**Clases en Java**

Para salvar unas dudas que surgieron durante las clases les armo esta pequeña introducción sobre lo que son las clases y métodos, es un tema que todavía no abordaremos, pero consideramos que es necesario que lo sepan para trabajar en el ambiente de Java.

**¿Qué es una Clase?**

Las **clases en Java** son plantillas para la creación de objetos, en lo que se conoce como programación orientada a objetos, considerada uno de los principales paradigmas de desarrollo de software en la actualidad.

La **clase es la esencia de Java**. Es la base sobre la cual se construye todo el [lenguaje Java](https://www.tokioschool.com/noticias/curiosidades-de-java/) porque la clase define la naturaleza de un objeto. Como tal, la **clase** forma la base para la programación orientada a objetos en Java.

Dentro de una **clase** se definen los datos y el código que actúa sobre esos datos. El código está contenido en métodos. Tanto las clases, como los objetos y los métodos son fundamentales para [Java](https://www.tokioschool.com/formaciones/cursos-programacion/java/). Tener una comprensión básica de estas características te permitirá escribir programas más sofisticados y comprender mejor el **lenguaje de programación Java.**

**Estructura de un programa - aplicación en Java**

Para crear programas en Java hay que tener en cuenta que toda implementación que se realice se efectuará encapsulada en una clase. Un programa aplicación fuente en Java se podría considerar formado por las siguientes partes:

* Una sentencia de paquete (package) que también puede ser omitida.
* Una, ninguna o varias sentencias de importación (import).
* Una serie de comentarios opcionales colocados en diversos lugares del pro-
* Declaraciones de las clases privadas deseadas; puede no haber ninguna.
* Una declaración de clase pública.

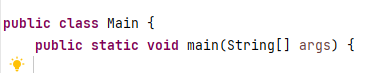
A su vez una declaración de clase comenzará con la sentencia class y podrá

contener:

* Declaraciones de variables de la clase (estáticas).
* Declaraciones de variables de instancia.
* Definiciones de constructores.
* Definiciones de métodos.

Dado que una clase es un modelo y las instancias son los objetos de esa clase, a las variables de instancia se las denomina así porque cada objeto contendrá una copia propia de dichas variables, de forma que los datos de un objeto se encontrarán separados de los de otro objeto, utilizándose los métodos para la modificación

de los valores de las variables de instancia. En consecuencia, un programa muy sencillo que se puede crear en Java es:



formado exclusivamente por una declaración de clase pública

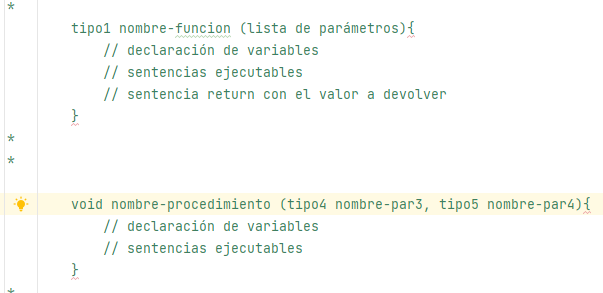
Compilación: javac Main.java

Ejecución: java Main

Las clases de un programa contienen métodos, interactúan entre sí y no necesitan contener un método main ( ) , pero éste sí será necesario en la clase que constituya el punto de partida del programa. Tenga también en cuenta que las *applets* no utilizan el método main ( ) .

La *declaración* de métodos en una clase se puede efectuar en cualquier orden y cada una de ellas se compone por una cabecera y un cuerpo. La cabecera del método debe contener su nombre, una lista de parámetros y el tipo de resultado. Se especificará void cuando el método no devuelva resultados. En la implementación del método, cuando éste no haya sido declarado void, se utilizará la instrucción return para devolver un valor al punto de llamada del método. La lista de parámetros consistirá en cero o más parámetros formales cada uno de ellos precedido por su tipo y separados por comas.

