



JAVA

PROGRAMACIÓN
Iº DAM - Iº DAW

TEMA 01:

**INTRODUCCIÓN AL LENGUAJE
DE PROGRAMACIÓN JAVA**



ÍNDICE

1. ¿Qué es JAVA?
2. ¿Por dónde empiezo?
3. Mi primer programa en JAVA
4. Documentar correctamente un archivo .java?
5. Ejercicios de consolidación



JAVA

INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN JAVA

I. ¿Qué es JAVA?

I.- ¿Qué es JAVA?

- JAVA es un lenguaje de programación de alto nivel con el que se pueden escribir tanto programas convencionales como para Internet.
- La ventaja de Java sobre otros lenguajes de programación es que los programas creados son independientes de la plataforma, ya que pueden transportarse a cualquier plataforma que tenga instalada una máquina virtual Java y ejecutarse.

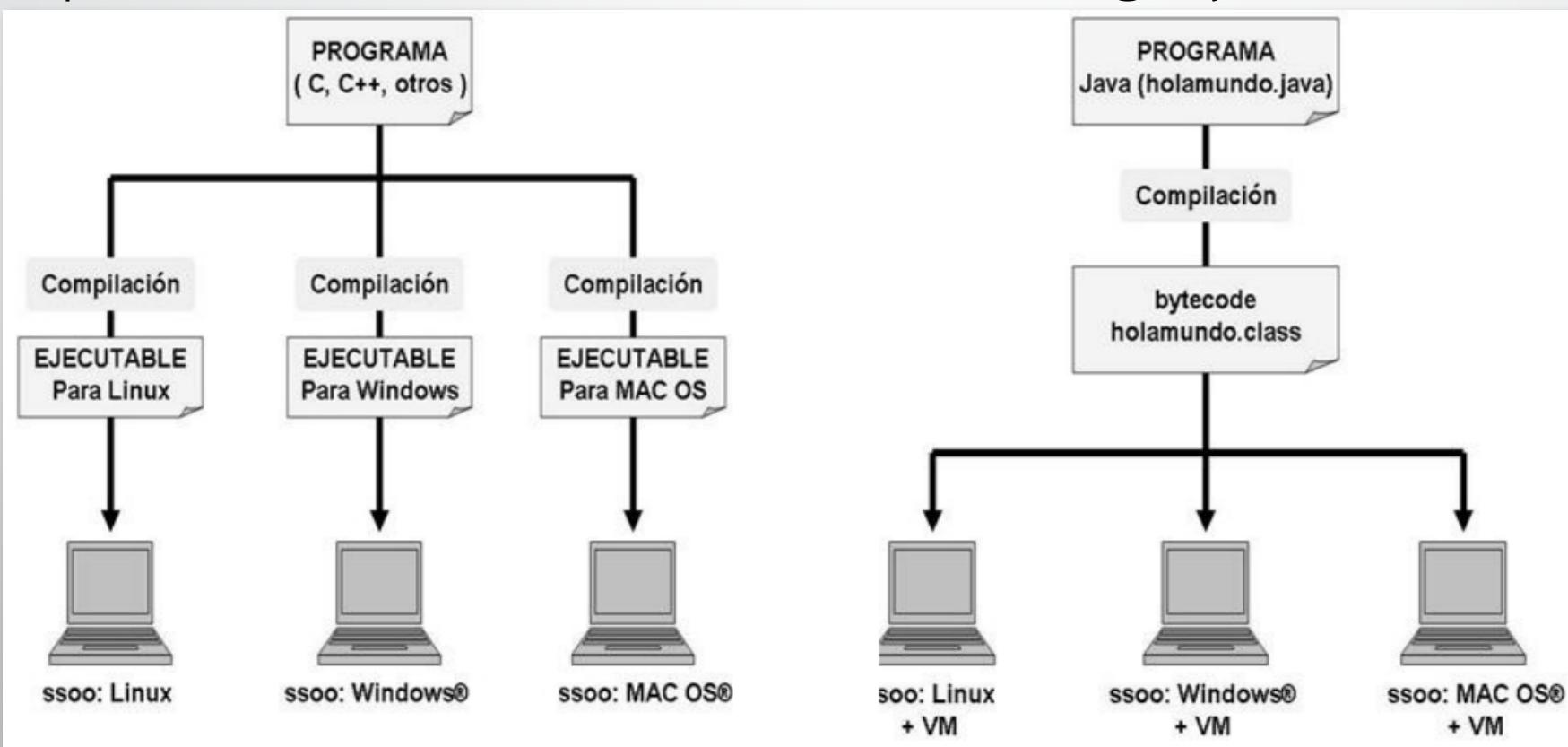
I.- ¿Qué es JAVA?

