

IDE NetBeans

Contenido

¿Qué es NetBeans?	2
Crear proyectos	2
Guardar proyecto	3
Versionar (GIT)	3
Recuperar desde el repositorio	5
Instalar plugins	8
Desarrollar	9
Compilar	10
Ejecutar	10
Probar	11
Documentar	11
Generar documentación	12
Modelar	13
Depurar	13
Inspección de variables	14
Ejecución paso a paso	15
Comparar código	16
Refactorizar	17
Generar código	18
Ingeniería inversa	19
Administrar base de datos	19
Ejecutar script sobre la base de datos	19

¿Qué es NetBeans?

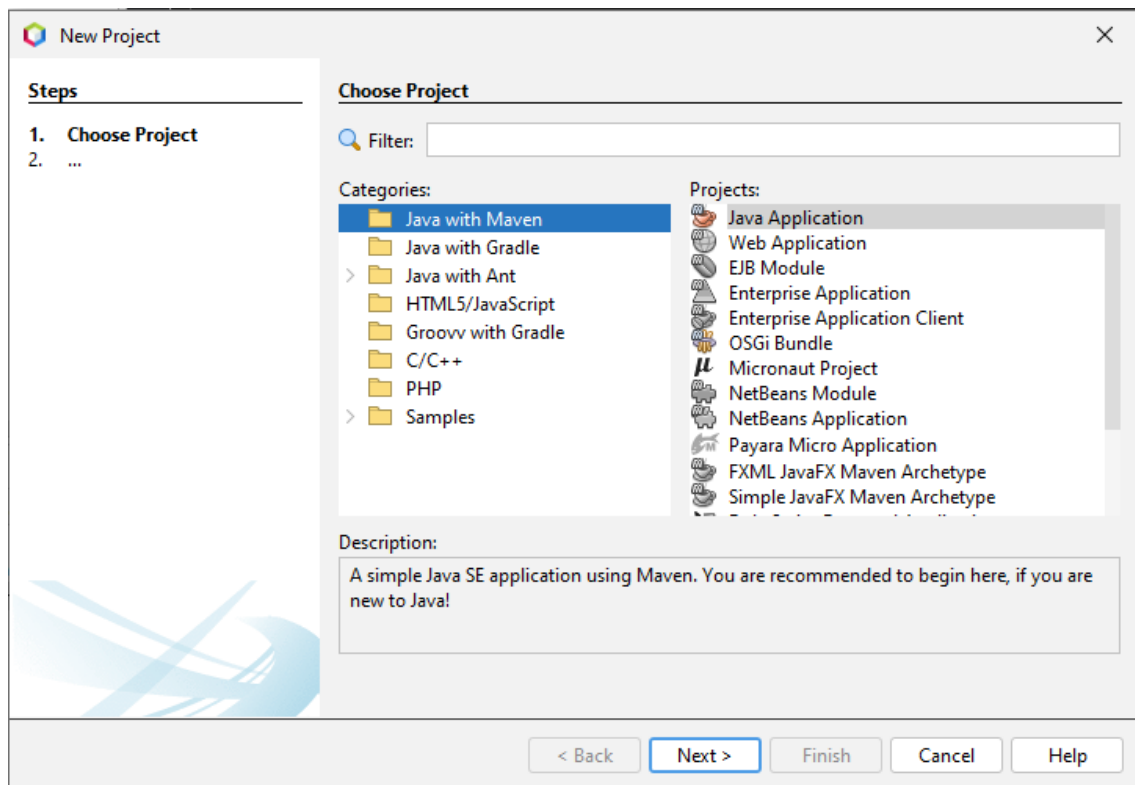
Es un entorno de desarrollo integrado lo que significa que, con él, se puede desarrollar cualquier tipo de aplicación informática, escribir código en la gran mayoría de lenguajes de programación, gestionar directorios y ficheros, instalar plugins oficiales y no oficiales (realizados por la comunidad), etc...

Crear proyectos

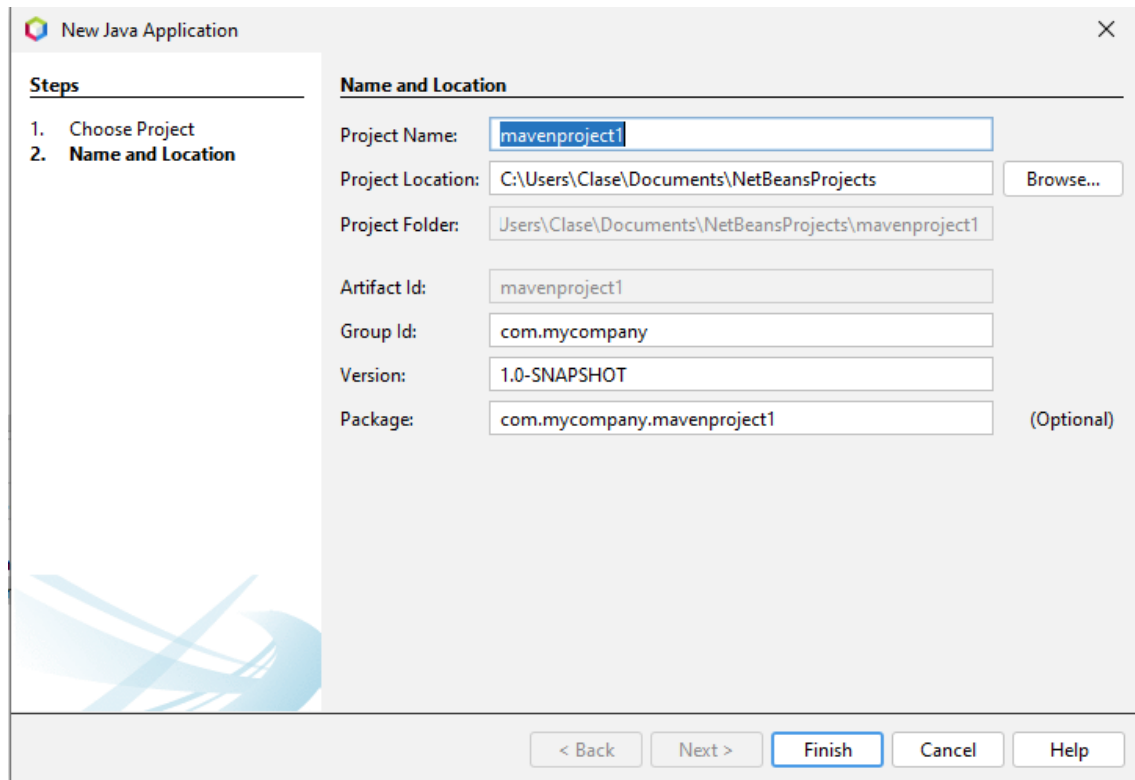
Usando los dos primeros iconos de la barra de herramientas podemos crear un nuevo proyecto. Se nos abrirá una pestaña en la que seleccionaremos el tipo de proyecto que queremos y, una vez escogido, nos pedirá un nombre para agregárselo al proyecto y guardarlo en un ruta también a gusto del desarrollador. Automáticamente se creará un archivo en el proyecto para poder empezar a trabajar. En el ejemplo que nos ocupa, al ser un proyecto Java se crea un archivo main con el nombre del proyecto.



1. Pulsamos este icono de un cuadrado amarillo y un + verde.



2. Nos aparecerá este cuadro donde podemos escoger el tipo de proyecto que queremos, filtrar con el buscador y donde nos describirá brevemente el tipo de proyecto escogido.



3. En el siguiente y último recuadro podemos escoger el nombre del proyecto, localización del proyecto, el id la versión y el paquete del proyecto.

