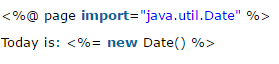
* **Apache tomcat (tomcat server)** – an open source Java servlet container developed by Apache software foundation (ASF).
* Implements Java EE specifications including: java servlet, javaServer pages, Java EL and websocket.
* **Catalina** – a tomcat servlet container. It implements sun microsystems specifications for servlet and JavaServer Pages (JSP). It can be represented as a hierarchy of objects that contain references to each other.
* J**asper** - is Tomcat's JSP Engine. .Jasper [parse](https://en.wikipedia.org/wiki/Parsing)s [JSP files](https://en.wikipedia.org/wiki/JSP_files) to compile them into Java code as servlets (that can be handled by Catalina). At runtime, Jasper detects changes to JSP files and recompiles them.
* **JavaServer Pages(JSP**) - only permit GET, POST and HEAD. is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications.
* **JSP comment** <%-- --%> - intended to follow developers
* **Directives** <%--@page --%> - these are messages that tells the web container how to translate a JSP page into a servlet.
* **3 Types**

**Page** - directive defines attributes that apply to an entire JSP page.



**Attributes:**

**import -** is used to import class,interface or all the members of a package.It is similar to import keyword in java class or interface.



**contentType -** defines the MIME(Multipurpose Internet Mail Extension) type of the HTTP response.The default value is "text/html;charset=ISO-8859-1".



**Extends** - defines the parent class that will be inherited by the generated servlet.It is rarely used.

**Info -** simply sets the information of the JSP page which is retrieved later by using getServletInfo() method of Servlet interface.



**Buffer -**  sets the buffer size in kilobytes to handle output generated by the JSP page.The default size of the buffer is 8Kb.

**Language -** specifies the scripting language used in the JSP page. The default value is "java".

**isELIgnored -**  We can ignore the Expression Language (EL) in jsp by the isELIgnored attribute. By default its value is false i.e. Expression Language is enabled by default. We see Expression Language later.

**isThreadSafe -** If you want to control this behaviour of JSP page, you can use isThreadSafe attribute of page directive.The value of isThreadSafe value is true.If you make it false, the web container will serialize the multiple requests, i.e. it will wait until the JSP finishes responding to a request before passing another request to it.

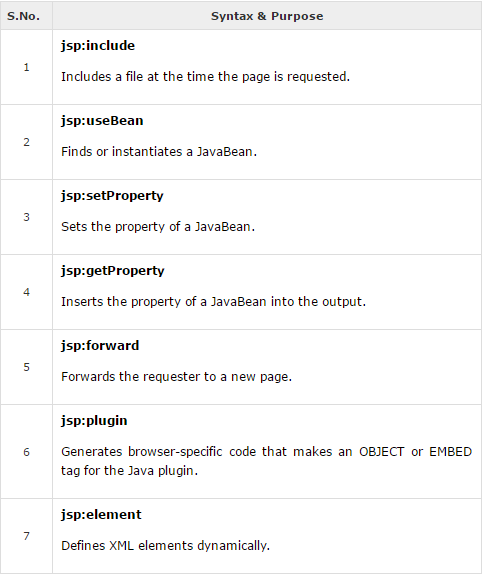
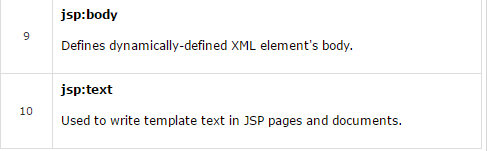
**errorPage -** is used to define the error page, if exception occurs in the current page, it will be redirected to the error page.

**isErrorPage -** The isErrorPage attribute is used to declare that the current page is the error page.

**Include -** is used to [include](https://www.javatpoint.com/jsp-include-directive) the contents of any resource it may be jsp file, html file or text file. The [include](https://www.javatpoint.com/jsp-include-directive) directive [include](https://www.javatpoint.com/jsp-include-directive)s the original content of the [include](https://www.javatpoint.com/jsp-include-directive)d resource at page translation time (the jsp page is translated only once so it will be better to [include](https://www.javatpoint.com/jsp-include-directive) static resource).

**Tag-library** - is used to define a tag library that defines many tags. We use the TLD (Tag Library Descriptor) file to define the tags. In the custom tag section we will use this tag so it will be better to learn it in custom tag.

* **Static content** – template text. is used to define a tag library that defines many tags. We use the TLD (Tag Library Descriptor) file to define the tags. In the custom tag section we will use this tag so it will be better to learn it in custom tag.
* **JSP standard actions** – indicated as element tags. These actions use constructs in XML syntax to control the behavior of the servlet engine. You can dynamically insert a file, reuse JavaBeans components, forward the user to another page, or generate HTML for the Java plugin.

**COMMON ATTRIBUTES:**

**ID attribute -** uniquely identifies the Action element, and allows the action to be referenced inside the JSP page. If the Action creates an instance of an object, the id value can be used to reference it through the implicit object PageContext.

**Scope attribute -** This attribute identifies the lifecycle of the Action element. The id attribute and the scope attribute are directly related, as the scope attribute determines the lifespan of the object associated with the id. The scope attribute has four possible values: (a) page, (b)request, (c)session, and (d) application.

<jsp:forward …> - passes servelets

Usually used together:

<jsp:attribute …>

<jsp:element …>

<jsp:body …>

<jsp:directive page …>

<jsp:scriptlet …>

<jsp:text …>

* **EL Expressions** – expression language. which provides an important mechanism for enabling the presentation layer (web pages) to communicate with the application logic (managed beans). The EL is used by both JavaServer Faces technology and JavaServer Pages (JSP) technology. The EL represents a union of the expression languages offered by JavaServer Faces technology and JSP technology.

