**Explicación De La Sobrecarga En Dart**

Jordy Sneider Arias Ramírez, Daniela García León, Camilo Guapacha y Jhoan Andres Hernandez posada Servicio Nacional de Aprendizaje, Pereira

Servicio Nacional De Aprendizaje

Análisis y Desarrollo de Software

Edison Sandoval

Pereira, Colombia

31 de Junio de 2024

**Resumen**

Esta documentaciòn trata acerca del uso de la Sobrecarga de métodos y operadores; Resuelve dudas acerca del cómo usarlos y una breve descripciòn acerca del funcionamiento de cada uno, así mismo como algunas aclaraciones, notas y varias capturas de pantalla que ayudarán al lector a comprender el uso de cada uno los métodos y un tema extra que se llama “Nulos” a pesar de no estar explícitamente en el tema, será brevemente explicado debido a su constante aparecimiento en el tema de Parámetros Opcionales Posicionales y Parámetros Nombrados; A su vez tendrá unos ejemplos básicos antes de iniciar el tema principal de “Sobrecarga de Métodos”.

**Introducción**

Este trabajo es una exposición que pretende enseñar y explicar el uso de la sobrecarga de métodos, operadores y constructores. Esto quiere decir que buscará explicar tanto la aplicación de cada una de las sobrecargas como añadir un par de aclaraciones de cómo usarlas correctamente.

*Tabla de contenido*

# 

[**Variables Nulas 5**](#_r8ooaf20g4mv)

[1.1 Signo ?? 5](#_z58k40oq5c5r)

[1.2 Signo ! 6](#_3l1rx4ko87x4)

[**Sobrecarga de Métodos en Dart 7**](#_qq3a89kmbca0)

**1.0 Sobrecarga de constructores 8**

[1.1 Parámetros Opcionales Posicionales :](#_g8rtha3l0t50) 9

[1.2 Parámetros Nombrados:](#_w5gl441bqg1l) 10

[**Sobrecarga de operadores 13**](#_7vj53wl3laib)

[1.1 Sobrecarga del operador “+” 13](#_7thdihpkkayx)

[1.2 Sobrecarga del operador ”-” 14](#_g2vc3msi1hju)

[1.3 Sobrecarga del operador “\*” 14](#_bx960po1alz0)

[1.3 Sobrecarga del operador “/” 15](#_alnm4olgf01b)

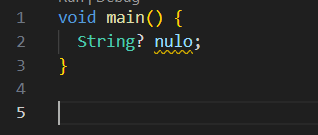
[**Bibliografía. 17**](#_ohzio3wd64a1)

# 

# **Variables Nulas**

Dart posee “Null safety”, quiere decir que previene errores al momento de compilación que pueden ser por una intención involuntaria al acceder al set de una variable.

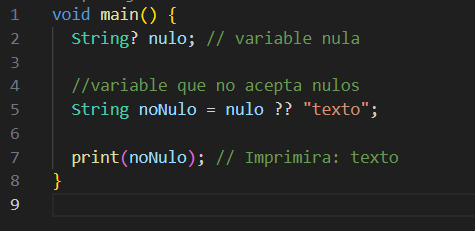
Ahora verás cómo crear una variable que puede ser nula:



El signo ? después de un tipo de dato, quiere decir que está variable puede tener tanto un *String* como un *nulo*. Ahora bien, como dart posee un “null safety” quiere decir que tendrás que ocuparte de controlar el uso de los nulos a través de los signos ?? y ! enseguida veremos el uso de ambos.

