# Typescript

* Typescript es un superset de Javascript, por lo tanto está escrito en Javascript.
* Es un agregado a JavaScript, no lo reemplaza. Algunas tareas las tendremos que hacer en Javascript.
* Agrega nuevas características:
  + Tipos
  + Clases
  + Interfaces
* Es transpilado a Javascript.
* Es el lenguaje con el que se programa para Angular4.

# Herramientas necesarias para utilizar Angular4

* Node.js
* Angular CLI

# Angular CLI

Es un set de herramientas de consola que sirve para crear proyectos y generar componentes, rutas, servicios, pipes, etc.

Permite correr nuestra app de forma local, hacer testing y preparar una app para producción.

<https://cli.angular.io/>

Para instalar Angular CLI se corre el siguiente comando:

**npm install -g @angular/cli**

# Crear un nuevo proyecto

**ng new <nombre de proyecto>**

Este comando crea la carpeta del proyecto y todas las subcarpetas y archivos necesarios.

# Correr proyecto

**ng serve**

Este comando (dentro de la carpeta de nuestro proyecto) transpila el código y corre un servidor en local.

# Módulos

Es la forma como se debe dividir un proyecto. Se declaran en app.module.ts, en la sección **imports** dentro de **@NgModule**.

# Componentes

Cada vista de la app deberá ser un componente distinto. Se declaran en app.module.ts, en la sección **declarations** dentro de **@NgModule**.

Cada componente deberá tener un **component.ts** en donde indica la información de nuestro componente. Esto se hace dentro del decorador **@Component**. Deberá tener obligatorio un **selector** y un **templateUrl,** opcional podemos tener un archivo de estilos en **styleUrls**. Para que se pueda utilizar ese componente se deberá exportar. Cuando se crea el proyecto con **ng new <nombre de proyecto>** se tiene por default un componente llamado app.component.ts.

# Data Binding

Comunicación entre el código que escribimos y lo que ve finalmente el cliente.

## Tipos

### String Interpolation {{}}

Tener TypeScript y presentar al usuario HTML.

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

a = 3;

b = 5;

}

**app.component.html**

<div style="text-align:center">

<h1>

Welcome to {{title}}!

</h1>

<h2>

{{a+b}}

</h2>

</div>

### Property Binding []

Tener información de HTML y que viaje a TypeScript.

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

listo = false;

constructor() {

setTimeout(()=>{this.listo = true;}, 3000);

}

}

**app.component.html**

<div style="text-align:center">

<h1>

Welcome to {{title}}!

</h1>

<hr/>

<button [disabled]="!listo">Click me!</button>

</div>

### Event Binding []

Tener evento en HTML y que viaje a TypeScript.

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

hacerAlgo() {

alert('Haciendo algo');

}

}

**app.component.html**

<div style="text-align:center">

<h1>

Welcome to {{title}}!

</h1>

<hr/>

<button (click)="hacerAlgo()">Click me!</button>

</div>

### Two Way Data Binding [()]

Comunicación de 2 vías.

**Ejemplo:**

**app.module.ts**

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';

import { FormsModule } from '@angular/forms';

@NgModule({

declarations: [

AppComponent

],

imports: [

BrowserModule,

FormsModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

nombre:string = '';

}

**app.component.html**

<div style="text-align:center">

<h1>

Welcome to {{title}}!

</h1>

<br/>

<input type="text" placeholder="Tu nombre" [(ngModel)]="nombre"/>

<br/>

Mi nombre es {{nombre}}

</div>

# Directivas

Forma elegante y rápida de manipular la información.

## Tipos

### Componentes

Directivas que siempre tienen asignado un template de HTML.

### Estructurales

Directivas que permite modificar el DOM.

#### \*ngFor

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

lugares:any = [

{nombre: 'Florería La Gardenia'},

{nombre: 'Donas La Pasadita'},

{nombre: 'Veterinaria Huellitas Felices'},

];

}

**app.component.html**

<ul>

<li \*ngFor="let lugar of lugares">{{lugar.nombre}}</li>

</ul>

#### \*ngIf

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

lugares:any = [

{active: true, nombre: 'Florería La Gardenia'},

{active: false, nombre: 'Donas La Pasadita'},

{active: true, nombre: 'Veterinaria Huellitas Felices'}

];

}

**app.component.html**

<ul>

<!--

No puede haber más de una directiva en un elemento,

pero se puede utilizar otra etiqueta llamada <ng-container>

para utilizar el ngFor.