Guardar proyecto

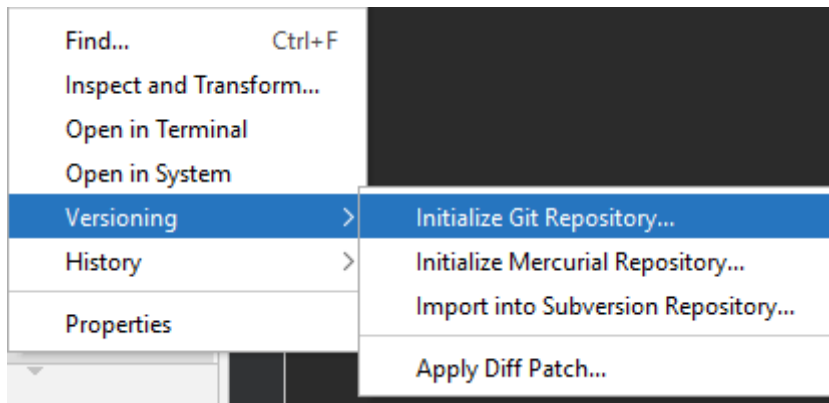
Una vez creado el proyecto ya se establece una ruta donde se almacenan todos los archivos. Después de trabajar sobre uno o varios archivos podemos guardar todos los cambios realizados en la sesión pulsando el botón con símbolo de disquetes en la barra superior.



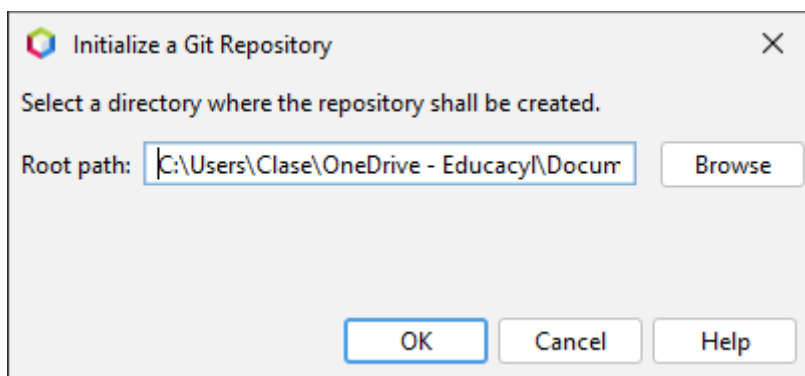
1. Este icono de dos disqueteras superpuestas sirve para guardar. Cuando un archivo tiene algún cambio no guardado el icono aparece “iluminado” y cuando ya está guardado el icono aparece grisáceo.

Versionar (GIT)

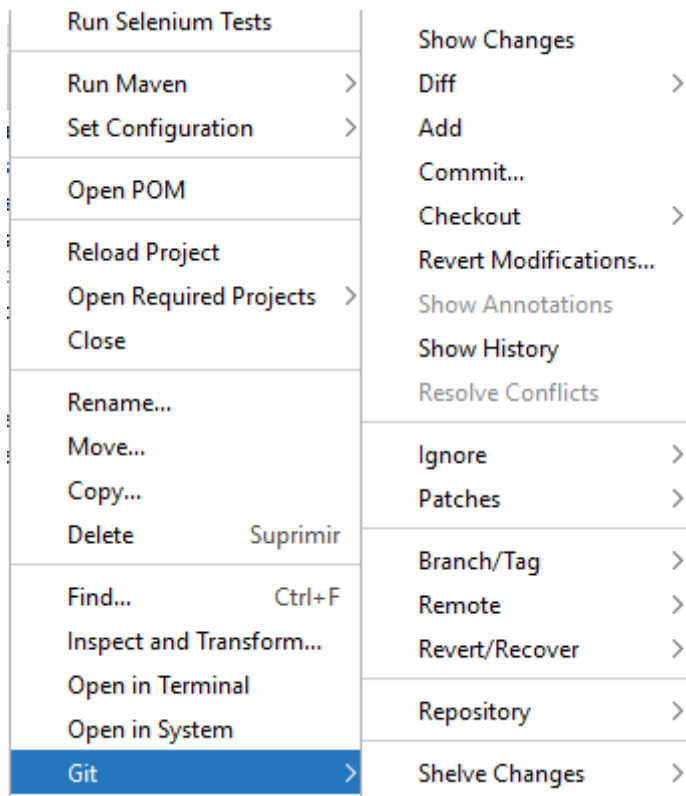
El uso de un repositorio es indispensable para un desarrollador de aplicaciones informáticas. Para inicializar un repositorio tenemos que pulsar botón derecho sobre el proyecto que queremos versionar e ir al apartado de “inicializar repositorio git”. Una vez creado el repositorio debemos hacer un primer commit y crear una rama de desarrollo para trabajar sobre ella en vez de usar constantemente la rama master. Esta rama solo se usará cuando el código este finalizado y probado correctamente.



1. Haciendo click derecho sobre el proyecto vamos al apartado Versioning y dentro de él, seleccionamos la primera opción: Initialize Git Repository.



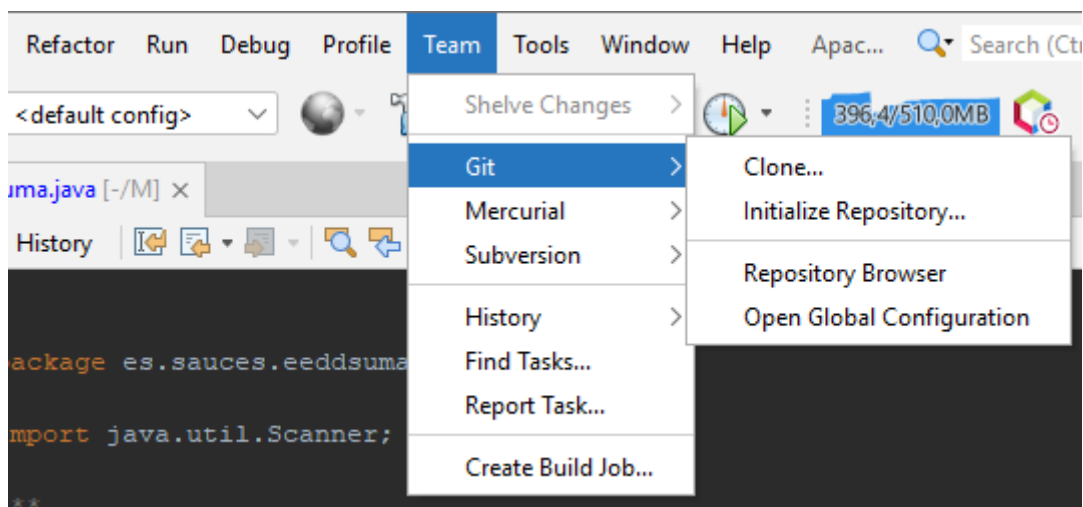
2. El paso siguiente es escoger una ruta donde guardar la configuración del repositorio el cual será la carpeta principal del proyecto.