- Para conseguir esto previamente el código fuente en Java se tiene que precompilar generando un código previo (que no es directamente ejecutable) conocido como bytecode o J-code. Ese código (generado normalmente en archivos con extensión class) es el que es ejecutado por la máquina virtual de Java que interpreta las instrucciones de los bytecodes.



I.- ¿Qué es JAVA?

- Comparativa de JAVA frente a otros lenguajes:



I.- ¿Qué es JAVA?



- ¡¡ Ojo!! →
- Una de las confusiones habituales la provoca el parecido nombre que tienen estos dos lenguajes. Sin embargo no tienen nada que ver entre sí:
 - Java es un lenguaje completo creado por SUN que permite realizar todo tipo de aplicaciones, tanto de escritorio como aplicaciones WEB (applets).
 - JavaScript es código que está inmerso en una página web y cuya finalidad es mejorar el dinamismo de las páginas web. Creado por Netscape.

JAVA *isto*
JAVASCRIPT
as HAM *isto*
HAMSTER



ILLUSTRATION BY SEGUÉ TECHNOLOGIES

JAVA

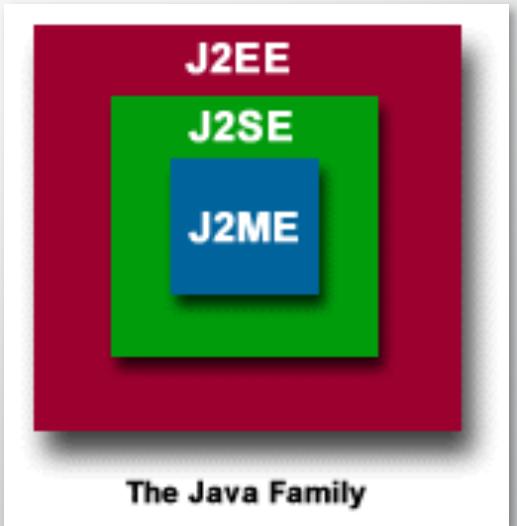
INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN JAVA

2. ¿Por dónde empiezo?



2.- ¿Por dónde empiezo?

- **Primero:** Elegimos una Plataforma de Java:
 - **Java SE:** Se denomina así al entorno de Sun relacionado con la creación de aplicaciones y applets en lenguaje Java. Este utilizaremos nosotros en clase.
 - **Java EE:** Pensada para la creación de aplicaciones Java empresariales y del lado del servidor.
 - **Java ME:** Pensada para la creación de aplicaciones Java para dispositivos móviles.



2.- ¿Por dónde empiezo?

- **Segundo:** Descargamos el Kit de Desarrollo Java (JDK)
 - Una vez que escribamos un programa en Java, hacen falta los programas que realizan el precompilado y la interpretación del código. Hay entornos que permiten la creación de los bytecodes y que incluyen herramientas con capacidad de ejecutar aplicaciones de todo tipo. El más famoso (que además es gratuito) es el **Java Developer Kit (JDK)** de Oracle, que se encuentra disponible en la dirección <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

2.- ¿Por dónde empiezo?

- El JDK no contiene ninguna herramienta gráfica para el desarrollo de programas. Solo consta de los mínimos componentes necesarios para ejecutar una aplicación Java, como son la máquina virtual y las librerías de clases, además de las siguientes herramientas de consola:
 - **java.** Es la máquina virtual de Java.
 - **javac.** Es el compilador de Java. Con él es posible compilar las clases que desarrollemos.
 - **javap.** Es un desensamblador de clases.
 - **jdb.** El depurador de consola de Java.
 - **javadoc.** Es el generador de documentación.
 - **appletviewer.** Visor de Applets.



2.- ¿Por dónde empiezo?

