Primero tenemos que tener instalado angularcli, node y npm

para instalar angular cli

\*\* npm install -g angular-cli

crear la carpeta donde estara el proyecto de la tienda ubicada

por medio del cmd entrar en ella con cd

crear proyecto

\*\* ng new TiendaAngular2

despues de tener creado el proyecto entramos en la carpeta creada desde el mismo cmd y ejecutamos el siguiente comando para correr el servidor

\*\* ng serve

en el navegador abrir localhost:4200

en el editor de codigo comenzamos a editar lo que necesitamos.

en la carpeta app estara todo lo que vamos a editar por el momento

para poder continuar con el trabajo tienes que tener conocimiento intermedio a avanzado de typescript, por el momento me aventurare sin conocer de este...

lectura:

1. Verifica si tienes Node instalado.  
   a. Ejecuta en la terminal el siguiente comando.

user$ node -v

b. Si el resultado del comando anterior es un mensaje de error indicando que no existe dicha sentencia o no se reconoce, quiere decir que no está instalado node en la máquina.  
En este caso debes dirigirte a la página oficial de Node.js y descargar el archivo de instalación, haciendo click en LTS para la última versión estable. Posteriormente debes seguir el proceso de instalación.

https://nodejs.org/es/



c. Si el resultado del comando en a. es un mensaje que indica la versión de node y ésta inicia con un número menor a 4, debes realizar los mismos pasos en b. para instalar una versión más reciente.  
  
d. Si el resultado del comando en a. es un mensaje que indica la versión de node y ésta inicia con un número mayor o igual a 4, cuentas con la versión correcta de node.

1. Verifica que tienes instalado npm con la versión correcta  
   a. Ejecuta en la terminal el siguiente comando.

user$ npm -v

b. Si el resultado del comando anterior es un mensaje de error indicando que no existe dicha sentencia o no se reconoce, quiere decir que no está instalado npm en la máquina. Para instalarlo debes ejecutar el siguiente comando:

user$ npm install -g

c. Si el resultado del comando en a. es un mensaje que indica la versión de node y ésta inicia con un número menor a 3, debes ejecutar el siguiente comando:

user$ npm install npm@latest -g

d. Si el resultado del comando en a. es un mensaje que indica la versión de node y ésta inicia con un número mayor o igual a 3, cuentas con la versión correcta de npm.

1. Instala el asistente Angular CLI  
   a. Ejecuta en la terminal el siguiente comando

user$ npm install -g angular-cli

b. Espera unos cuantos minutos, puesto que la instalación de AngularCLI toma algún tiempo.  
c. Si durante la instalación observas un error que se presente, lee bien la descripción del mismo. Generalmente los errores que se presentan son relacionados con versiones incorrectas de node o npm.  
d. Al finalizar la instalación de manera exitosa se mostrará un mensaje y se muestra nuevamente el indicador de un nuevo comando en la terminal.

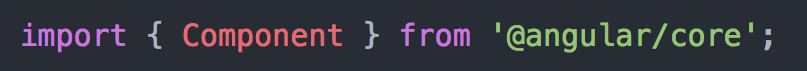
1. Comprueba la correcta instalación de AngularCLI creando un nuevo proyecto  
   a. El nuevo proyecto llevará el nombre “tareas” e indicamos un prefijo para todos los componentes generados a través de AngularCLI. Este prefijo se usa para evitar duplicidad en los selectores únicos de componentes para que no coincidan con elementos propios de HTML. Ejecuta el siguiente comando:

user$ ng new tareas --prefix t

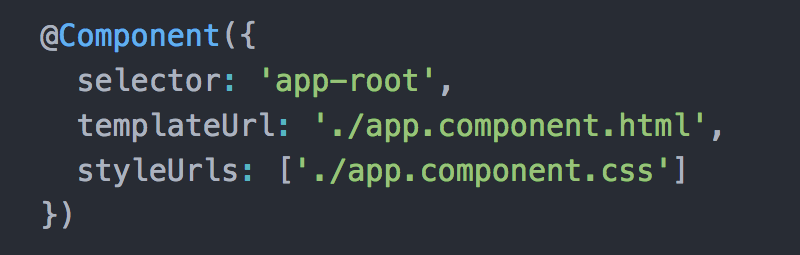
**Diferencias entre TypeScript y JavaScript plano**

Como ya lo mencionamos, TypeScript corresponde a un set adicional de herramientas para JavaScript. Dentro de las principales novedades que se encuentran en este lenguaje están:

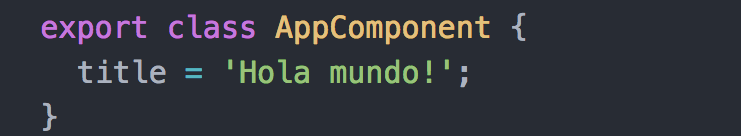
* Imports: la manera de importar código desde otro módulo es muy sencilla, haciendo que la modularidad en los proyectos sea fluida y eficiente.



