

DAM  
Desarrollo de Aplicaciones Multiplataforma  
2º Curso

AD  
Acceso a Datos

UD 0  
Iniciación a Java  
(Parte 2)

IES BALMIS  
Dpto Informática  
Curso 2021-2022  
Versión 1 (09/2021)

## UD0 – Iniciación a Java

### ÍNDICE

1. Motivación
2. Esqueleto básico
3. Compilar (Javac y jGRASP)
4. Leer desde consola
5. Condiciones
6. Bucles
7. Tipos de datos básicos
8. Arrays
9. Funciones
10. Estructuras dinámicas
11. Clases
12. Interfaces
13. Java en Linux
14. Geany
15. NetBeans
16. Otros IDE (IntelliJ y Eclipse)
17. Spring

## 13. Java en Linux

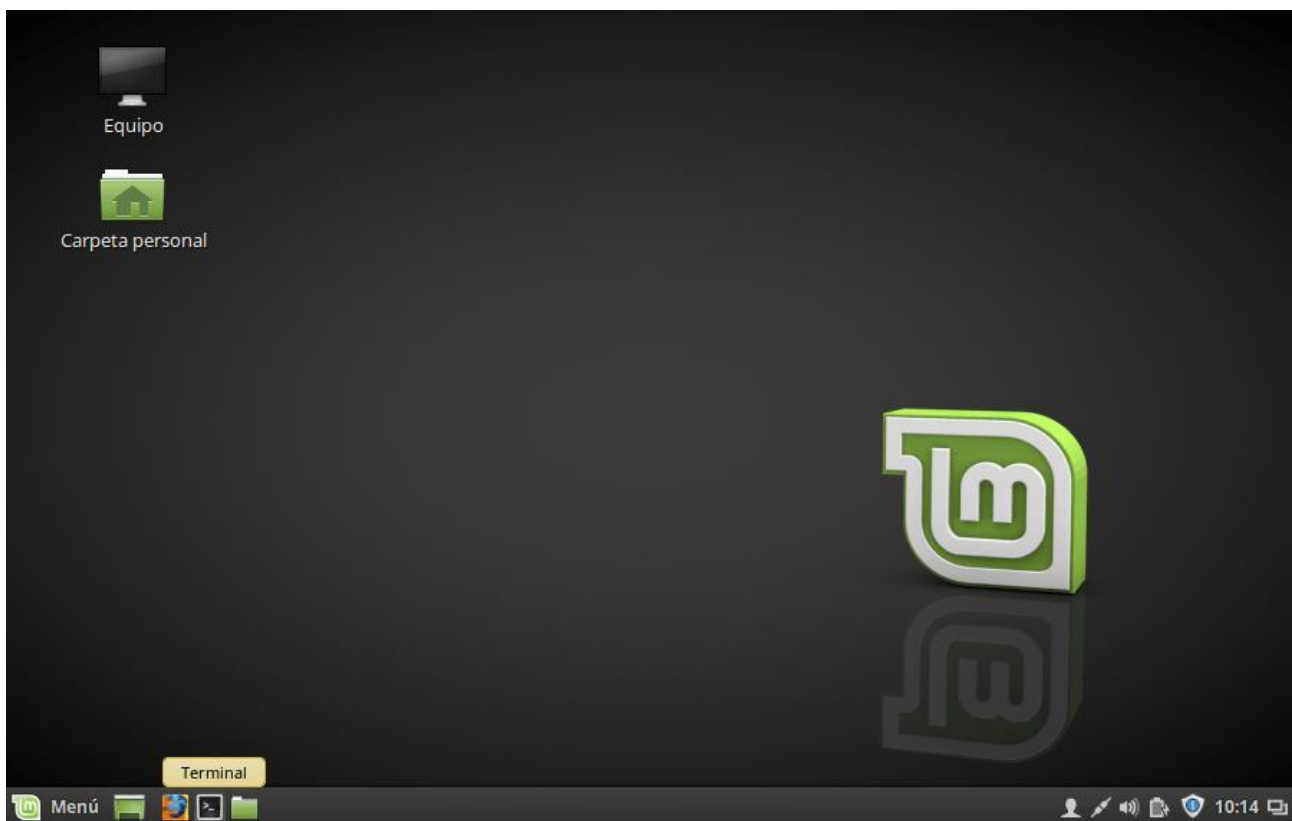
Como ya hemos comentado, para compilar los **pasos** son sencillos, aunque en la práctica se pueden complicar un poco:

1. Teclear el **fuelle** usando cualquier **editor** de **texto**.
2. **Compilar** con "javac", por ejemplo:  
**javac HolaMundo.java**
3. **Lanzar** con "java", por ejemplo:  
**java HolaMundo**

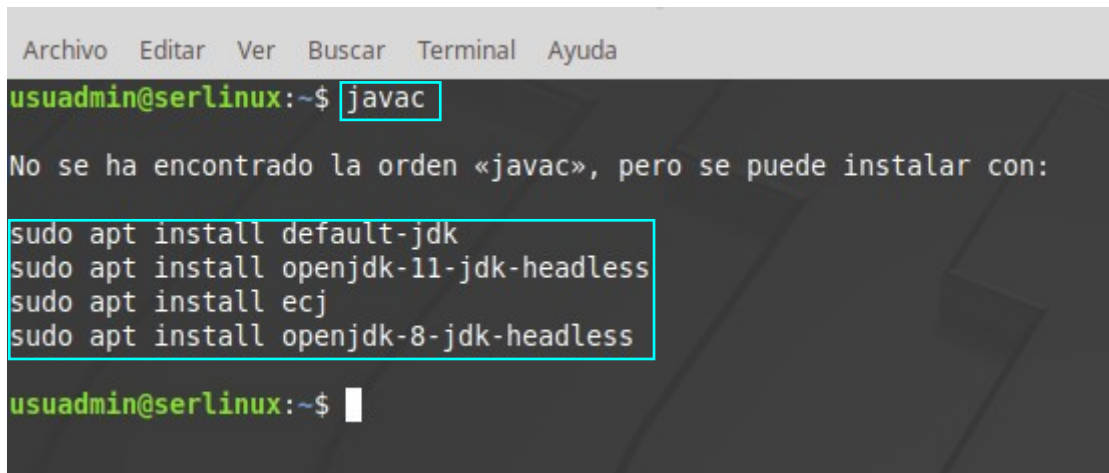
En la **práctica** pueden aparecer **distintos** **problemas**, como **no** tener **Java** **instalado**, que el **nombre** del **fichero** sea **incorrecto** o (en programas más complejos) no encontrar alguna de las **bibliotecas** que el **programa** **necesite**.

El **problema** más **habitual**, y el **único** que nos **interesa** por ahora, es que **no** esté **instalado** **Java**. Nos centraremos en una **solución** para el **caso** de **Linux**, en **concreto** con un **Linux Mint Cinnamon**, que se basa en Ubuntu pero tiene un entorno de escritorio más "normal".

En entornos un poco más convencionales, como Linux Mint Cinnamon, es habitual que haya un acceso directo al terminal directamente en la barra inferior:



El siguiente paso sería probar a lanzar el compilador de Java, tecleando "javac". Es habitual que, en caso de que éste no se encuentre instalado, se nos muestren pistas de cómo instalarlo:



```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
usuadmin@serlinux:~$ javac

No se ha encontrado la orden «javac», pero se puede instalar con:

sudo apt install default-jdk
sudo apt install openjdk-11-jdk-headless
sudo apt install ecj
sudo apt install openjdk-8-jdk-headless

usuadmin@serlinux:~$
```

Como se nos indica, lo podemos instalar con "**sudo apt-get install default-jdk**". Al final de la instalación podremos lanzar el editor "gedit" seguido del nombre del programa que deseamos crear o editar, como "**gedit HolaMundo.java**". Si ese editor no está instalado, podemos optar por usar el del sistema (que quizá no realice la sintaxis en colores) o por instalarlo del mismo modo.