Cuando se llama a un método los parámetros actuales se asignan a los parámetros formales correspondientes. Entre los parámetros actuales (los de la llamada) y formales (los de la declaración) debe existir concordancia en cuanto a número, tipo y orden. Formatos de métodos pueden ser los siguientes:



En el primer ejemplo, tipo1 representa el tipo de dato devuelto por 'el método; cuando el método no devuelve ningún valor se especificará void como tipo devuelto. La lista de parámetros es una secuencia de parejas tipo-identificador separadas por comas. Observe el formato expuesto en el segundo ejemplo.

En Java es posible agrupar sentencias simples, encerrándolas entre una pareja de llaves para formar un bloque o sentencia compuesta; las variables declaradas dentro de un bloque sólo son válidas en dicho bloque y, si éste tuviera otros anidados, en los interiores a él. En un programa Java se podrán declarar variables tanto dentro como fuera de los métodos.

Las variables declaradas dentro del cuerpo de un método se crean cuando se ejecuta el cuerpo del método y desaparecen después. Las variables que se declaran fuera del cuerpo de un método son globales a la clase. Las que se declaran como final y static son, en realidad, constantes.

**¿Qué es el método Main?**

El método main es el punto de entrada para todos los programas Java. Es el método que se ejecuta cuando se inicia un programa Java.

El método main debe tener la siguiente firma:



**public**: El método main debe ser público para que pueda ser invocado desde cualquier parte.

**static**: El método main debe ser estático para que pueda ser invocado sin crear una instancia de la clase.

**void**: El método main no devuelve ningún valor.

**main**: El nombre del método main debe ser "main".

**String[] args**: El parámetro args es un arreglo de cadenas que representa los argumentos que se pasan al programa al ejecutarlo.

El método main puede contener cualquier código que desee ejecutar el programa. Por ejemplo, el código puede crear objetos, llamar a otros métodos, leer o escribir datos, o realizar cualquier otra tarea.

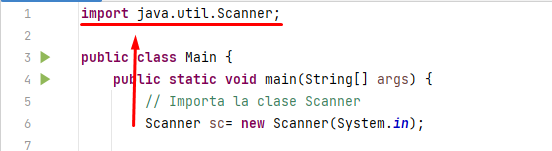
Pero ¿Por qué el método Main es void? El método main es void porque no devuelve ningún valor. Esto se debe a que el método main es el punto de entrada para todos los programas Java, y no tiene sentido que devuelva un valor, ya que no hay ningún código anterior que pueda hacer algo con ese valor.

Si el método main devolviera un valor, ese valor tendría que ser utilizado por algún otro código. Sin embargo, como el método main es el primer código que se ejecuta, no hay ningún otro código que esté disponible para utilizar ese valor.

¿Por qué los imports van fuera de la Clase?

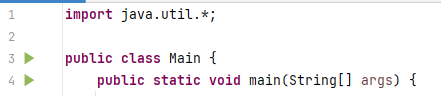
El import va fuera de la clase en Java porque es una declaración de alcance global. Esto significa que afecta a todas las clases del archivo de código fuente. Si el import estuviera dentro de la clase, solo afectaría a esa clase. Esto podría causar confusiones, ya que sería difícil saber qué clases están disponibles en un programa.

Además, el import va fuera de la clase para facilitar la organización del código. Al colocar el import al principio del archivo, es fácil ver qué clases están disponibles en un vistazo.



En este ejemplo, el import está en la primera línea del archivo. Esto indica que la clase Scanner está disponible para todas las clases en el archivo.

También es posible utilizar el asterisco (\*) para importar todas las clases de un paquete. Esto puede ser útil si se está utilizando un paquete de clases grande.



Bibliografía:

JAVA 2 - MANUAL DE PROGRAMACIÓN - Luis Joyanes Aguilar - Matilde Fernandez Azuela

https://www.tokioschool.com/noticias/que-es-clase-java/