



# **Introducción a TypeScript**

**Antonio Risueño Espinosa**



# Índice

<b>Introducción a TypeScript</b>	<b>1</b>
<b>Introducción a TypeScript</b>	<b>3</b>
<b>Tipos Básicos</b>	<b>3</b>
<b>Objetos, Arreglos e Interfaces</b>	<b>4</b>
<b>Funciones y sus Argumentos</b>	<b>4</b>
<b>Desestructuración de Arreglos y Objetos</b>	<b>4</b>
<b>Importaciones y Exportaciones</b>	<b>5</b>
<b>Clases y Constructores</b>	<b>5</b>
<b>Tipos Genéricos</b>	<b>6</b>
<b>Encadenamiento Opcional</b>	<b>6</b>

# Introducción a TypeScript

```
let texto: string = "hola";  
texto = 10;    Type 'number' is not assignable to type 'string'.
```

## Tipos Básicos

```
let texto: string = "hola";  
texto = 10;    Type 'number' is not assignable to type 'string'.  
  
let numero: number = 90;  
numero = "hola";    Type 'string' is not assignable to type 'number'.  
  
let tOf: boolean = false;  
tOf = "hola";    Type 'string' is not assignable to type 'boolean'.  
  
let arreglo: number[] = [1, 2, 3];  
arreglo = "hola";    Type 'string' is not assignable to type 'number[]'.  
  
let tupla: [string, number] = ["texto", 42];    Either use this collection's contents  
tupla = [42, "texto"];    Type 'number' is not assignable to type 'string'.  
  
enum Color {  
    Rojo,  
    Verde,  
    Azul  
}  
let colorFavorito: Color = Color.Rojo;  
colorFavorito = "rojo";    Type '"rojo"' is not assignable to type 'Color'.  
  
let cualquiera: any = "puede ser cualquier cosa";  
cualquiera = 42;  
  
let sinValor: void;  
sinValor = undefined;  
sinValor = null;    Type 'null' is not assignable to type 'void'.  
  
let nulo: null = null;  
nulo = undefined;    Type 'undefined' is not assignable to type 'null'.  
  
let indefinido: undefined = undefined;  
indefinido = null;    Type 'null' is not assignable to type 'undefined'.  
  
function errorFatal(): never {  
    throw new Error("Esto siempre falla");  
}  
let nunca: never;  
nunca = errorFatal();    Remove this use of the output from "errorFatal"; "errorFa  
nunca = 42;    Type 'number' is not assignable to type 'never'.
```

## Objetos, Arreglos e Interfaces

```
interface Producto {  
    nombre: string;  
    precio: number;  
    categoria: string;  
}  
  
const productos: Producto[] = [  
    { nombre: "Laptop", precio: 1200, categoria: "Electrónica" },  
    { nombre: "Camiseta", precio: 20, categoria: "Ropa" },  
    { nombre: "Libro", precio: 15, categoria: "Educación" }  
];  
  
productos[0].precio = 1000;  
productos[1].nombre = 50; | Type 'number' is not assignable to type 'string'.
```


## Funciones y sus Argumentos

```
function multiplicar(a: number, b: number): number {  
    return a * b;  
}  
  
const resultado1 = multiplicar(5, 10);  
const resultado2 = multiplicar(5, "10"); | Argument of type 'string' is not assignable to parameter of type 'number'  
const resultado3 = multiplicar("5", 10); | Argument of type 'string' is not assignable to parameter of type 'number'
```

## Desestructuración de Arreglos y Objetos

```
const persona = { nombre: "Juan", edad: 30 };  
const { nombre, edad } = persona;  
  
const numeros = [10, 20, 30, 40];  
const [primero, segundo, ...resto] = numeros;
```

## Importaciones y Exportaciones

```
ImportacionesExportaciones > ts utils.ts >  saludar
1  export function saludar(nombre: string): string {
2      return `Hola, ${nombre}!`;
3  }
```

```
ImportacionesExportaciones > ts main.ts > ...
1  import { saludar } from './utils';
2
3  const mensaje = saludar("Juan");
4  console.log(mensaje);
5
```

## Clases y Constructores

```
ClasesConstructores > ts ClasesConstructores.ts > ...
1  class Animal {
2      nombre: string;
3      especie: string;
4
5      constructor(nombre: string, especie: string) {
6          this.nombre = nombre;
7          this.especie = especie;
8      }
9  }
10
11  const miAnimal = new Animal("Leo", "León");
12  console.log(`Nombre: ${miAnimal.nombre}, Especie: ${miAnimal.especie}`);
13
```

```
C:\Medac\Diseño de Interfaces\TS\ClasesConstructores>node animal.js
Nombre: Leo, Especie: León
```

## Tipos Genéricos

```
function primero<T>(arreglo: T[]): T {  
    return arreglo[0];  
}  
  
const arregloNumeros = [1, 2, 3, 4];  
const primerNumero = primero(arregloNumeros);  
  
const arregloStrings = ["a", "b", "c"];  
const primerString = primero(arregloStrings);  
  
console.log(primerNumero);  
console.log(primerString);
```

## Encadenamiento Opcional

```
const usuario = {  
    perfil: {  
        nombre: 'Juan',  
        direccion: {  
            calle: 'Av. Principal',  
            ciudad: 'Sevilla'  
        }  
    }  
};  
  
const nombreUsuario = usuario.perfil?.nombre;  
console.log(nombreUsuario);  
  
const codigoPostal = usuario.perfil?.direccion?.codigoPostal; Property 'codigoPostal' does not exist on type 'Dirección'.  
console.log(codigoPostal);
```

```
C:\Program Files\nodejs\node.exe .\Encadenamiento\usuario.js  
Juan  
undefined
```