Chapter

7

SECTION II: JAVA SERVLETS

Working With Servlets

Working with Servlets is an important step in the process of application development and delivery through the Internet. A Servlet as explained earlier is a Java program that must be run by a Java Servlet engine on a Java enabled Web server. The Servlet's output is delivered to a Web browser.

GlassFish Server 4.0 that comes bundled with the **NetBeans IDE 7.4** has built-in support for running Java Servlets. **This book uses the bundled server to demonstrate and run Servlets.**

Getting Started

Working with Servlets requires the knowledge of**:**

- What is a web application
- The organization of a web application
- How to create a Servlet file
- Compiling and building the application

- Deploying the application
- Executing the Servlet / web application

What Is A Web Application?

A Web application, sometimes called a Web app, is an application that is accessed using a Web browser over a network such as the Internet or an Intranet. While many Web applications are written directly in PHP or Perl, Java remains a commonly used programming language for writing Web applications. This is especially true for Web based enterprise applications [usually referred to as enterprise Web applications].

Java EE 7 provides several useful components such as JavaServer Pages, JavaServer Faces, Servlets, client-side applets, Enterprise JavaBeans, JDBC and several other Web service technologies for writing enterprise Web applications. This chapter focuses on writing Servlets as the Web application of choice.

Basically, a Web application is defined as a hierarchy of directories and files in a standard layout. There are two ways in which such a hierarchy can be accessed**:**

- The first is where each directory and file exists in the file system separately. This is termed as an application in its **unpacked** form. It is used during application development
- The second is where all the sub directories are zipped together in a **packed** form known as a **W**eb **AR**chive or WAR file. It is used when distributing the application to be installed

Organization Of A Web Application

Before beginning to work with Servlets, it is useful to examine the runtime organization of a Web application. Before the Servlet API Specification, version 2.2, across servers there was little consistency in the WAR file structure.

However, Web servers that conform to the Servlet API version 2.2 [or later] the **W**eb application **AR**chive must be in a standard format before it is accepted. These are discussed below.

HINT

In addition to the above specified files and directories, application specific directories can also be created either in the document root directory or the **/WEB-INF/classes/** directory. For example, all HTML files can be placed under **html** directory, similarly all the images can be placed under a directory named pictures and so on.

Deployment Descriptor

Servlet API **version 3.0** onwards, the **deployment descriptor** [**web.xml**] takes precedence over **annotations**.

The deployment descriptor overrides configuration information specified through the annotation mechanism.

Version 3.0 of the web deployment descriptor contains a new attribute called **metadata-complete** on **<web-app>**.

This attribute defines whether the web descriptor is complete or whether the class files of the web application should be examined for annotations that specify deployment information.

If this attribute is set to **true**, the deployment tool must ignore any servlet annotations present in the class files and use only the configuration details mentioned in the descriptor. Otherwise, if the value is **not specified** or set to **false**, the container must scan all class files of the application for annotations.

This provides a way to enable or disable scanning of the annotation and its processing during the startup of the application.

Annotations introduced in Servlet 3.0 are explained later in this chapter.

Context Path

When the application is being deployed on a particular Web server, a context path to the application has to be assigned. This means if the application is assigned to the context path as /bookshop, then a request URI referring to /bookshop/index.html will retrieve the index.html file from that document root.

Creating A Web Application Using NetBeans

Let's create a simple HTML form that accepts user's name and submits the same to the Servlet. The Servlet in response welcomes the user based on the user name it receives.

Since NetBeans is the IDE of choice throughout this book, let's use it to create a new Web Application Project called **MyFirstServlet**.

Run the NetBeans IDE and select **File → New Project...** to create a new **Web Application** project in it, as shown in diagram 7.2.1.

