**ANGULAR**

**Comandos**

Verfificar versión node: node -v

Verificar versión npm: npm -v

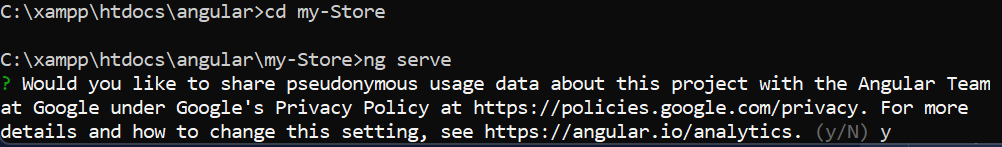
**Crear aplicación**

ng new my-store



**Correr servidor para ver proyecto**

ng serve



Correr servidor y que abra automáticamente en el navegador

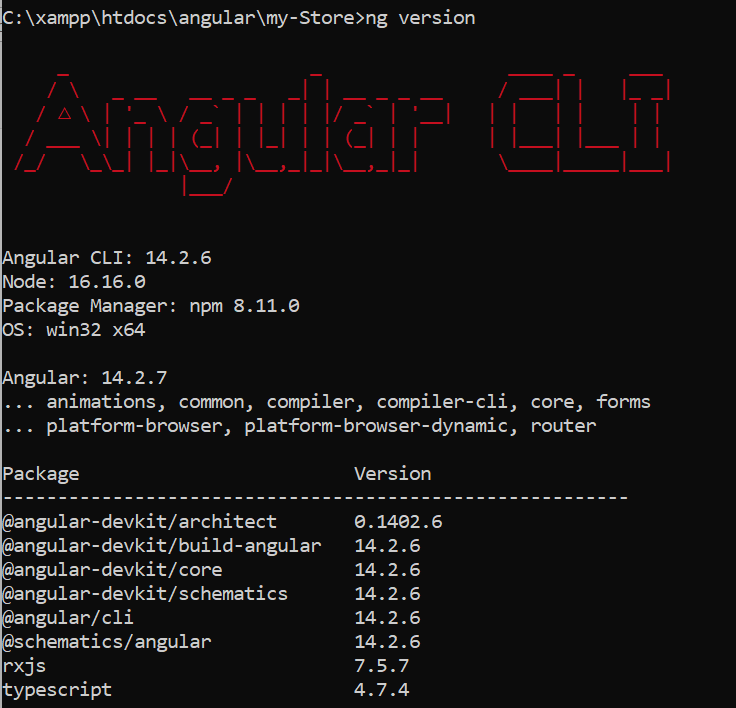
ng serve -o

abrir con otro puerto, en el que se desee

ng serve -o --port=3500

Ver version instalada y dependencias dentro del proyecto

ng version

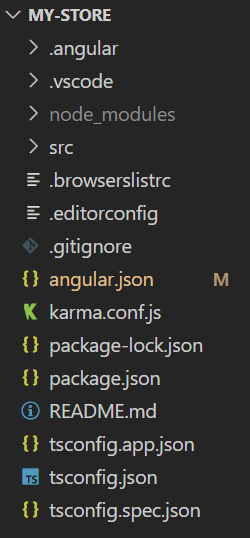


**Abrir editor desde la terminal dentro del proyecto**

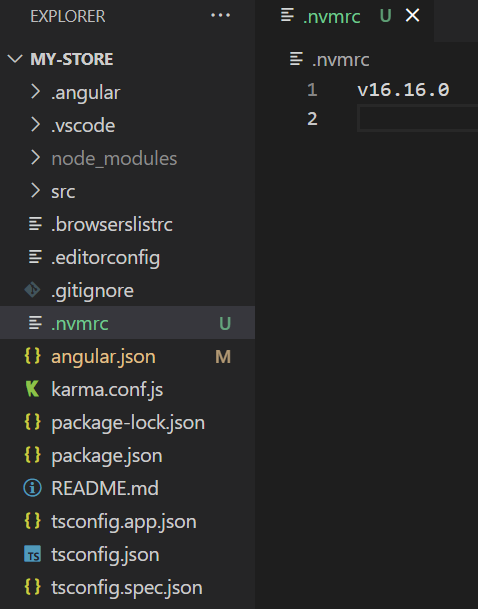
code .