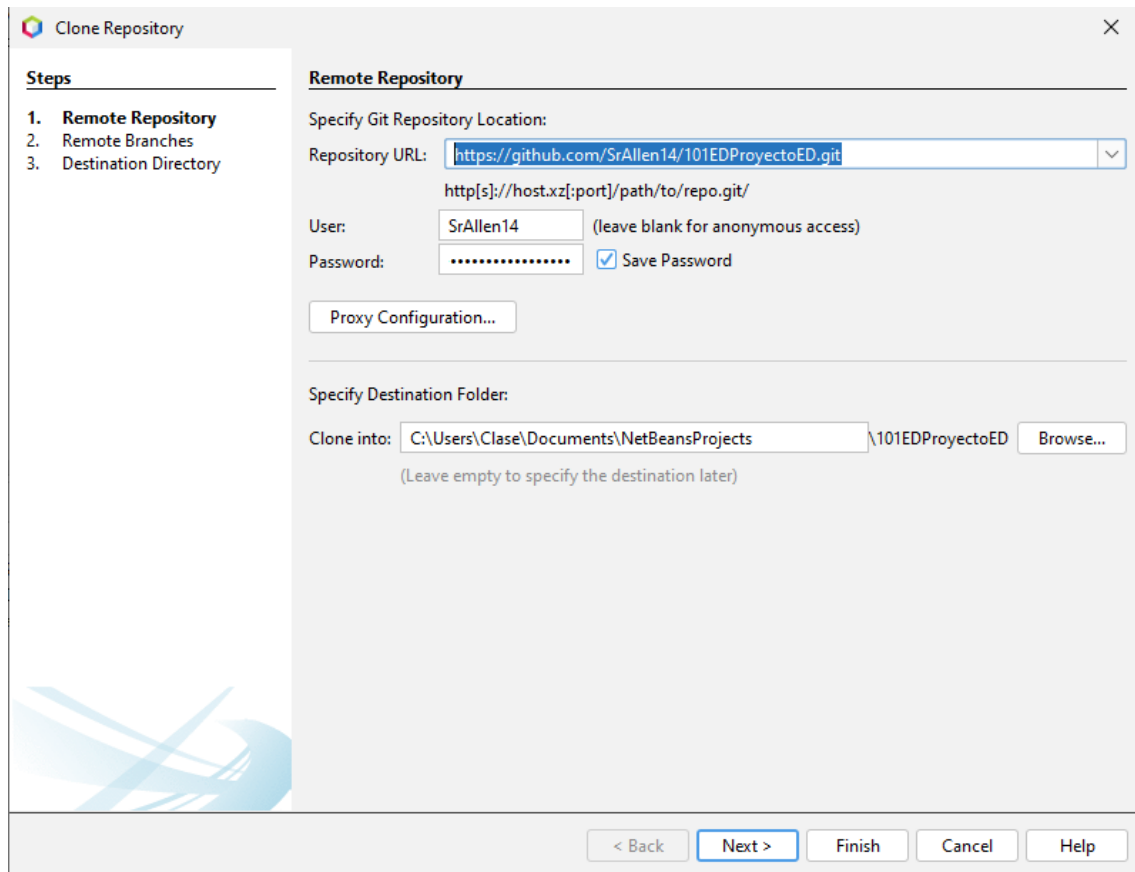
3. Después de inicializar el repositorio aparecerá este menú. Para terminar de configurar el repositorio crearemos una rama en la pestaña de Branch/Tag>Create Branch y realizaremos el primer commit en la pestaña Commit.

Recuperar desde el repositorio

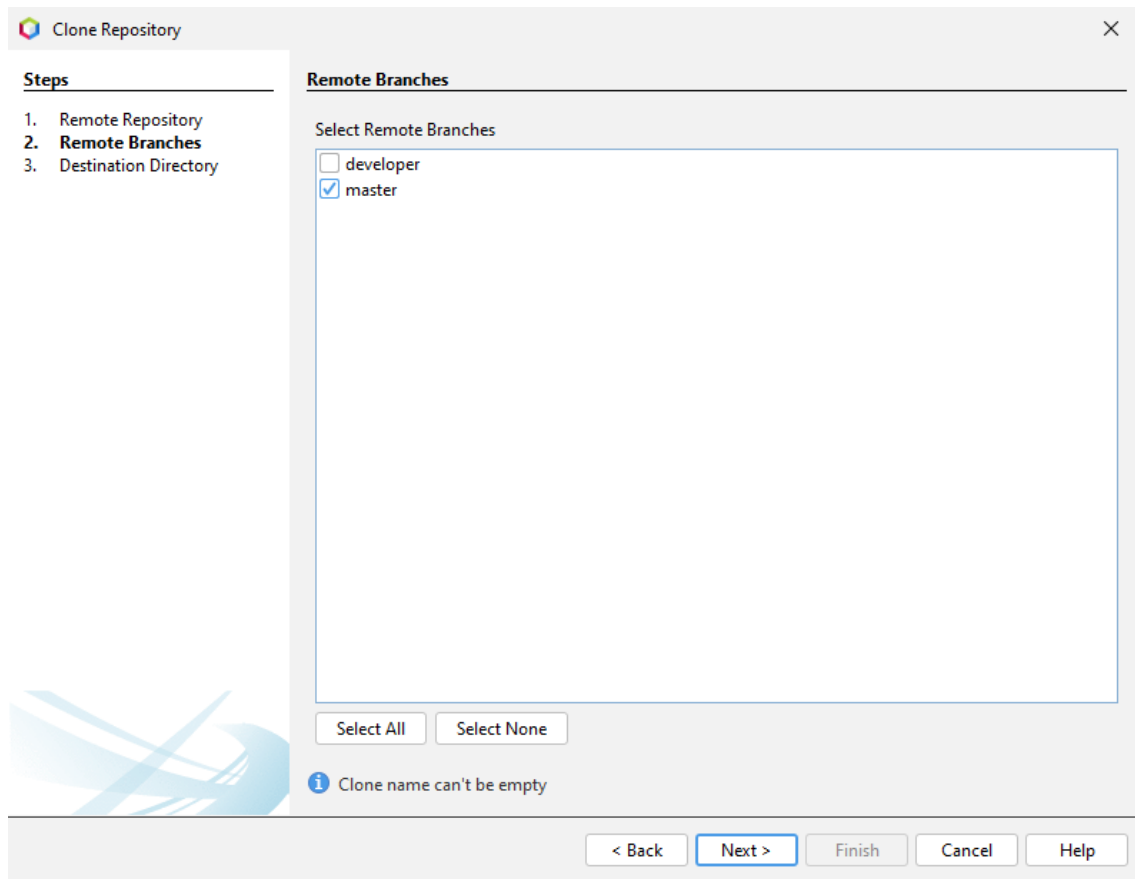
Podemos recuperar un repositorio desde NetBeans. Necesitamos un repositorio guardado en local o en algún repositorio online como GitHub. Esto nos ayudará a que, en caso de perder un proyecto o que se corrompa algún archivo del proyecto, podamos recuperarlo de forma fácil y segura.



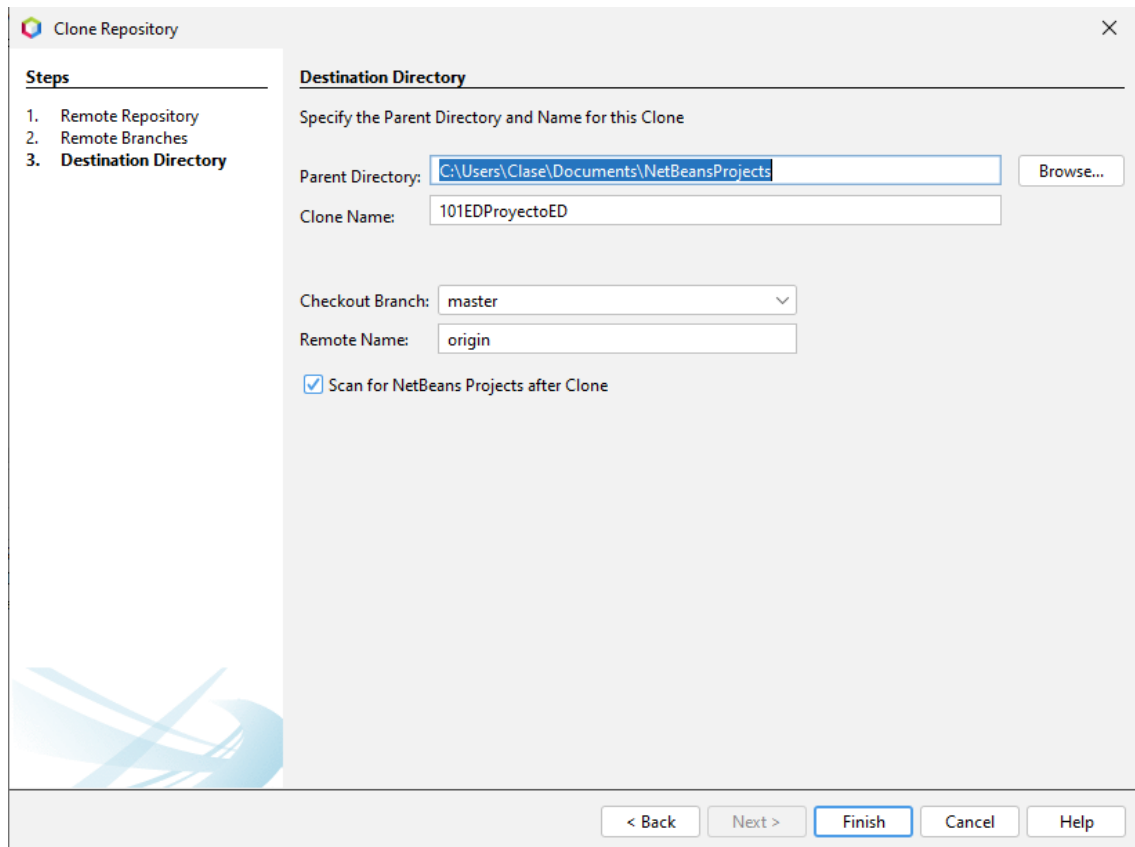
1. Para clonar un repositorio debemos ir a la pestaña Team>Git>Clone en la barra superior de la ventana de NetBeans.



