

Práctica 1. RESTfull Hello World

DESARROLLO DE SERVICIOS REST JERSEY Y CLIENTES JQUERY



Guadalupe Ortiz Bellot
Departamento de Ingeniería Informática

PRÁCTICA 1. RESTFULL HELLO WORLD

Contenido

- Restful Hello World
- Anexo 1. Cómo evitar el mensaje de error al desplegar

PRÁCTICA 1. RESTFULL HELLO WORLD

Contenido

- **Restful Hello World**
 - **Ejercicio 1. Pasos a seguir**
 1. Creación de un servidor Tomcat en Eclipse
 2. Creación de un proyecto web dinámico
 3. Inclusión de las librerías Jersey
 4. Creación del paquete y de la clase
 5. Creación del archivo web.xml
 6. Despliegue del servicio en el servidor
 7. Prueba desde un cliente de navegador
- Anexo 1. Cómo evitar el mensaje de error al desplegar

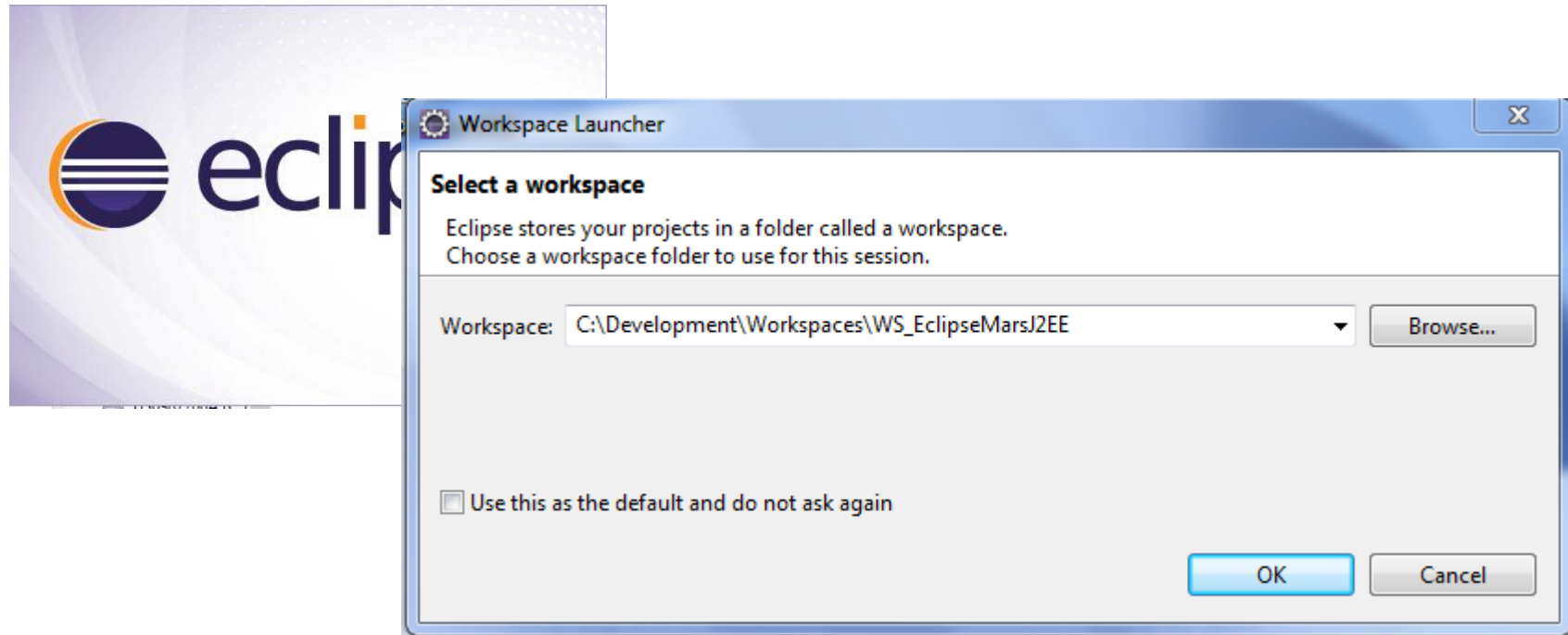
RESTFUL HELLO WORLD

Ejercicio 1. Pasos a seguir:

1. Crear un servidor Apache Tomcat en Eclipse
2. Crear un Dynamic Web Project
3. Copiar las librerías de Jersey en WebContent/Web-Inf/Lib
4. Crear un paquete dentro de JavaResources/src y la clase Hello dentro del paquete.
5. Crear el archivo web.xml en WebContent/Web-Inf
6. Desplegar el servicio en el servidor (Run as->Run on Server)
7. Probar desde el cliente REST