-->

<ng-container \*ngFor="let lugar of lugares">

<li \*ngIf="lugar.active">{{lugar.nombre}}</li>

</ng-container>

</ul>

### Atributos

Directivas implementadas a través de atributos que pueden modificar el elemento al que están asignadas.

#### [ngStyle]

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

lugares:any = [

{active: true, nombre: 'Florería La Gardenia'},

{active: false, nombre: 'Donas La Pasadita'},

{active: true, nombre: 'Veterinaria Huellitas Felices'}

];

}

**app.component.html**

<ul>

<ng-container \*ngFor="let lugar of lugares">

<li [ngStyle]="{color:(lugar.active) ? 'black' : 'gray'}">{{lugar.nombre}}</li>

</ng-container>

</ul>

#### [ngClass]

**Ejemplo:**

**app.component.css**

.numero\_par{

background-color: aliceblue;

color: blue;

}

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

lugares:any = [

{nombre: 'Florería La Gardenia'},

{nombre: 'Donas La Pasadita'},

{nombre: 'Veterinaria Huellitas Felices'}

{nombre: 'Florería La Gardenia 2'},

{nombre: 'Donas La Pasadita 2'},

];

}

**app.component.html**

<ul>

<ng-container \*ngFor="let lugar of lugares; let i = index">

<li [ngClass]="{numero\_par: (i+1) % 2 !== 0}">{{lugar.nombre}}</li>

</ng-container>

</ul>

#### [ngSwitch]

**Ejemplo:**

**app.component.ts**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'JazzoSquare';

lugares:any = [

{cercania: 1, distancia: 1, active: true, nombre: 'Florería La Gardenia'},

{cercania: 1, distancia: 1.8, active: false, nombre: 'Donas La Pasadita'},

{cercania: 2, distancia: 5, active: true, nombre: 'Veterinaria Huellitas Felices'},

{cercania: 3, distancia: 10, active: true, nombre: 'Florería La Gardenia 2'},

{cercania: 3, distancia: 35, active: true, nombre: 'Donas La Pasadita 2'},

{cercania: 3, distancia: 120, active: false, nombre: 'Veterinaria Huellitas Felices 2'}

];

constructor() { }

}

**app.component.html**

<ul>

<ng-container \*ngFor="let lugar of lugares; let i = index" [ngSwitch]="lugar.cercania">

<li [ngStyle]="{color:(lugar.active) ? 'black' : 'gray'}" [ngClass]="{numero\_par: (i+1) % 2 !== 0}">{{lugar.nombre}} -

<span \*ngSwitchCase="1" [ngStyle]="{color:'green'}">Muy cercano</span>

<span \*ngSwitchCase="2" [ngStyle]="{color:'black'}">Cercano</span>

<span \*ngSwitchCase="3" [ngStyle]="{color:'red'}">Lejano</span>

</li>

</ng-container>

</ul>

## Crear nuestras propias directivas

1. Crear carpeta directives dentro de app.
2. Crear archivo <nombre-de-directiva>.directive.ts

import {Directive, OnInit, ElementRef, Renderer2, Input} from "@angular/core";

@Directive({

selector: '[resaltar]'

})

export class ResaltarDirective implements OnInit {

constructor(private elRef: ElementRef, private renderer: Renderer2) {}

@Input('resaltar') plan:string = '';

ngOnInit() {

if(this.plan === 'pagado') {

this.renderer.setStyle(this.elRef.nativeElement, 'background-color', 'yellow');

this.renderer.setStyle(this.elRef.nativeElement, 'font-weight', 'bold');

}

}

}

1. Agregarlo a nuestro app.module.ts

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';

import { FormsModule } from '@angular/forms';

import { AgmCoreModule } from '@agm/core';

import { ResaltarDirective } from './directives/resaltar.directive';

@NgModule({

declarations: [

AppComponent,

ResaltarDirective

],

imports: [

BrowserModule,

FormsModule,

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

1. Implementarla en app.component.html

<ul>

<ng-container \*ngFor="let lugar of lugares; let i = index" [ngSwitch]="lugar.cercania">