### **Configurar Proxy en Programas/Aplicaciones**

Para configurar el proxy en comandos como apt-get, es necesario editar la configuración.

#### **Paso 1. Editar el archivo de configuración**

```
sudo gedit /etc/apt/apt.conf
```

#### **Paso 2. Añadir la siguiente línea con los datos de IP y port de tu proxy**

```
Acquire::http::proxy "http://yourproxyaddress:proxyport";
Acquire::https::proxy "https://yourproxyaddress:proxyport";
Acquire::ftp::proxy "ftp://yourproxyaddress:proxyport";
```

**Nota.** En caso de necesitar usuario y contraseña se sustituye

```
"http://yourproxyaddress:proxyport";
```

por:

```
"http://username:password@yourproxyaddress:proxyport";
```

## Configurar Proxy de Sistema

La configuración del proxy de sistema se realiza desde los asistentes del entorno gráfico tanto en Linux como en Windows, asignando la IP y el puerto del servidor proxy.

En Windows 10 podemos configurar el proxy de sistema desde el Configurador de Windows 10:

- Paso 1. Acceda a la configuración de Windows 10. Para ello, abres el menú Inicio y haces clic en "Configuración" en el lado izquierdo.
- Paso 2. En la ventana Configuración del sistema, haz clic en "Redes e Internet".
- Paso 3. Ahora, en la barra lateral izquierda, haz clic en "Proxy" ya a la derecha, activa la opción "Usar un servidor proxy". Por último, introduce los datos del servidor (dirección y puerto) y haz clic en "Guardar".

En Windows 10 también podemos acceder directamente a través de los nuevos comandos ms-settings. En el caso del proxy el comando es:

```
start ms-settings:network-proxy
```


En Linux, el asistente nos aparece desde "Preferencias de Red".

En las aplicaciones podremos indicar que se use el proxy del sistema o bien configurarlo manualmente.

En los navegadores de los ordenadores del aula se debe configurar para poder navegar.

En Linux tanto en los navegadores como aplicaciones como synaptic (Asistente para la instalación de aplicaciones) se realiza la configuración mediante asistente.

En la ventana de "gedit" podremos escribir el programa. Como ya le hemos indicado que se trata de un programa en Java, se mostrará la sintaxis realzada en colores, para ayudarnos a detectar fallos:



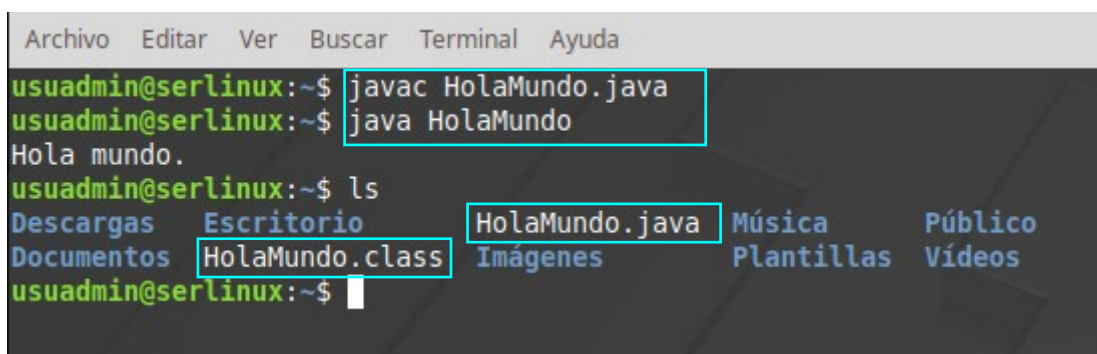
```
Abrir [icon] *HolaMundo.java Guardar - + x
Archivo Editar Ver Buscar Herramientas Documentos Ayuda
// Fichero HolaMundo.java
// Compilar con: javac HolaMundo.java
// Ejecutar con: java HolaMundo

public class HolaMundo {
    public static void main(String [] args) {
        System.out.println("Hola mundo.");
    }
}
```

Java Anchura de la pestaña: 8 Ln 10, Col 1 INS

Si, por el contrario, hubiéramos abierto directamente "gedit" (o cualquier otro editor de textos como "nano") sin indicarle nombre de ficheros, es probable que la sintaxis no se muestre en colores, como mínimo hasta que guardemos el fichero con un nombre terminado en ".java".

