contains plain text with html tags
	For this processing is not required and it will become argument to write() inside _jspService() method.

eg:
```
	<h1>The server time is <%= new java.util.Date() %></h1>
```

The translated servlet code is
```
	public final class index_jsp extends HttpJspBase....
	{
		public void _jspService(){
			out.write("<h1>The server time is ");
			out.print(new java.util.Date());
			out.write("</h1>");
		}
	}
```

Q>Template text will become argument to write() method where as expression value will become argument to print(), what
is the reason?
Answer: write() -> it can take only character data as a argument.
print() -> It can take any type of data as an argument.

Directives
========
Directives are used to provide general information about jsp page to the JSPEngine.
Directives are translation time instructions to JSP Engine.
There are 3 types of directives
a. page directive
b. include directive
c. taglib directive

page directive
============
=> This directive is used to specify overall properties of jsp page to JSP Engine.
=> syntax: <%@ page [attribute_Name]=[attribute_Value] %>
=> There are 13 attributes associated with page directive
a. import
b. session
c. contentType
d. isElIgnored
e. isThreadSafe
f. language
g. extends
h. buffer
i. autoFlush
j. info
k. pageEncoding
l. isErrorPage
m. errorPage

1. import
This attribute is used to import the classes being used in our jsp pages.
We can use import attribute as shown below
1. <%@ page import = "java.util.*"%>
2. <%@ page import = "java.util.Date, java.util.ArrayList" %>
3. <%@ page import = "java.util.Date" import = "java.util.ArrayList"%>
4. <%@ page import ="java.util.Date"%>
<%@ page import = "java.util.ArrayList"%>
Default value of import attribute is
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;
import java.lang.*;
Note: with in jsp page, we are not allowed to take any attribute multiple times with mulitple values.
But we can take multiple times with same value, This rule is not applicable only for "import" attribute.

eg:
```
<%@ page language ="java" import = "java.util.Date" import = "java.util.ArrayList"
%>
<h1>The server time is :: <%= new Date()%></h1>
```

```
<h2>Array List object creation is :: <%= new ArrayList().size()%></h2>

output
The server time is :: Tue Feb 14 22:37:24 IST 2023
Array List object creation is :: 0

2. session
       By default session object is available in every jsp page.
       session -> HttpSession
       If we don't want session object to be used inside jsp page then we can make
it unavilable using page directive
       <%@ page session = "true" %>
                                          |
            javax.servlet.http.HttpSession session = null;
            session  = pageContext.getSession();

      <%@ page sesion = "false" %>
         Then the above 2 lines won't be available in the translated servlet code.

Note:
      1. <%@ page session = "true" session = "true" %>(valid)
      2. <%@ page session = "true" session = "false" %>(invalid)
      3. <%@ page import = "java.util.Date" import = "java.util.ArrayList"%>(valid)
      4. <%@ page session ="true" %>(valid)
      5. <%@ page imoprt = 'java.util.Date' %>(valid)
      6. <%@ page session = "divya" %>(invalid)
      7. <%@ page session = "False" %>(invalid)
      8. <%@ page session = "TRUE" %>(invalid)
```