Estructura inicial de angular



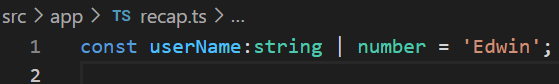
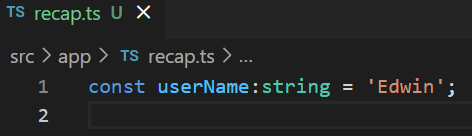
Crear archivo en la raíz del proyecto llamado “.nvmrc” este archivo es recomendable para colocar la versión de node con la que esta el proyecto.



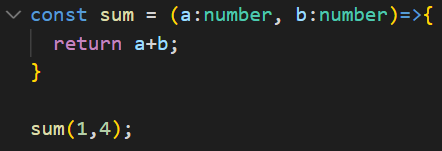
Recomendaciones

**Colocar tipado tanto a variables, funciones, clases**

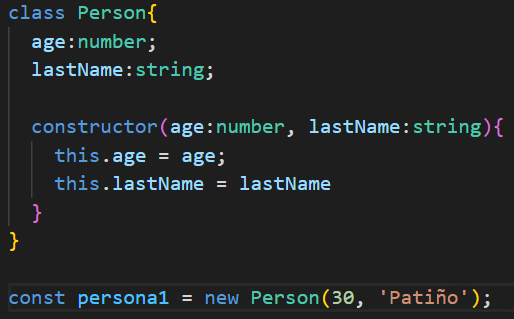
Variable en un solo tipo en este caso string, o doble tipado.



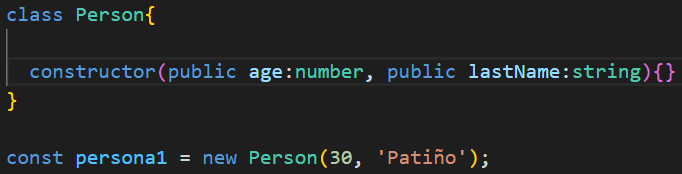
Tipado funciones



Tipado clases



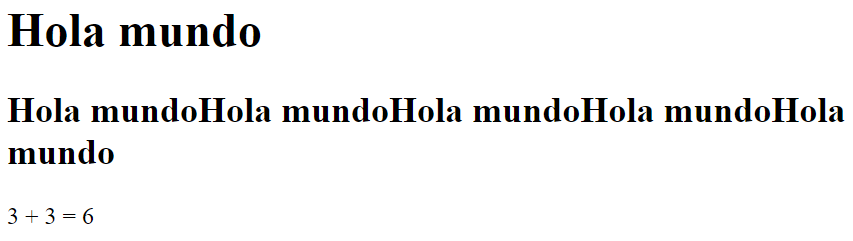
La clase se puede reducir, esto lo permite typescript



