



Universidad  
Andrés Bello®  
Conectar · Innovar · Liderar

# DESARROLLO DE APLICACIONES WEB DINÁMICAS JAVA

## El entorno JEE y sus componentes

### ¿Qué es la tecnología JEE?

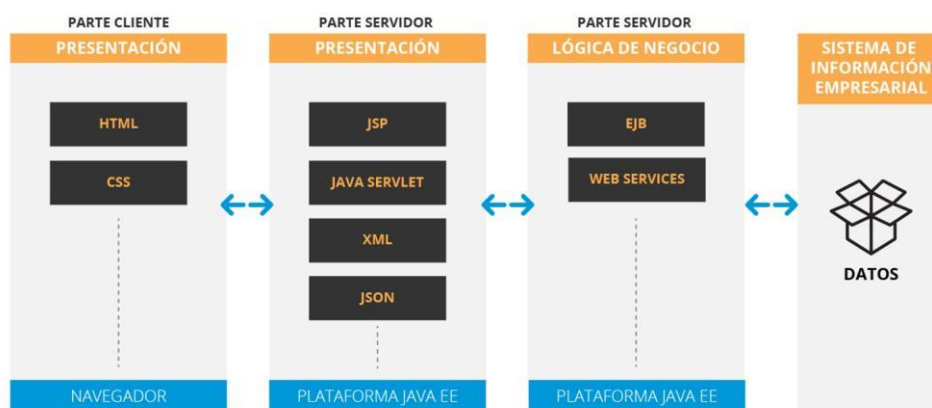
Java Enterprise Edition, Java EE en adelante, es un conjunto de estándares de tecnologías dedicadas al desarrollo de Java del lado del servidor. La plataforma Java EE consta de un conjunto de servicios, API y protocolos que proporcionan la funcionalidad necesaria para desarrollar aplicaciones basadas en web de varios niveles. Es decir, desarrollaremos aplicaciones empresariales distribuidas, con arquitecturas multicapa, escritas en Java y que se ejecutan en un servidor de aplicaciones.

Java es un lenguaje orientado a objetos de alto nivel. Sin duda, es uno de grandes lenguajes de programación de los que se disponen, con una alta demanda en el mercado laboral.

Necesitamos un buen nivel de Java que adquiriremos bajo la plataforma Java SE (Estándar Edition): Sintaxis, características, objetos, etc. Java EE incluye muchos componentes de Java Standard Edition (Java SE)

### ¿Qué es una Arquitectura multicapa?

Una arquitectura multicapa es un conjunto ordenado de subsistemas, cada uno de los cuales está constituido en términos de los que tiene por debajo y proporciona la base de la implementación de aquellos que están por encima de él.



## Servlets y Vistas JSP

### ¿Qué es un Servlet?

Los Servlets son módulos escritos en Java que se utilizan en un servidor, que puede ser o no ser servidor web, para extender sus capacidades de respuesta a los clientes al utilizar las potencialidades de Java. Los Servlets son para los servidores lo que los applets para los navegadores, aunque los Servlets no tienen una interfaz gráfica.

Los Servlets pueden ser incluidos en servidores que soporten la API de Servlet. La API no realiza suposiciones sobre el entorno que se utiliza, como tipo de servidor o plataforma, ni del protocolo a utilizar, aunque existe una API especial para HTTP.

Los Servlets son un reemplazo efectivo para los CGI (Common Gateway Interface) en los servidores que los soporten ya que proporcionan una forma de generar documentos dinámicos utilizando las ventajas de la programación en Java como conexión a alguna base de datos, manejo de peticiones concurrentes, programación distribuida, etc. Por ejemplo, un Servlet podría ser responsable de procesar los datos desde un formulario enHTML como registrar la transacción, actualizar una base de datos, contactar algún sistemaremoto y retornar un documento dinámico o redirigir a otro servlet u alguna otra cosa.

### ¿Cómo es un Servlet?

Un pequeño Servlet de ejemplo es el siguiente:

```
public class SimpleServlet extends HttpServlet {  
    // Maneja el método GET de HTTP para  
    // construir una sencilla página Web.  
  
    public void doGet (HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
        PrintWriter out;  
        String title = "Simple Servlet Output";  
  
        // primero selecciona el tipo de contenidos y otros campos de cabecera de la respuesta  
        response.setContentType("text/html");  
        // Luego escribe los datos de la respuesta  
        out = response.getWriter();  
        out.println("<HTML><HEAD><TITLE>");  
        out.println(title);  
        out.println("</TITLE></HEAD><BODY>");  
        out.println("<H1>" + title + "</H1>");  
        out.println("<P>This is output from SimpleServlet.");  
        out.println("</BODY></HTML>");  
        out.close();  
    }  
}
```

Este Servlet puede ser puesto en un servidor web ya que utiliza el protocolo HTTP para comunicarse. Primero es necesario señalar que el Servlet será del tipo HTTP por lo que se extiende de la clase HttpServlet.

