**TypeScript**

**Any y Casting.**

**Documentación:**

[**https://www.typescriptlang.org/es/play?q=414#example/any**](https://www.typescriptlang.org/es/play?q=414#example/any)

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

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

El tipo de dato <any> es muy importante comprenderlo y al momento de programar tenemos que utilizarlo lo menos posible que se pueda. Hay que tratar de nunca usar este tipo de dato. En el archivo de configuración **TSConfig.json** existe una propiedad **"noImplicitAny"** con el valor de **true,** es aconsejable no cambiar el valor de esta propiedad, ya que esta propiedad habilita el informe de errores para expresiones y declaraciones con un tipo "cualquiera" (any) implícito.

Pero algunas veces tendremos que utilizarlo, ya que será nuestra única alternativa de momento rápido para hacer que nuestro programa funcione y luego implementar una interface, implementar alguna clase o alguna solución con un tipado estricto.

Al utilizar el tipo de dato <any> es decirle a TS que se olvide de todo lo que sabe de tipos de datos y que nos deje hacer lo que queramos.

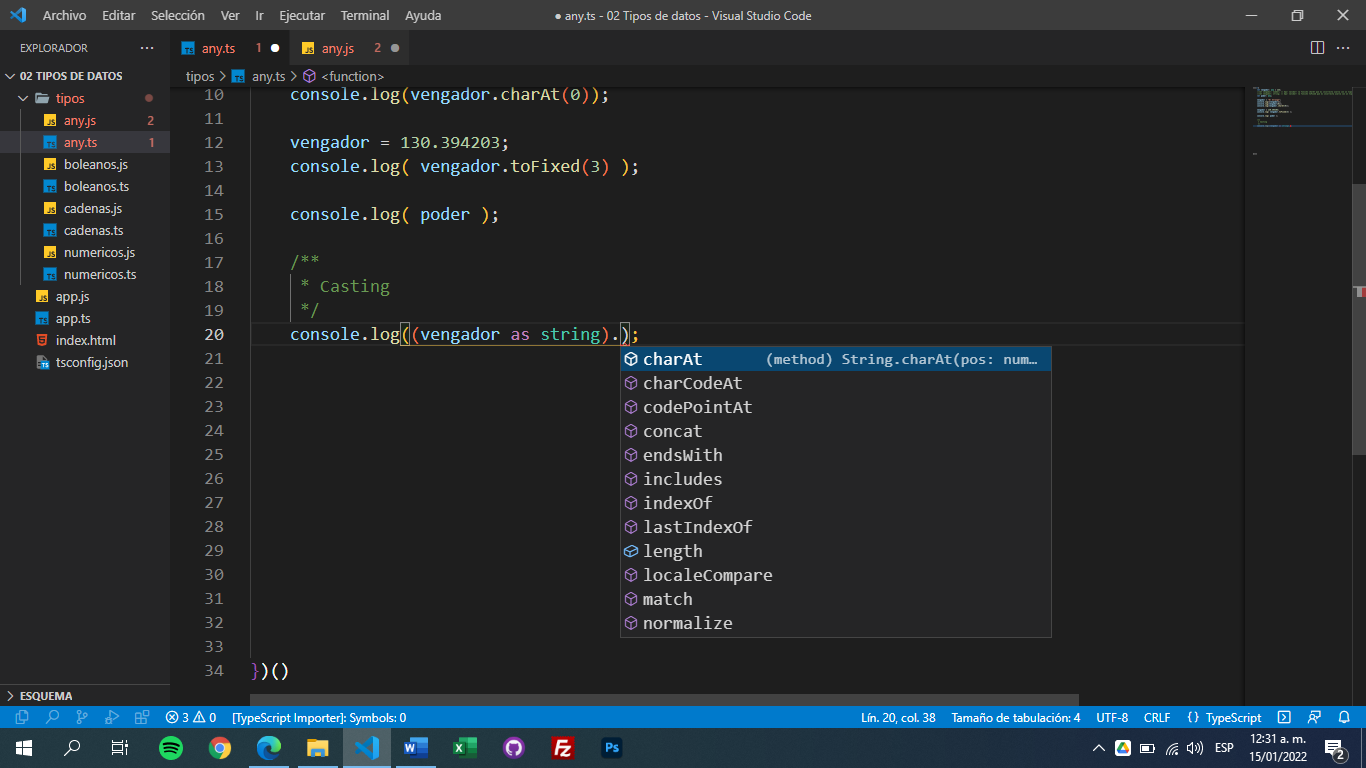
**Es importante saber que cuando utilizamos este tipo de dato al poner un “.” (punto) en las variables, no obtendremos la ayuda del autocompleto de VSCode esta ayuda no se habilitará. Esto puede ser un gran problema porque es muy fácil no acordarse de los nombres de las funciones o métodos. En cambio si nosotros especificamos exactamente el tipo de dato, TS nos validará las funciones que se pueden utilizar con cada tipo de dato.**