## 1.1 Signo ??

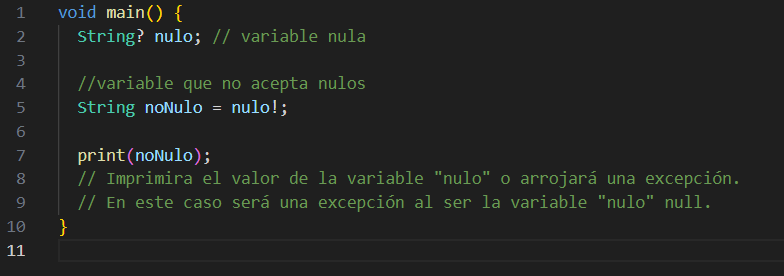
El signo ?? funciona como un condicional en caso que la variable sea nula, ejemplo:



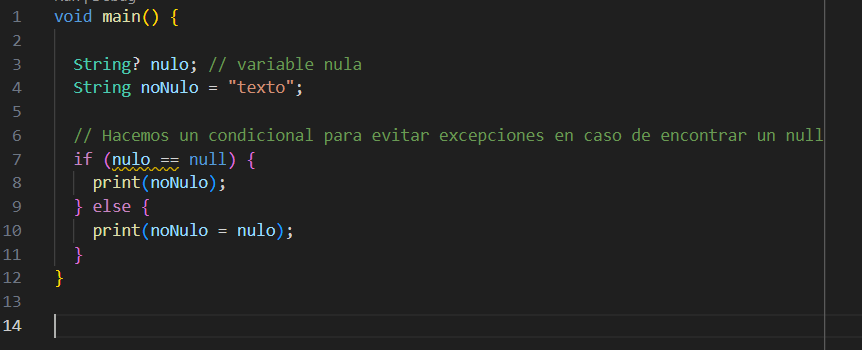
En caso de que la variable “nulo” posea un valor, el condicional ?? no será ejecutado.

## 1.2 Signo !

El signo ! obliga a la variable que no va a poseer un valor *null* y en caso de tenerlo, el programa lanzará una excepción, ejemplo:

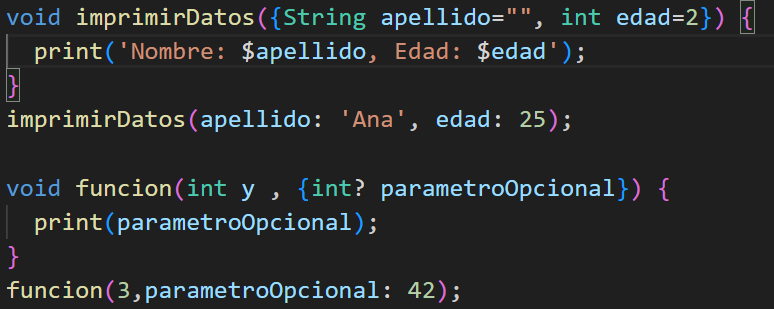


No obstante, el uso del signo ! no es recomendado, en su lugar se recomienda usar los signos ?? o el condicional if( ), ejemplo con if( ):



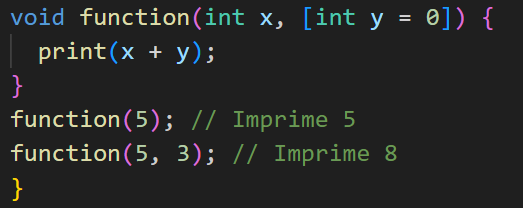
## 1.3 Signo {}

se pueden utilizar para definir parámetros de función con nombre en Dart, lo que permite pasar argumentos a funciones de manera más explícita, también para denotar que un parámetro es opcional



## 1.4 Signo []

Sirven para inicializar un parámetro opcional



# **Sobrecarga De constructores**

En Dart, la sobrecarga de constructores no se implementa de la misma manera que en algunos otros lenguajes de programación como Java o C#. En lugar de permitir múltiples constructores con diferentes parámetros, Dart permite la definición de constructores con nombre. Esto significa que puedes tener varios constructores en una clase, pero cada uno debe tener un nombre único. Esta característica permite inicializar objetos de una clase de diferentes maneras, dependiendo de los parámetros proporcionados.