Al extender de esta clase es necesario definir el método `doGet` para responder la petición. Este método recibe los parámetros dados por el cliente a través de la clase `HttpServletRequest` y encapsula la respuesta que se le dará al cliente a través de la clase `HttpServletResponse`. El Servlet puede retornar al cliente cualquier tipo de información, desde texto plano hasta un ejecutable, por lo que es necesario señalar inicialmente qué tipo de respuesta se dará a través del método `setContentType`. Luego se obtiene el objeto para poder escribir texto al cliente a través del método `getWriter` con el cual se puede retornar una página web llamado sucesivamente el método `println` hasta terminar con `close`.



## Propiedades

- **Manejo de Sesiones:** Se puede hacer seguimiento de usuarios a través de distintos servlets a través de la creación de sesiones.
- **Utilización de Cookies:** Las cookies son pequeños datos en texto plano que pueden ser guardados en el cliente. La API de servlets permite un manejo fácil y limpio de ellas.
- **Multi-thread:** Los servlets soportan el acceso concurrente de los clientes, aunque hay que tener especial cuidado con las variables compartidas a menos que se utilice la interfaz `SingleThreadModel`.
- **Programación en Java:** Se obtienen las características de multiplataforma o acceso a APIs como JDBC, RMI, etc.

## ¿Qué es una vista JSP?

### JSP (Java Server Page)

Una de las tecnologías más ampliamente utilizadas para el desarrollo de aplicaciones Web bajo ambiente Java es JSP (Java Server Pages). Esta herramienta permite crear una serie de plantillas HTML que incluyen código incrustado (llamados snippets) mediante marcadores especiales.

El uso de JSP simplifica la programación de servlets pues no es necesario compilar código ya que esto se realiza de forma automática. Además, debido a que la plantilla se escribe directamente en HTML es factible utilizar editores y diseñadores de páginas Web para programar la aplicación.

Una página JSP es una página (X)HTML que incorpora ciertos elementos dinámicos: etiquetas especiales y pequeños fragmentos de código.

- El código HTML aparece a la salida sin modificaciones.
- Los elementos dinámicos se evalúan o ejecutan en el servidor en el momento de construcción de la respuesta.
- Aunque no es estrictamente obligatorio, una página JSP se suele transformar en el código fuente de un servlet, que después se compila y ejecuta.

## Ejemplo JSP

```
<%@ page language='java' contentType='text/html; charset=iso-8859-1'%>
<%@ page import='java.util.Date' %>
<html>
  <head>
    <title>Hola Mundo</title>
  </head>
  <body>
    <p>Hola, esto es una página JSP.</p>
    <p>La hora del servidor es <%= new Date() %></p>
  </body>
</html>
```

## JSP ejemplo: documento recibido por el cliente

```
<html>
  <head>
    <title>Hola Mundo</title>
  </head>
  <body>
    <p>Hola, esto es una página JSP.</p>
    <p>La hora del servidor es Wed Nov 06 13:25:34 CET 2002</p>
  </body>
</html>
```

## Directivas JSP: page

Todas las páginas JSP deberían incluirla. Atributos habituales:

- **language:** lenguaje de programación (java por defecto).
- **contentType:** tipo de contenido de la página (text/html por defecto).
- **isErrorPage:** indica si es una página de error (false por defecto).
- **errorPage:** página a la que dirigirse si ocurre una excepción procesando esta página.

```
<%@ page language='java' contentType='text/html'
        isErrorPage='false' errorPage='error.jsp' %>
```

## Java Beans

- Un Java Bean es una clase Java que cumple el siguiente convenio:
  - Contiene propiedades (normalmente atributos de instancia privados).
  - El acceso a las propiedades se realiza mediante métodos de acceso get, set e is.
  - Tiene siempre un constructor sin argumentos (aunque podría tener más constructores)

## Java Beans: ejemplo

```
public class UserInfoBean implements java.io.Serializable
{
```

```
private String firstName;
private boolean registered;

public String getFirstName() {
    return firstName;
}
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
public boolean isRegistered() {
    return registered;
}
public void setRegistered(boolean registered) {
    this.registered = registered;
}
}
```

## Java Beans en JSP

- JSP proporciona instrucciones especiales para trabajar más cómodamente con Java Beans.
- La acción `jsp:useBean`:
  - Si el bean aún no existe en el contexto:
  - Declara, crea e inicializa el bean.
  - Crea una referencia al bean con el nombre dado por id.
  - Si el bean ya existe en el contexto:
  - Obtiene una referencia al mismo con el nombre dado por id.

```
<jsp:useBean id='user' class='foo.UserInfoBean'
scope='session'>
    <jsp:setProperty name='user' property='name'
value='Pepe' />
</jsp:useBean>
```

## ¿Qué es un servidor de aplicaciones?

### Servidores de aplicaciones

Un servidor de aplicaciones es un programa de servidor en un equipo en una red distribuida que proporciona la lógica de negocio para un programa de aplicación.

