## **Funciones con Parámetros Opcionales en TypeScript**

En TypeScript, una función puede tener parámetros que son opcionales. Esto significa que al llamar a la función, no es obligatorio proporcionar un valor para esos parámetros. Si no se proporciona un valor, el parámetro toma un valor por defecto, que puedes especificar.

**¿Cómo se declaran los parámetros opcionales?**

Se indica que un parámetro es opcional colocando un signo de interrogación (?) después del nombre del parámetro y su tipo.

TypeScript

function saludar(nombre: string, apellido?: string): string { // ... }

Use code [with caution.](file:///C:/faq#coding)

En este ejemplo, el parámetro apellido es opcional. Puedes llamar a la función de esta manera:

TypeScript

saludar("Juan"); // Solo se proporciona el nombre saludar("Juan", "Pérez"); // Se proporcionan tanto el nombre como el apellido

Use code [with caution.](file:///C:/faq#coding)

**Ejemplos con nombres descriptivos y comentarios detallados**

### **Ejemplo 1: calcularAreaCirculo.ts**

TypeScript

function calcularAreaCirculo(radio: number, pi: number = 3.1416): number { // Calcula el área de un círculo dado su radio. // El valor de pi es opcional y por defecto se toma como 3.1416. return pi \* radio \* radio; }

Use code [with caution.](file:///C:/faq#coding)

### **Ejemplo 2: crearUsuario.ts**

TypeScript

function crearUsuario(nombre: string, edad: number, correoElectronico?: string): { nombre: string, edad: number, correoElectronico?: string } { // Crea un objeto de usuario con las propiedades proporcionadas. // El correo electrónico es opcional. return { nombre, edad, correoElectronico }; }

Use code [with caution.](file:///C:/faq#coding)

### **Ejemplo 3: generarMensaje.ts**

TypeScript

function generarMensaje(mensaje: string, destinatario: string, asunto?: string): string { // Genera un mensaje de correo electrónico. // El asunto es opcional. if (asunto) { return `Para: ${destinatario}\nAsunto: ${asunto}\n\n${mensaje}`; } else { return `Para: ${destinatario}\n\n${mensaje}`; } }

Use code [with caution.](file:///C:/faq#coding)

### **Ejemplo 4: sumarNumeros.ts**

TypeScript

function sumarNumeros(...numeros: number[]): number { // Suma todos los números proporcionados como argumentos. // Utiliza el operador rest (...) para aceptar un número variable de argumentos. return numeros.reduce((total, numero) => total + numero, 0); }

Use code [with caution.](file:///C:/faq#coding)

**Explicación de los ejemplos:**

* **calcularAreaCirculo.ts:** Calcula el área de un círculo, permitiendo al usuario especificar un valor de pi si lo desea.
* **crearUsuario.ts:** Crea un objeto de usuario con propiedades básicas, permitiendo agregar un correo electrónico opcional.
* **generarMensaje.ts:** Genera un mensaje de correo electrónico simple, con un asunto opcional.
* **sumarNumeros.ts:** Suma un número variable de argumentos numéricos utilizando el operador rest.

**Resumen:**

Los parámetros opcionales en TypeScript proporcionan flexibilidad al definir funciones. Permiten que las funciones sean llamadas con diferentes números de argumentos, lo que facilita su uso en diversas situaciones.

**Puntos clave:**

* Se indican con un signo de interrogación (?).
* Tienen un valor por defecto si no se proporciona un valor al llamar a la función.
* Pueden ser utilizados en combinación con otros tipos de parámetros.

**Beneficios:**

* **Mayor flexibilidad:** Las funciones pueden adaptarse a diferentes escenarios.
* **Código más conciso:** Se evitan comprobaciones innecesarias de si un parámetro está definido.
* **Mejor legibilidad:** El código se vuelve más claro y fácil de entender.