**TypeScript**

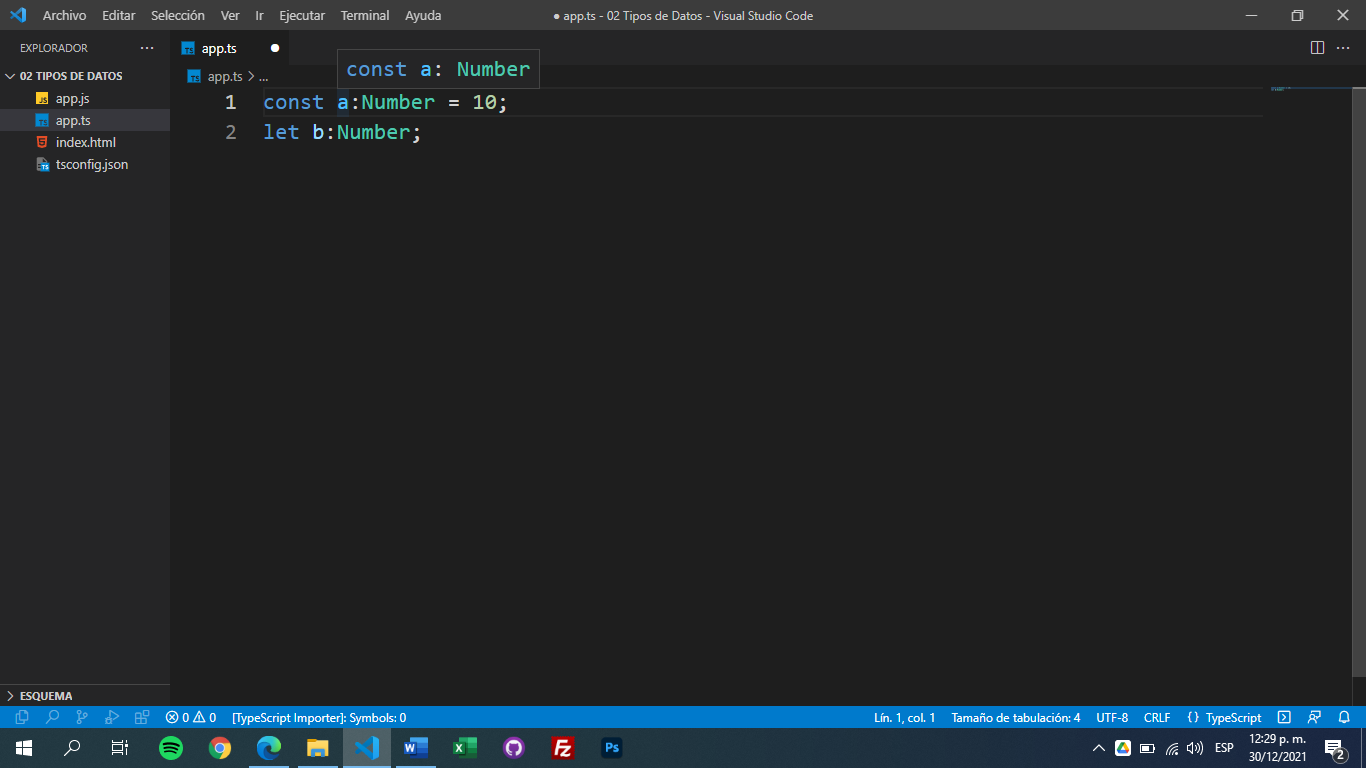
**Inferir tipos y modo estricto.**

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

El TS va a tratar de inferir hasta donde sea posible los tipos de datos, lo mismo que hace JS sola que en TS va a tratar de poner ciertas reglas para evitar que se deforme el código de manera que no esperamos.