RESTFUL HELLO WORLD

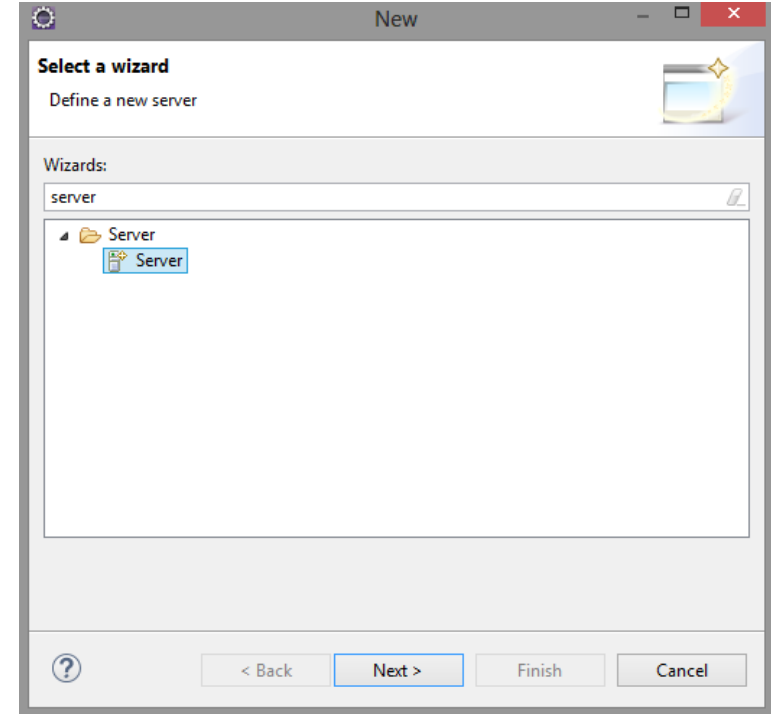
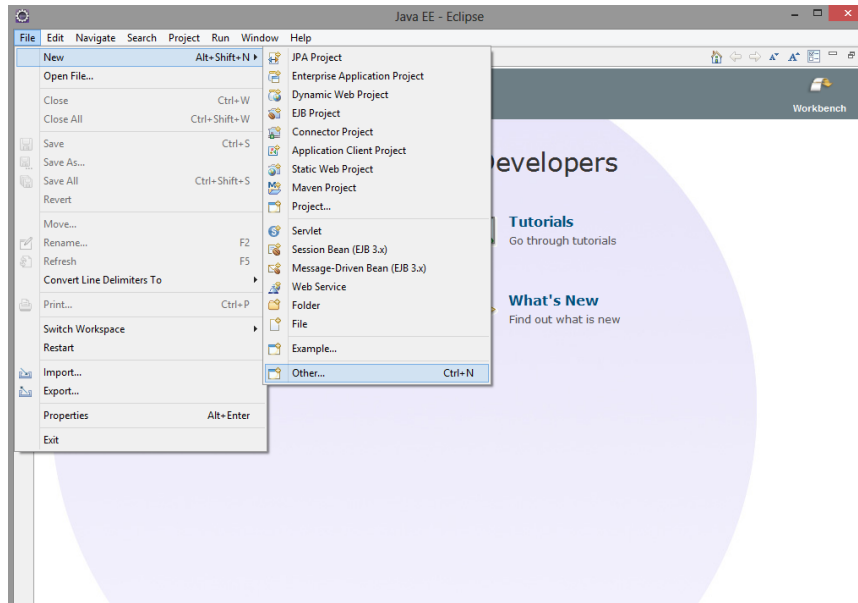
1. Creación de un servidor Tomcat en Eclipse (i)



- **RECORDATORIO:** se recomienda crear una carpeta para el workspace por ejemplo dentro de la misma carpeta raíz creada para la instalación del software (Development)
- Abrimos Eclipse y seleccionamos el workspace donde almacenaremos los proyectos que se creen.

RESTFUL HELLO WORLD

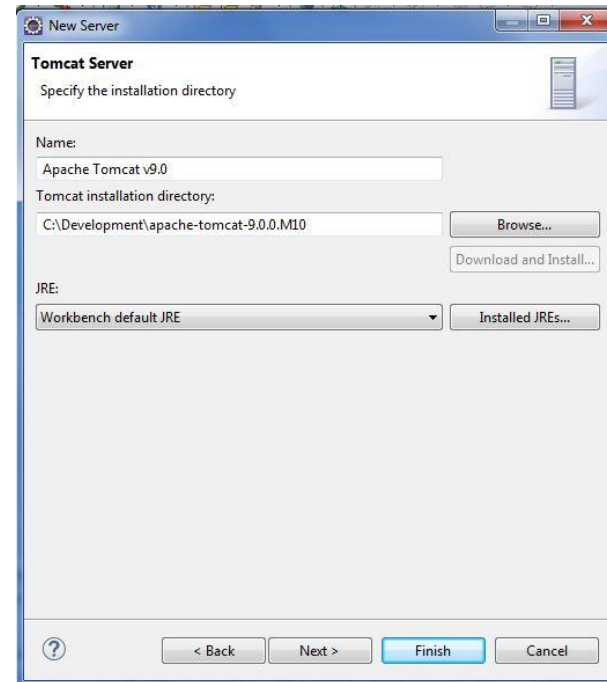
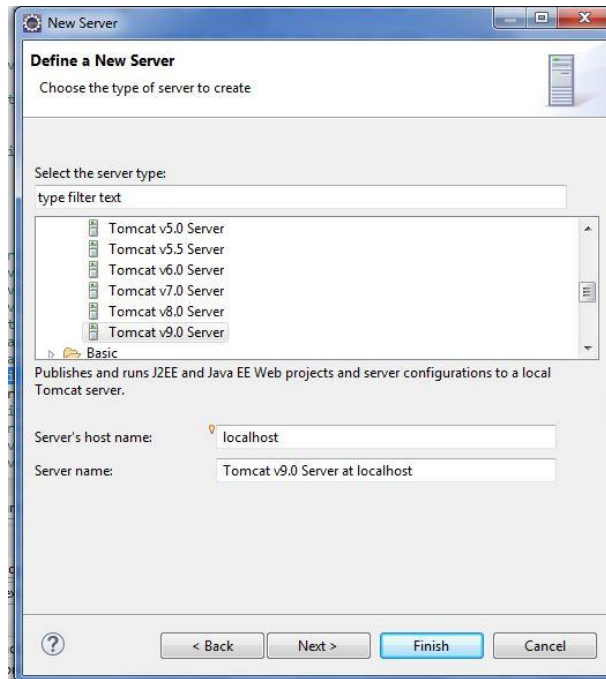
1. Creación de un servidor Tomcat en Eclipse (ii)



- Seguimos las capturas para crear el Servidor Tomcat:
File → New → Other → Server → Server