**Sobrecarga de Métodos en Dart**

En Dart, la sobrecarga de métodos no se soporta de la misma manera que en algunos otros lenguajes de programación, como C ++ o Java. En lugar de permitir múltiples métodos con el mismo nombre pero diferentes firmas, Dart emplea otros mecanismos como los parámetros opcionales y los parámetros con nombre para lograr una funcionalidad similar. A continuación, se proporciona una explicación detallada con ejemplos.

## 1.1 Parámetros Opcionales Posicionales :

En Dart, puedes definir parámetros opcionales usando corchetes []. Estos parámetros pueden omitirse cuando se llama al método.

void printInfo(String name, [int? age]) {

if (age != null) {

print('Name: $name, Age: $age');

} else {

print('Name: $name');

}

}

void main() {

printInfo('Alice');

printInfo('Bob', 30);

}

## 

## 

## 

## 

## 1.2 Parámetros Nombrados:

Dart permite definir parámetros con nombre usando llaves {}. Los parámetros con nombre pueden ser opcionales y pueden tener valores predeterminados.

void printInfo(String name, {int? age, String country = 'Unknown'}) {

print('Name: $name, Age: ${age ?? 'Not specified'}, Country: $country');

}

void main() {

printInfo('Alice');

printInfo('Bob', age: 30);

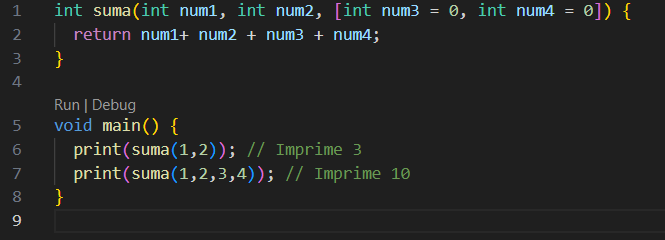
printInfo('Charlie', country: 'USA');

printInfo('David', age: 25, country: 'Canada');

}

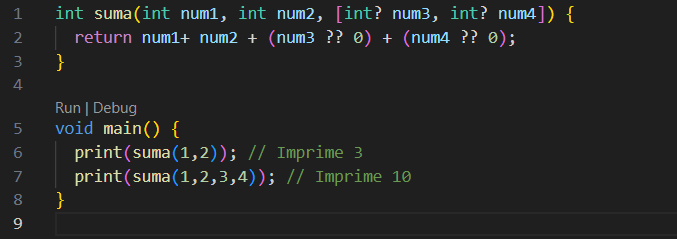
1.4 Parámetros Declarados:

Hemos apreciado que en dart los parámetros pueden ser opcionales u nombrados, no obstante, puedes declararlos con un valor en específico como en el siguiente ejemplo con **Parámetros Posicionales Opcionales**:



Lo anterior muestra que en caso de no pasar los parámetros *num3* y *num4,* tomarán su valor por defecto que es *cero.*

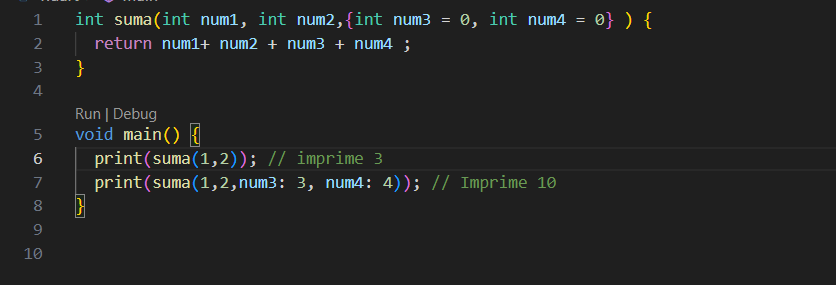
Aclaración: En caso de no proveer un valor por defecto, sus valores tendrán que ser nulos. Ejemplo:



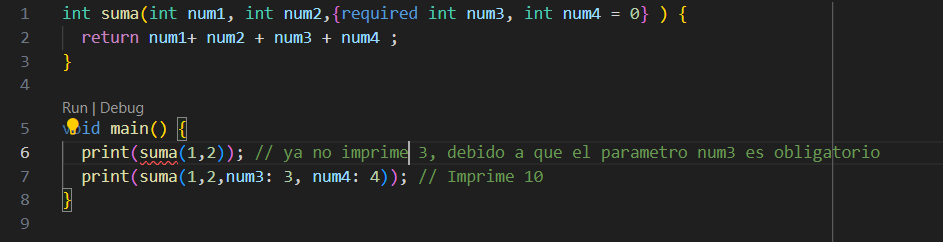
Funcionará similar, solo que habrá que implementar un control de nulos ( ?? o ! ).

En el caso de los **Parámetros Nombrados** será similar, solo hay que tener en cuenta que estos permiten el uso de la palabra reservada “required”, esta palabra obliga al programador a especificar el valor de este parámetro al llamar la función.

Ejemplo sin required:

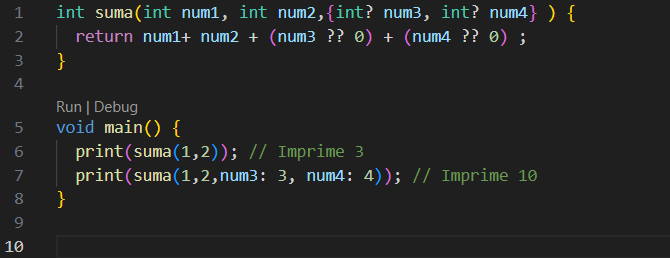


Ejemplo con required:



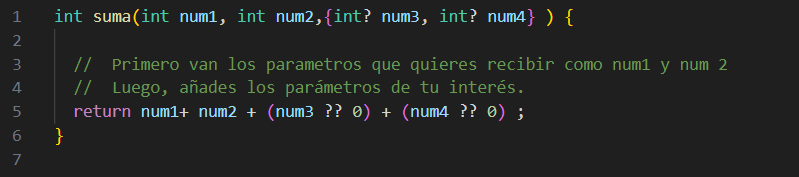
En caso de no proveer un valor por predeterminado o el nombre de un parámetro marcado como “required” << ( ... ) sus valores tendrán que ser nulos >>

Ejemplo:

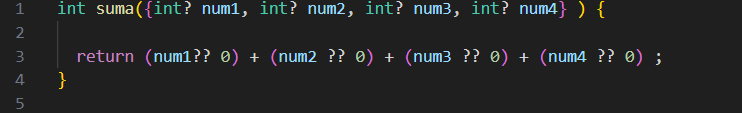


Aclaración: Dart no permite usar **Parámetros Nombrados** y **Parámetros Posicionales Opcionales** al mismo tiempo; así que tendrás que pensar cuáles parámetros quieres usar antes de construir tu función, teniendo en cuenta sus ventajas y desventajas.

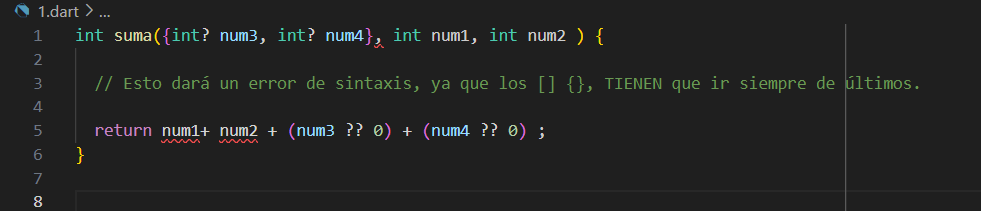
Aclaración: Al usar cualquiera de los parámetros anteriormente mencionados, siempre tendrás que aplicarlos correctamente de la siguiente manera:



También podrás crearlos de la siguiente manera:



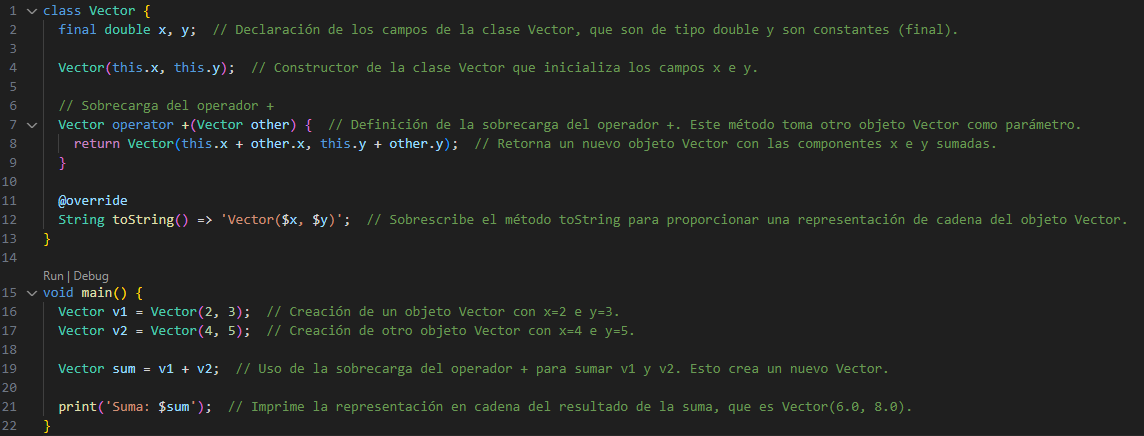
Pero núnca así:



# Sobrecarga de operadores

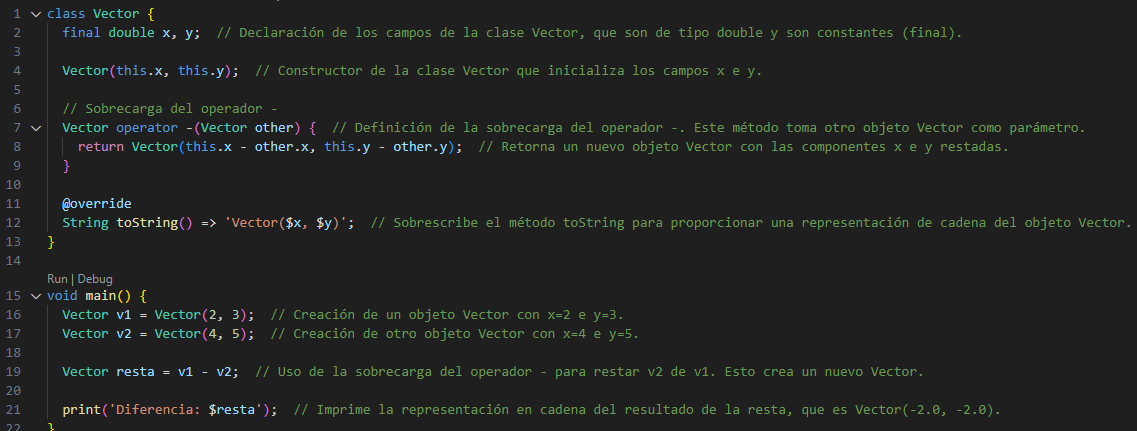
La sobrecarga de operadores permite hacer el código más intuitivo y fácil de leer, especialmente cuando se trabaja con operaciones matemáticas o lógica personalizada en clases. En Dart es una característica que permite a los desarrolladores definir o redefinir el comportamiento de los operadores para clases personalizadas. Esto significa que puedes especificar cómo debe comportarse un operador.