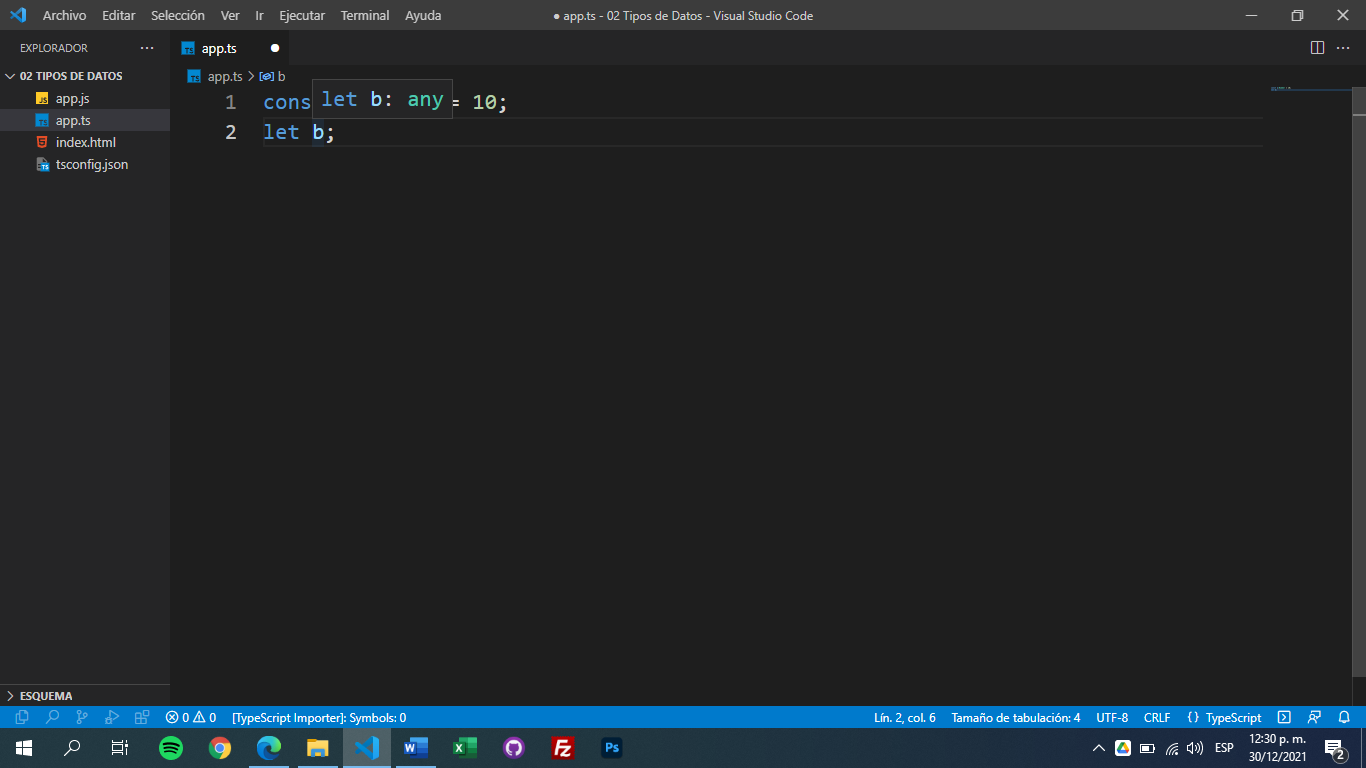
Cuando declaramos los tipos de datos en TS, es muy recomendable declarar exactamente que tipo de dato es cada variable. Para que cualquier persona que lea nuestro código se entere que tipos de datos estamos utilizando específicamente.

Ejemplo:



Si nosotros no especificáramos que tipos de datos tendrían las variables, para el TS entraría un tipo de dato muy famoso llamado <any> para este tipo de dato en TS significa cualquiera.