**Casting**

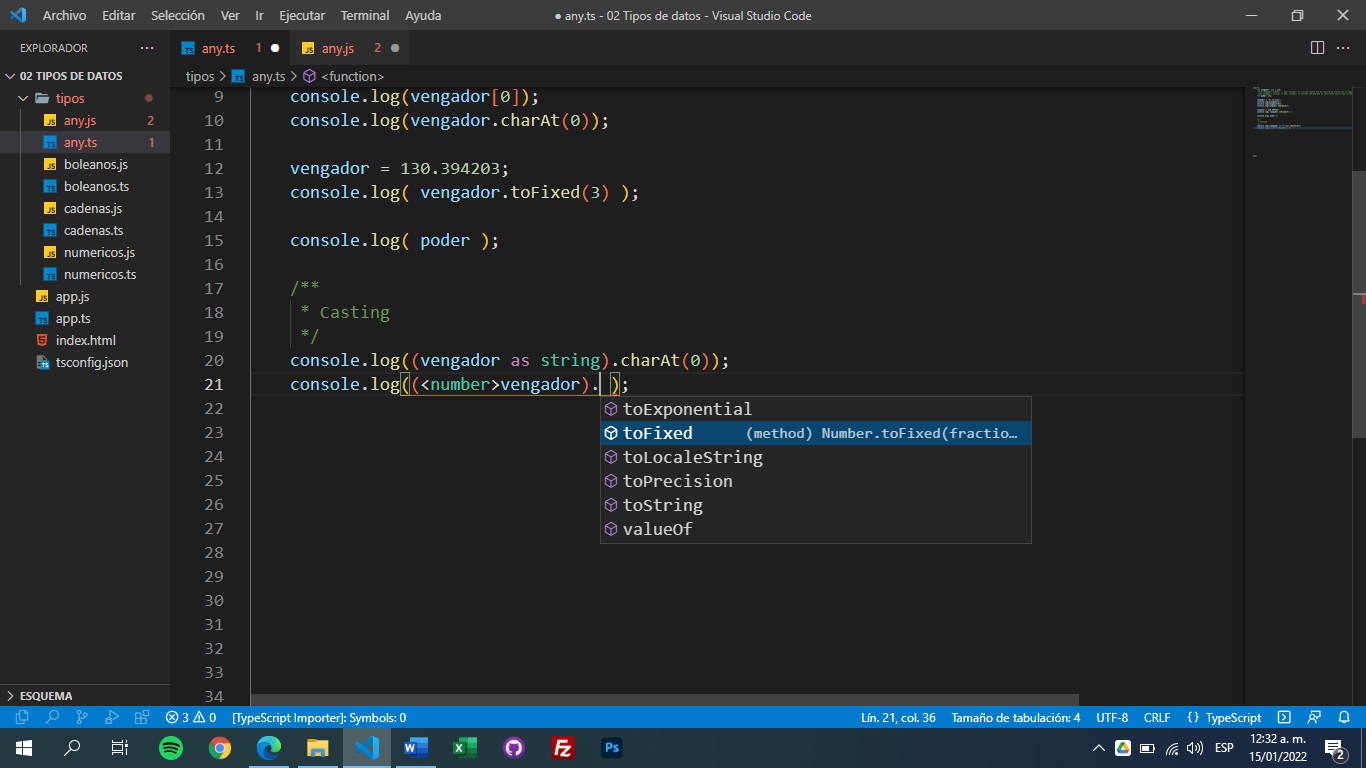
Otra forma de validar los tipos de datos <any> es realizar un casting. **El casting es decirle a TS que tipo de dato será la variable por ese momento, para que sea tratado con el tipado que tiene TS.**

Existen 2 formas de realizar casting:

1. Con el **“as”,** TS estará haciendo un casteo de la variable y tratará la variable como el tipo de dato que se le indique, pero sin cambiar su tipo de dato que es el <any>. Al hacer eso y poner un **"." (punto)**  podremos obtener la ayuda del autocompletado para las variables. ***Este código es propio de TS por lo que en JS no tenemos su sinónimo.*** Ejemplo:



1. Con el **“<>”** la otra forma de hacer un casteo es indicar del lado izquierdo de la variable entre **“<>”** el tipo de dato que será por el momento. Ejemplo:



**Nota.**

**Como consejo, es más recomendable el utilizar la forma del casteo con el “as”, puede resultar más fácil de leer.**