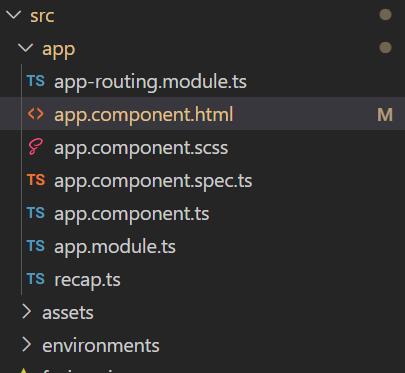
**String interpolation “{}”**

Lo que esta en llaves lo opera y lo que no lo renderiza normalmente

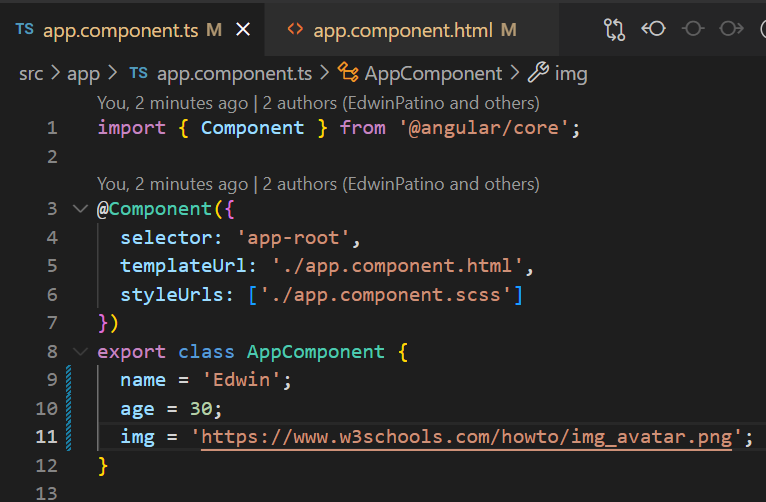




app.component.html se encarga de el renderizado

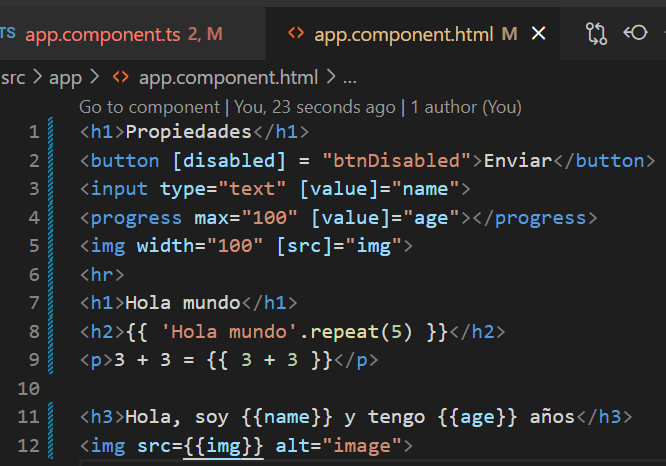
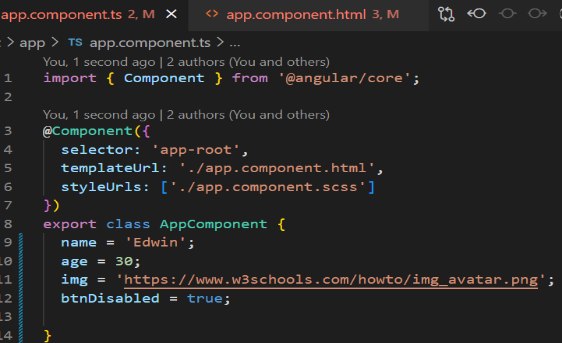
,

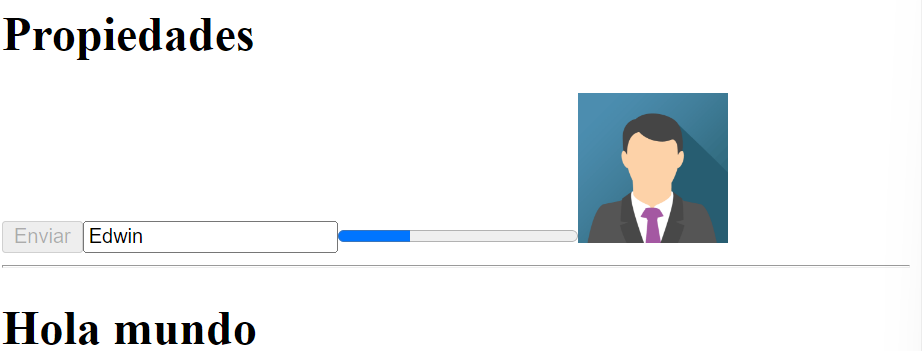
app.component.ts tenemos decoradores “@” y clases, este archivo dice como debe comportarse a que template pertenece “vista”, las funciones y variables deben ser publicas, también nos indica los estilos del componente.