Ejemplo:

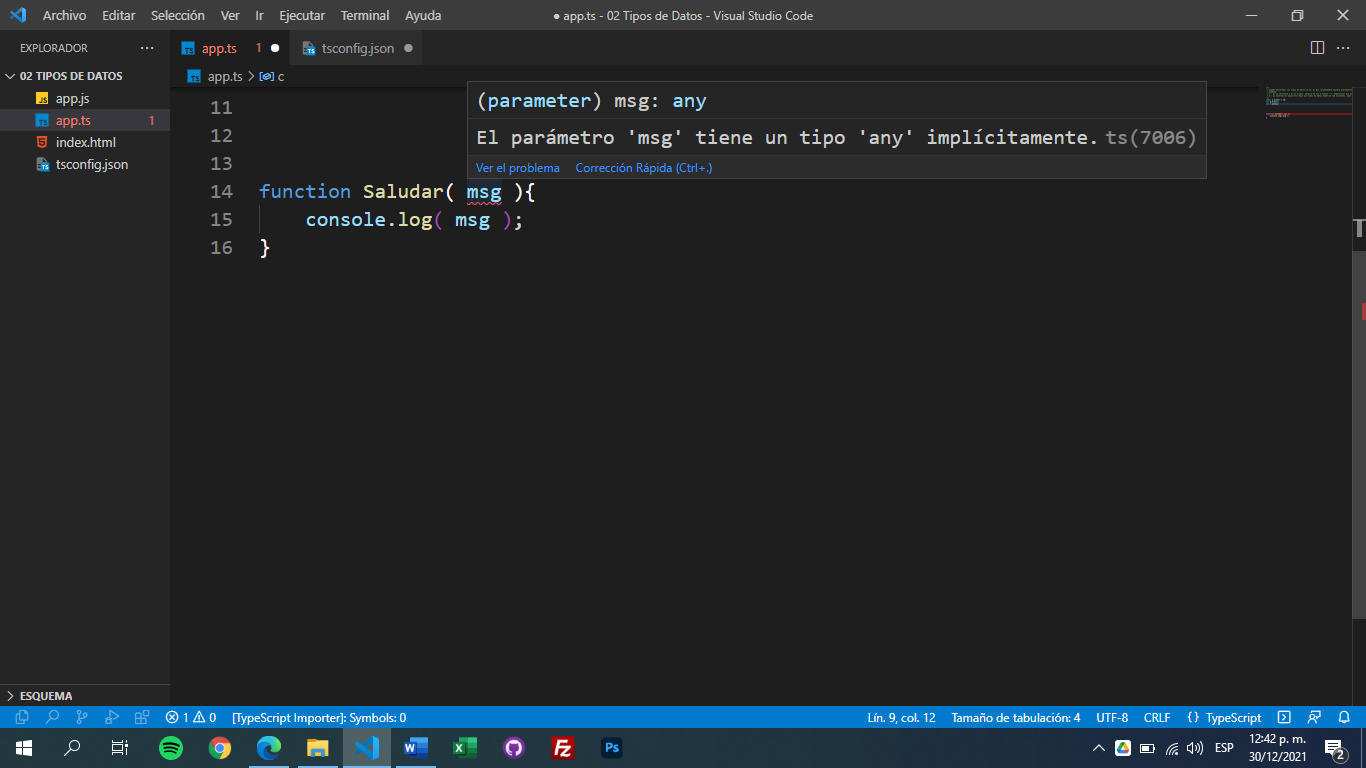


**Funciones**

Para las funciones, por defecto en los argumentos que enviamos tiene marcado el tipo de dato <any> y nos lo marca como error, esto es porque en el archivo de configuración de **TSConfig.json** en la propiedad de **"noImplicitAny":** viene marcado por defecto el valor de verdadero **(true).**

Esto significa que TS habilita el informe de errores para expresiones y declaraciones con un tipo implícito "<any>". Por eso TS estipula como error los argumentos que se encuentren marcados como un tipo de dato <any>. Esto es una regla muy útil, por lo cual no se recomienda poner en valor de **(false)** la propiedad de **"noImplicitAny":** en el **TSConfig.json**

Ejemplo:



El ejemplo de la imagen de arriba con la función declarada sería una mala práctica, ya que no tendríamos la ayuda de TS con la forma de autocompletar el código dependiendo de qué tipo de dato sea la variable.