El servidor de aplicaciones se ve frecuentemente como parte de una aplicación de tres niveles, que consta de un servidor gráfico de interfaz de usuario (GUI), un servidor de aplicaciones (lógica empresarial) y un servidor de bases de datos y transacciones.

}De manera más descriptiva, se puede visualizar como la división de una aplicación en:

1. Una interfaz gráfica de usuario de primer nivel, de front-end, basada en el navegador web, normalmente en un equipo de cómputo personal o una estación de trabajo.
2. Una aplicación de lógica de negocio de nivel medio o conjunto de aplicaciones, posiblemente en una red de área local o un servidor de intranet.
3. Un servidor de back-end, base de datos y transacciones de tercer nivel, a veces en un mainframe o servidor grande.

El servidor de aplicaciones, en el contexto de la especificación de Java Enterprise Edition (EE), se encarga de gestionar y ejecutar todos los componentes de la aplicación, escritos en lenguaje Java, como Servlets, Vistas JSP (Java Server Pages), WebSockets o EJB's Enterprise JavaBeans)

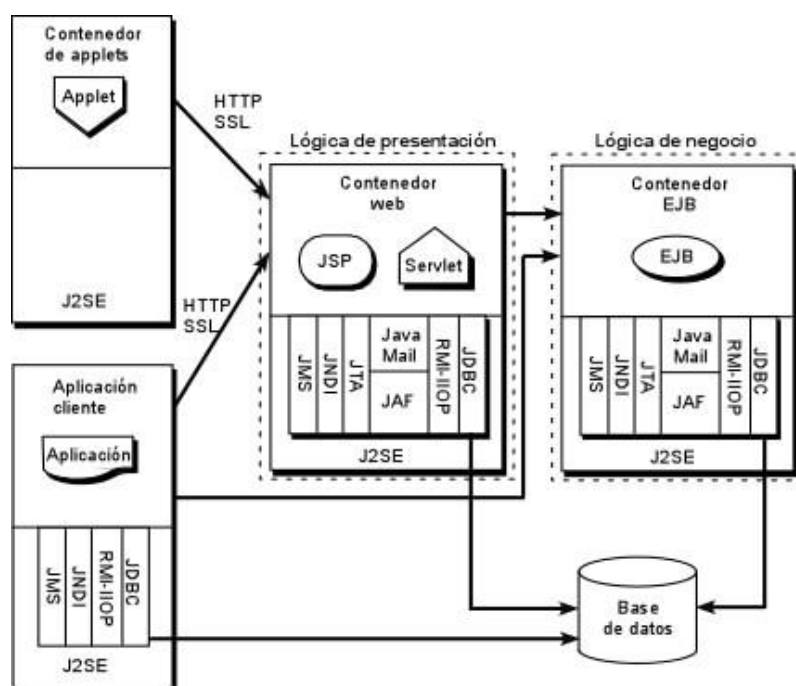


Figura 1.- Arquitectura J2EE.

Definimos a continuación algunos de los conceptos que aparecen en la figura 1:

- **Ciente web (contenedor de applets):** Es usualmente un navegador e interactúa con el contenedor web haciendo uso de HTTP. Recibe páginas HTML o XML y puede ejecutar applets y código JavaScript.
- **Aplicación cliente:** Son clientes que no se ejecutan dentro de un navegador y pueden utilizar cualquier tecnología para comunicarse con el contenedor web o directamente con la base de datos.
- **Contenedor web:** Es lo que comúnmente denominamos servidor web. Es la parte *visible* del servidor de aplicaciones. Utiliza los protocolos HTTP y SSL (seguro) para comunicarse.
- **Servidor de aplicaciones:** Proporciona servicios que soportan la ejecución y disponibilidad de



las aplicaciones desplegadas. Es el corazón de un gran sistema distribuido.

Frente a la tradicional estructura en dos capas de un servidor web (ver Figura 2) un servidor de aplicaciones proporciona una estructura en tres capas que permite estructurar nuestro sistema de forma más eficiente. Un concepto que debe quedar claro desde el principio es que no todas las aplicaciones de empresa necesitan un servidor de aplicaciones para funcionar. Una pequeña aplicación que acceda a una base de datos no muy compleja y que no sea distribuida probablemente no necesitará un servidor de aplicaciones, tan solo con un servidor web (usando servlets y jsp) sea suficiente.