2. Aquí es donde podemos escoger el repositorio que queremos clonar escribiendo la URL del archivo. Debemos facilitar el usuario y la clave de acceso del repositorio. También podemos establecer la ruta del proyecto donde la guardaremos en local.



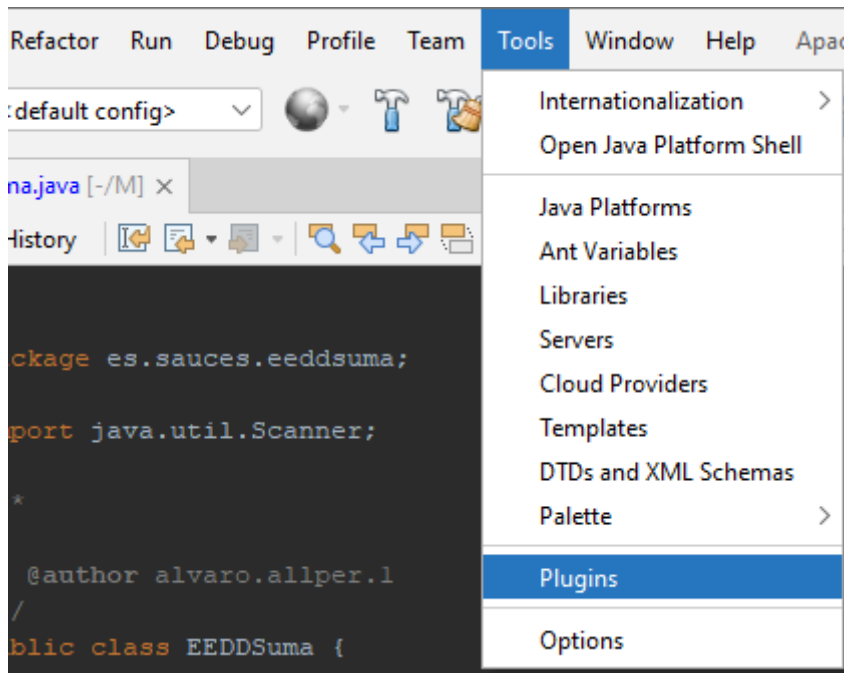
3. En este apartado podemos escoger las ramas que queremos clonar en local. Como mínimo debemos seleccionar una rama y como máximo todas las que haya.



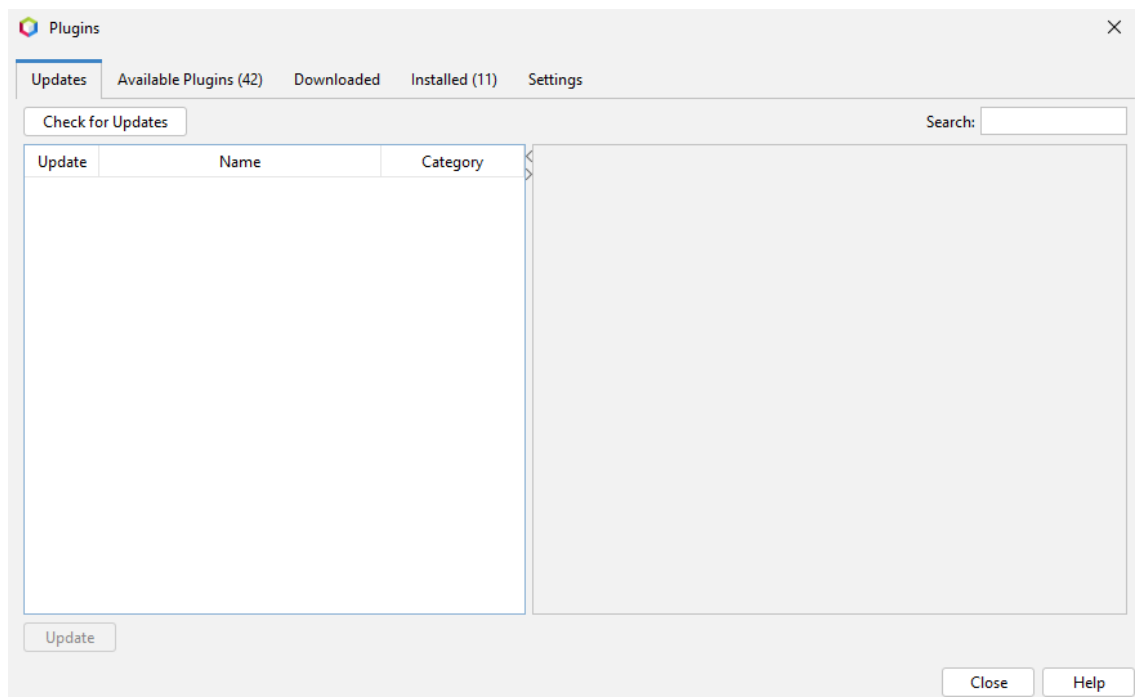
4. Por último debemos escribir el nombre de la carpeta clonada y la ruta donde la guardaremos. Por defecto viene introducido el nombre del repositorio.

Instalar plugins

Desde el apartado Tools>Plugins se puede, mediante el uso de un menú, instalar, actualizar, desinstalar y gestionar todo un catalogo de plugins. Los plugins son herramientas adicionales que mejorar la experiencia a la hora de desarrollar implementando nuevas funciones al IDE. Como ayuda en la búsqueda del plugin que queremos tenemos un buscador para filtrar la lista.



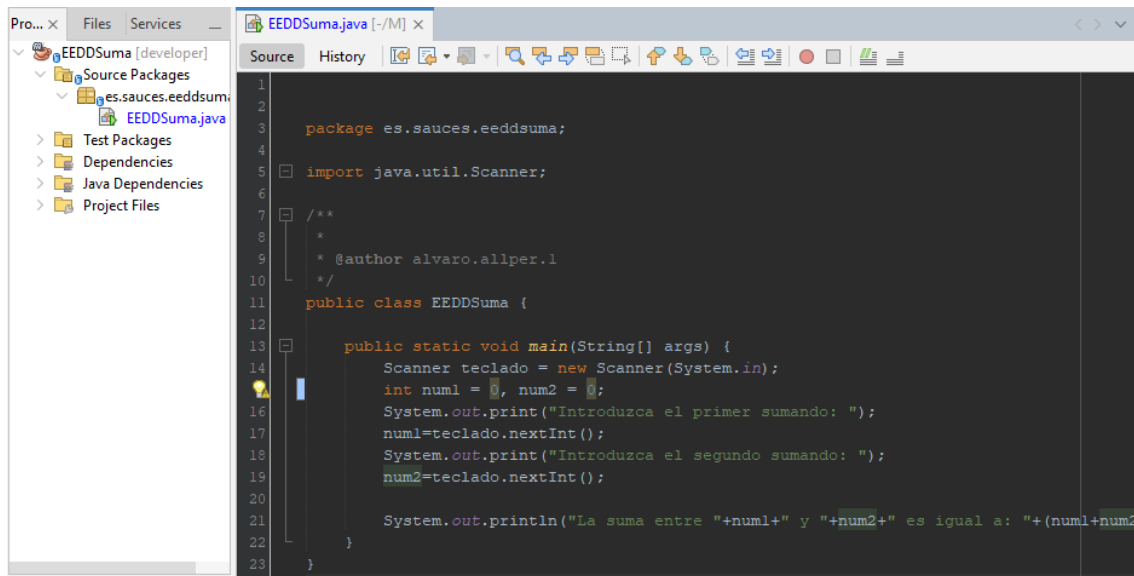
1. En la barra de herramientas superior vamos a Tools>Plugins.



