**3.11 Inicio de proyecto – Bases de TypeScript**

Para iniciar un nuevo proyecto, en la consola del sistema (CMD), nos posicionamos en la carpeta donde queremos agregar el proyecto y ejecutamos:   
*npm create vite@latest.*

Elegimos un nombre para el proyecto y seleccionamos:

*√ Project name: ... typscript-intro*

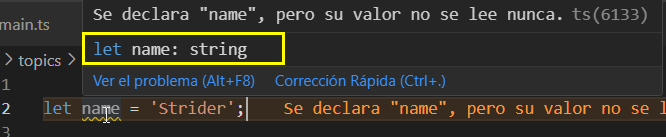
*√ Select a framework: » Vanilla*

*√ Select a variant: » TypeScript*

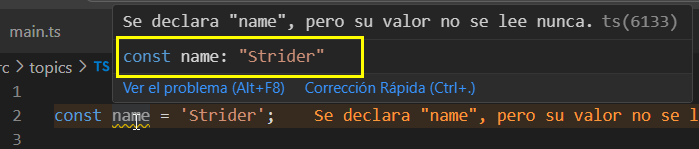
Ya dentro de la carpeta en VSC instalamos los modulo con *npm install* y a continuación arrancamos el proyecto con *npm run dev*

**3.12 Tipos básicos y conceptos generales**

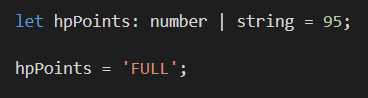
Creamos una carpeta que se llama “Topics” y dentro un archivo que se llama 01-basic-types.

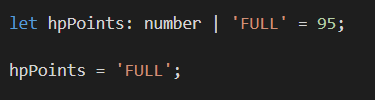
TS siempre nos va a inferir el tipo de una variable, es decir, según el valor que tenga esa variable nos va a indicar qué tipo de dato contiene:   


En este caso nos está indicando que el tipo de dato es un string.



Si la declaramos como constante, directamente ya sabe el valor que le estamos pasando.

Aunque no es lo habitual, podemos indicar que una variable puede tener 2 tipos de datos:   


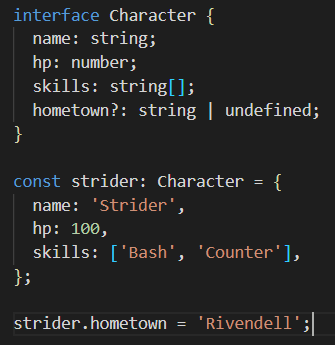
O decirle que puede ser un tipo de dato o un valor directamente:   
  


**3.13 Objetos, arreglos e interfaces**

Agregamos el archivo 02-object-interfaces.ts.

Puedo determinar de que tipo va a ser mi arreglo si lo defino después de ponerle nombre:   


Este arreglo de ejemplo, es un arreglo de tipo string, es decir, no me va a permitir tener dentro ningún otro tipo de dato que no sea un string.

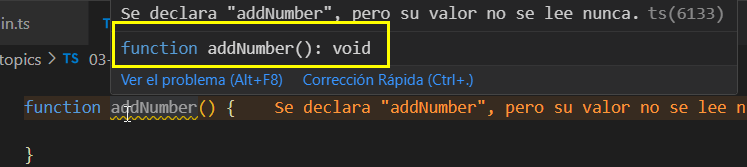
Con las interfaces puedo determinar qué tipo de datos va a tener un objeto:   


Estoy definiendo la interfaz Character, que va a tener 4 propiedades, pero hometown será opcional, y además puede ser un string o undefined.

Cuando creo la constante “strider” le indico que lo que contenga va a ser de tipo Character.

3.14 Funciones

Creamos un nuevo archivo con el nombre: 03-functions.ts

:void 🡪 significa que esta función no regresa nada, no hay un return en la función.

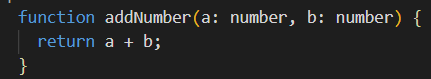
() 🡪 contiene los parámetros que tiene la función.

La forma más básica de definir una función sería:

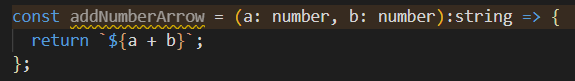
*function addNumber() {*

*}*

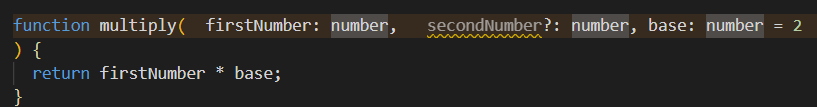
Cuando agregamos un parámetro a una función le tenemos que especificar que tipo de dato es. Si no sabemos que tipo de dato es, al menos tenemos que indicarle que sea :any



En la definición de las funciones podemos definir el tipo de dato que queremos que nos devuelva.

Ejemplo:  
Esta función recibe dos parámetros de tipo number. Lo normal sería que devolviese un number, pero si queremos que devuelva un string lo podemos hacer de este modo.

En una función también puedo definir que va a recibir 3 parámetros: uno de ellos será obligatorio (firstNumber), el segundo va a ser opcional (secondNumber) y el tercero va a tener un valor fijo (base = 2)



Al haber definido en el return firstNumber \* base, si le paso lo siguiente, me devolverá 10 porque está multiplicando el valor de firstNumber \* base:   