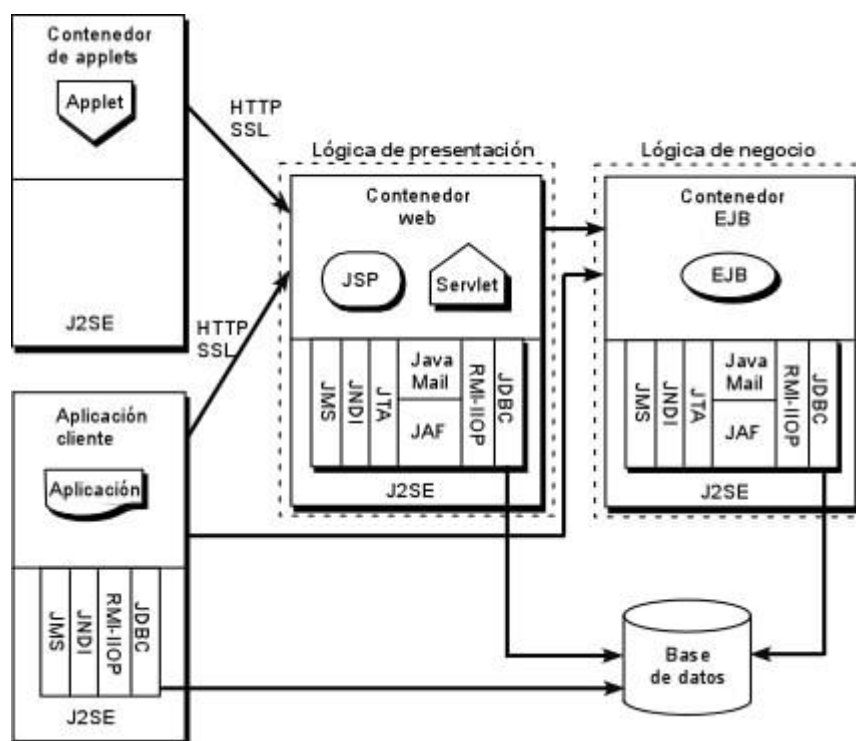


Figura 2.- Arquitectura en dos capas frente a tres capas utilizando el servidor de aplicaciones

Como hemos comentado, un servidor de aplicaciones es una implementación de la especificación J2EE. Existen diversas implementaciones, cada una con sus propias características que la pueden hacer más atractiva en el desarrollo de un determinado sistema. Algunas de las implementaciones más utilizadas son las siguientes:

- BEA WebLogic
- IBM WebSphere
- Sun-Netscape IPlanet
- Sun One
- Oracle IAS
- Borland AppServer
- HP Bluestone



## Diferencias entre front-end y back-end

Para comenzar, podríamos plantear la diferencia principal entre Front-end y Back-end en los siguientes términos: los desarrolladores frontend trabajan en lo que el usuario puede ver, mientras que los backend construyen la infraestructura que sienta las bases de la página.

Ambos componentes trabajan de forma conjunta para crear una aplicación o sitio web de alto rendimiento. Es bastante común que las empresas confundan o mezclen las tareas de un desarrollador frontend con las de un backend.

Aquello en parte se debe a que cada vez existe un mayor número de herramientas que pueden aplicarse en ambos espectros de desarrollo web. Por lo tanto, es comprensible que las personas ajenas a este campo asuman que no existe mucha diferencia entre estos especialistas.

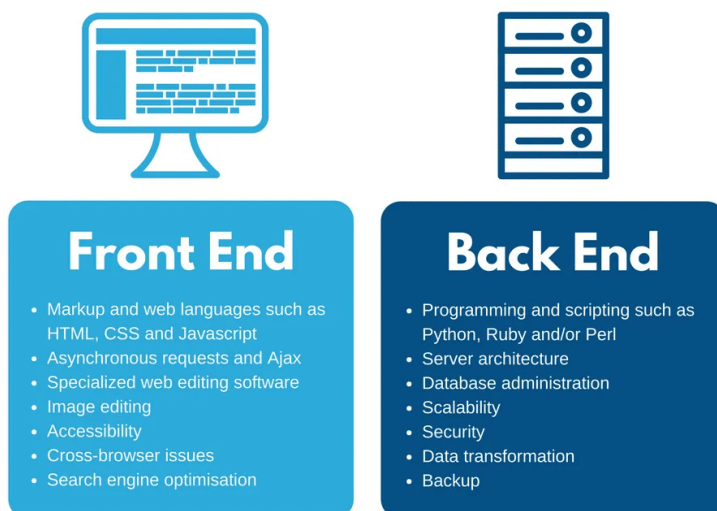
A pesar de que trabajan juntos, cada uno tiene prioridades diferentes. Frontend se refiere a la interfaz con la que el usuario interactúa, mientras que el backend significa el servidor, aplicación y base de datos que trabaja tras bastidores para entregar la información al usuario.