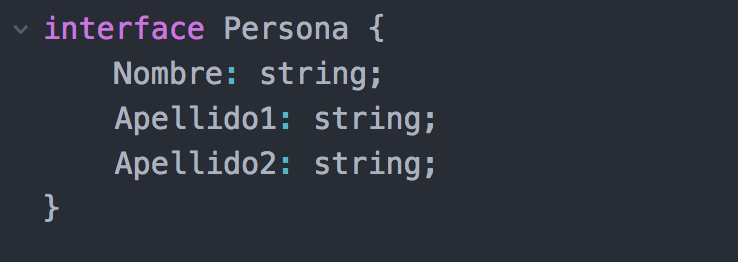
* Decoradores: los decoradores son funciones adjuntas a un objeto que realizan acciones en segundo plano y modifican el objeto al que están unidos para un fin específico.



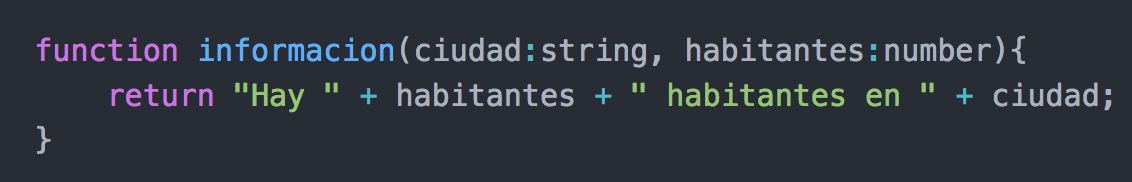
* Clases: la forma en que TypeScript define las clases permite que la programación orientada a objetos sea muy sencilla y correcta.



* Interfaces: elemento que no existe en JavaScript, sirve para forzar la implementación de atributos y métodos en una clase determinada.



* Tipos: como bien sabes, en JavaScript no se definen los tipos de datos para las variables creadas, y por lo tanto, éstas pueden tener cualquier tipo de valor. TypeScript añade un nivel adicional de seguridad en el código mediante la asignación de tipos a las variables o atributos de un objeto. De allí viene su nombre, ya que Type en inglés es Tipo.



**¿Por qué usar TypeScript?**

Si el código TyeScript es compilado en JavaScript plano, ¿por qué no hacerlo todo en JavaScript desde el principio? En realidad hay muchas razones por las cuales se usa TypeScript en el desarrollo de aplicaciones con Angular 2:

1. Lenguaje fuertemente tipado: TypeScript añade tipos de datos a variables y a atributos de objetos, esto aumenta el nivel de seguridad en el código evitando que valores inapropiados sean asignados y permitiendo fácilmente la detección de errores en la compilación, o que incluso el mismo IDE pueda identificar errores y alertas que te permitan, como desarrollador, detectar estas situaciones con gran facilidad.
2. Características de nueva generación: las herramientas de TypeScript mejoran en gran medida algunos elementos de JavaScript, lo que aumenta la calidad del código escrito y las posibilidades que se pueden alcanzar con él.
3. Nuevas características: TypeScript, además de mejorar lo existente en JavaScript, también añade nuevos elementos como las interfaces y los genéricos, ampliando la gama de herramientas para que el desarrollador cree scripts de alto rendimiento y funcionalidad.
4. Lenguaje principal de Angular 2: los creadores de Angular 2 eligieron a TypeScript como el lenguaje principal del framework, lo que significa que crear aplicaciones en Angular 2 usando TypeScript produce una fluidez y compatibilidad óptimas en la codificación y la compilación.
5. Documentación y soporte: al ser TypeScript el lenguaje principal para Angular 2, la gran mayoría de documentación y soporte que encontrarás por parte de la comunidad de desarrolladores, estará en dicho lenguaje; permitiendo encontrar ayuda de los expertos con gran facilidad.

