Aplicación de Servlets de Java en WebSphere Liberty Profile

Aquí tienes una guía completa para ejecutar una aplicación de Servlet de Java muy simple en WebSphere Liberty Profile (WLP). Este ejemplo crea un Servlet básico que devuelve "Hello World!"y lo despliega en WLP utilizando el directorio `dropins`. Supongamos que ya has configurado WLP con la característica `servlet-4.0` habilitada, ya que esto proporciona el soporte de Servlet necesario. Esta guía evita el uso de Maven y proporciona todo el código y los pasos de despliegue necesarios.

Guía Paso a Paso

1. Crear la Estructura de la Aplicación Servlet Crea una estructura de directorios para tu aplicación Servlet manualmente. Puedes usar cualquier nombre de carpeta, pero para este ejemplo, la llamaremos `SimpleServletApp`.

- **Estructura de Directorios:**

```
SimpleServletApp/
  src/
    com/
      example/
        HelloServlet.java
  webapp/
    WEB-INF/
      web.xml
```

2. Escribir el Código del Servlet Crea un archivo llamado `HelloServlet.java` en `SimpleServletApp/src/com/example` con el siguiente contenido:

```java
package com.example;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

public class HelloServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws IOException {
```

```
        resp.setContentType("text/plain");
        resp.getWriter().write("Hello World!");
    }
}
```

- **Explicación:** Este Servlet responde a las solicitudes HTTP GET con “Hello World!”en texto plano. Estamos utilizando un simple método `doGet` sin anotaciones para obtener la máxima compatibilidad y simplicidad.

3. Crear el Descriptor de Despliegue `web.xml` Crea un archivo llamado `web.xml` en `SimpleServletApp/webapp/WEB-I` con el siguiente contenido:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app 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_4_0
         version="4.0">
    <servlet>
        <servlet-name>HelloServlet</servlet-name>
        <servlet-class>com.example.HelloServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HelloServlet</servlet-name>
        <url-pattern>/hello</url-pattern>
    </servlet-mapping>
</web-app>
```

- **Explicación:** El archivo `web.xml` mapea la clase `HelloServlet` al patrón de URL `/hello`. Esto es necesario ya que no estamos utilizando anotaciones como `@WebServlet`.

4. Compilar el Servlet Compila el archivo `HelloServlet.java` en un archivo `.class` utilizando `javac`. Necesitarás la biblioteca `javax.servlet-api` en tu classpath, que está proporcionada por WLP pero debe estar disponible durante la compilación.

- **Pasos:**
    1. Localiza el JAR de la API Servlet en tu instalación de WLP. Por ejemplo, si WLP está instalado en `/opt/ibm/wlp`, el JAR generalmente está en:

       `/opt/ibm/wlp/dev/api/spec/com.ibm.websphere.javaee.servlet.4.0_1.0.x.jar`

       El nombre del archivo puede variar según la versión de WLP.

2. Ejecuta el siguiente comando desde el directorio `SimpleServletApp`:

   ```
   javac -cp "/opt/ibm/wlp/dev/api/spec/com.ibm.websphere.javaee.servlet.4.0_1.0.x.jar" src/com/example/
   ```

3. Esto crea `HelloServlet.class` en `SimpleServletApp/src/com/example/`.

5. Empaquetar la Aplicación en un Archivo WAR Organiza los archivos compilados y crea un archivo WAR manualmente.

- **Mover la Clase Compilada:** Crea un directorio `WEB-INF/classes` y mueve los archivos de clase compilados:

  ```
  mkdir -p webapp/WEB-INF/classes/com/example
  mv src/com/example/HelloServlet.class webapp/WEB-INF/classes/com/example/
  ```

- **Crear el Archivo WAR:** Desde el directorio `SimpleServletApp`, usa el comando `jar` para empaquetar la carpeta `webapp` en un archivo WAR:

  ```
  cd webapp
  jar -cvf ../myapp.war .
  cd ..
  ```

  Esto crea `myapp.war` en el directorio `SimpleServletApp`.

6. Desplegar el Archivo WAR en WLP Despliega el archivo WAR en WLP utilizando el directorio `dropins` para el despliegue automático.

- **Localizar el Directorio `dropins`:** Encuentra el directorio `dropins` de tu servidor WLP. Si WLP está instalado en `/opt/ibm/wlp` y tu servidor se llama `myServer`, la ruta es:

  ```
  /opt/ibm/wlp/usr/servers/myServer/dropins
  ```

- **Copiar el Archivo WAR:** Mueve el archivo WAR al directorio `dropins`:

  ```
  cp myapp.war /opt/ibm/wlp/usr/servers/myServer/dropins/
  ```

- **Iniciar el Servidor (si No Está en Ejecución):** Si WLP no está en ejecución, inícialo:

  ```
  /opt/ibm/wlp/bin/server start myServer
  ```

  Si ya está en ejecución, detectará y desplegará el archivo WAR automáticamente.

- **Verificar el Despliegue:** Verifica los registros del servidor o la consola para un mensaje como:

  ```
  [AUDIT   ] CWWKT0016I: Web application available (default_host): http://localhost:9080/myapp/
  ```

  Los registros están en `/opt/ibm/wlp/usr/servers/myServer/logs/console.log`.

7. **Acceder a la Aplicación** Prueba el Servlet desplegado en un navegador.

- **Abrir tu Navegador:** Navega a:

  ```
  http://localhost:9080/myapp/hello
  ```

  - `9080` es el puerto HTTP predeterminado de WLP.
  - `/myapp` es la raíz del contexto del archivo WAR.
  - `/hello` es el patrón de URL definido en `web.xml`.

- **Resultado Esperado:** Deberías ver:

  ```
  Hello World!
  ```

  mostrado como texto plano.

Notas

- **Sin JSP Incluido:** La consulta solicitó archivos JSP, pero para mantener esto *muy simple*, me he centrado en una aplicación Servlet solo. Los JSPs requerirían archivos adicionales y complejidad, que se pueden agregar si es necesario.
- **Raíz del Contexto:** La raíz del contexto (`/myapp`) proviene del nombre del archivo WAR. Renombra `myapp.war` para cambiarlo (por ejemplo, `app.war` para `/app`).
- **Número de Puerto:** WLP predetermina `9080` para HTTP. Verifica tu `server.xml` si es diferente.
- **Versión de Java:** Usa una versión de Java compatible con WLP (por ejemplo, Java 8 o 11).
- **Solución de Problemas:** Si no funciona, verifica `/opt/ibm/wlp/usr/servers/myServer/logs/messages.log` para errores, asegúrate de que el archivo WAR esté en `dropins` y verifica que el servidor esté en ejecución.

Resumen

Esta guía muestra cómo: 1. Crear un Servlet simple (`HelloServlet.java`) y `web.xml`. 2. Compilar el Servlet sin Maven utilizando `javac`. 3. Empaquetarlo en un archivo WAR (`myapp.war`) manualmente. 4. Desplegarlo en el directorio `dropins` de WLP. 5. Acceder a “Hello World!”en `http://localhost:9080/myapp/hello`.

Esto proporciona una aplicación Servlet mínima y directa ejecutándose en WebSphere Liberty Profile sin herramientas de construcción como Maven.