RESTFUL HELLO WORLD

1. Creación de un servidor Tomcat en Eclipse (iii)



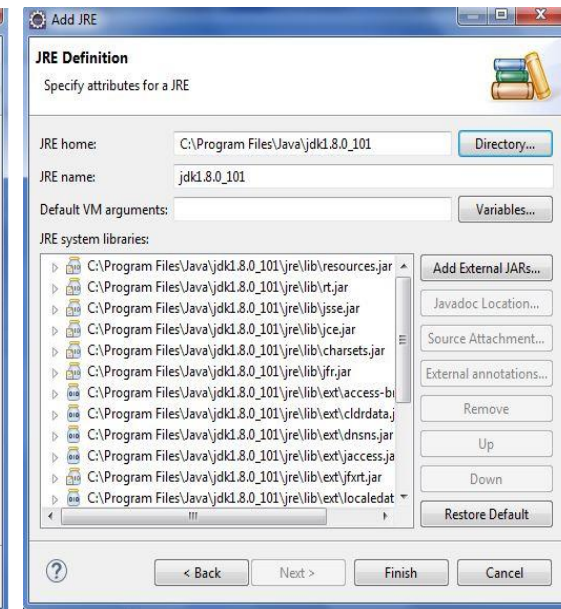
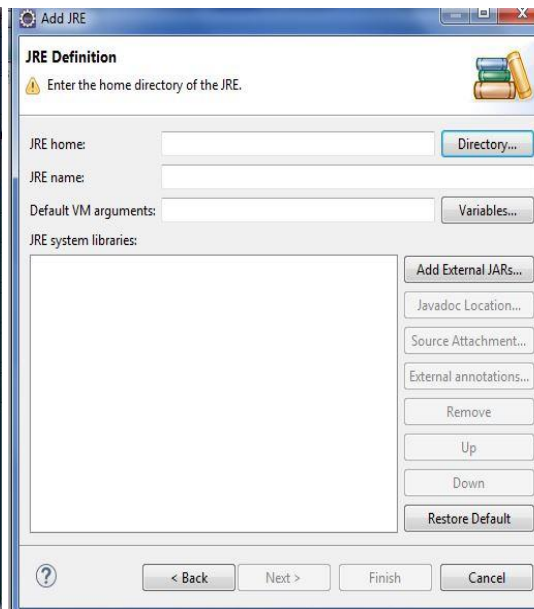
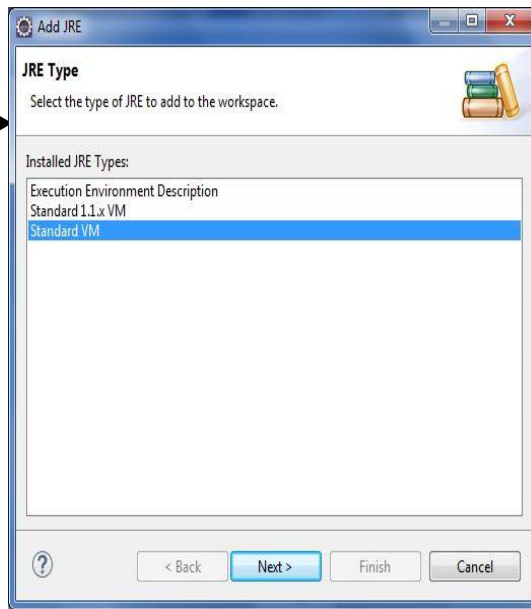
- Seleccionamos Tomcat 9 y le damos a Next
- Seleccionamos el directorio donde hemos instalado el Tomcat y el JRE que queremos usar.
- Para el JRE debemos seleccionar el JDK instalado. Habitualmente por defecto sale seleccionado el JRE, en tal caso sigue los pasos de la siguiente transparencia.

RESTFUL HELLO WORLD

1. Creación de un servidor Tomcat en Eclipse (Iv)

Cómo seleccionamos el JDK: Pinchar en Installed JRE

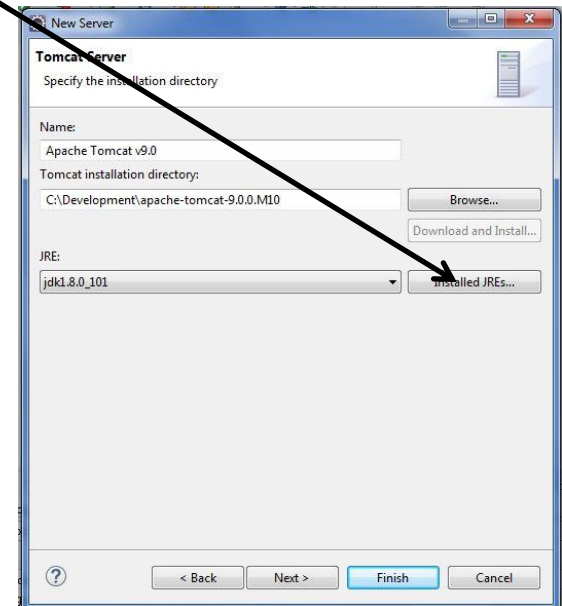
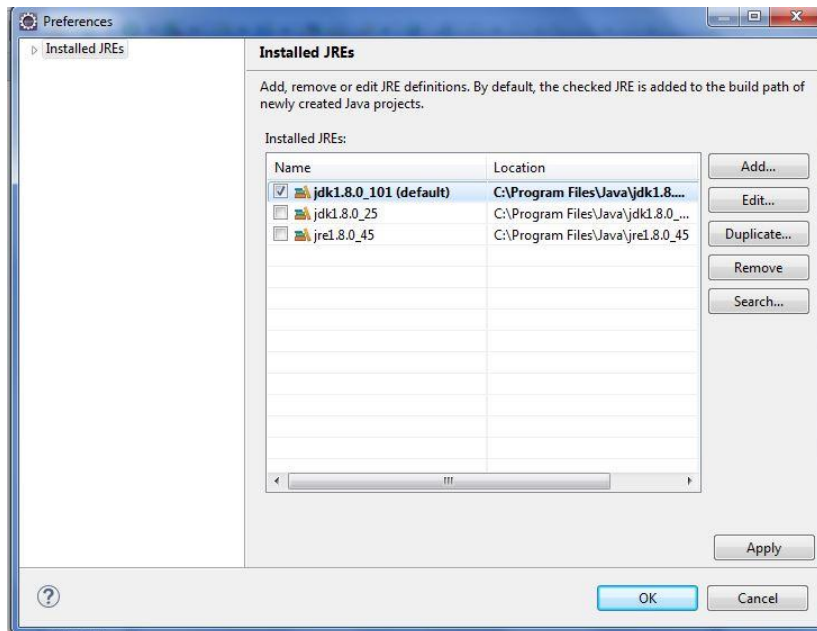
- OPCIÓN 1. Si no aparece en la lista el JDK :
 - Pinchar en Installed JRE → Add → Estándar VM → Next
 - En la pantalla que sale seleccionar el directorio donde se ha instalado el JDK y Finish
 - Ahora nos saldrá la pantalla de la opción 2 (**VER y SEGUIR LOS PASOS DE LA TRASPARENCIA ANTERIOR**: seleccionar el JDK y OK y seleccionar de nuevo el JDK en el desplegable y Finish).



