//https://people.apache.org/~jim/NewArchitect/webrevu/2001/02\_02/developers/index03.html

How JSP works?

The primary component of a servlet-based implementation of JavaServer Pages is a servlet which is the page compiler. The container is configured to call this servlet for all requests with URLs that match the JSP file extension, and it is the presence of this servlet and its associated Java classes that turns a servlet container into a JSP container. The task of this servlet is not just finding JSP pages in response to such requests, but actually compiling them: each JSP page is compiled into a page-specific servlet whose purpose is to generate the dynamic content specified by the original JSP document.

Whenever the HTTP server receives a request for a URL corresponding to a JSP, that request is sent to the JSP container, which invokes the page compiler servlet to handle the request. If this is the first time a request has been received for a particular JSP file, this servlet compiles the JSP file into a servlet.

To compile a page, the JSP page compiler parses through its contents, looking for JSP tags. As it parses the file, it translates its contents into the equivalent Java source code which, when executed, will generate the output indicated by the contents of the original file. Static HTML is translated into Java strings, which will be written unmodified and in their original sequence into an output stream. JSP tags are translated into Java code for generating dynamic content: Bean tags are translated into the corresponding object and property calls, while scripting elements are transferred as is. This code will be mixed in with the output of the original static HTML, so that the dynamic content is inserted into the output in the correct location. This source code is then used to write the service methods for a servlet, such that running it for a request has the effect of producing the content specified by the original JSP file. Once all the servlet code has been constructed, the page compiler servlet calls the Java compiler to compile this source code and add the resulting Java class file to the appropriate directory in the JSP container's class path.

Once the compiled JSP page servlet is in place, the page compiler servlet then invokes this new servlet to generate the response for the original request. Of course, this parsing, code generation, and compiling incurs quite a bit of overhead. Fortunately, these steps are required only the first time a request for a given JSP page is received. All subsequent requests can be passed directly to the already-compiled page servlet for immediate processing.

As long as the contents of the original JSP page remain unchanged, there is no need to generate a new servlet, since the Java code corresponding to those contents remains the same. For this reason, the very first step taken by the JSP page compiler when it receives a request for a JSP is to check the time stamp for the JSP file corresponding to the requested URL, to determine when that file was modified or created. The page compiler will also check the time stamp on the compiled servlet for this JSP page. If no compiled servlet is found, or if the time stamp on the JSP file is more recent than the one on the compiled page servlet, then a new servlet must be generated. This means that the (new or modified) JSP file must be parsed and translated into source code, and this new source code must be compiled. If the compiled servlet is newer than the JSP file, however, no new compilation is required and control can be transferred directly to the servlet to finish processing the request, saving considerable time. So while the first request for a new or recently modified JSP page will be slow, all later requests go straight to the compiled servlet for response generation.