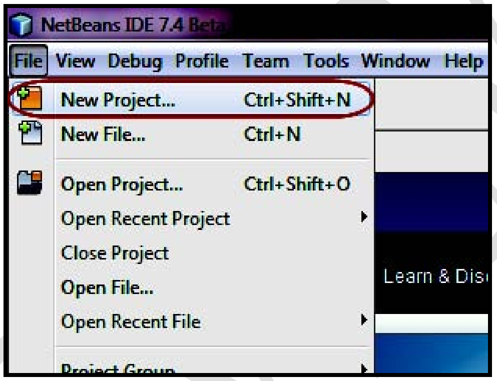


Diagram 7.2.1: File → New Project

New Project dialog box appears, as shown in diagram 7.2.2.

Select **Java Web** option available under **Categories** list and **Web Application** option available under **Projects** list, as shown in diagram 7.2.2.

This page is not part of the book preview.

This page is not part of the book preview.

Explanation:

web.xml is a deployment descriptor XML file that consists of several elements.

REMINDER

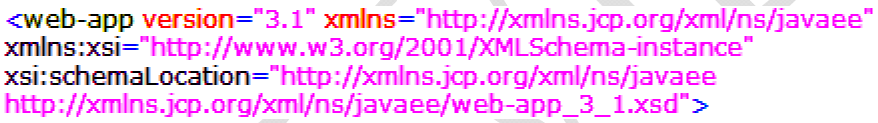
Servlet API **version 3.0** onwards, the **deployment descriptor** [**web.xml**] takes precedence over **annotations**. This means it is not mandatory to have a deployment descriptor, the annotation mechanism can be used instead.

Depending on the components in the Web application, there is no need to include all of the elements to configure and deploy the Web application. **<servlet>** is the most important element for describing the Servlets in an application.

<servlet> consists of many sub elements among which the required ones are <servlet-name> and <servlet-class>.

The <web-app> Element

All the entries are wrapped within a pair of opening and closing **<web-app>** elements:



```
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">
```

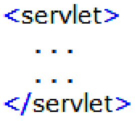
The body of <web-app> itself contains additional elements that determine how the Web application will run in the GlassFish Server.

REMINDER

In XML, properties are defined by surrounding a property name or value with opening and closing tags. The opening tag, the body [the property name or value] and the closing tag are collectively called an element. Some of the elements do not use the surrounding tags, but instead use a single tag that contains attributes called an empty tag.

The <servlet> Element

Servlets are registered and configured as a part of Web application. To register a Servlet, add several entries to the Web Application deployment descriptor.



```
<servlet>
   . . .
   . . .
</servlet>
```