Cuando salgamos del editor ya podremos usar la orden "javac" para compilar el programa y "java" para lanzarlo. La ejecución de "javac" nos creará el archivo **HolaMundo.class** que es que ejecutaremos con "java".



```
Archivo Editar Ver Buscar Terminal Ayuda
usuadmin@serlinux:~$ javac HolaMundo.java
usuadmin@serlinux:~$ java HolaMundo
Hola mundo.
usuadmin@serlinux:~$ ls
Descargas Escritorio HolaMundo.java Música Público
Documentos HolaMundo.class Imágenes Plantillas Vídeos
usuadmin@serlinux:~$
```

## 14. Geany

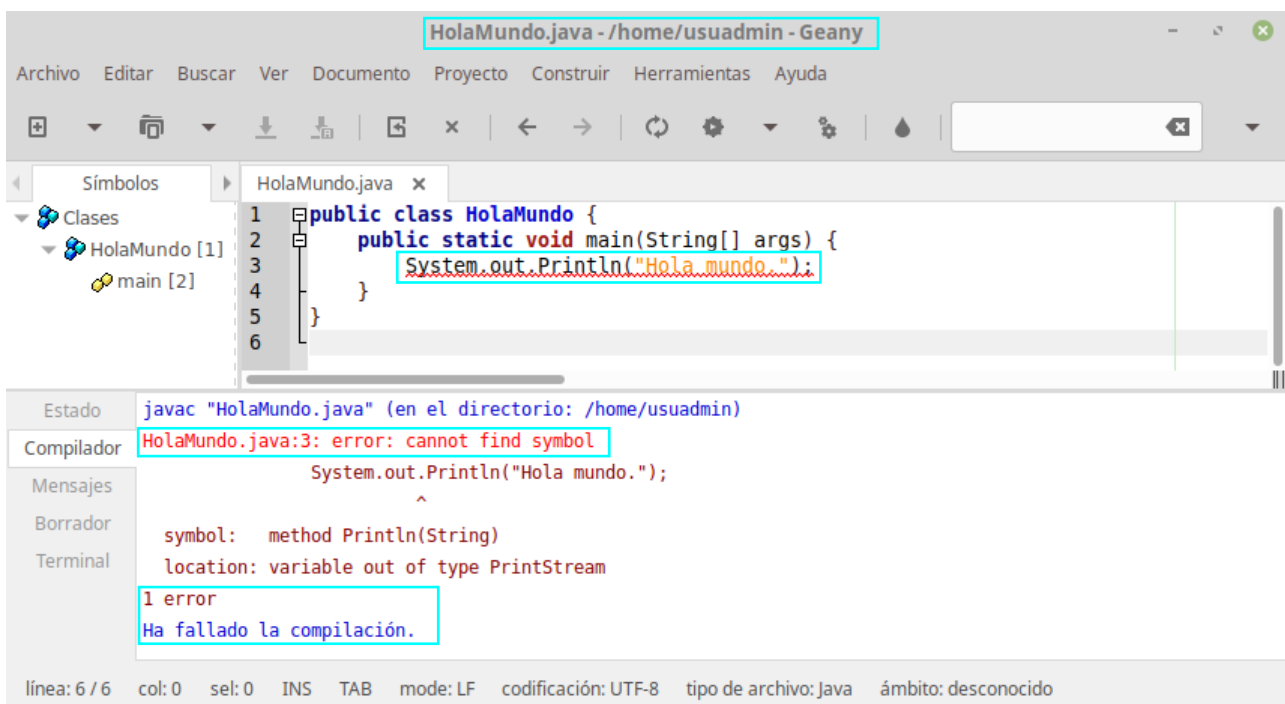
Hay alternativas más avanzadas, que permiten ahorrar algo de trabajo al no tener que teclear siempre **"javac"** y **"java"**, pero que no suelen venir preinstaladas en el sistema.