RESTFUL HELLO WORLD

1. Creación de un servidor Tomcat en Eclipse (v)

- OPCIÓN 2. Si ya está aparece en la lista el JDK en Eclipse (si no, ve a la siguiente transparencia):
 - Pinchar en Installed JRE → Seleccionar el JDK y OK.
 - Aún dándole a OK es posible que en la siguiente ventana siga seleccionado el JRE, despliega y elije el JDK y Finish

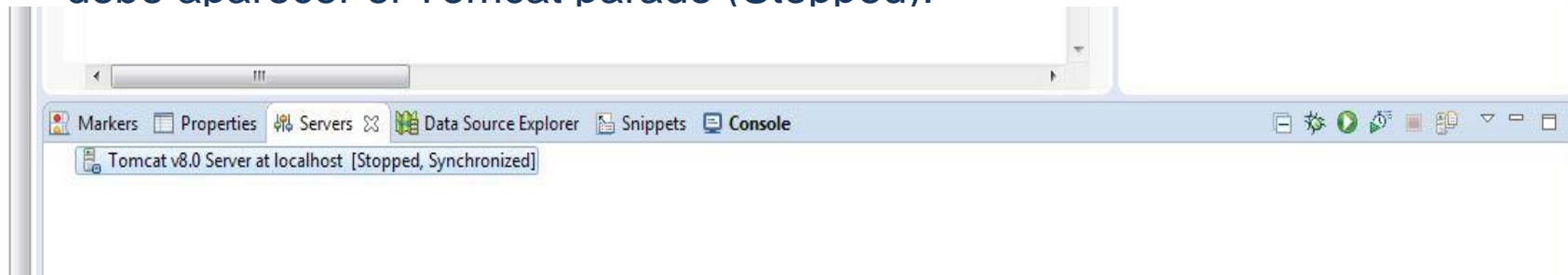


RESTFUL HELLO WORLD

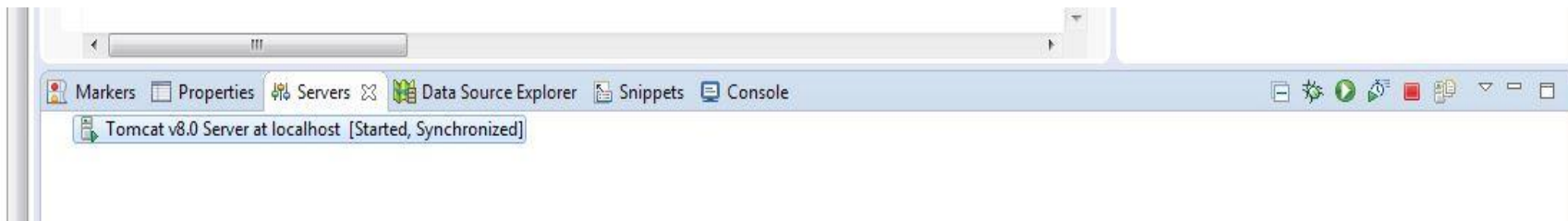
1. Creación de un servidor Tomcat en Eclipse (v)

Arrancamos el servidor Apache Tomcat:

- En la parte inferior del Eclipse, hacemos click sobre la pestaña Server y ahí debe aparecer el Tomcat parado (Stopped):



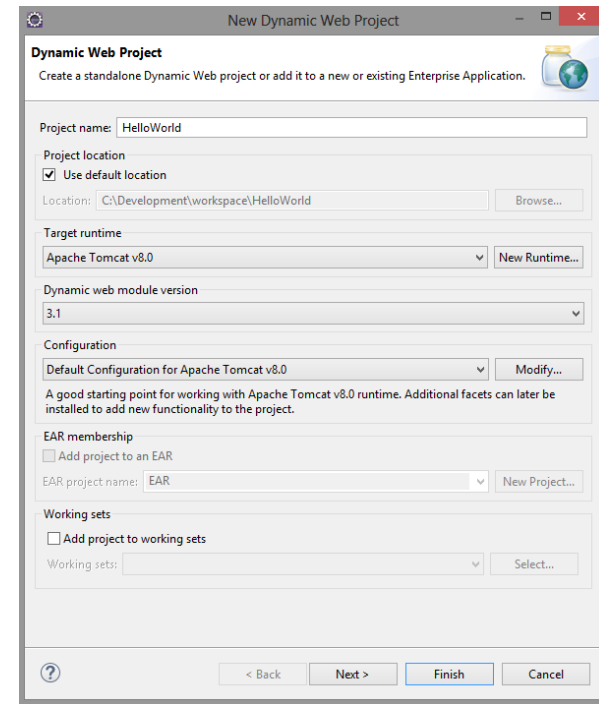
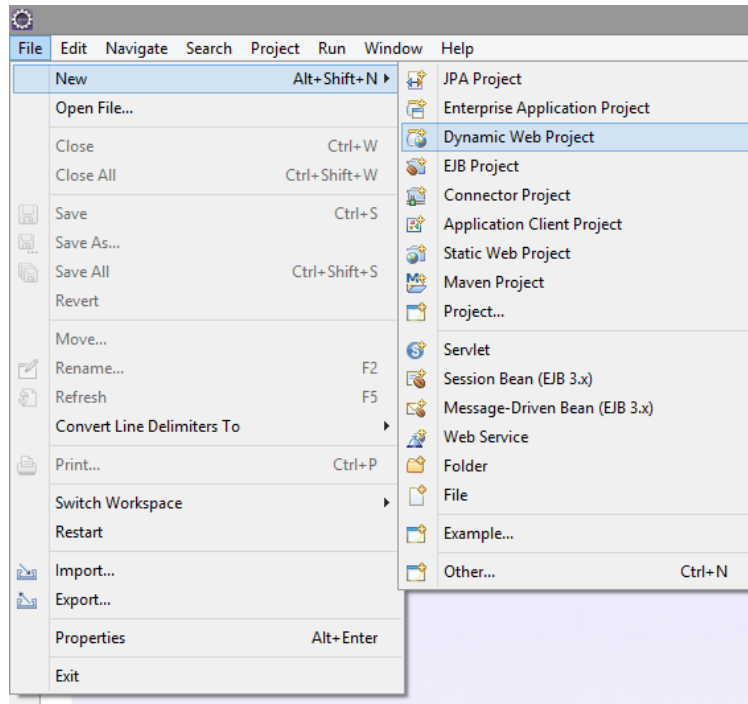
- Para arrancarlo, pinchamos sobre él con el botón derecho y seleccionamos la opción Start. Dejamos que complete el arranque, cuando haya terminado aparecerá como Started, Synchronized:



- NOTA: Para ver la vista de servidores en Eclipse, si no sale por defecto: Window → Show View → Server → Servers.