- **Tercero:** Elegiríamos uno de los entornos de desarrollo (IDE) Java y nos lo descargaríamos. Este tercer punto sería opcional pero muy recomendable.
 - El código en Java se puede escribir en cualquier editor de texto, y para compilar el código en bytecodes, sólo hace falta descargar la versión del JDK deseada.
 - La escritura y compilación de programas utilizando “a pelo” el JDK es un poco incomoda, por lo que numerosas empresas fabrican sus propios entornos de edición, algunos incluso incluyen el compilador (Microsoft), y otras utilizan el propio JDK de Oracle.

2.- ¿Por dónde empiezo?

- Ejemplos: NetBeans, Eclipse, JBuilder, JCreator, JDeveloper, Microsoft Visual J++, Sun ONE Studio....



Apache
NetBeans IDE



Visual Studio®



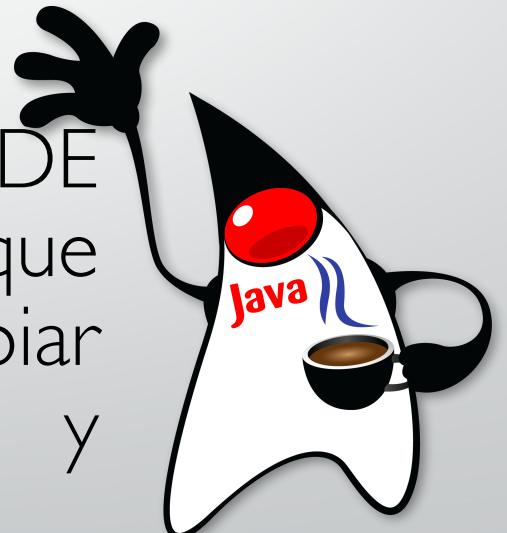
JAVA

INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN JAVA

3. Mi primer programa en JAVA

3.- Mi primer programa en JAVA

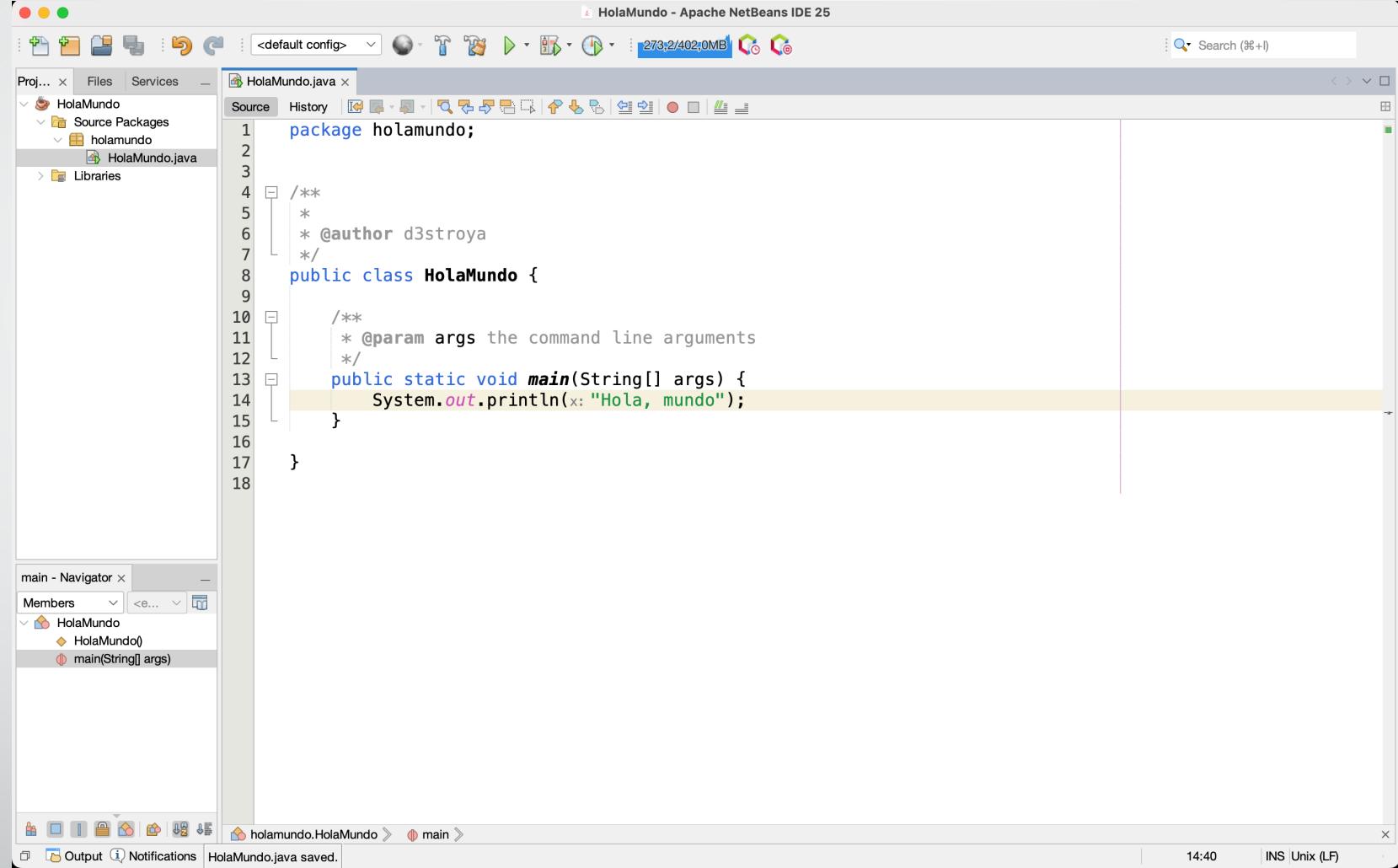
- El código fuente JAVA lo podemos escribir en un simple documento de texto del bloc de notas poniéndole extensión .java, por ejemplo, HolaMundo.java.
- En este primer ejemplo utilizaremos el IDE instalado en nuestros ordenadores, por lo que lo único que tendremos que hacer será copiar todo el código fuente en nuestro IDE y ejecutarlo (pulsando “Run Project”).