Fin de lectura

Creamos los componentes

Los componentes son la vista y el controlador

Componentes que necesitamos

Carrito

Catalogo

Login

navbar

Detalleitem

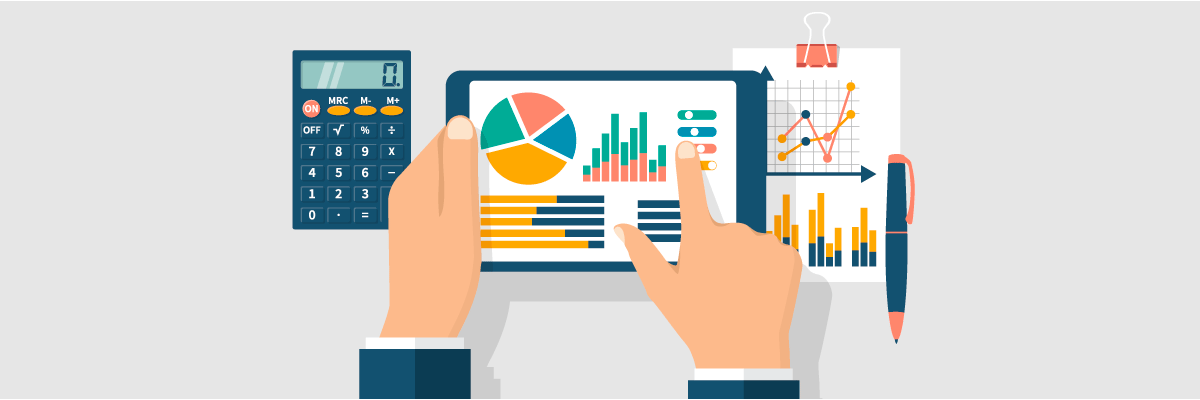
Dasboard

\*\* ng g component carrito

En app.component.ts colocamos todos los import de los componentes

Lectura

## **Databinding**

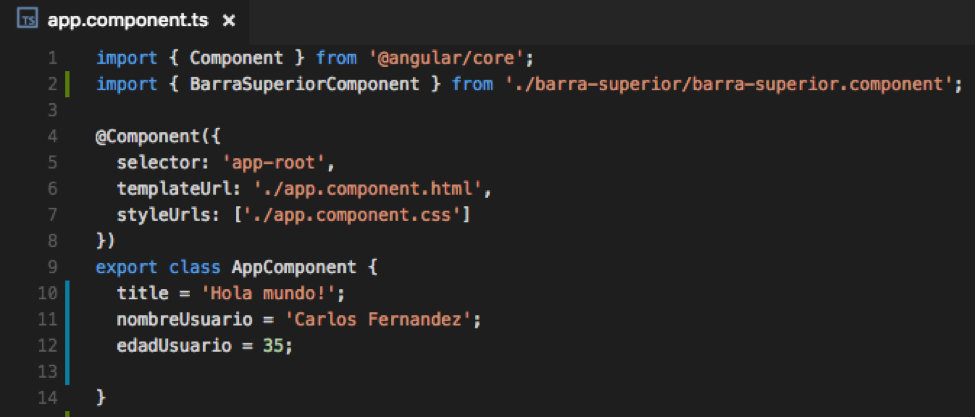


Una de las características principales en el desarrollo de aplicaciones usando Angular 2 es el concepto de databinding. Este término puede traducirse como enlace de datos y hace referencia a la comunicación que existe entre la vista y la lógica en una aplicación, es decir, la manera en la que se pasan datos desde las clases TypeScript a las plantillas HTML de los componentes.  
  
Existen 4 tipos de databinding:

* String Interpolation
* Property Binding
* Event Binding
* Two-way Binding

##### String Interpolation

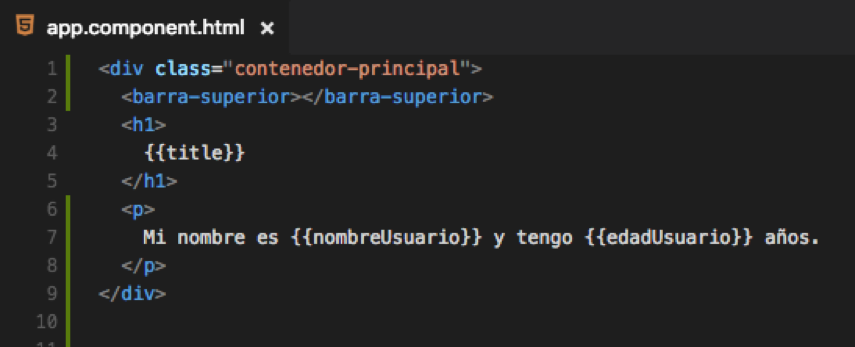
La interpolación de cadenas de caracteres, o como mejor se conoce por los desarrolladores que usan este framework string interpolation, es una forma de comunicación entre la lógica en la vista para pasar valores de tipo string o que pueden ser convertidos a string. A continuación, verás una clase en TypeScript correspondiente a un componente, en ella se define una variable con el nombre de **nombreUsuario,** y otra con el nombre de **edadUsuario,** y se le asignan valores a ambas.



Para realizar una interpolación de cadena se usa el operador doble llave y, en su interior, el nombre de la variable que queremos asignar a esa parte de la plantilla.

{{ variable }}

Ahora veamos la plantilla HTML correspondiente a este componente. Hemos creado un párrafo y, mediante el operador de doble llave, hemos indicado que deben interpolarse las variables **nombreUsuario y edadUsuario**.



Si vemos el resultado en el navegador, observamos que las variables se han interpolado y hemos pasado a la vista un valor asignado desde la clase TypeScript.



##### Property Binding:

El enlace de propiedades, o property binding, hace referencia a pasar datos desde la clase TypeScript de un componente a una plantilla HTML para asignar propiedades a los elementos de dicha plantilla. Funciona de manera muy similar a la interpolación de cadenas, con la diferencia que ésta se enfoca sólo en propiedades de los elementos HTML. Si añadimos, por ejemplo, un elemento de tipo input a la plantilla del componente app, podemos hacer un enlace de propiedad usando el operador corchete:

[nombre\_propiedad] = “variable o expresión”



Si vemos el resultado en el navegador, podemos observar que la propiedad value del campo de texto tomó el valor de la variable **nombreUsuario** asignada en la clase TypeScript del componente.



##### Event binding:

La asignación de eventos, o event binding como es mejor conocida, es el método por el cual se asignan eventos a los elementos de las plantillas HTML de los componentes, asociándolos con los métodos que deben disparar dichos eventos. Vamos a definir primero un método en la clase del componente app que muestre una alerta, indicando que se disparó el evento click.



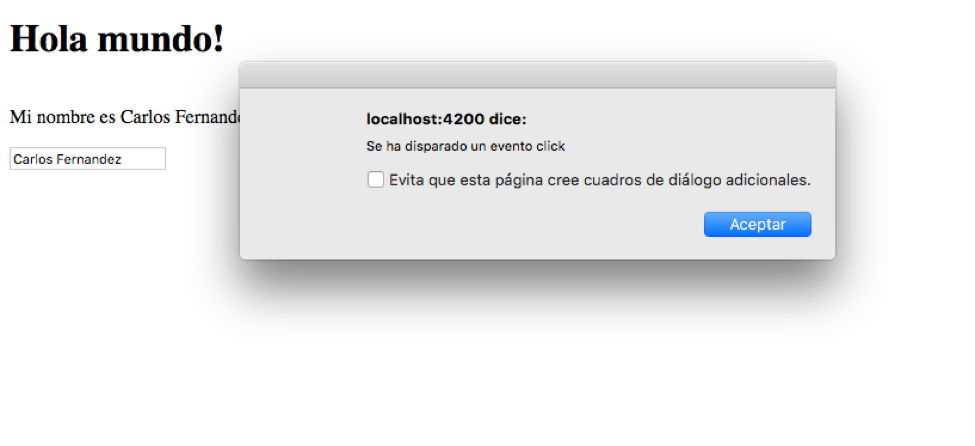
Ahora vamos a la plantilla HTML de este componente y realicemos un enlace de evento. Para enlazar un evento se usa el operador paréntesis para encerrar el nombre del evento que dicho elemento va a disparar, y se hace igual al método que debe ejecutarse al dispararse el evento indicado.

(nombre\_evento) = “método a ejecutar”

Por ejemplo, al elemento h1 de nuestra plantilla HTML vamos a asignarle un evento click, y lo hacemos igual al método que acabamos de crear: titleClicked().