El proceso es el siguiente: el usuario ingresa la información por medio de la interfaz, la cual es verificada y comunicada al servidor. Este extrae la data necesaria y la envía de vuelta al usuario. Como puedes ver, el proceso es bastante simple. Básicamente, cuando hablamos de "detrás de escena", es decir, el servidor y la base de datos que ayudan a entregar información del usuario desde una interfaz, hablamos del backend. Es la parte del sitio con la que los usuarios no tienen contacto.

El back-end es una parte fundamental de cualquier sitio web o aplicación web. Si está leyendo este texto, por ejemplo, es una señal de que la comunicación con el servidor fue exitosa y esto probablemente se deba al buen trabajo del programador Web Full Stack.

Cualquiera que prefiera especializarse como desarrollador de back-end actuará con lógica, funcionalidad del sitio, reglas, seguridad e integridad de la base de datos. Es decir, vivir detrás de escena de Internet requiere mucha paciencia, cuidado y concentración constante.

Si el back-end es el desarrollo del elemento web que no vemos, el front-end es toda la parte visible de las aplicaciones y sitios web. Esta área no trata directamente con bases de datos, servidores y todas las aplicaciones de back-end complejas, pero aborda la usabilidad, los efectos visuales y la velocidad de carga, entre otros detalles.



Más directamente, el Desarrollador Front End es responsable de la interacción directa del usuario, por lo que se desarrolla cuidando el lado más visual de las aplicaciones, como el cuidado de los colores, botones, enlaces, menús y todo lo que vemos en una página cuando estamos accediendo.

Precisamente por esto, un profesional de front-end necesita tener un ojo constante para la mejor experiencia de usuario. Es decir, las preocupaciones de front-end y back-end son opuestas pero complementarias. Los desarrolladores front-end y back-end siempre deben trabajar juntos para que la aplicación o el sitio funcionen correctamente.

Los programadores pueden trabajar tanto en el back-end como en el front-end, por eso a estos profesionales se los llama Desarrolladores Web Full Stack. En definitiva, se trata de personas con una visión más completa del negocio que trabaja de principio a fin de un proyecto. Para eso, tiene conocimientos de diferentes tecnologías de programación y lenguajes, especialmente si actúa solo.

## Preparando el Entorno Integrado de Desarrollo3.1.- Creación de un proyecto web dinámico

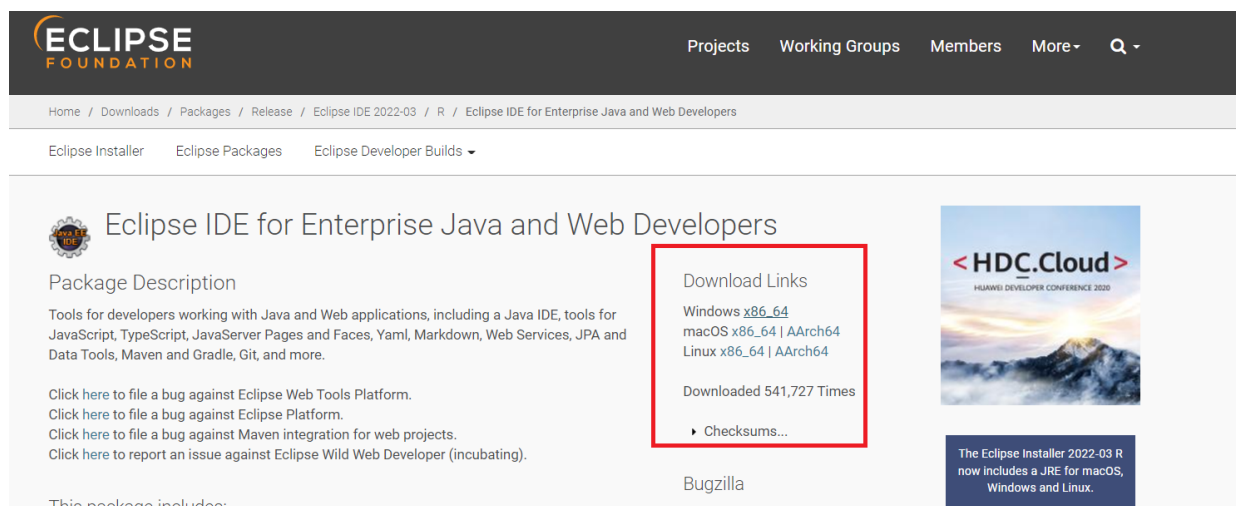
### Creando un proyecto dinámico en Eclipse

Para crear un proyecto web dinámico en Eclipse, lo puede hacer a través de las siguientes opciones:

- **File → New → Dynamic Web Project o**
- **File → New → Other → Web → Dynamic Web Project**

Para desarrollar aplicaciones que se ejecuten en un servidor web se debe utilizar la versión de Eclipse que viene con todos los complementos que facilitan el desarrollo.