2. En este cuadro podemos gestionar los plugins descargados, actualizarlos, desinstalarlos y deshabilitarlos. También podemos descargar plugins nuevos y configurar su funcionamiento.

Desarrollar

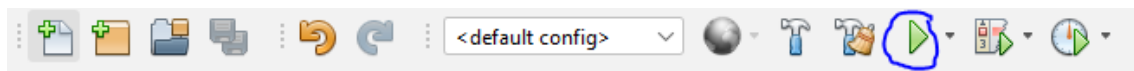
Tenemos un editor de texto donde poder redactar todo el código del programa así como una zona en la barra de herramientas para poder probar el código y depurarlo. También vienen implementadas funciones de autocompletado y un gestor de proyectos y archivos donde poder crear nuevos directorios y gestionar sus contenidos.



1. Aquí tenemos el editor de texto y el gestor de archivos y proyectos para desarrollar en NetBeans. En el editor de texto podemos escribir todas las líneas de código y cambiar entre archivos. En el gestor de archivos y proyectos podemos crear, borrar y modificar archivos y carpetas.

Compilar

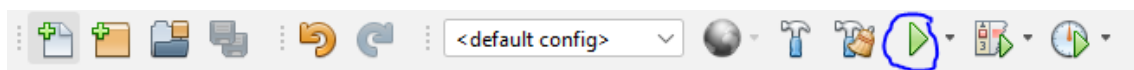
Usando el botón “Run” de la barra de herramientas se compila y ejecuta el programa automáticamente mostrando el resultado en la consola integrada de NetBeans.



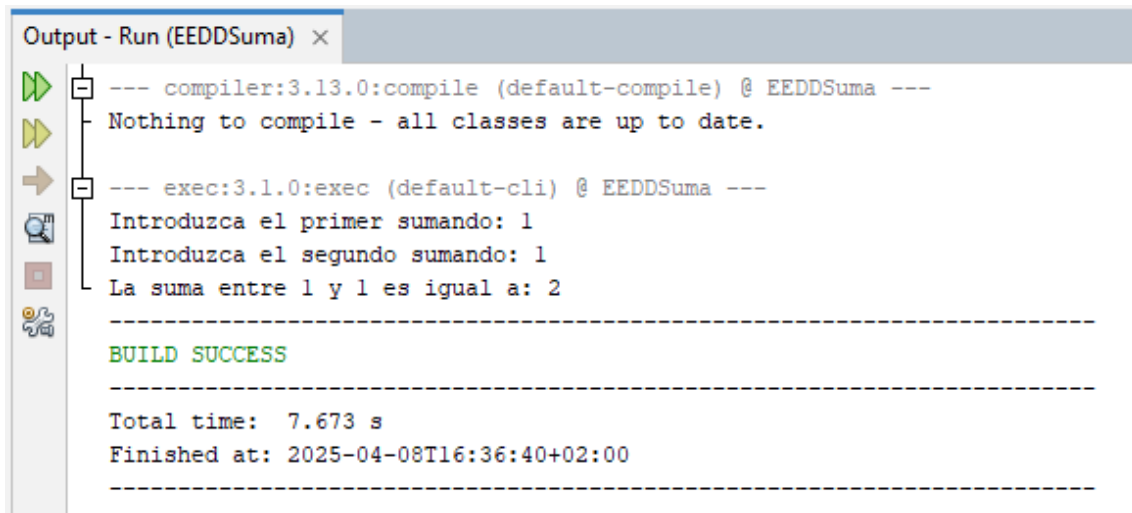
1. Dándole al botón de la flecha verde se compilan todos los archivos del proyecto java. Se mostrará en el cuadro de ejecución un mensaje de confirmación de compilación. En caso de fallo al compilar se mostrará un mensaje de error.

Ejecutar

Igual que en el apartado de compilar, dando al botón de “Run” de la barra de herramientas se ejecutan los programas después de ser compilados. También se puede ejecutar por pasos o, dicho de otra forma, depurar el código, para comprobar el valor de las variables y comprobar el funcionamiento de cada parte del código.



1. Igual que al compilar, dándole al botón de la flecha verde se ejecutara el proyecto después de ser compilado.



```
Output - Run (EEDDSuma) x
--- compiler:3.13.0:compile (default-compile) @ EEDDSuma ---
Nothing to compile - all classes are up to date.
--- exec:3.1.0:exec (default-cli) @ EEDDSuma ---
Introduzca el primer sumando: 1
Introduzca el segundo sumando: 1
La suma entre 1 y 1 es igual a: 2
-----
BUILD SUCCESS
-----
Total time: 7.673 s
Finished at: 2025-04-08T16:36:40+02:00
-----
```

2. Aquí aparece un ejemplo de ejecución de un programa que suma dos números recibidos por teclado y muestra por teclado un mensaje con el resultado.

Probar

Es la comprobación del funcionamiento de nuestro programa, realizando múltiples ejecuciones, abarcando todo el abanico de posibilidades. Sirve para advertir de fallos de ejecución, errores en el algoritmo que no sean de carácter sintáctico y para solventar dichos errores hasta que deje de haberlos. La herramienta principal en la comprobación del programa es la depuración.

Documentar

Consiste en la realización de un “manual de uso” de la aplicación. Toda aplicación debe tenerlo ya que, solo el programador sabe como funciona su código y que hace cada variable y, para que otro programador sepa que hace cada parte del código, se requiere un documento o comentarios en el propio código que expliquen que hace cada método, que devuelven y que argumentos necesita, si lanza o no alguna excepción, etc...

```
//Definición del objeto de la clase Scanner para recibir valores por teclado.
//Definición de dos variables de tipo entero inicializadas a 0.
//Mensaje informativo.
//Insercción de un valor recibido por teclado a la variable num1.
//Mensaje informativo.
//Insercción de un valor recibido por teclado a la variable num2.

//Mensaje informativo donde se calcula el resultado de la suma de ambas variables. |
```

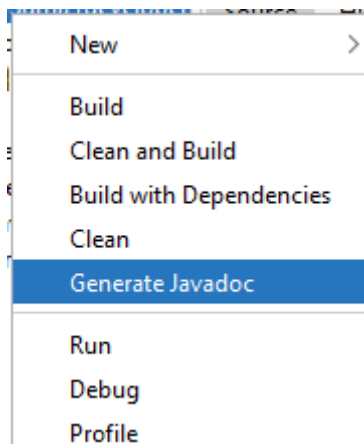
1. En este caso la documentación es básica al ser solo un archivo y no tener métodos.

```
/**
 * @param num1
 * @param num2
 * @return un entero con la suma de los numeros recibidos como parametros.
 */
private static int sumarNumeros(int num1, int num2){
    return num1+num2;
}
```

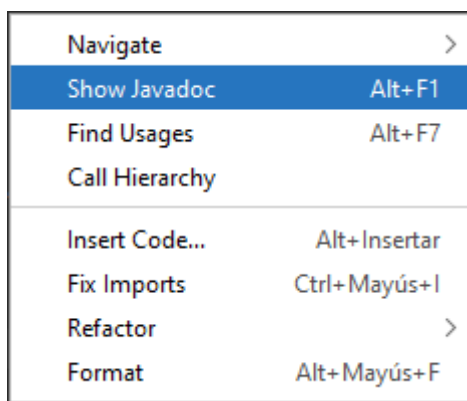
2. Este sería la documentación de un método privado que recibe dos parametros enteros y devuelve un entero con la suma de los dos parámetros.