<li [ngClass]="{numero\_par: (i+1) % 2 !== 0}" [resaltar]="lugar.plan">{{lugar.nombre}} -

<span \*ngSwitchCase="1" [ngStyle]="{color:'green'}">Muy cercano</span>

<span \*ngSwitchCase="2" [ngStyle]="{color:'black'}">Cercano</span>

<span \*ngSwitchCase="3" [ngStyle]="{color:'red'}">Lejano</span>

</li>

</ng-container>

</ul>

# Instalar librerías externas con NPM

## Google Maps

1. Ir a <https://angular-maps.com/>
2. Getting started.
3. Seguir instrucciones de Angular Google Maps modificando app.mudule.ts, app.component.ts, app.component.html y app.component.css.

# Host Listeners

Ayuda para que cuando se crea una directiva a algún elemento HTML se puedan escuchar los eventos de una manera muy sencilla.

Ejemplo:

**contar-clicks.directive.ts**

import {Directive, HostListener} from "@angular/core";

@Directive({

selector: 'a[contar-clicks]'

})

export class ContarClicksDirective {

clickN = 0;

@HostListener('click', ['$event.target']) onClick(btn) {

console.log('a', btn, "Número de clicks: ", this.clickN++);

}

}

**app.module.ts**

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';

import { FormsModule } from '@angular/forms';

import { ContarClicksDirective } from './directives/contar-clicks.directive';

@NgModule({

declarations: [

AppComponent,

ContarClicksDirective

],

imports: [

BrowserModule,

FormsModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

**app.component.html**

<ul>

<ng-container \*ngFor="let lugar of lugares; let i = index" [ngSwitch]="lugar.cercania">

<a contar-clicks>

<li [ngStyle]="{color:(lugar.active) ? 'black' : 'gray'}" [ngClass]="{numero\_par: (i+1) % 2 !== 0}">{{lugar.nombre}} -

<span \*ngSwitchCase="1" [ngStyle]="{color:'green'}">Muy cercano</span>

<span \*ngSwitchCase="2" [ngStyle]="{color:'black'}">Cercano</span>

<span \*ngSwitchCase="3" [ngStyle]="{color:'red'}">Lejano</span>

</li>

</a>

</ng-container>

</ul>

# Host Binding

Permite modificar el elemento del DOM desde la propia directiva.

Ejemplo:

**contar-clicks.directive.ts**

import {Directive, HostListener, HostBinding} from "@angular/core";

@Directive({

selector: 'li[contar-clicks]'

})

export class ContarClicksDirective {

clickN = 0;

@HostBinding('style.opacity') opacity: number = .1;

@HostListener('click', ['$event.target']) onClick(btn) {

console.log('a', btn, "Número de clicks: ", this.clickN++);

this.opacity += .1;

}

}

**app.module.ts**

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';

import { FormsModule } from '@angular/forms';

import { ContarClicksDirective } from './directives/contar-clicks.directive';

@NgModule({

declarations: [

AppComponent,

ContarClicksDirective

],

imports: [

BrowserModule,

FormsModule,

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

**app.component.html**

<ul>

<ng-container \*ngFor="let lugar of lugares; let i = index" [ngSwitch]="lugar.cercania">

<a>

<li contar-clicks [ngStyle]="{color:(lugar.active) ? 'black' : 'gray'}" [ngClass]="{numero\_par: (i+1) % 2 !== 0}" [resaltar]="lugar.plan">{{lugar.nombre}} -

<span \*ngSwitchCase="1" [ngStyle]="{color:'green'}">Muy cercano</span>

<span \*ngSwitchCase="2" [ngStyle]="{color:'black'}">Cercano</span>

<span \*ngSwitchCase="3" [ngStyle]="{color:'red'}">Lejano</span>

</li>

</a>

</ng-container>

</ul>

# Bootstrap

<https://getbootstrap.com>

## Instalar

npm install bootstrap (Correr comando dentro de la carpeta de nuestro proyecto).

## Configurar

1. Modificar **.angular-cli.json** agregando **bootstrap.min.css** en la sección **styles**.

"styles": [

"../node\_modules/bootstrap/dist/css/bootstrap.min.css",

"styles.css"

],

1. Reiniciar la ejecución del proyecto.
2. Usar componentes (ver ejemplos en página oficial).