3.- Mi primer programa en JAVA

- En la primera línea se declara la clase de objetos HolaMundo ya que el esqueleto de cualquier aplicación Java se basa en la definición de una clase.
- A continuación se escribe el cuerpo de la clase que va entre los caracteres { y }.
- Las líneas encerradas entre /* y */ son comentarios que ayudan a entender un programa cuando se lee. Son ignorados por el compilador.
- Por último se escribe el método principal e imprescindible llamado main. Fíjate que tiene el modificador () después de su nombre y que su bloque de código va entre llaves { }.

El código fuente del programa HolaMundo.java, lo único que hace es mostrar por la pantalla del ordenador el mensaje Hola mundo



The screenshot shows the Apache NetBeans IDE interface with the following details:

- Title Bar:** HolaMundo - Apache NetBeans IDE 25
- Toolbar:** Standard NetBeans toolbar with icons for file operations, search, and project navigation.
- Project Explorer:** Shows a project named "HolaMundo" with a "Source Packages" folder containing "holamundo" and "HolaMundo.java".
- Code Editor:** Displays the Java code for "HolaMundo.java".

```
1 package holamundo;
2
3 /**
4  * 
5  * @author d3stroya
6  */
7 public class HolaMundo {
8
9     /**
10      * @param args the command line arguments
11     */
12    public static void main(String[] args) {
13        System.out.println("Hola, mundo");
14    }
15
16
17 }
18
```
- Navigator:** Shows the members of the "HolaMundo" class: "HolaMundo", "HolaMundo()", and "main(String[] args)".
- Status Bar:** Shows the message "HolaMundo.java saved." and the system information "14:40 INS Unix (LF)".



JAVA

INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN JAVA

4. Documentar correctamente un archivo .java

4.- Documentar correctamente un archivo java

- A la hora de realizar un proyecto en Java, donde pueden aparecer cientos de objetos y de clases, es muy importante documentar correctamente el código para luego poder presentar de manera adecuada los archivos que componen el programa.
- Documentar sirve también para que una persona que lea el código de un programa le resulte más sencillo entender lo que hace el programa. Esto es fundamental en empresas y grupos de trabajo donde la gente de tu alrededor necesitará entender tu código para luego ellos, por ejemplo, añadir nuevos módulos a tu programa.

4.- Documentar correctamente un archivo java

- Incluso, para el propio autor del programa, en el caso de retomarlo pasado un tiempo, le viene bien que todo esté documentado para recordar el funcionamiento de su programa con rapidez.