Syntax = ${ …} (example: ${param.name}! – retrieve incoming request name)

${header[‘User-Agent’]} – retrieve incoming header

${paramValues} – returns collection for param

${requestScope}

* JSP
* Scriptlet scriptlet – arbitrary java chunk codes. <% …… %>
* Scriptlet expressions - <%= …….. %>
* Scriptlet declaration - <%! ……… %>

JSTL Library – allows programmatic constructs

<%@taglib uri = “ “ prefix = “xml”>

Core, fmt, functions, sql, xml

* ForEach – iterate set of items
* ForTokens – iterate strings

<sql :

* setDataSource – allows to connect to a backend connection
* param
* dateParam
* transaction

<fmt:

* setBundle
* **Session handling** – set of request that are correlated with each other. Becomes mandatory when a requested data need to be sustained for further use. Since http protocol considers every request as a new one, session handling becomes important.

HTTP – stateless protocol

* Server doesn’t keep any information about the client.
* **Cookie** – text information (small text file). Used to establish session.  (also called **web cookie**, **Internet cookie**, **browser cookie** or simply **cookie**) is a small piece of data sent from a website and stored on the user's computer by the user's [web browser](https://en.wikipedia.org/wiki/Web_browser) while the user is browsing. Cookies were designed to be a reliable mechanism for websites to remember [stateful](https://en.wikipedia.org/wiki/Program_state" \o "Program state) information (such as items added in the shopping cart in an online store) or to record the user's browsing activity (including clicking particular buttons, [logging in](https://en.wikipedia.org/wiki/Access_control), or recording which pages were visited in the past). They can also be used to remember arbitrary pieces of information that the user previously entered into form fields such as names, addresses, passwords, and credit card numbers.
* **Non persistent cookie** – when closed the cookie jar will be empty.
* **Persistent cookie** – not for session handling (can construct cookie when in HTTPS). Also called a *permanent cookie*, or a *stored cookie*, a [cookie](http://www.webopedia.com/TERM/C/cookie.html) that is stored on a users [hard drive](http://www.webopedia.com/TERM/H/hard_drive.html) until it expires (persistent cookies are set with expiration dates) or until the user deletes the cookie. Persistent cookies are used to collect identifying information about the user, such as Web surfing behavior or user preferences for a specific Web site.
* Jsession id (java)

Scoped objects

.invalidate()

* Session

.getLastAccessedTime()

.setMaxInActive() – set timeout

* applicationScope – largest scope
* webapplicationListener
* contextInitialized – start session
* contextDestroyed – stop session
* web.xml

.<content param>

* **PHP** – server side scripting – general purpose programming language. Recursive acronym for *PHP: Hypertext Preprocessor*) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

<?php ……. ?> - php script

Echo – print

$prodID – variables are declared with dollar sign ($)

\_construct – special way to introduce constructions

$this 🡪items = []; - array - dynamic in size.

* Superglobals – available in all scope. Are built- in variables that are always available in all scope.

$Global

$\_SERVER – incoming request from the server

“ “ – embed variable names

‘ ‘ – treated literally

$\_GET

$\_POST

$\_FILES

* Node.js – mean stack (for monggo db)

BSON – binary object notation

* Express.js – backend
* Angular.js – frontend
* **Webserver Security**

OWASP

* **Hash** – function that is to be/can be applied to a string.  is any function that can be used to map data of arbitrary size to data of fixed size. The values returned by a **hash** function are called **hash** values, **hash** codes, digests, or simply **hashes**.

**MD5** - is a widely used hash function producing a 128-bit hash value. Although MD5 was initially designed to be used as a cryptographic hash function, it has been found to suffer from extensive vulnerabilities.

**SHA -** (Secure Hash Algorithm 1) is a cryptographic hash function designed by the United States National Security Agency and is a U.S. Federal Information Processing Standard published by the United States NIST.

* **Salted Hash -**
* **Cross-site Request forgery -** also known as one-click attack or session riding and abbreviated as CSRF(sometimes pronounced sea-surf) or XSRF, is a type of malicious exploit of a website where unauthorized commands are transmitted from a user that the web application trusts.
* **Cross- site scripting -** is a type of computer security vulnerability typically found in web applications. XSS enables attackers to inject client-side scripts into web pages viewed by other users. Across-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy.

**REFERENCES:**

[**http://php.net/manual/en/intro-whatis.php**](http://php.net/manual/en/intro-whatis.php)

[**https://en.wikipedia.org**](https://en.wikipedia.org/wiki/Hash_function)

[**https://security.stackexchange.com/questions/51959/why-are-salted-hashes-more-secure-for-password-storage**](https://security.stackexchange.com/questions/51959/why-are-salted-hashes-more-secure-for-password-storage)

[**https://www.maxcdn.com/one/visual-glossary/static-content/**](https://www.maxcdn.com/one/visual-glossary/static-content/)

[**https://www.tutorialspoint.com**](https://www.tutorialspoint.com/jsp/jsp_actions.htm)

[**http://docs.oracle.com/javaee/6/tutorial/doc/gjddd.html**](http://docs.oracle.com/javaee/6/tutorial/doc/gjddd.html)

[**https://www.hscripts.com/tutorials/jsp/jsp-session.php**](https://www.hscripts.com/tutorials/jsp/jsp-session.php)

[**http://www.webopedia.com/TERM/P/persistent\_cookie.html**](http://www.webopedia.com/TERM/P/persistent_cookie.html)

[**https://www.javatpoint.com/exception-handling-in-jsp**](https://www.javatpoint.com/exception-handling-in-jsp)

[**https://tomcat.apache.org/tomcat-5.5-doc/catalina/funcspecs/mbean-names.html**](https://tomcat.apache.org/tomcat-5.5-doc/catalina/funcspecs/mbean-names.html)