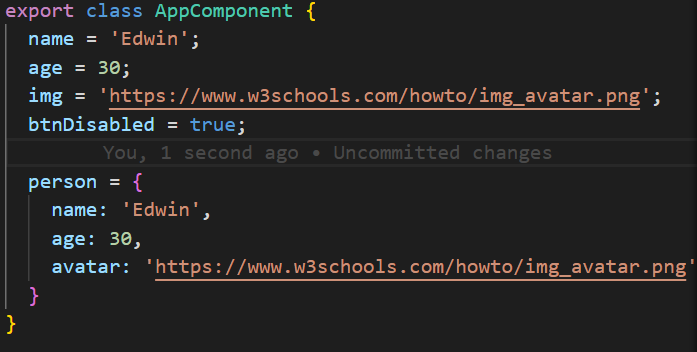
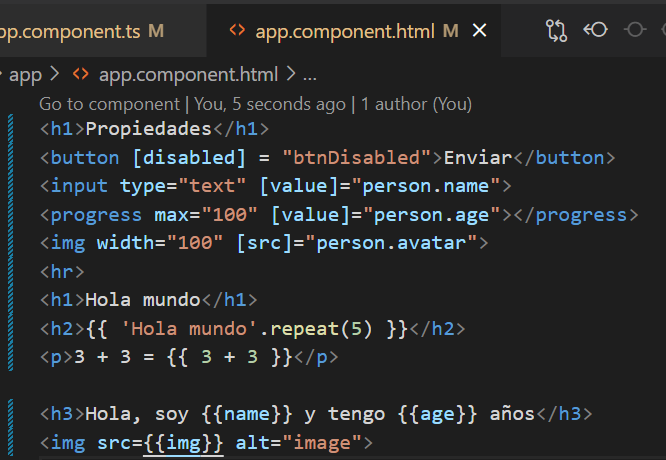
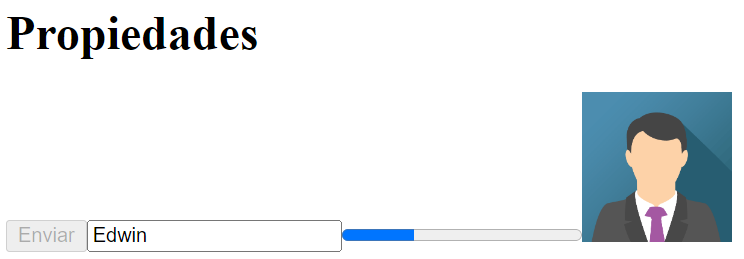

**Property Binding “[]”**

Modificar propiedades nativas de html y enviarle estados controlados desde nuestro componente, diferencia con string interpolation, property binding es para propiedades y string, solo usar para ingresar contenido, un h1 texto párrafo, etc



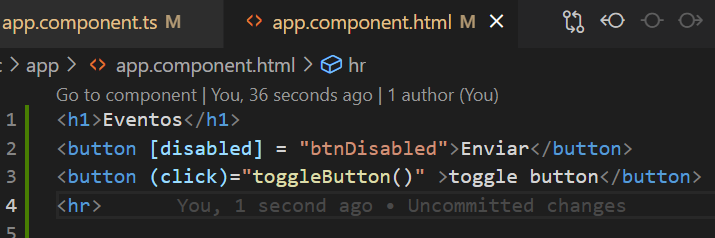


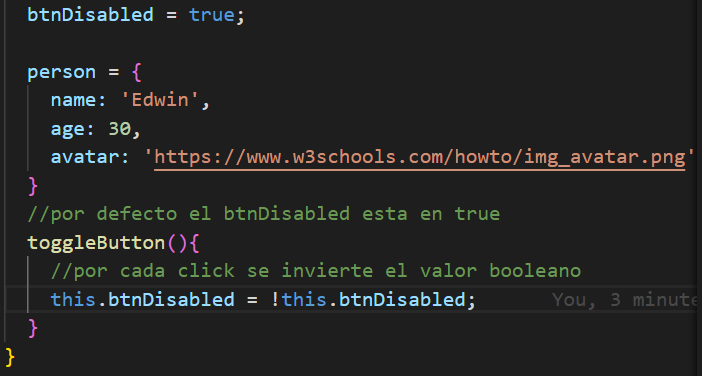
Enviarle objetos

**Event Binding**

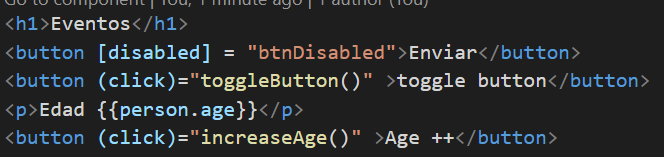
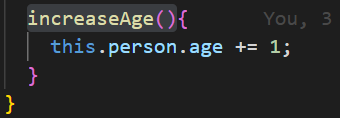
**Evento click:** Habilitar un botón deshabilitado, y cada vez que se da click cambia de estado