4.- Documentar correctamente un archivo java

- Javadoc es una herramienta muy interesante del kit de desarrollo de Java para generar automáticamente documentación Java. Genera documentación para paquetes completos o para archivos java.
- Su sintaxis básica es: **javadoc archivo.java** (o paquete)



4.- Documentar correctamente un archivo java

- El funcionamiento es el siguiente:
 - Los comentarios javadoc comienzan con el símbolo `/**` y terminan con `*/` y serán utilizados por los programas de generación de documentación javadoc.
 - Cada línea javadoc se inicia con un símbolo de asterisco `*`. Dentro se puede incluir cualquier texto. (Incluso se pueden utilizar códigos HTML)
 - En el código javadoc se pueden usar etiquetas especiales y comienzan con el símbolo `@`. Pueden ser:

4.- Documentar correctamente un archivo java

- **@author.** Tras esta etiqueta se indica el autor del documento.
- **@version.** Tras la cual sigue el número de versión de la aplicación
- **@see.** Tras esta palabra se indica una referencia a otro código Java relacionado con éste.
- **@since.** Indica desde cuándo esta disponible este código
- **@deprecated.** Palabra a la que no sigue ningún otro texto en la línea y que indica que esta clase o método está obsoleta u obsoleto.

4.- Documentar correctamente un archivo java

- **@throws.** Indica las excepciones que pueden lanzarse en ese código.
- **@param.** Le sigue una descripción de los parámetros que requiere el código para su utilización. Cada parámetro se coloca en una etiqueta **@param** distinta, por lo que puede haber varios **@param** para el mismo método.
- **@return.** Tras esta palabra se describe los valores que devuelve el código.

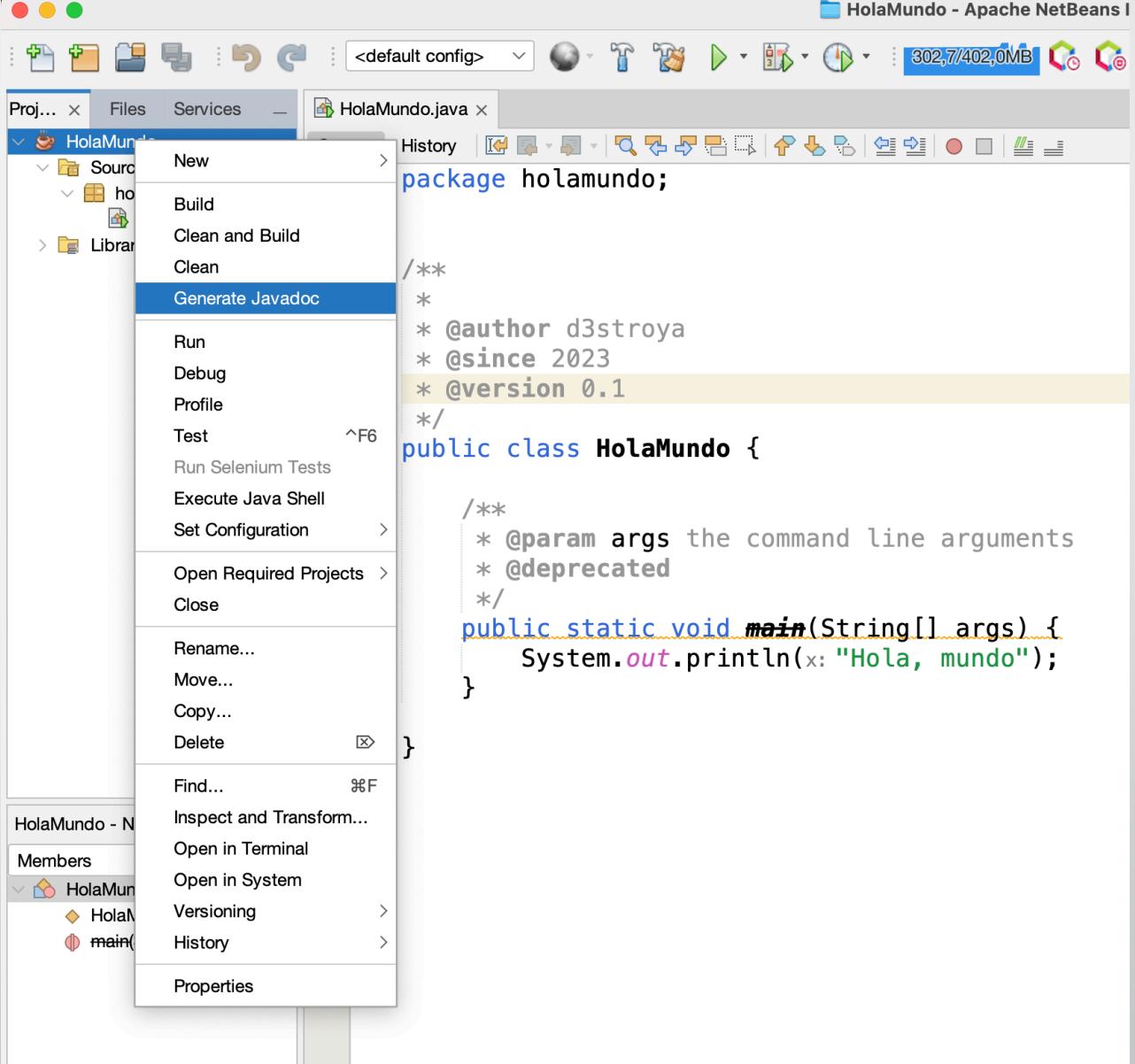
4.- Documentar correctamente un archivo java