Generar documentación

Es la automatización de la documentación. Para que sea más completa se necesita una documentación bien detallada del programa. Por tanto, mientras se desarrolla el programa, hay que ir comentando cada uno de los métodos.



1. Haciendo click derecho en el proyecto y seleccionando Generate Javadoc se genera automaticamente el Javadoc en forma de HTML.



2. Haciendo click derecho sobre cualquier zona blanca del editor de texto dentro de cualquier .java del proyecto y seleccionando Show Javadoc podremos ver el Javadoc generado.

Modelar

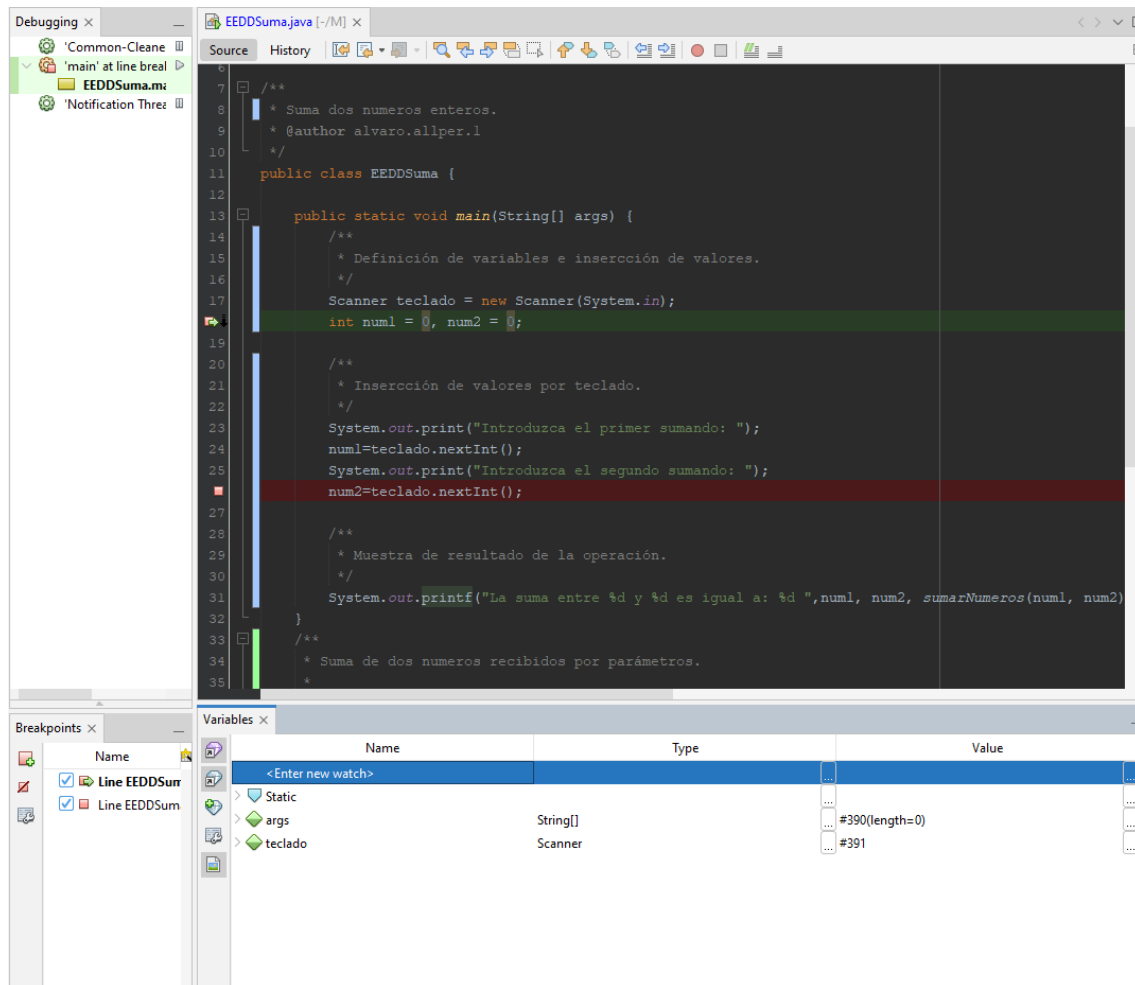
Consiste en diseñar la estructura del software antes de codificarlo, usando herramientas como diagramas UML para representar clases, relaciones y lógica del sistema. Es necesario la instalación de plugin.

Depurar

Sirve para ejecutar con interrupciones establecidas a lo largo del código y comprobar el funcionamiento del código y el valor de las variables en cada interrupción.

```
13 public static void main(String[] args) {
14     /**
15     * Definición de variables e insercción de valores.
16     */
17     Scanner teclado = new Scanner(System.in);
18     int num1 = 0, num2 = 0;
19
20     /**
21     * Insercción de valores por teclado.
22     */
23     System.out.print("Introduzca el primer sumando: ");
24     num1=teclado.nextInt();
25     System.out.print("Introduzca el segundo sumando: ");
26     num2=teclado.nextInt();
27
28     /**
29     * Muestra de resultado de la operación.
30     */
31     System.out.printf("La suma entre %d y %d es igual a: %d ",num1, num2, sumarNumeros(num1, num2));
32 }
```

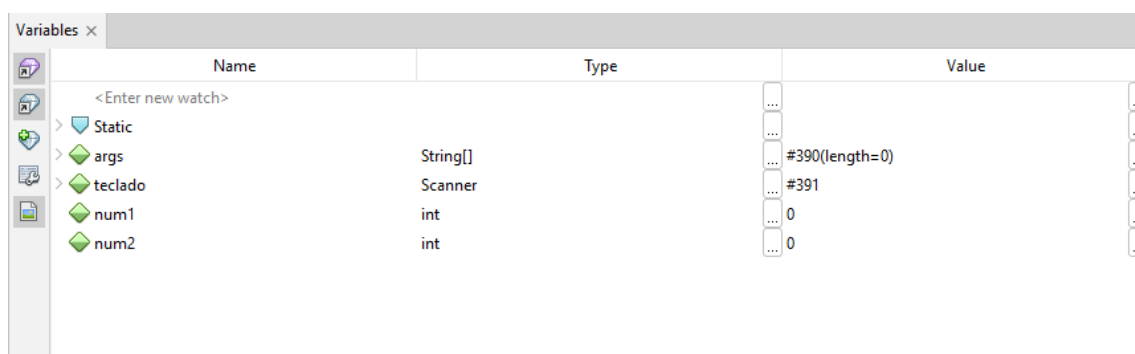
1. Primero escogeremos las líneas de código donde realizaremos las paradas de ejecución para comprobar el valor de las variables y el funcionamiento de cada parte del código.



- Primera parada antes de crear las variables num1 y num2.

Inspección de variables

Durante la depuración, comprobar el valor de las variables en cada interrupción.