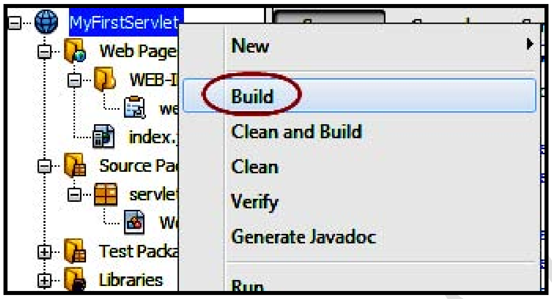
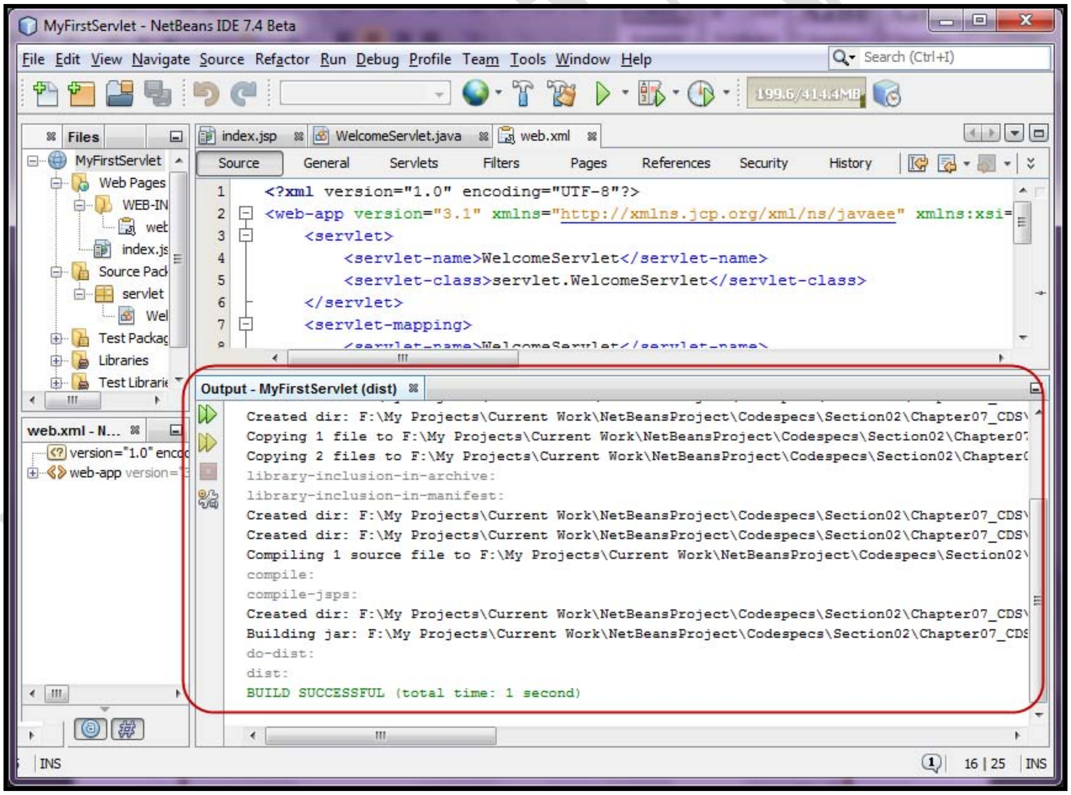


Diagram 7.6.1: Clicking Build

Once **Build** is clicked, the details about the compilation and building of the MyFirstServlet Web application appear in the **Output - MyFirstServlet (dist)** window, as shown in diagram 7.6.2.

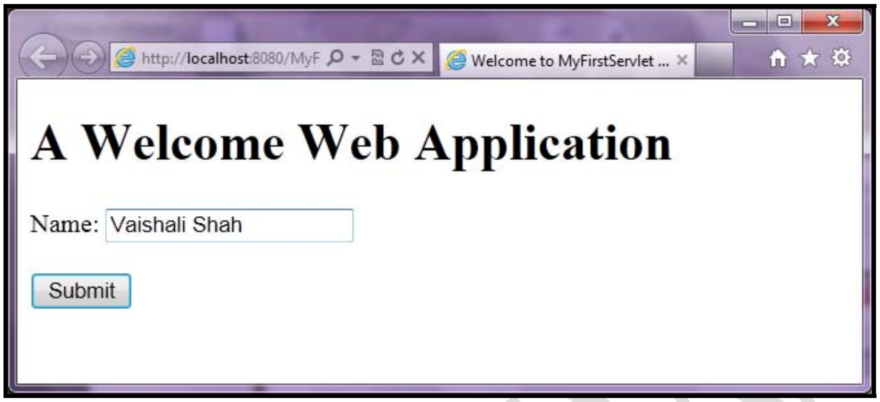


```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:xsi=
    <servlet>
        <servlet-name>WelcomeServlet</servlet-name>
        <servlet-class>servlet.WelcomeServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>WelcomeServlet</servlet-name>
```

```
Created dir: F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter07_CDS\
Copying 1 file to F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter07
Copying 2 files to F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter0
library-inclusion-in-archive:
library-inclusion-in-manifest:
Created dir: F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter07_CDS\
Created dir: F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter07_CDS\
Compiling 1 source file to F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\
compile:
compile-jsps:
Created dir: F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter07_CDS\
Building jar: F:\My Projects\Current Work\NetBeansProject\Codespecs\Section02\Chapter07_CDS
do-dist:
dist:
BUILD SUCCESSFUL (total time: 1 second)
```

Diagram 7.6.2: The details of the Build been successful

This page is not part of the book preview.