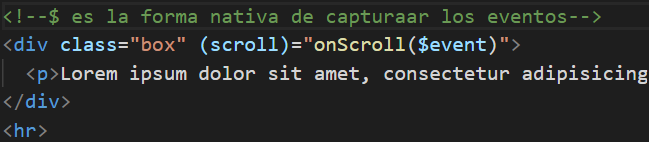
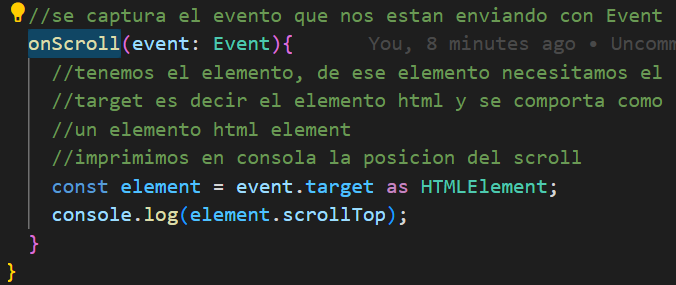


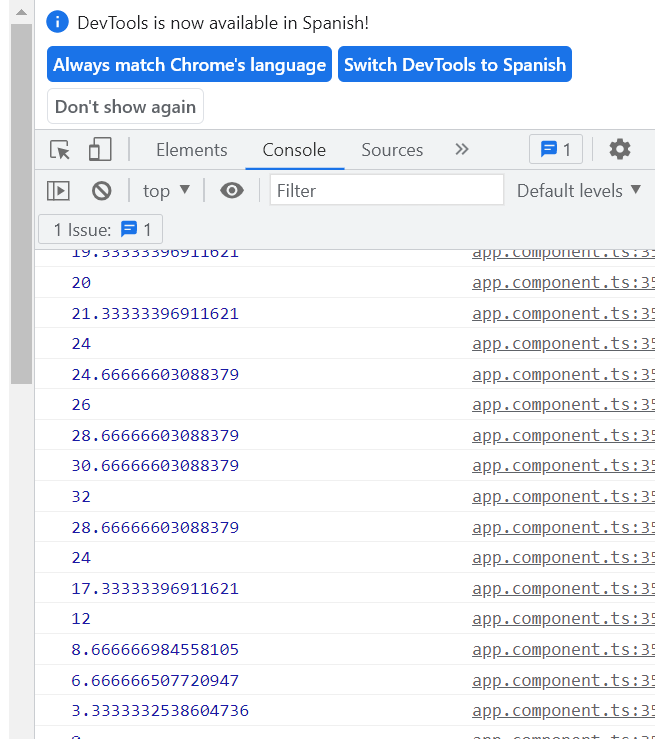
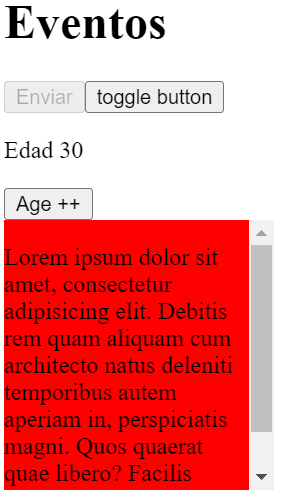


Cada click en el botón age aumenta la edad en uno

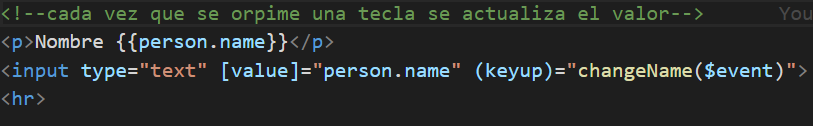
  

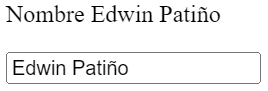
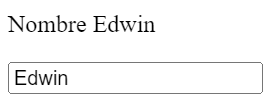
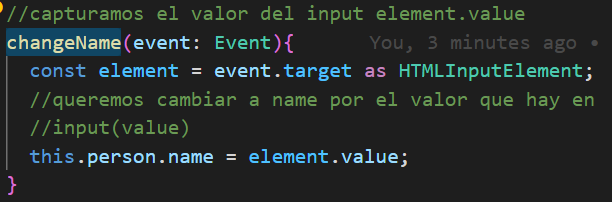