- El código javadoc hay que colocarlo en tres sitios distintos dentro del código java de la aplicación:
 - I. Al principio del código de la clase (antes de cualquier código Java). En esta zona se colocan comentarios generales sobre la clase que se crea mediante el código Java. Dentro de estos comentarios se pueden utilizar las etiquetas: **@author, @version, @see, @since y @deprecated**

4.- Documentar correctamente un archivo java

2. Delante de cada método. Los métodos describen las cosas que puede realizar una clase. Delante de cada método los comentarios javadoc se usan para describir al método en concreto. Además de los comentarios, en esta zona se pueden incluir las etiquetas: **@see, @param, @exception, @return, @since y @deprecated**.
3. Delante de cada atributo. Se describe para qué sirve cada atributo en cada clase. Puede poseer las etiquetas: **@since y @deprecated**

Para generar la documentación, haz clic derecho en el proyecto (ícono de la taza) y selecciona la opción Generate Javadoc.



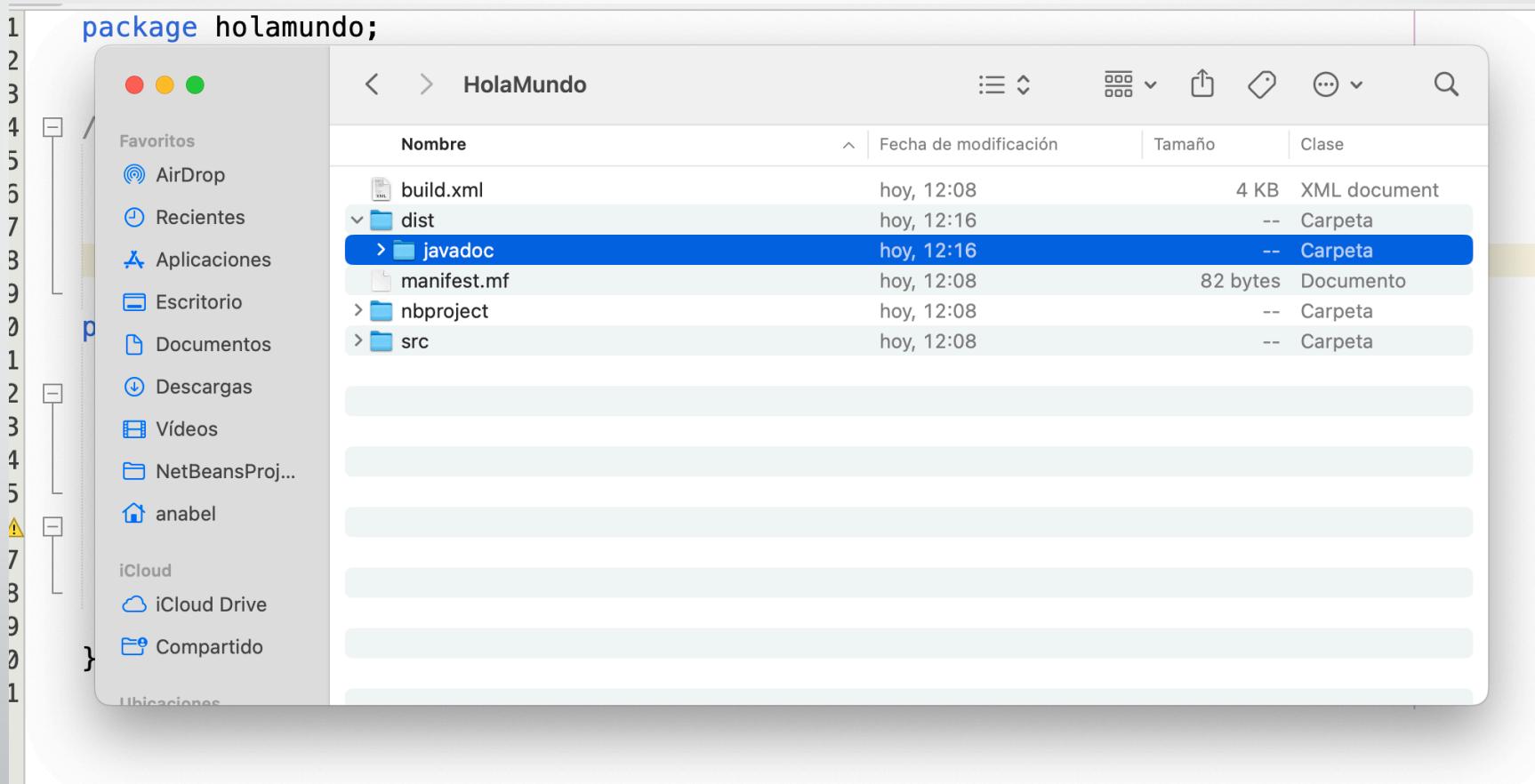
The screenshot shows the Apache NetBeans IDE interface. A context menu is open over the 'HolaMundo' project in the Project Explorer. The 'Generate Javadoc' option is selected. The code editor on the right displays a Java class 'HolaMundo' with Javadoc annotations and a main method. The status bar at the bottom indicates '302,7/402,0MB'.

```
package holamundo;

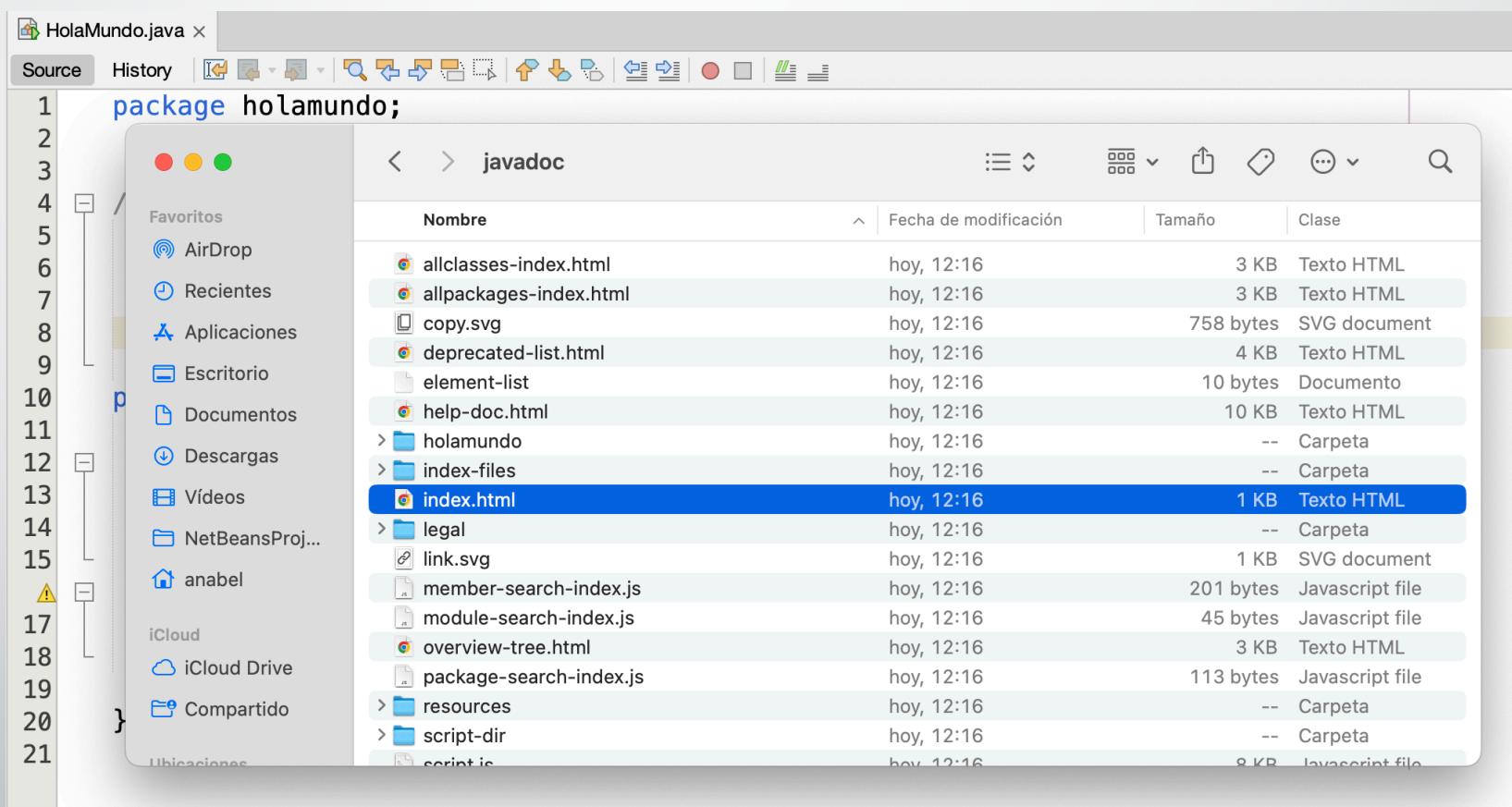
/**
 * @author d3stroya
 * @since 2023
 * @version 0.1
 */
public class HolaMundo {

    /**
     * @param args the command line arguments
     * @deprecated
     */
    public static void main(String[] args) {
        System.out.println("Hola, mundo");
    }
}
```