Es el caso de **Geany**, un editor para **Java** más potente que **"gedit"** con licencia **GNU GPL** (software libre y gratuito), y que permite compilar y lanzar el ejecutable simplemente con un clic. También permite saltar a las líneas que tienen un cierto error, en caso de que el programa no haya compilado correctamente. Es un entorno muy adecuado para programas sencillos, formados por un único fuente.

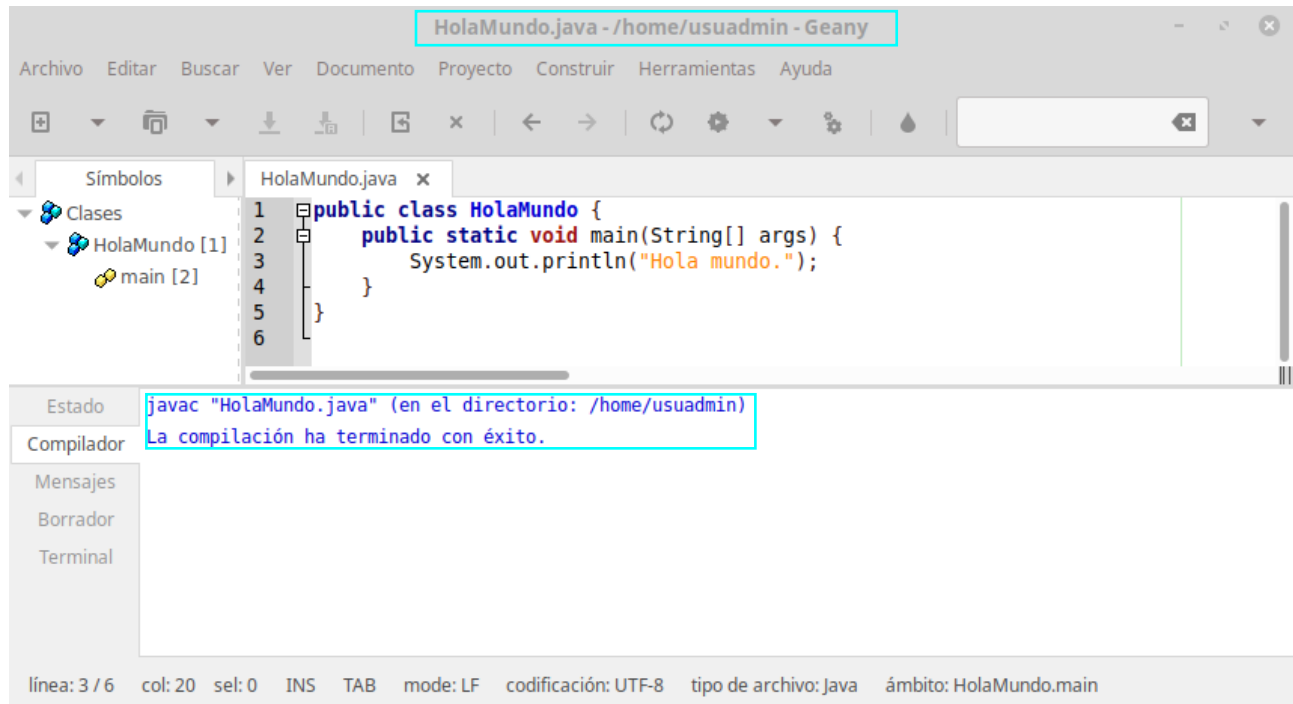
Para instalarlo en **Windows** bastaría con descargar el programa de su web oficial <https://www.geany.org/> y ejecutarlo.

Para instalarlo en **Linux** se puede ejecutar **"sudo apt-get install geany"**.