La versión que deberá descargar es Eclipse IDE for Enterprise Java and Web Developer disponible en el sitio oficial de Eclipse Foundation en el siguiente link: <https://www.eclipse.org/downloads/packages/>



**ECLIPSE FOUNDATION** Projects Working Groups Members More Q

Home / Downloads / Packages / Release / Eclipse IDE 2022-03 / R / Eclipse IDE for Enterprise Java and Web Developers

Eclipse Installer Eclipse Packages Eclipse Developer Builds

### Eclipse IDE for Enterprise Java and Web Developers

**Package Description**

Tools for developers working with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer Pages and Faces, YAML, Markdown, Web Services, JPA and Data Tools, Maven and Gradle, Git, and more.

Click here to file a bug against Eclipse Web Tools Platform.  
Click here to file a bug against Eclipse Platform.  
Click here to file a bug against Maven integration for web projects.  
Click here to report an issue against Eclipse Wild Web Developer (incubating).

This package includes:

**Download Links**

Windows x86\_64  
macOS x86\_64 | AArch64  
Linux x86\_64 | AArch64

Downloaded 541,727 Times

Checksums...

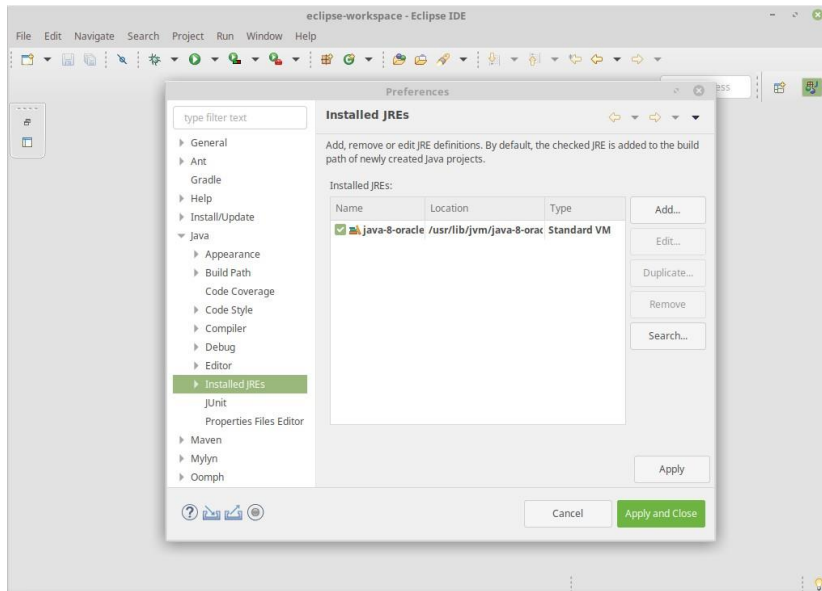
Bugzilla

**<HDC.Cloud>**  
HUAWEI DEVELOPER CONFERENCE 2020

The Eclipse Installer 2022-03 R now includes a JRE for macOS, Windows and Linux.