## 1.1 Sobrecarga del operador “+”



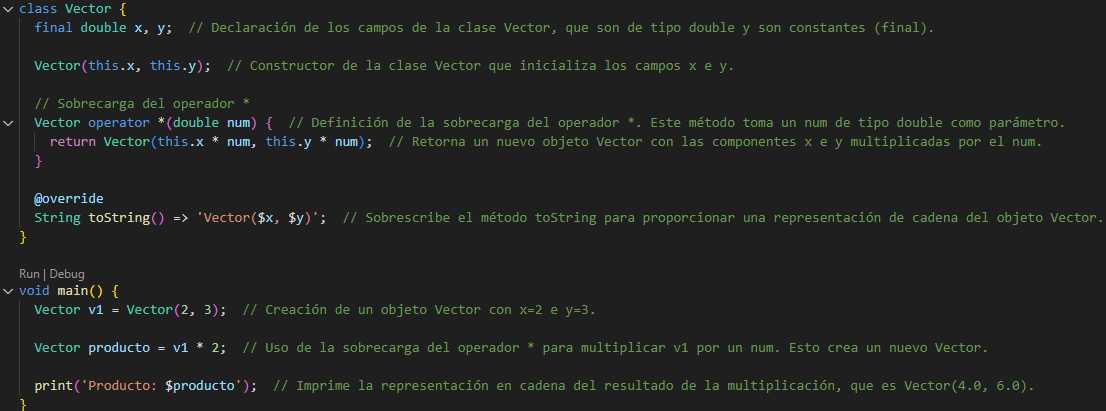
Se define un método “operator +” que toma otro objeto de tipo vector y devuelve un nuevo vector con los componentes sumados.

## 1.2 Sobrecarga del operador ”-”



Se define un método “operator -” que toma otro objeto de tipo vector y devuelve un nuevo vector con los componentes sumados.

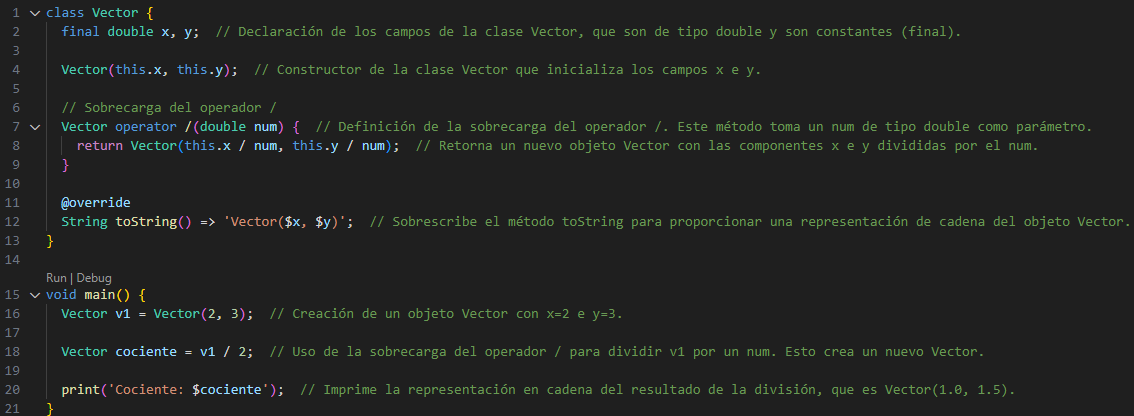
## 1.3 Sobrecarga del operador “\*”



Se define un método “operator \*” que toma otro dato de tipo Double y devuelve un nuevo vector con los componentes multiplicados.

También es posible multiplicar 2 vectores usando la misma estructura que en la suma y la resta.

## 1.3 Sobrecarga del operador “/”



Se define un método “operator /” que toma otro dato de tipo Double y devuelve un nuevo vector con los componentes divididos.

También es posible dividir 2 vectores usando la misma estructura que en la suma y la resta.

# 

# 

# 

# Bibliografía.

[Functions | Dart](https://dart.dev/language/functions)

[Variables | Dart](https://dart.dev/language/variables)

[Seguridad contra nulos en Dart - Introducción - YouTube](https://www.youtube.com/watch?v=iYhOU9AuaFs)

<https://dart.dev/language/operators>