There are 3 ways to deploy a web application to a server.

1. Placing the deployment directory structure inside *webapps* directory of tomcat.
2. Creating a WAR file and then deploy it.

There are 2 ways to create a WAR file:

* 1. Create a WAR file manually and deploy it.
  2. Use a build tool like ANT to automatically create a WAR and deploy it.

There are 2 ways to deploy a WAR file to a server:

1. By server console panel.
2. By copying the war file in specific folder of server. The server will extract the file automatically.

Placing deployment directory structure inside *webapps* directory:

1. Create the directory structure as above.
2. *src* directory contains the source code for the application. It is not mandatory to put *src* folder here, but for the sake of simplicity, we are keeping it at this location.
3. To compile the source code:

$ javac -classpath /path-to/tomcat/lib/servlet-api.jar -d classes src/java-program.java

In our example, open the command prompt at *Servlets-Study* directory:

$ TOMCAT\_HOME\webapps\Servlets-Study> javac -classpath ../../lib/servlet-api.jar -d WEB-INF/classes src/BeerServlet.java

1. The compiled classes will land up in WEB-INF/classes directory.
2. Now, we can simply start the tomcat server and access the application.

Creating a WAR file manually

1. If using command line, go inside the directory *Servlets-Study*. To create a WAR file, we should be one level above WEB-INF directory.
2. Use the command to create a WAR: $ jar -cvf Servlets-Study.war \*

-c = Create a file

-v = Generate a verbose output

-f = Specify an archive file name

1. \* will copy all the files & directories in *Servlets-Study* directory into the WAR file. Since our example contains source code files as well, we need to exclude those files.
2. The jar program *by itself* will not be able to do this. We need to add in some other type of script or build tool like ANT to accomplish what we are looking for.