En la carpeta de tu proyecto, dentro del directorio dist, aparecerá una nueva carpeta llamada javadoc.



Dentro de la carpeta javadoc, verás el archivo index.html. Ábrelo para ver la página web de la documentación.



API de Java

- Java tiene su propia documentación, que puedes consultar siempre que lo necesites en esta [página web](#) o buscando en Google “API Java”.

JAVA

INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN JAVA

5. Ejercicios de consolidación

EJERCICIOS

- **Ejercicio 01.- (OPTATIVO)** Escribe en Netbeans el siguiente programa llamado **CAritmetica.java**, compílalo y ejecútalo en el propio IDE.

EJERCICIOS



```
public class CAritmetica {  
    /**  
     * @author Oscar Laguna Garcia  
     * Operaciones Aritméticas  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        int dato1; //Declaro la variable entera dato1  
        int dato2, resultado; //Declaro, a la vez, dos variables enteras: dato2 y resultado  
  
        dato1 = 20; //Asigno el valor 20 a la variable dato1  
        dato2 = 10;  
  
        //Suma  
        resultado = dato1 + dato2; //Guardo la suma de las dos variables en la variable resultado  
        System.out.println(dato1 + " + " + dato2 + " = " + resultado); /* El método println escribe  
           por pantalla tanto el valor de las variables así como las cadenas que están entre  
           comillas. Para unir los 5 elementos se ha utilizado el operador + */  
  
        //Resta  
        resultado = dato1 - dato2;  
        System.out.println(dato1 + " - " + dato2 + " = " + resultado);  
  
        //Producto  
        resultado = dato1 * dato2;  
        System.out.println(dato1 + " * " + dato2 + " = " + resultado);  
  
        //Cociente  
        resultado = dato1 / dato2;  
        System.out.println(dato1 + " / " + dato2 + " = " + resultado);  
    }  
}
```

EJERCICIOS

- Ejercicio 02.- (**OBLIGATORIO**) Crea un programa similar al del ejercicio anterior (CAritmetica.java), el cual realizará las operaciones de sumar, restar y multiplicar con 3 datos: dato1, dato2 y dato3. Guárdalo con el nombre CAritmetica2.java, compílalo y ejecútalo.

EJERCICIOS

- **Ejercicio 03.- (OPTATIVO)** Escribe una aplicación que visualice en pantalla los siguientes mensajes: (cada uno en una línea distinta)

Bienvenido al mundo de Java.

El camino será duro, pero la recompensa merecerá la pena.