**TypeScript**

**Null y Undefined.**

**Documentación:**

**<https://www.typescriptlang.org/docs/handbook/advanced-types.html#nullable-types>**

[**https://www.typescriptlang.org/docs/handbook/release-notes/typescript-2-0.html**](https://www.typescriptlang.org/docs/handbook/release-notes/typescript-2-0.html)

[**https://www.typescriptlang.org/docs/handbook/release-notes/typescript-2-0.html#null--and-undefined-aware-types**](https://www.typescriptlang.org/docs/handbook/release-notes/typescript-2-0.html#null--and-undefined-aware-types)

[**https://www.typescriptlang.org/docs/handbook/advanced-types.html#nullable-types**](https://www.typescriptlang.org/docs/handbook/advanced-types.html#nullable-types)

[**https://www.typescriptlang.org/docs/handbook/2/everyday-types.html**](https://www.typescriptlang.org/docs/handbook/2/everyday-types.html)

Para los tipos de datos null y undefined no hay mucho de qué hablar de ellos en TS, cosa que no hayamos visto en JS. Exceptuando de que los null y undefined podrían ser considerados valores permitidos a un string, un bolean u otro tipo de dato. Pero en las ultimas versiones de TS eso ya no es permitido.

A los tipos de datos números y boléanos no se les puede asignar el valor de undefined, porque TS nos mandaría un error. Esto es algo que TS ha venido incorporando en sus últimas versiones de manera obligatoria, ya que te exige el asignarle un valor del tipo indicado. Ejemplo:

// Error: Se declara "numero", pero su valor no se lee nunca

// Error: El tipo 'undefined' no se puede asignar al tipo 'number'

let numero: number = undefined;

// Error: Se declara "esActivo", pero su valor no se lee nunca.

// Error: El tipo 'undefined' no se puede asignar al tipo 'boolean'

let esActivo: boolean = undefined;

Si quisiéramos cambiar esta configuración por defecto de TS, en nuestro archivo de configuración **TSConfig.json** existe la propiedad **"strictNullChecks: true"** que podríamos poner como **"false"** para cambiar la configuración y que al verificar el tipo de dato pueda tener en cuenta el tipo `null` e `undefined`. **Pero esto es algo que no se recomienda en lo absoluto.**

**Nota importante.**

**Si nosotros sabemos que en nuestra variable sería posible recibir el tipo de dato undefined, la mejor recomendación sería indicar que en esa variable el tipo de dato también se podría recibir el undefined. Ejemplo:**

let calificacion: (number | undefined) = undefined; // Todo Ok

let esAdministrador: (boolean | undefined) = undefined; // Todo Ok

**Todas estas restricciones que se emplean al tipo de dato undefined, aplicarían exactamente igual con el tipo de dato null.**

**Por último hay que recordar que NULL no es lo mismo que UNDEFINED [(null ¡== undefined) === true] y también tenemos que hacer que TS sea lo más restrictivo posible con los tipos de datos que vamos a manejar.**