Si probamos esto en el navegador, observamos que se ha asignado el evento y se dispara el método que creamos en la clase del componente.

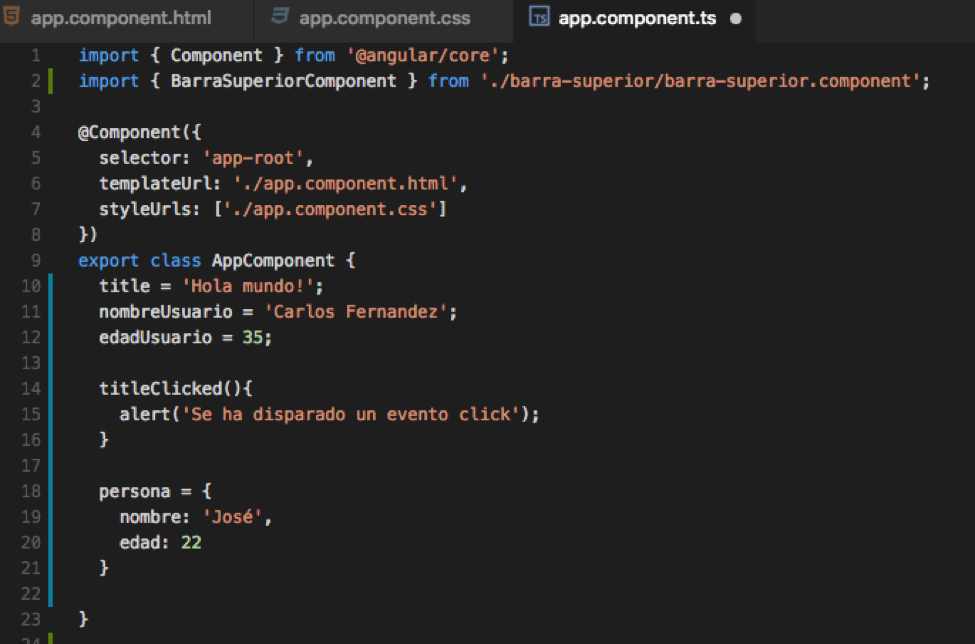


##### Two-way Binding:

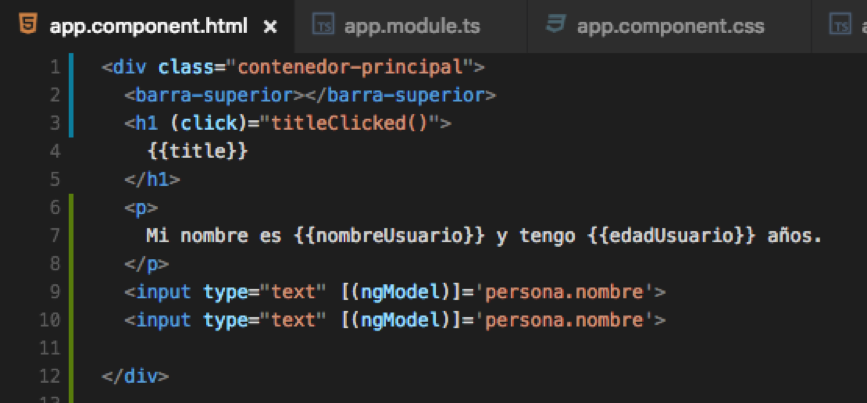
El enlace bidireccional, o two-way binding como es conocido en Angular 2, hace referencia a una combinación de los dos tipos de Databinding vistos previamente: property binding y event binding. El concepto bidireccional indica que se envían datos desde la clase TypeScript y de la misma manera se reciben desde la plantilla HTML. Para definir un enlace bidireccional se deben usar, tanto los corchetes como los paréntesis, y al interior usar la palabra NgModel. A continuación, se debe asignar una propiedad de algún objeto que interactúe con la asignación:

[(NgModel)] = “objeto.propiedad”

Añadimos a continuación un objeto a la clase del componente llamado persona y le definimos dos atributos, nombre y edad.



Ahora agregaremos dos elementos de tipo input a la plantilla HTML y en sus propiedades añadimos la especificación de un enlace bidireccional.



Si probamos esto en el navegador, podremos observar que, al modificar el valor de uno de los dos campos de texto, modifica el valor del otro, ya que un evento se envía al cambiar el valor del input, y al tiempo se envía el nuevo valor del atributo nombre de persona al otro campo de texto en la vista.