Por ejemplo, ésta sería la apariencia de **Geany** tras intentar compilar un programa con errores:



Y ésta tras **compilar un fuente sin errores:**





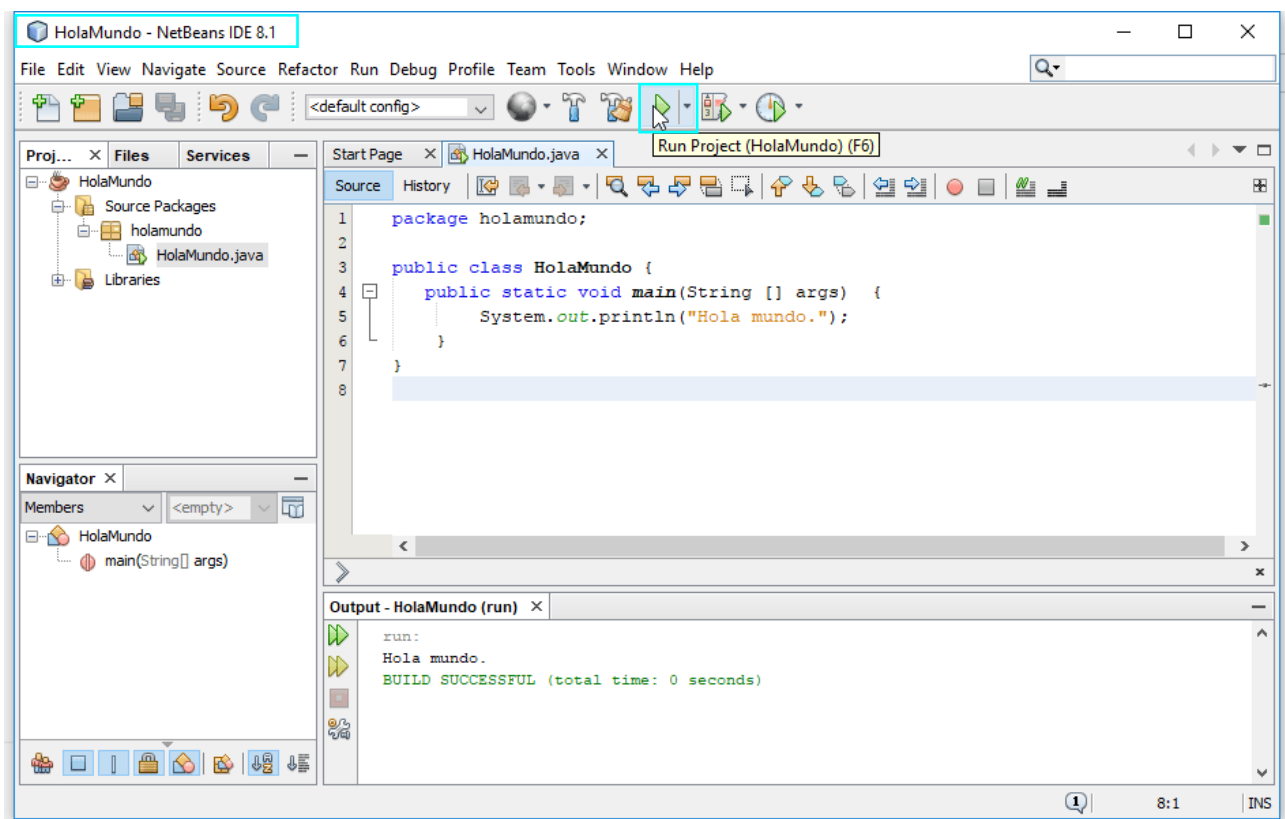
## 15. Netbeans

**NetBeans**, al igual que otros IDE como **Eclipse**, es de mucho mayor tamaño, pero mucho más adecuado para programas formados por varios fuentes, ya que permite crear y mantener proyectos con facilidad.

Para instalarlo en Windows accederemos al portal oficial para descargar la versión 11.3 por su compatibilidad con **Hibernate** que usaremos más tarde:

<https://netbeans.org/downloads>

En el caso de usar (por ejemplo) NetBeans, bastaría con instalarlo, crear un nuevo proyecto (de tipo "Java Application") y lanzarlo con el botón "Run Project":



El programa creado desde NetBeans deberá formar parte de un "package", o no se podrá compilar correctamente:

```
package holamundo;  
  
public class HolaMundo {  
    public static void main(String [] args) {  
        System.out.println("Hola mundo.");  
    }  
}
```

Un **package** (en español: "paquete") es un contenedor de un grupo de clases, interfaces, etc. relacionadas entre si. Un paquete es a Java lo que una librería es a otro lenguaje (por ejemplo: C). Estos pueden contener clases, interfaces, enumerados, anotaciones e incluso otro tipo de archivos.