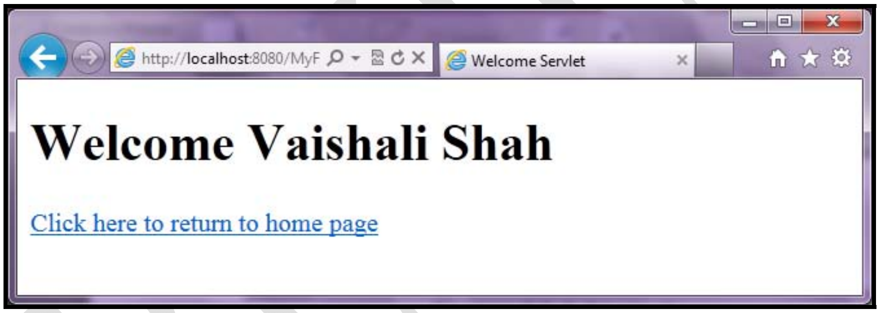


A Welcome Web Application

Name: Vaishali Shah

Diagram 7.7.4: Entering the name

Click Submit. WelcomeServlet is called, which displays the welcome message along with the name entered, as shown in diagram 7.7.5.



Welcome Vaishali Shah

Diagram 7.7.5: Welcome message displayed

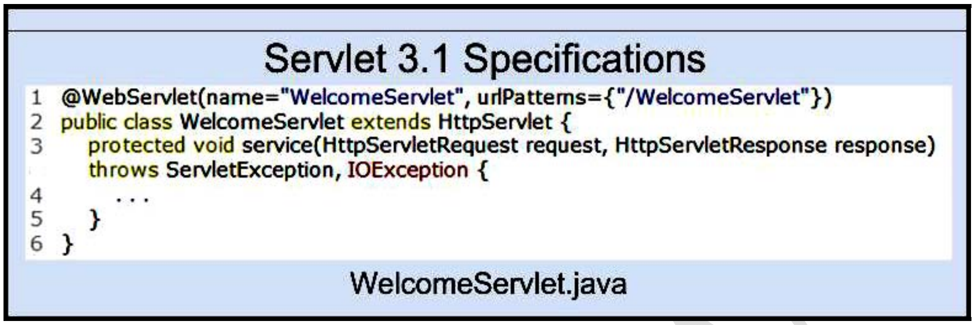
Click Click here to return to home page to return to the main page i.e. the index.jsp page.

Using Annotations Instead Of Deployment Descriptor

Servlet API 3.0 onwards, instead of making an entry in a deployment descriptor [web.xml], annotations can be used to mark a Servlet.

With the introduction of annotations, the deployment descriptor web.xml becomes **optional**.

All annotations introduced in Servlet 3.0 are available under the package javax.servlet.http.annotation.



Servlet 3.1 Specifications

```
@WebServlet(name="WelcomeServlet", urlPatterns={"/WelcomeServlet"})
public class WelcomeServlet extends HttpServlet {
   protected void service(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {
      . . .
   }
}
```

WelcomeServlet.java

Diagram 7.9: Servlet 3.1 specifications where deployment descriptor is optional

Running The Web Application

Since the Servlet i.e. WelcomeServlet.java and the JSP i.e. index.jsp files are in place, it's time to compile and build the Web application.

Follow the steps shown in the section **Compiling And Building The Web Application** to compile and build the Web application named MySecondServlet.

Follow the above steps shown in the section **Running The Web Application** to run the Web application named MySecondServlet.

The Book CDROM holds the complete application source code built using the NetBeans IDE for the following applications:

- Codespecs \ Section02 \ Chapter07_CDS \ **MyFirstServlet**
- Codespecs \ Section02 \ Chapter07_CDS \ **MySecondServlet**

The web application can be directly used by compiling, building and executing it.