RESTFUL HELLO WORLD

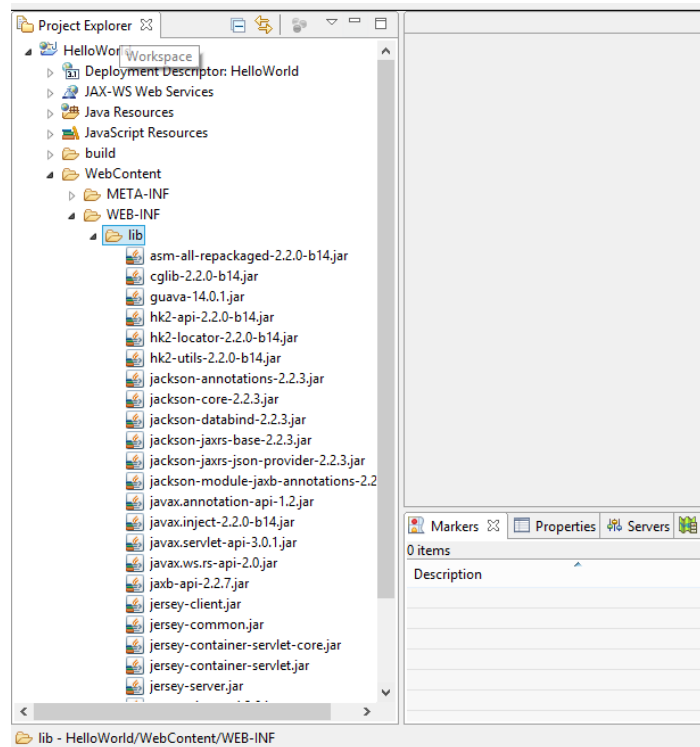
2. Creación de un proyecto web dinámico



- Creamos un Dynamic Web Project llamado HelloWorld:
 - New → Dynamic Web Project
 - Rellenar Project name y seleccionar el Tomcat que acabamos de crear en el campo Target runtime y → Finish
- **MUY IMPORTANTE: Respetar mayúsculas y minúsculas, especialmente las convenciones de Java para la inicial**

RESTFUL HELLO WORLD

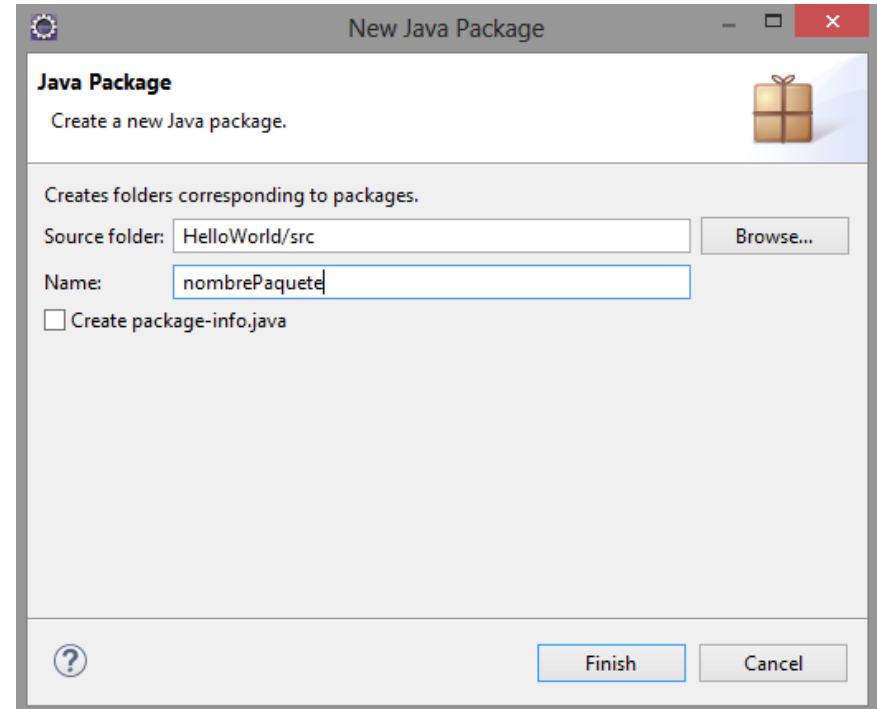
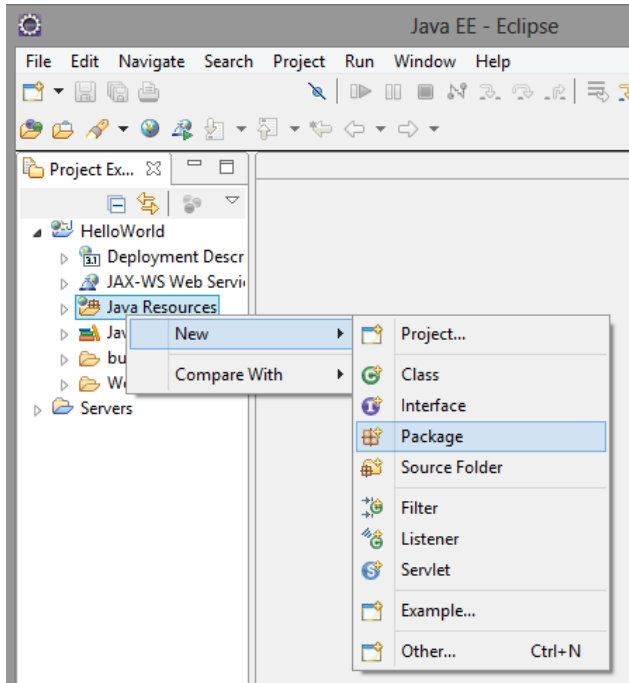
3. Inclusión de las librerías Jersey