Las ventajas de organizar nuestros archivos en paquetes:

- Agrupar clases que tienen algo en común.
- Reutilización del código.
- Seguridad gracias a los modificadores de acceso.

El package se mostrará como una carpeta en el explorador del proyecto.

Para nuestros primeros proyectos crearemos una estructura sencilla basada en la siguiente estructura:

- Proyecto
  - Source Packages
    - app
      - Main.Java
    - clases

**app:** será el package para los archivos de nuestro proyecto, siendo Main.java el archivo principal que contendrá el método main.

**clases:** será el package que contenga las clases definidas para el proyecto

Además, crearemos dos carpetas específicas:

- **datos:** donde se alojarán los archivos de datos utilizados en el proyecto
- **lib:** donde se copiarán las librerías usadas por el proyecto

## 16. Otros IDE (IntelliJ y Eclipse)

IntelliJ IDEA y Eclipse son entornos de desarrollo integrados (IDE) que permiten trabajar con lenguaje Java. Analizamos sus características y ventajas para ayudar a seleccionar el IDE de Java adecuado para sus necesidades de desarrollo de aplicaciones java.

### Netbeans

- Un IDE de Oracle
- Se desarrolla para versiones de Java desde Java ME hasta Java EE
- Una gran variedad de plugins (aunque no tantos como eclipse).
- Los diferentes paquetes tienen varias funciones.
- Herramientas y editores para HTML, PHP, XML, JavaScript y más.
- Soporte para HTML5 y otras tecnologías web.
- Soporte de base de datos con controladores para Java DB, MySQL, PostgreSQL y Oracle.
- El Explorador de BD crea, modifica y elimina tablas y bases de datos.

#### Netbeans

<https://netbeans.org/>

Nosotros usaremos Netbeans por:

- simplicidad
- compatibilidad
- trabajo sin conexión a internet
- fácil exportación y reutilización de proyectos

### Eclipse

- Un IDE muy estable y ampliamente usado
- Muchos complementos lo hacen versátil y personalizable.
- El IDE está organizado en Perspectivas
- Diseñado para grandes proyectos de desarrollo.
- Maneja análisis y diseño.
- Se ocupa de la gestión de productos.
- Maneja la implementación.
- Maneja el desarrollo de contenido, pruebas y documentación.

#### Eclipse

<https://www.eclipse.org/downloads/>

## IntelliJ

- Tiene una edición comercial y una versión de código abierto.
- Admite Java, Scala, Groovy, Clojure y más.
- Maneja la finalización de código inteligente.
- Maneja el análisis de código.
- Soporta refactorización avanzada.
- La versión comercial es compatible con SQL, ActionScript, Ruby, Python y PHP.
- La versión 12 admite el desarrollo de aplicaciones de Android.
- 900 complementos (más en la versión comercial).

### IntelliJ

<https://www.jetbrains.com/idea/download>

## 17. Spring

**Spring** es un framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.

Si bien las características fundamentales de Spring Framework pueden ser usadas en cualquier aplicación desarrollada en Java, existen variadas extensiones para la construcción de aplicaciones web sobre la plataforma Java EE. A pesar de que no impone ningún modelo de programación en particular, este framework se ha vuelto popular en la comunidad al ser considerado una alternativa, sustituto, e incluso un complemento al modelo EJB (Enterprise JavaBean).

**Spring Tools Suite** es una aplicación parametrizada para usar Spring en el desarrollo de proyectos Java.

Existe una versión de Spring Tools Suite integrada en Eclipse, lo que influye en las empresas a la hora de elegir Eclipse como IDE de desarrollo.

### Spring Tools Suite

<https://spring.io/tools>