


CREACIÓN DE APLICACIONES WEB CON ECLIPSE

1. Partimos que ya tenemos instalado Java y Eclipse, por lo que lo primero que debemos hacer es instalar un servidor de aplicaciones, en nuestro caso Tomcat. Para ello debemos ir a la siguiente página web y descargar el paquete correspondiente, según nuestro sistema operativo (<https://tomcat.apache.org/download-90.cgi>)
2. Una vez que lo descarguemos, tenemos que descomprimirlo en una carpeta.
3. Nos vamos a Eclipse y en **Help** elegimos **Eclipse Marketplace**. Buscamos **Tomcat** Elegimos



Eclipse Tomcat Plugin 9.1.4

The Eclipse Tomcat Plugin provides simple integrati applications. This project is the successor of... [mor](#)

by [Markus Keunecke](#), Apache 2.0
[tomcatplugin](#) [tomcat](#) [java](#) [JDT](#) [plugin](#)

★ 145

🔄 Installs: **217K** (5.764 last month)

Nos pedirá que aceptemos las condiciones y el certificado. Cuando termine la instalación nos informará que debemos reiniciar Eclipse. Reiniciamos.

4. También tendremos que instalar el plugin **Eclipse Java EE Developer**. Lo buscamos en el eclipse Marketplace y lo instalamos. Tendremos que reiniciar eclipse cuando termine.



Eclipse Java EE Developer Tools 3.15

Enables Enterprise Java Bean, Java Enterprise Application, Fragments, and Connector, Java Web Application, JavaServer Faces (JSF), Java Server Pages (JSP), Java... [more info](#)

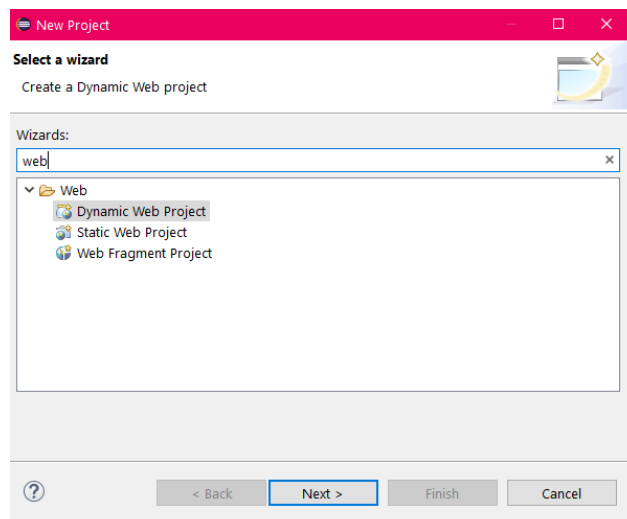
by [The Eclipse Foundation](#), EPL
[xml](#) [html](#) [CSS](#) [js](#) [jsp](#) ...

★ 590

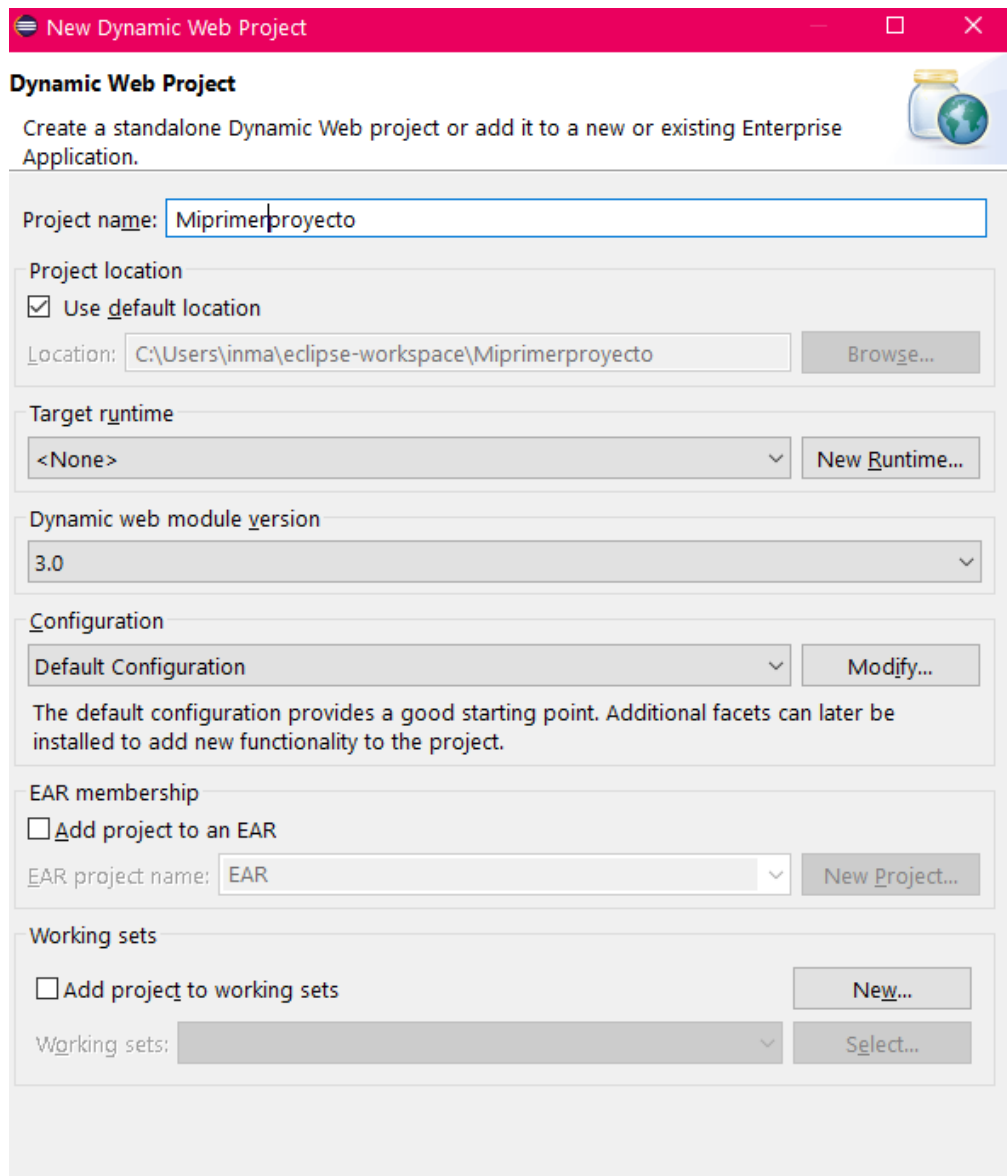
🔄 Installs: **289K** (12.840 last month)