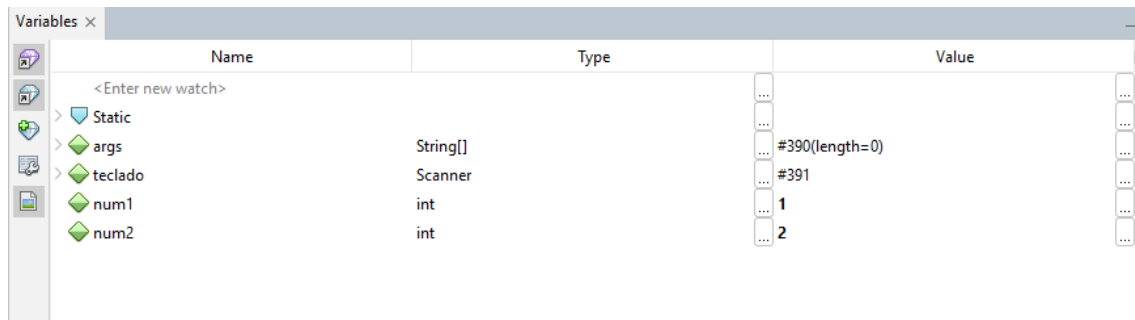


- En este momento las variables son creadas e inicializadas a 0.

```

| --- exec:3.1.0:exec (default-cil) @ K
| Introduzca el primer sumando: 1
| Introduzca el segundo sumando: 2

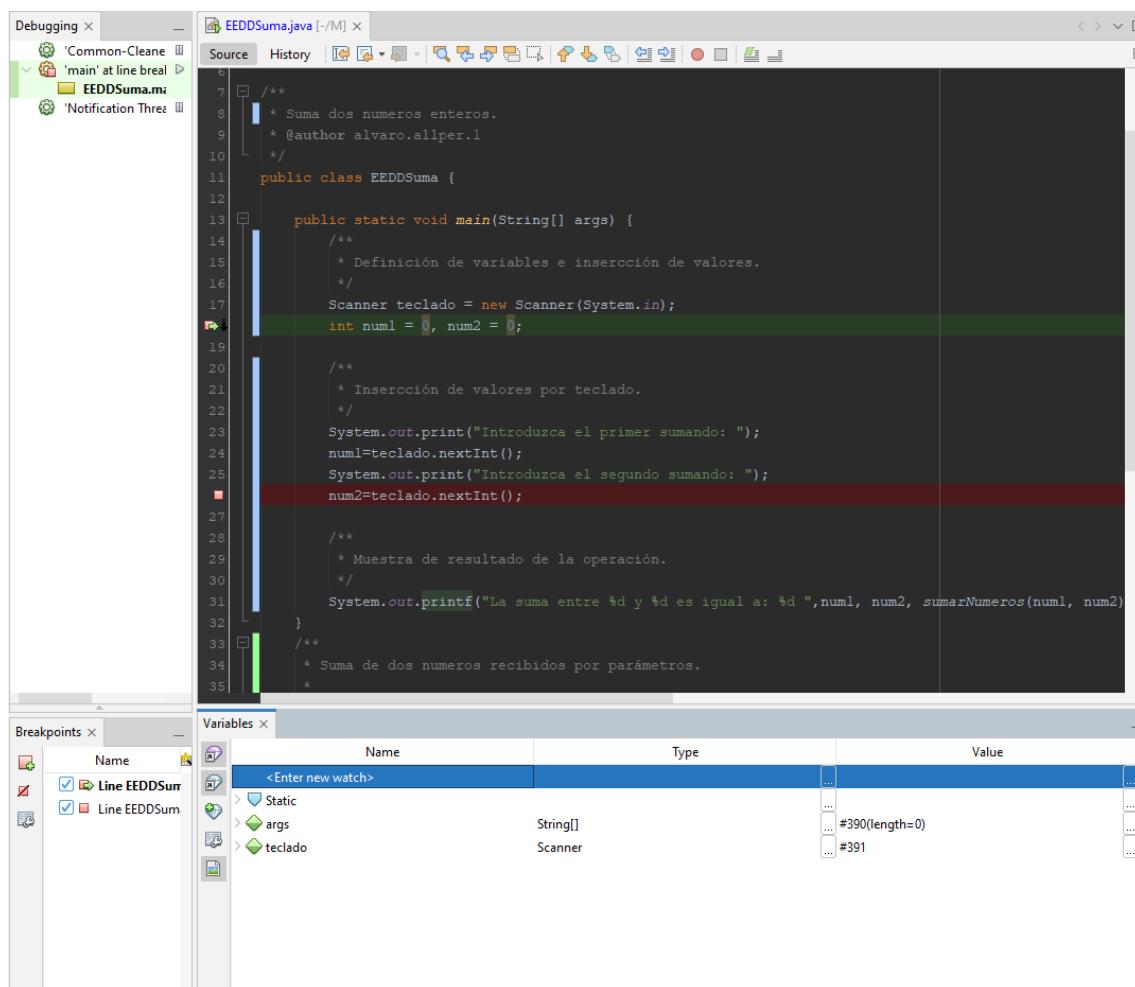
```



2. Aquí estan los valores establecidos por teclado.

Ejecución paso a paso

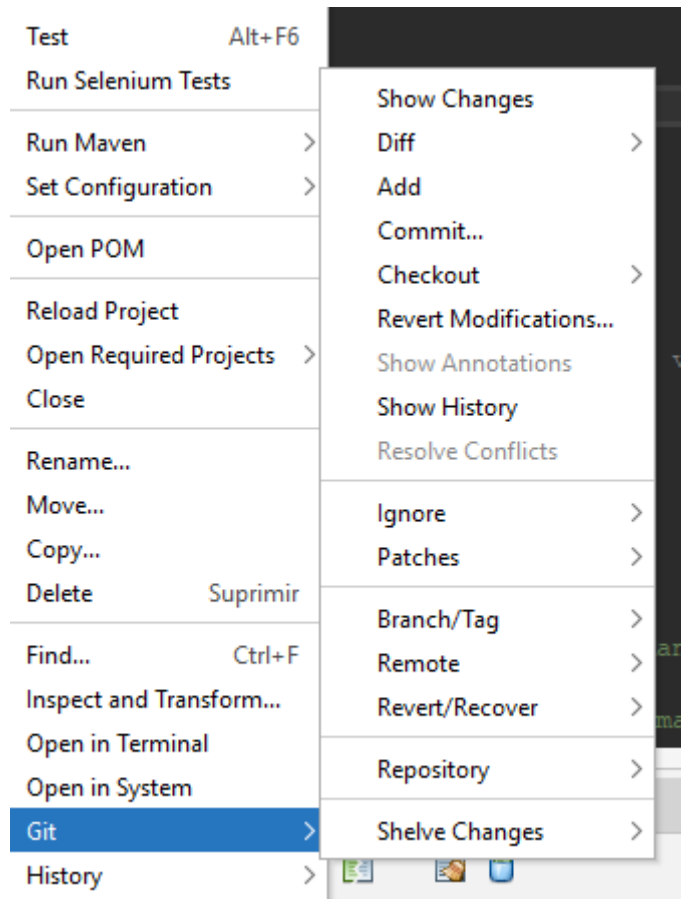
Durante la depuración, ejecutar el programa con puntos de parada para comprobar si hay fallos de ejecución y, en caso de haberlo, poder saber donde sucede para poder solucionarlo.