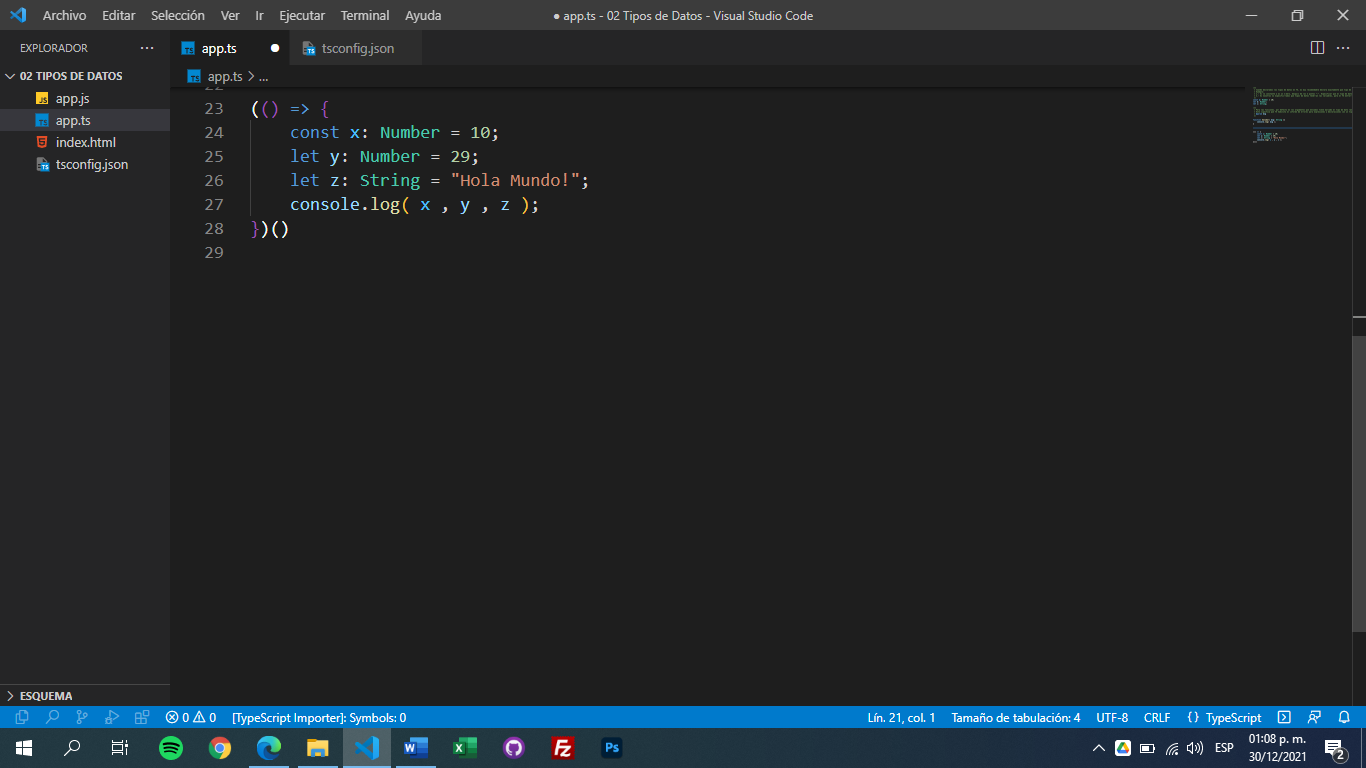
Ejemplo:

|  |  |
| --- | --- |
| Ayuda del TS | Sin ayuda del TS |

**Funciones anónimas autoinvocadas**

Este tipo de funciones se crean para encapsular el código por bloques, esta es una técnica que se usa bastante cuando estamos trabajando con JS y queremos que el código de nuestro proyecto no choque entre si con otros archivos del JS. Para crear este tipo de funciones se hace de la siguiente manera.

Ejemplo:



**Nota Importante.**

**El objetivo de las funciones anónimas autoinvocadas es para que podamos crear varios archivos del JS en el mismo proyecto y que de esta manera nuestro código no caiga dentro del scope global de nuestra aplicación y de esta manera TS no nos marcara error.**