**Evento scroll:** cada vez que se mueve el scroll por consola sale el valor numerico

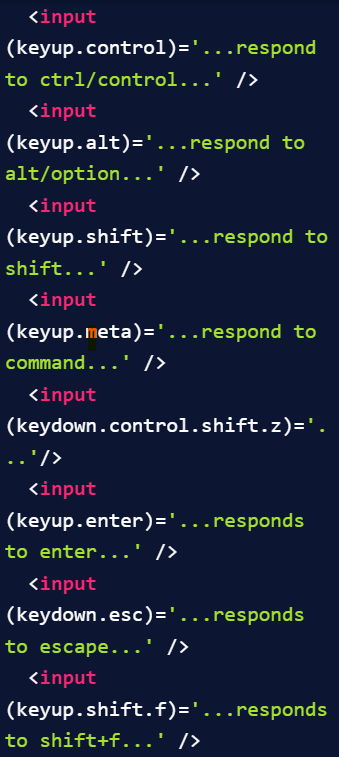
  




Evento keyup: cada vez que teclee se actualiza el valor



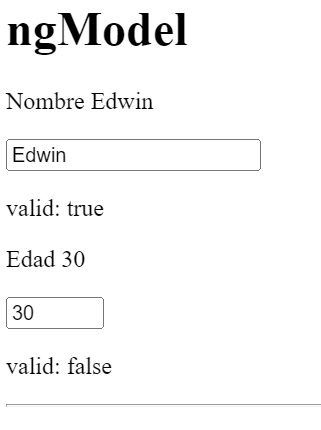
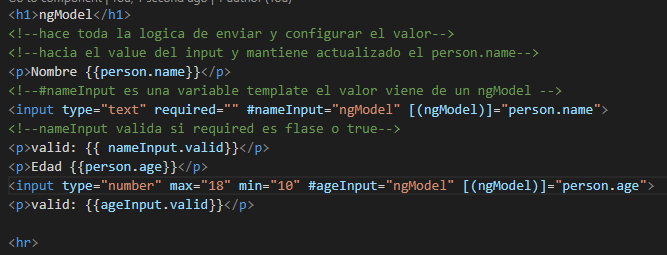




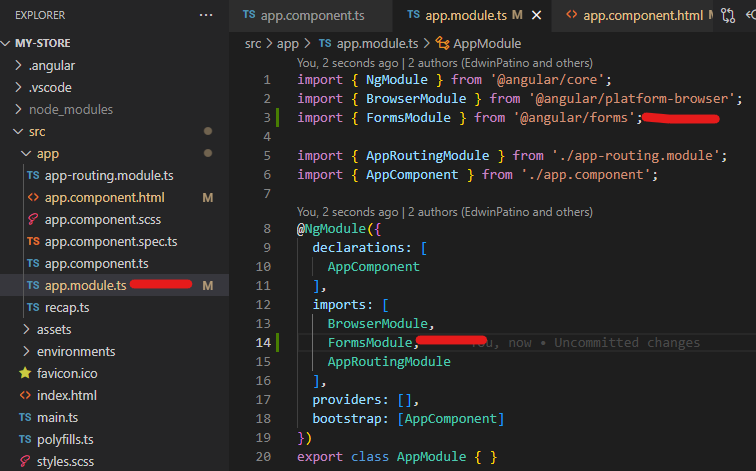
**Data binding [()]**

Se refiere a la fusión de escuchar un evento y setear una propiedad, es un enlace de doble via

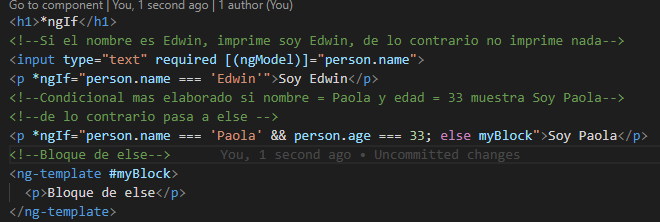
[(ngModel)] = “someValue”



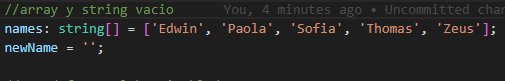
Primero hay que habilitar forms, se importa formsModel y se agrega a @NgModule, esto se hace en archivo app.module.ts