## Configuración de ambiente Eclipse IDE for Java EE Developers para crear sitio webdinámico

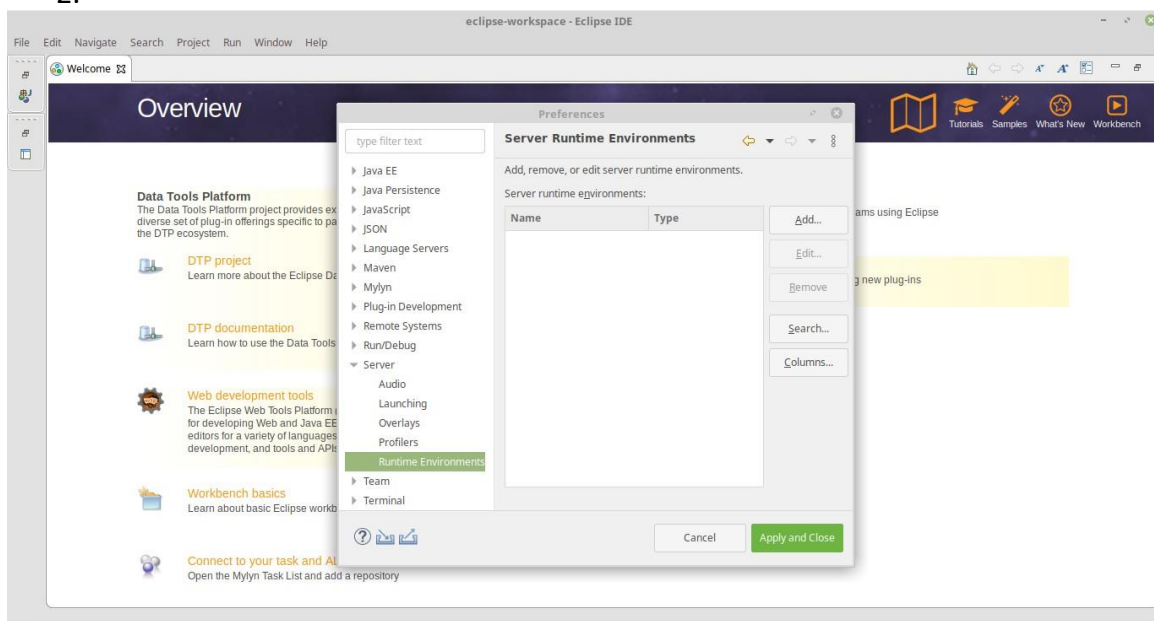
Para realizar una aplicación en Java ya sea web o escritorio se deberá configurar el JRE. Para esto ingresamos a la opción de menú Window → Preferences



1. Se mostrará una ventana con todas las configuraciones de Eclipse. Elegimos Java → Installed JREs, y se mostrarán los JRE configurados hasta ese momento. Se debe tener en cuenta que se pueden configurar diferentes versiones según el o los desarrollos que se hagan.

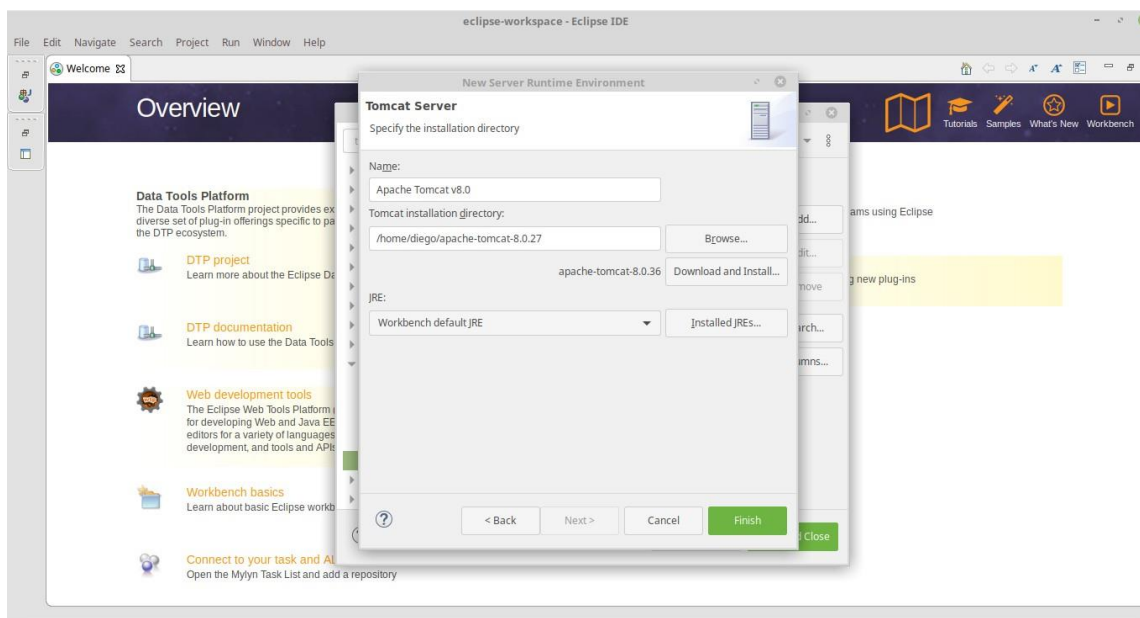
## Configuración del server dentro del proyecto

1. Ir a la opción Window → Preferences → Server → Runtime Environments.
- 2.





3. En la ventana que se desplegará, seleccionar Add → Apache → Version (por ejemplo apache Tomcat v9).
4. En el nuevo diálogo que aparece seleccionamos de la carpeta "Apache".