- Copiamos las librerías de Jersey en WebContent/WEB-INF/lib (previamente las habremos descomprimido)
- Si no aparecen dentro de la carpeta después de pegarlas, refrescar el proyecto (Botón derecho sobre el proyecto → Refresh)

RESTFUL HELLO WORLD

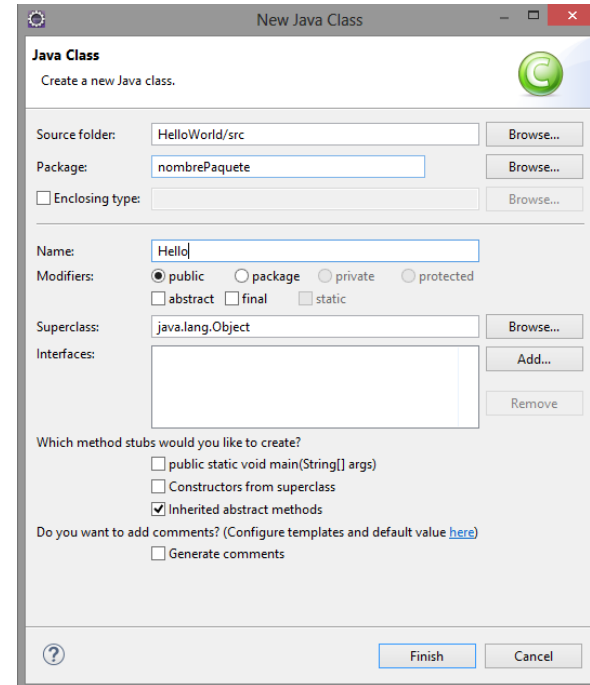
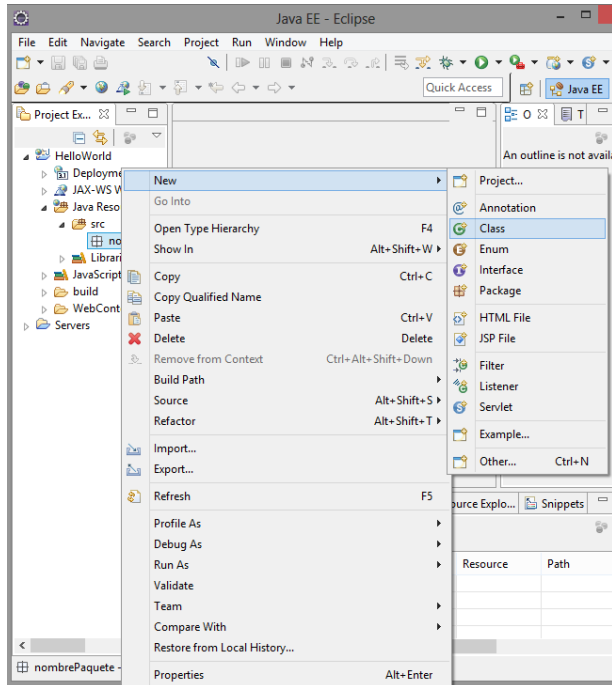
4. Creación del paquete y de la clase (i)



- Creamos un paquete **dentro de JavaResources/src** llamado **nombrePaquete**.
- **MUY IMPORTANTE: Respetar mayúsculas y minúsculas, especialmente las convenciones de Java para la inicial**

RESTFUL HELLO WORLD

4. Creación del paquete y de la clase (ii)



- Creamos la clase Hello e introducimos el código de la siguiente transparencia.
- **MUY IMPORTANTE: Respetar mayúsculas y minúsculas, especialmente las convenciones de Java para la inicial**

RESTFUL HELLO WORLD

4. Creación del paquete y de la clase (iii)

```
package nombrePaquete;
```

```
import javax.ws.rs.GET;
```

```
import javax.ws.rs.Path;
```

```
import javax.ws.rs.Produces;
```

```
import javax.ws.rs.core.MediaType;
```

```
@Path("/hello")
```

```
public class Hello {
```

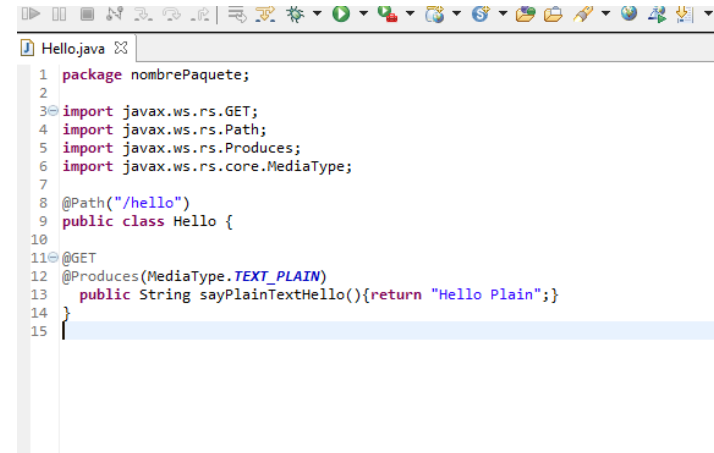
```
@GET
```

```
@Produces(MediaType.TEXT_PLAIN)
```

```
    public String sayPlainTextHello(){return "Hello Plain";}
```

```
}
```