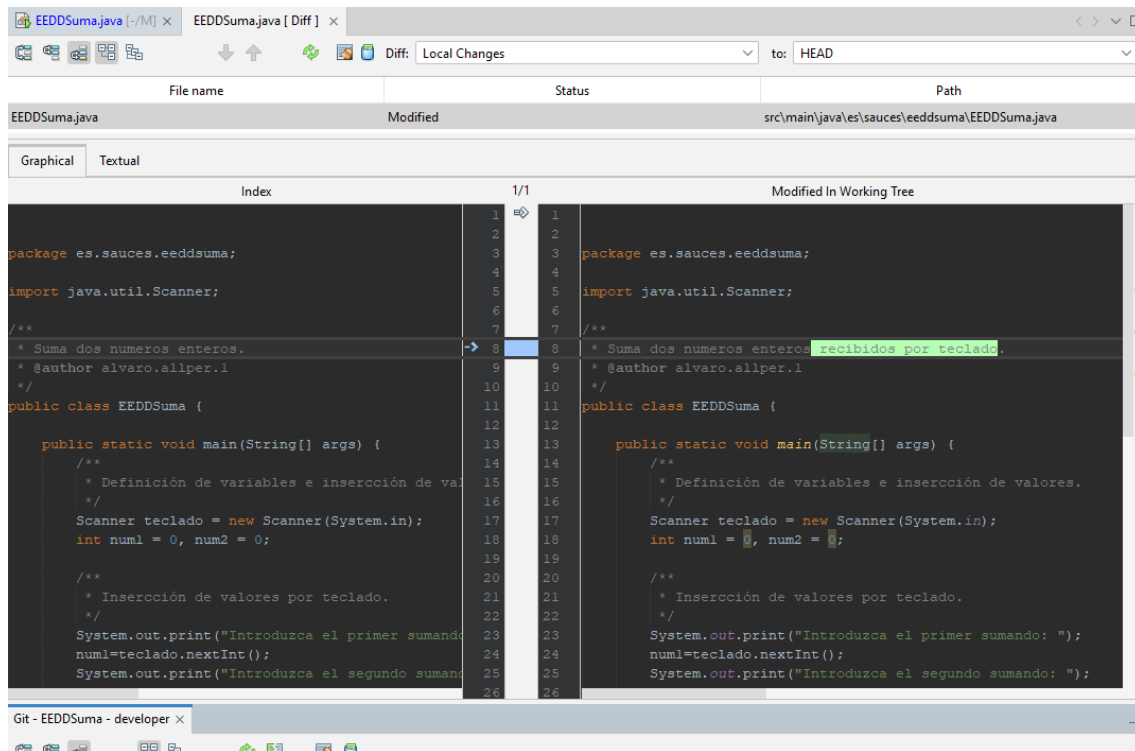
1. En la zona de la numeración de las líneas de código podemos seleccionar las zonas de parada que se marcarán con un cuadrado rojo y subrayando la línea con rojo.

Comparar código

Sirve para ver las diferencias entre dos archivos o dos versiones de un mismo archivo. Usando Tools>Diff para revisar cambios línea por línea. Ideal para controlar versiones y detectar modificaciones.



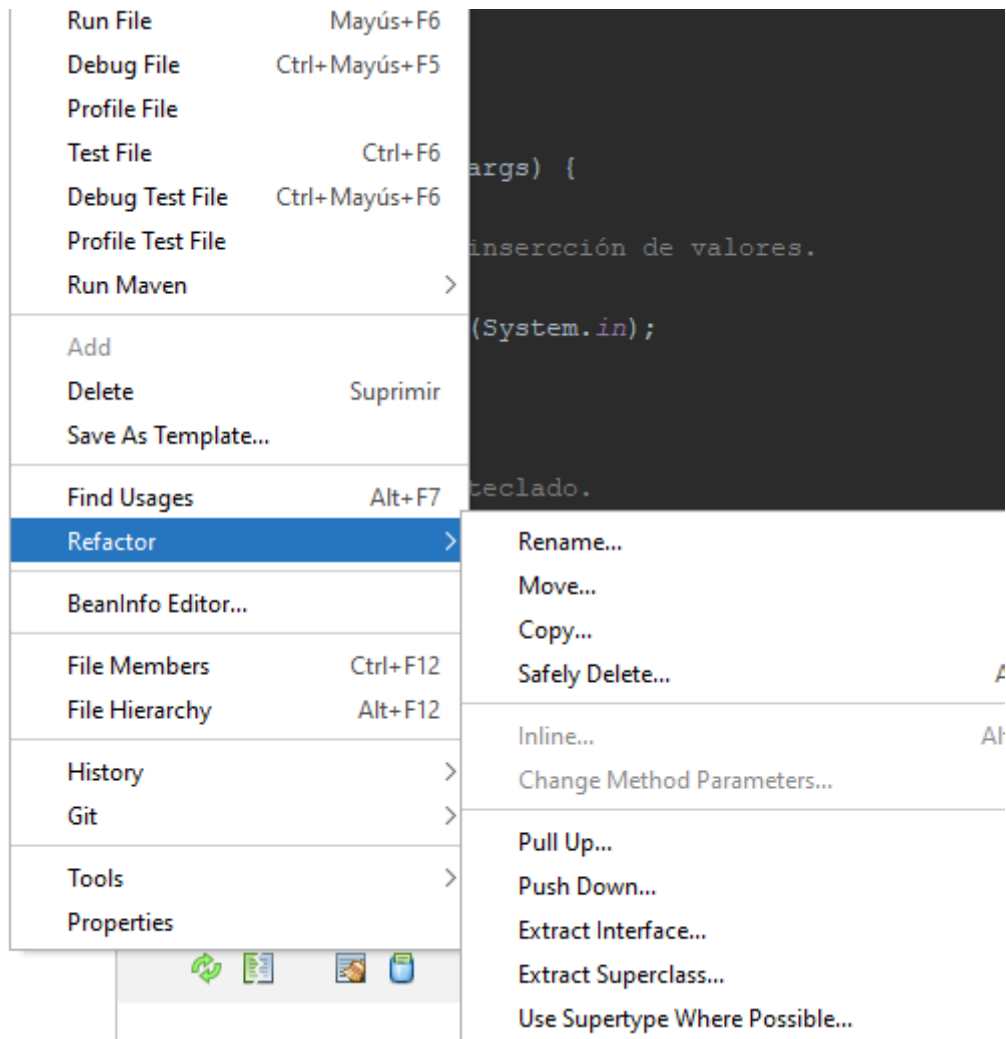
1. En caso de cambio en un archivo y antes de hacer el commit en el repositorio podemos observar los cambios entre el archivo cambiado y la última versión del commit. Haciendo click derecho en el proyecto y llenando a Git>Show Changes y seleccionando el archivo cambiando.



2. Aquí tenemos la pantalla donde se ven los cambios realizados.

Refactorizar

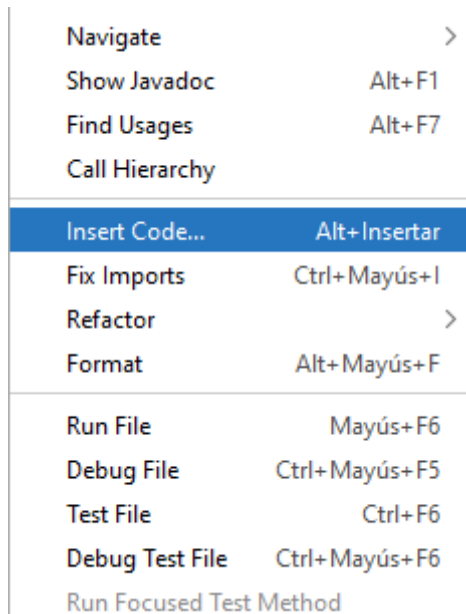
Sirve para modificar errores de sintaxis en el código o para modificar el nombre del archivo y a su vez modificar el nombre de la clase “main”.



1. Haciendo click derecho en el archivo que queremos y llendo a Refactor podemos cambiar de una forma segura el nombre del archivo. Tambien podemos mover el archivo copiarlo y borrarlo de forma segura.

Generar código

Haciendo click derecho sobre el código y usando el apartado "insert code" podemos implementar código de forma automática.



1. Haciendo click derecho sobre una zona sin texto en el editor de texto podremos seleccionar Insert Code... para insertar de forma automática partes del código como los Setter y los Getter o el toString de cualquier clase. También podremos modificar las partes del código generadas.

Ingeniería inversa

Es la generación de un diagrama UML usando el código. Se dice que es “inversa” porque lo normal es primero tener el diagrama y desarrollar código.

Administrar base de datos

NetBeans permite conectar, consultar y modificar bases de datos directamente desde el IDE. Se hace desde la pestaña “Services” donde se conecta a un servidor de base de datos. También se puede ejecutar consultas SQL y ver tablas.

Ejecutar script sobre la base de datos

Desde el mismo apartado “Services”, se pueden ejecutar comandos de creación de tablas e inserción de datos seleccionando “Execute Command” donde se escribe y ejecuta el script.