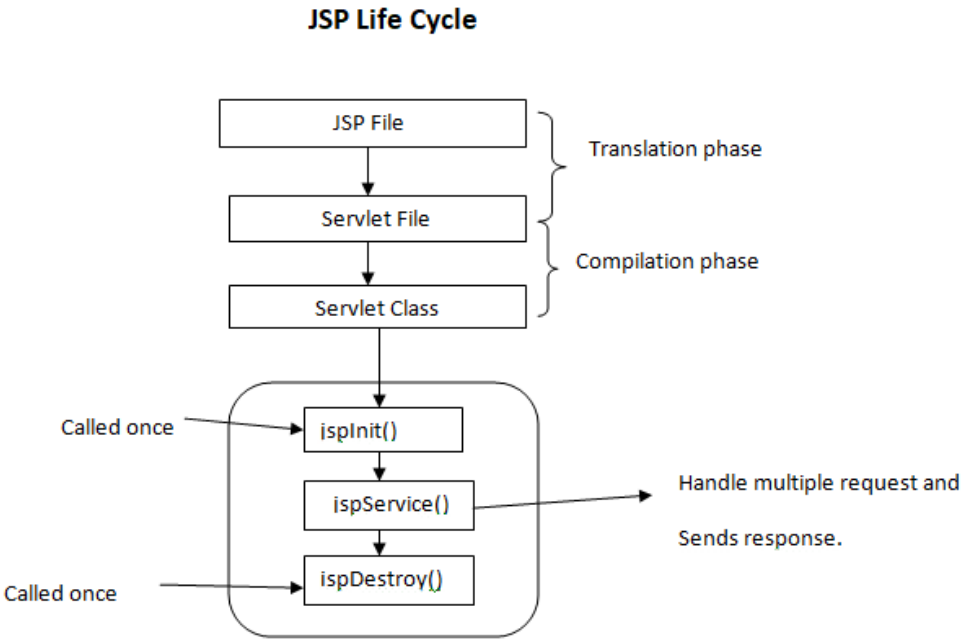
Life cycle of JSP

A Java Server Page life cycle is defined as the process that started with its creation which later translated to a servlet and afterward servlet lifecycle comes into play. This is how the process goes on until its destruction.



Following steps are involved in the JSP life cycle:

- Translation of JSP page to Servlet
- Compilation of JSP page(Compilation of JSP into test.java)
- Classloading (test.java to test.class)
- Instantiation(Object of the generated Servlet is created)
- Initialization(jspInit() method is invoked by the container)
- Request processing(_jspService()is invoked by the container)
- JSP Cleanup (jspDestroy() method is invoked by the container)

We can override jspInit(), jspDestroy() but we can't override _jspService() method.

Translation of JSP page to Servlet :
This is the first step of the JSP life cycle. This translation phase deals with the Syntactic correctness of JSP. Here test.jsp file is translated to test.java.

Compilation of JSP page :
Here the generated java servlet file (test.java) is compiled to a class file (test.class).

Classloading :
Servlet class which has been loaded from the JSP source is now loaded into the container.

Instantiation :
Here an instance of the class is generated. The container manages one or more instances by providing responses to requests.

Initialization :
jspInit() method is called only once during the life cycle immediately after the generation of Servlet instance from JSP.

Request processing :
_jspService() method is used to serve the raised requests by JSP. It takes request and response objects as parameters. This method cannot be overridden.

JSP Cleanup :
In order to remove the JSP from the use by the container or to destroy the method for servlets jspDestroy()method is used. This method is called once, if you need to perform any cleanup task like closing open files, releasing database connections jspDestroy() can be overridden.

The JspPage interface

According to the JSP specification, all the generated servlet classes must implement the JspPage interface. It extends the Servlet interface. It provides two life cycle methods.

Methods in this Interfaces:

jspInit(), jspDestroy()

Method of HttpJspPage interface:

public void _jspService(): It is invoked each time when request for the JSP page comes to the container. It is used to process the request. The underscore _ signifies that you cannot override this method.

Some classes are :

JspWriter

PageContext

JspFactory

JspEngineInfo

JspException

JspError

JSP implicit object:

Object	Type
out	JspWriter
request	HttpServletRequest
response	HttpServletResponse
config	ServletConfig
application	ServletContext
session	HttpSession
pageContext	PageContext
page	Object
exception	Throwable

Jsp request implicit object:

1. **getParameter(String name)** – This method is used to get the value of a request's parameter. For example at login page user enters user-id and password and once the credentials are verified the login page gets redirected to user information page, then using request.getParameter we can get the value of user-id and password which user has input at the login page.

```
String Uid= request.getParameter("user-id");
String Pass= request.getParameter("password");
```

2. **getParameterNames() –** It returns enumeration of all the parameter names associated to the request.

```
Enumeration e= request.getParameterNames();
```

3. **getParameterValues(String name) –** It returns the array of parameter values.

```
String[] allpasswords = request.getParameterValues("password");
```

4. **getAttribute(String name) –** Used to get the attribute value.
   request.getAttribute("admin") would give you the value of attribute admin.
5. **getAttributeNames() –** It is generally used to get the attribute names associated to the current session. It returns the enumeration of attribute names present in session.

```
Enumerator e = request.getAttributeNames();
```

6. **setAttribute(String,Object) –** It assigns an object's value to the attribute. For example I have an attribute **password** and a String object str which has a value **"admin"** then calling request.setAttribute("password", str) would assign a value **admin** to the attribute **password**.
7. **removeAttribute(String) –** By using this method a attribute can be removed and cannot be used further. For e.g. If you have a statement **request.removeAttribute("userid")** on a JSP page then the userid attribute would be completely removed and request.getAttribute("userid") would return **NULL** if used after the removeAttribute method.
8. **getCookies() –** It returns an array of cookie objects received from the client. This method is mainly used when dealing with cookies in JSP.
9. **getHeader(String name) –** This method is used to get the header information of the request.
10. **getHeaderNames() –** Returns enumerator of all header names. Below code snippet would display all the header names associated with the request.

```
Enumeration e = request.getHeaderNames();
     while (enumeration.hasMoreElements()) {
         String str = (String)e.nextElement();
         out.println(str);
     }
```

11. **getRequestURI() –** This method (request.getRequestURI()) returns the URL of current JSP page.
12. **getMethod() –** It returns HTTP request method. request.getMethod(). For example it will return GET for a Get request and POST for a Post Request.
13. **getQueryString() –** Used for getting the query string associated to the JSP page URL. It is the string associted to the URL after question mark sign (?).

Exception Handling:

https://www.javatpoint.com/exception-handling-in-jsp

Scripting Element:

Derective:

https://www.studytonight.com/jsp/jsp-directive-tag.php

jsp:forward action tag

The jsp:forward action tag is used to forward the request to another resource it may be jsp, html or another resource.

index.jsp

```
<html>
<body>
<h2>this is index page</h2>
<jsp:forward page="printdate.jsp" />
</body>
</html>
```

printdate.jsp

```
<html>
<body>
<% out.print("Today is:"+java.util.Calendar.getInstance().getTime()); %>
</body>
</html>
```

index.jsp

```
<html>
<body>
<h2>this is index page</h2>
<jsp:forward page="printdate.jsp" >
<jsp:param name="name" value="javatpoint.com" />
</jsp:forward>
</body>
</html>
```

Include action tag:

https://www.javatpoint.com/jsp-include-action

MVC

https://www.javatpoint.com/MVC-in-jsp