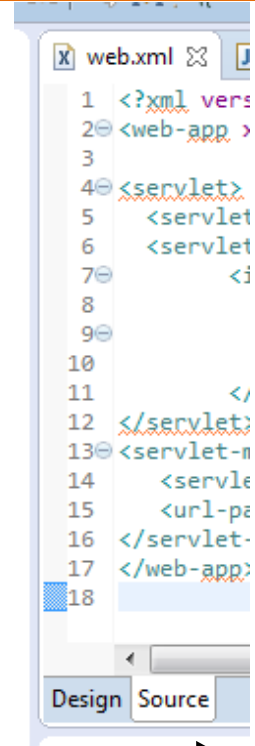
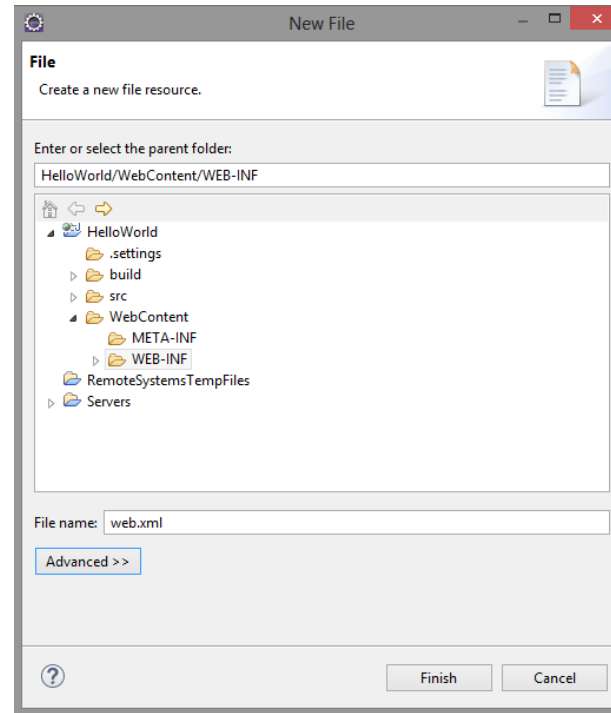
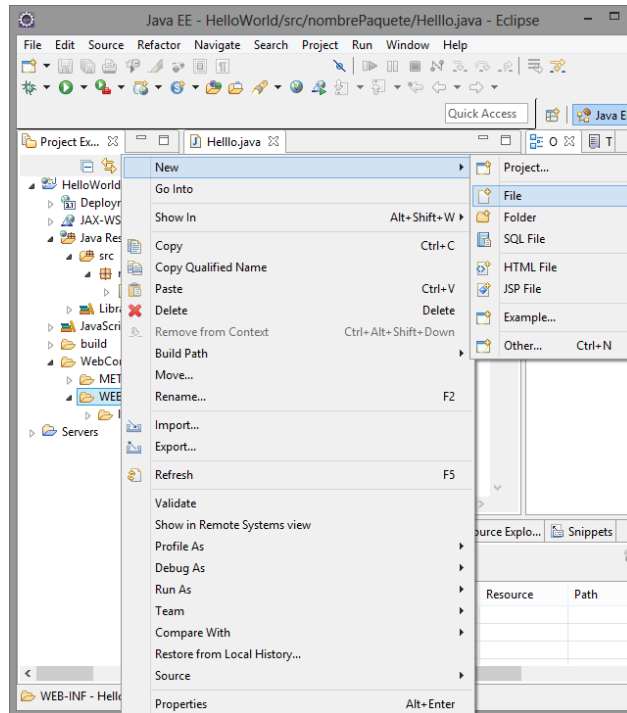
NOTA: No olvidéis salvar el archivo tras pegar o modificar el código.



```
Hello.java
1 package nombrePaquete;
2
3 import javax.ws.rs.GET;
4 import javax.ws.rs.Path;
5 import javax.ws.rs.Produces;
6 import javax.ws.rs.core.MediaType;
7
8 @Path("/hello")
9 public class Hello {
10
11 @GET
12 @Produces(MediaType.TEXT_PLAIN)
13     public String sayPlainTextHello(){return "Hello Plain";}
14 }
15
```

RESTFUL HELLO WORLD

5. Creación del archivo web.xml (i)



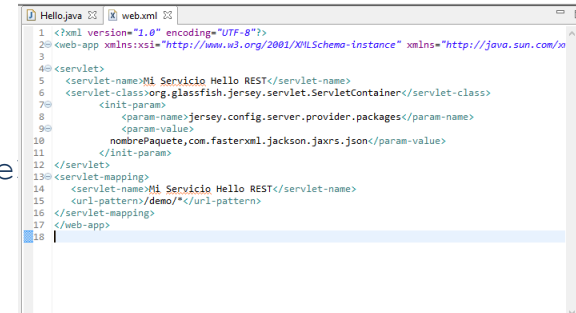
- Creamos el archivo web.xml en WebContent/WEB-INF
- Se abre automáticamente con el editor XML. Pinchar en la pestaña inferior “Source”, para verlo en modo texto
- Copiamos en él código de la siguiente transparencia y salvamos.

RESTFUL HELLO WORLD

5. Creación del archivo web.xml (ii)

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" id="WebApp_ID" version="3.0">