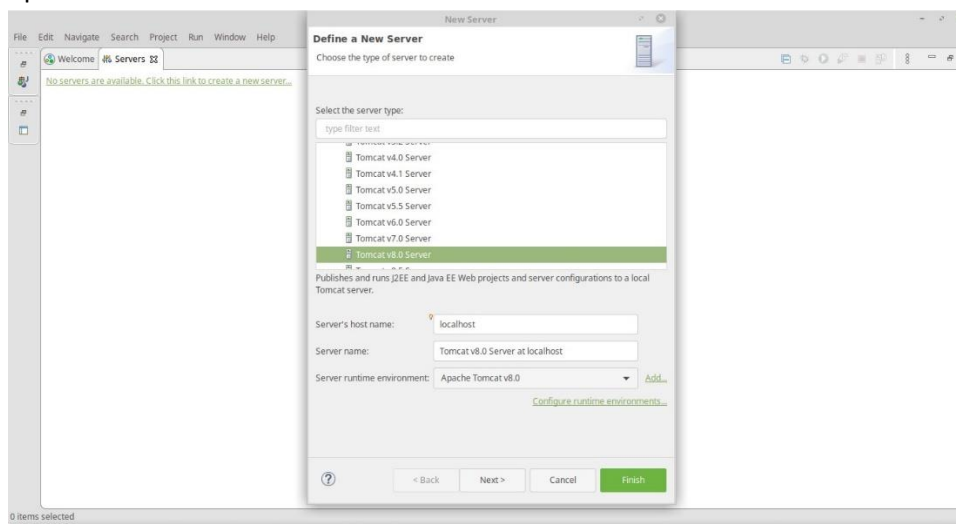


5. En el último texto que aparece debemos seleccionar la carpeta donde hemos descomprimido el "Apache Tomcat" y presionar el botón "Finish".
6. Ahora debemos iniciar los servicios del servidor "Apache Tomcat", lo que permitirá hacer aplicaciones que hagan peticiones a un servidor local.
7. Para arrancar el servidor Tomcat, debemos presionar el botón derecho del mouse sobre la ventana "Server"; si no aparece esta ventana podemos activarla desde el menú (Window → Show View → Servers) y seguidamente seleccionar del menú contextual la opción

New

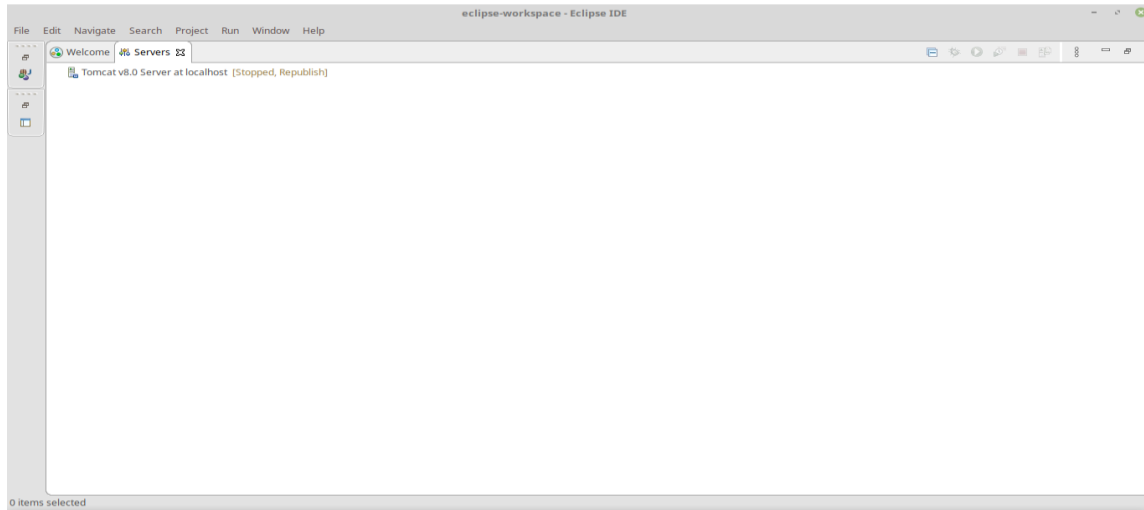
->

Server.





8. Comprobar que ya tenemos el servidor "Tomcat" listo para poderlo utilizar en los distintos proyectos que implementaremos.



**[1] Qué es Java Enterprise (J2ee, JEE)**

Referencia: <https://www.fundesem.es/bt/publicacion-java-ee-y-el-desarrollo-web-un-enfoque-de-aprendizaje>

**[2] Qué son los Servlets**

Referencia: <https://users.dcc.uchile.cl/~jbarrios/servlets/general.html>

**[3] Utilizando métodos post y get**

Referencia: <https://www.ecodeup.com/enviar-parametros-servlet-desde-una-vista-utilizando-post-get/>

**[4] Sesiones y cookies**

Referencia: <https://ricardogeek.com/manejo-de-sesiones-y-cookies-en-servlets-java/>

**[5] Etiquetas JSTL**

Referencia: <http://www.jtech.ua.es/ayto/ayto2008/jsp/sesion07-apuntes.html>