Install

5. Ya podemos crear nuestro primer proyecto Web. Elegimos Nuevo proyecto y buscamos web

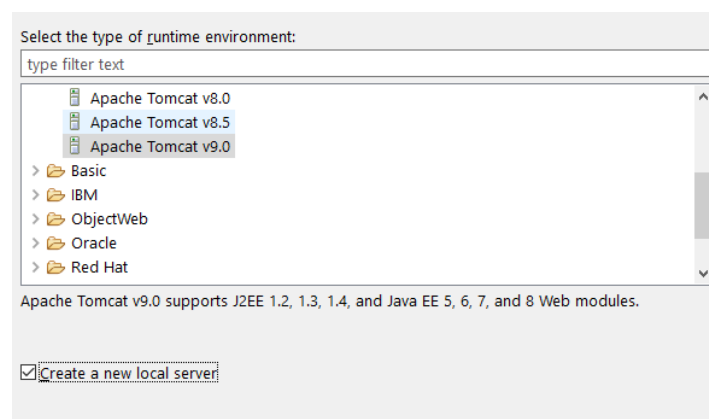


6. Nos pedirá el nombre del proyecto y el servidor de aplicaciones que vamos a utilizar. En **Target runtime** debemos crear uno nuevo, por lo que debemos pulsar **New Runtime**



The screenshot shows the 'New Dynamic Web Project' dialog box. The 'Project name' field is filled with 'Miprimerproyecto'. The 'Project location' section has 'Use default location' checked, and the 'Location' field shows 'C:\Users\inma\eclipse-workspace\Miprimerproyecto'. The 'Target runtime' dropdown is set to '<None>', and the 'New Runtime...' button is visible. The 'Dynamic web module version' is set to '3.0'. The 'Configuration' section shows 'Default Configuration' selected. The 'EAR membership' section has 'Add project to an EAR' unchecked. The 'Working sets' section has 'Add project to working sets' unchecked. The 'New Runtime...' button is highlighted.

Elegimos la versión 9.0 de Tomcat y pulsamos que cree un servidor local



The screenshot shows the 'Select the type of runtime environment' dialog. The 'type filter text' field is empty. The list of runtime environments includes 'Apache Tomcat v8.0', 'Apache Tomcat v8.5', and 'Apache Tomcat v9.0'. The 'Apache Tomcat v9.0' option is selected. Below the list, it states 'Apache Tomcat v9.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5, 6, 7, and 8 Web modules.' The 'Create a new local server' checkbox is checked.

A continuación nos pedirá la carpeta donde hemos descomprimido el Tomcat que nos hemos bajado.

Le damos terminar y nos dirá que requiere cambiar de perspectiva, aceptamos.

7. A continuación vamos a “maverizar” nuestro proyecto. Maven se ha a encargar de bajarse todas las librerías que necesitemos en nuestro proyecto. Para ello marcamos el nombre del proyecto y pulsamos el botón derecho del ratón. Elegimos la opción **Configure** y dentro de Configure seleccionamos **Convert to Maven Project**. Nos dirá que va a crear un fichero (pom.xml), le damos a aceptar. Ahora en el menú desplegable del proyecto nos debe salir la opción Maven, le damos a actualizar el proyecto y nos desaparecerá el error que tenía el proyecto.
8. El proyecto tendrá por defecto una carpeta llamada **Java Resources**. En ella es donde debemos crear nuestros paquetes y nuestras clases de Java.
9. Tenemos que crear “la vistas” que es la parte gráfica de nuestra aplicación. Las vistas son simples archivos con extensión .jsp, en las cuales usamos JSTL (Java Server Pages Tag Library) que no son más que etiquetas y que nos permiten manipular o embeber código java dentro de una página JSP. Para poder hacer uso de estas etiquetas, se debe incluir una declaración en el inicio de la página JSP donde vayamos a usarlas. Esta parte se tiene que crear en WebContent. Para ello debemos marcar nuestro proyecto y seleccionamos nuevo JSP File (por defecto lo crea en WebContent). Lo llamaremos index.jsp (nos debe crear lo siguiente)

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>

</body>

</html>
```

En las primeras líneas le decimos que vamos a utilizar Java. Si queremos escribir Java tendremos que poner el carácter <% Código java/>. Lo único que cambia es que para escribir pondremos out.print(cadena);

Por ejemplo, con este será nuestra primera aplicación “hola mundo”

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<% out.print("hola mundo");%>
</body>

</html>
```

Le damos a ejecutar y nos preguntará si queremos lanzar el servidor, aceptamos y ya lo tenemos.