<servlet>
  <servlet-name>Mi Servicio Hello REST</servlet-name>
  <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  <init-param>
    <param-name>jersey.config.server.provider.packages</param-name>
    <param-value>
      nombrePaquete, com.fasterxml.jackson.jaxrs.json</param-value>
  </init-param>
</servlet>
<servlet-mapping>
  <servlet-name>Mi Servicio Hello REST</servlet-name>
  <url-pattern>/demo/*</url-pattern>
</servlet-mapping>
</web-app>
```

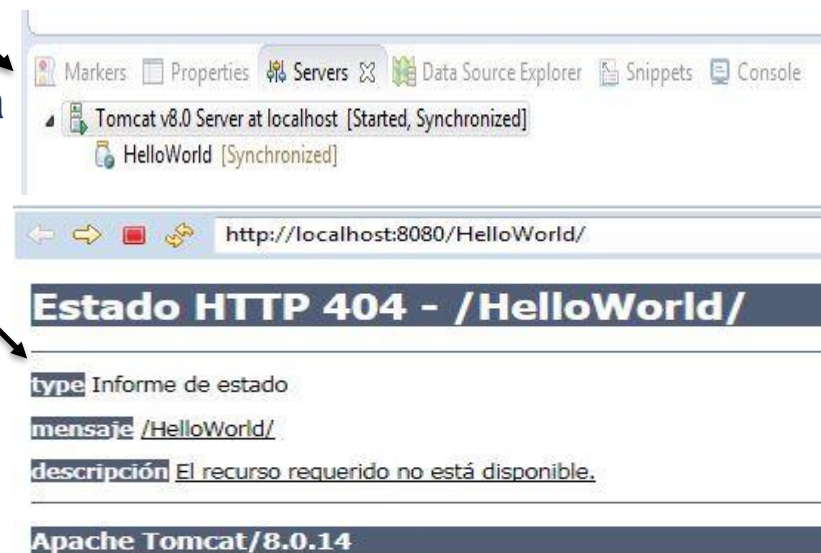
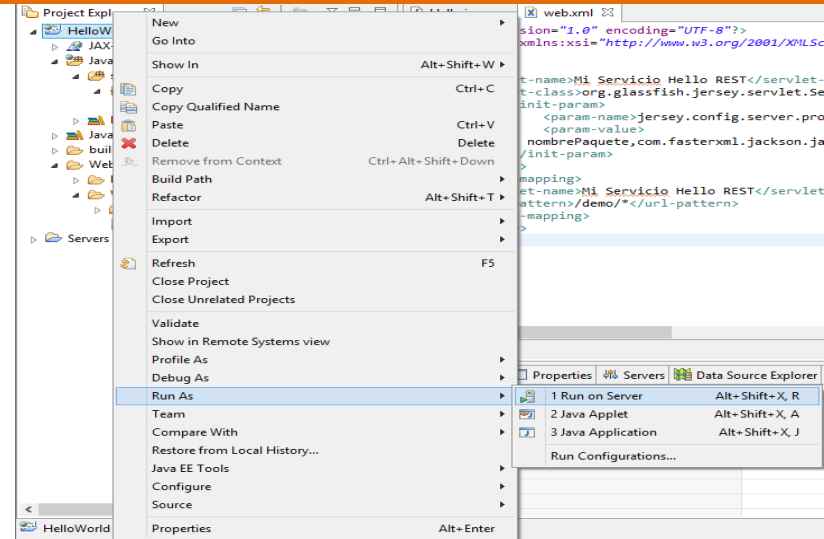
A screenshot of a code editor window titled 'Hello.java' and 'web.xml'. The editor displays the XML code for the web.xml file, with line numbers from 1 to 18. The code is identical to the one shown in the main text block. The editor has a dark background and a light-colored text color.

NOTA: dependiendo del sistema operativo las comillas puede que se copien mal. Si da error revisar las comillas.

RESTFUL HELLO WORLD

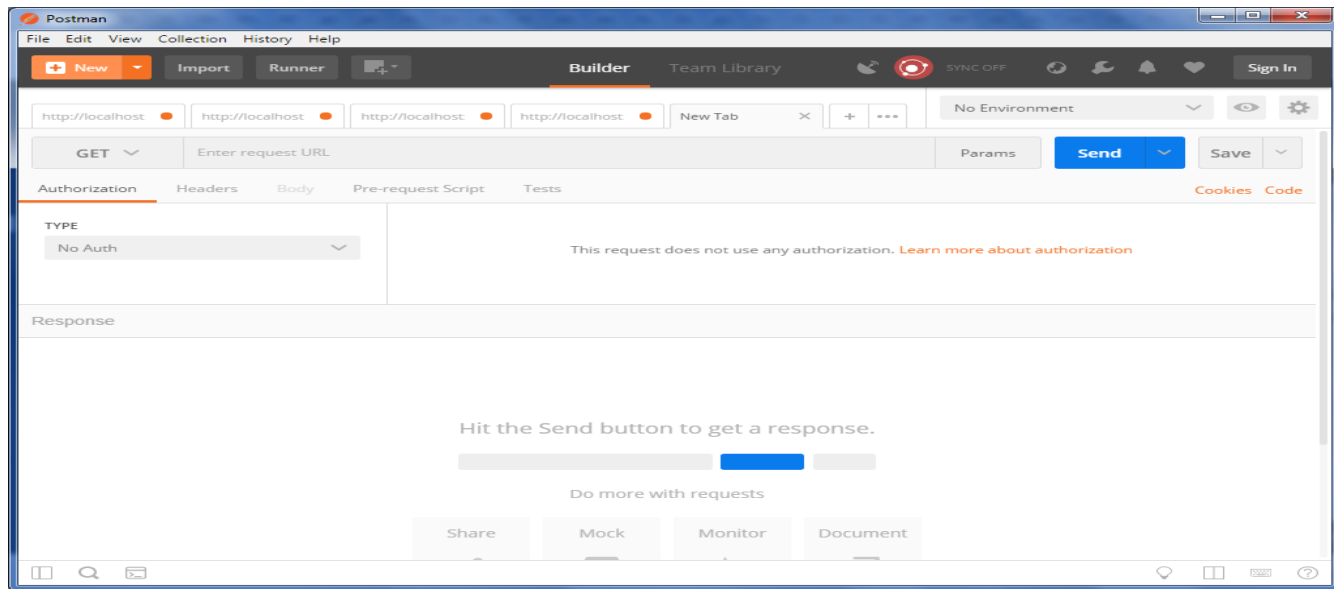
6. Despliegue del servicio en el servidor

- Desplegamos en el servidor Tomcat que hemos creado anteriormente: Botón derecho sobre el proyecto → Run as → Run on Server → Elegimos el servidor creado → Finish
- Preguntará si queremos reiniciar el servidor, le damos a OK
- Si todo ha ido bien al desplegar el contenido del servidor, saldrá debajo el servicio sincronizado.
- Si no habría que pinchar en la pestaña de la consola para ver el error.
- No hay que preocuparse si en el navegador de Eclipse sale Error 404.
- NOTA: Para ver la ventana de consola, problemas o log de errores: Window → Show View → General → la opción correspondiente



RESTFUL HELLO WORLD

7. Probar desde el cliente REST



- En la request url de Postman ponemos la siguiente URI: <http://localhost:8080/HelloWorld/demo/hello> , seleccionamos GET y pulsamos sobre Send. Nos debería dar el resultado siguiente:
- La URI se ha formado con la dirección del servidor Tomcat (localhost:8080), el nombre del proyecto (HelloWorld), el nombre de la raíz que establecimos en el archivo web.xml (demo) y el nombre del path indicado en la clase (hello).

Anexo 1

Cómo evitar el mensaje de error al desplegar

- Debes tener en cuenta que este error no es relevante ya que trata de cargar una web “en parte de cliente”, a pesar de tratarse de un servicio.
- En cualquier caso, si no quieres que aparezca este error:
 1. Dentro de la carpeta Webcontent del Proyecto añade un archivo Index.html con el texto que quieres que aparezca en la web.
 2. Dentro del web.xml antes de cerrar la etiqueta webapp añade lo siguiente:

```
<welcome-file-list>  
  <welcome-file>Index.html</welcome-file>  
</welcome-file-list>
```