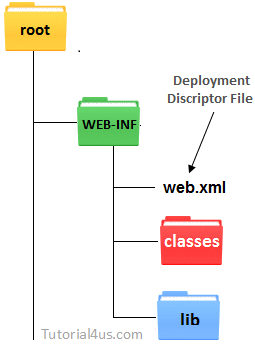
**What is web.xml?**

Java **web** applications use a deployment descriptor file to determine how URLs map to servlets, which URLs require authentication, and other information. This file is named **web**.**xml** , and resides in the app's WAR under the **WEB**-INF/ directory. **web**.**xml** is part of the servlet standard for **web** applications. It is a web application deployment descriptor file, contains detail description about web application like configuration of Servlet, Session management, Startup parameters, Welcome file..etc. We cannot change the directory or extension name of this **web.xml** because it is standard name to recognized by container at run-time.

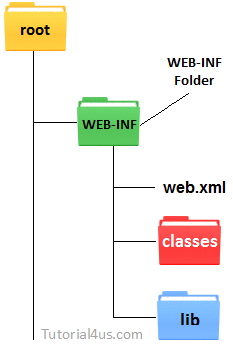
web.xml is present inside the Web-INF folder.



**what is WEB-INF?**

This is a web application initialization folder to recognized by the web container at run time then folder name should be **Web-INF** to deployed the web application successfully otherwise deployment name unsuccessful.

In side **Web-INF** folder we put **web.xml** file, classes folder, lib folder and any user defined folder.



The **WEB-INF** directory contains the deployment descriptors for the Web application (**web.xml** and **weblogic.xml**) and two subdirectories for storing compiled Java classes and library JAR files. These subdirectories are respectively named **classes** and **lib**. JSP taglibs are stored in the **WEB-INF** directory at the top level of the staging directory.

**what is META-INF?**

The **META**-**INF** folder is the home fort the MANIFEST.MF file. This file contains **meta**data about the contents of the JAR. For example, there is an entry called Main-Class that specifies the name of the Java class with the static main() for executable JAR files.

**what is MAINFEST.MF?**

The **manifest** is a special **file** that can contain information about the files packaged in a JAR **file**. By tailoring this "meta" information that the **manifest** contains, you enable the JAR **file** to serve a variety of purposes. The manifest file that is used to define extension and package related data. So mostly it can contain human readable info like version, producer.

Working with Manifest Files:The manifest is a special file that can contain information about the files packaged in a JAR file. By tailoring this "meta" information that the manifest contains, you enable the JAR file to serve a variety of purposes.

**Generate MANIFEST.MF entries using Maven**

Maven projects that use the Maven JAR, EJB or WAR packaging plug-ins can generate a MANIFEST.MF file with a Dependencies entry. This does not automatically generate the list of dependencies, this process only creates the MANIFEST.MF file with the details specified in the pom.xml.

Prerequisites

1. You must already have a working Maven project.
2. The Maven project must be using one of the JAR, EJB, or WAR plug-ins ( maven-jar-plugin, maven-ejb-plugin, maven-war-plugin).

**classes dir in Web app?**

A web application typically consists of a structured hierarchy of directories. Within the web application directory hierarchy, a special directory named WEB-INF must be created. This directory is the repository for meta-information relating to the web application. It is a private directory; no resource within it is accessible to a client. However, the resources in the WEB-INF directory are visible to servlets and Java classes that reside within the web application.