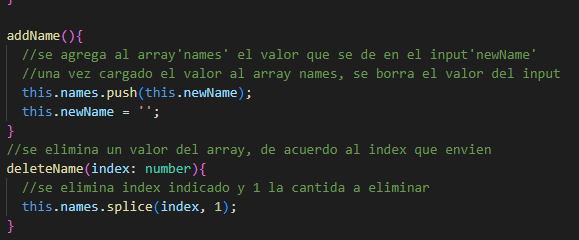


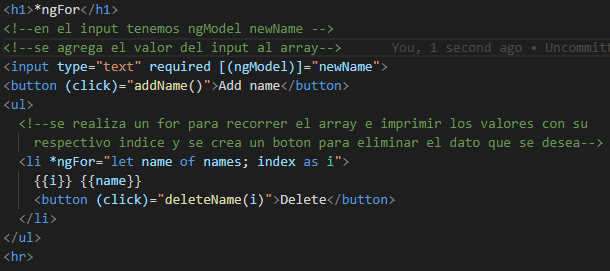
**Uso de \*ngIf y else**

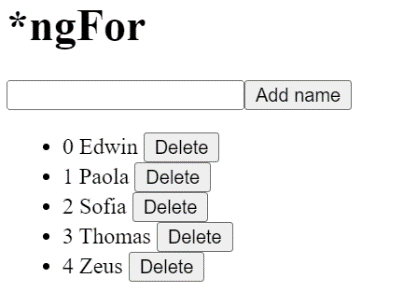
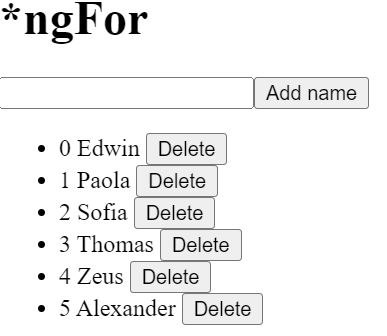
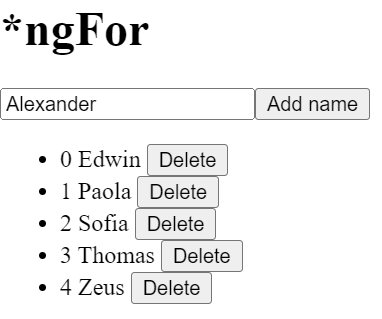
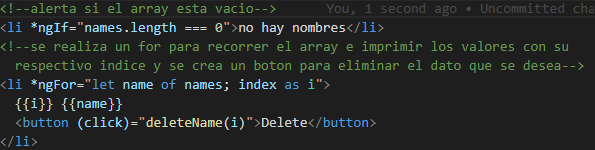


**Uso \*ngFor**



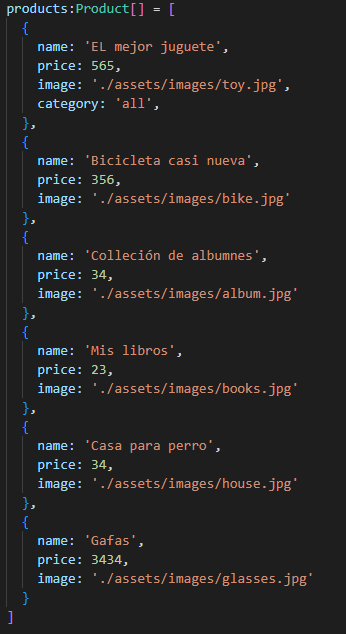
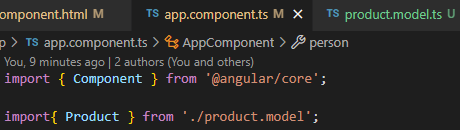
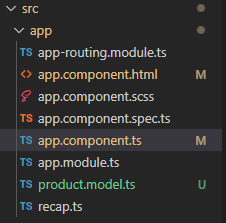


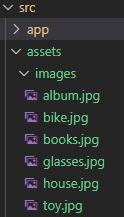
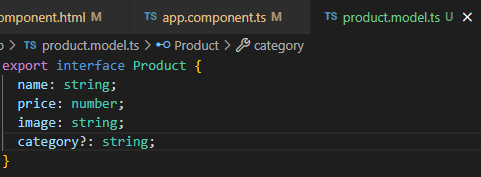


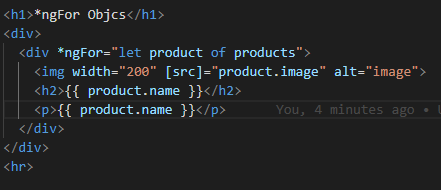
  


**Uso de \*ngFor Objects**

Se crea el archivo producto.model.ts para qrear una interfaz para poder tipar el objeto products, es importante exportar la interface para que pueda ser importada, en producto.model.ts el signo de interrogación en category significa opcional. Las imágenes se guardaron en src/assets/images







**Uso de ngSwitch**

