1.- Introducción

Esta guía mostrará cómo construir y ejecutar una aplicación en Angular para su versión 8. Se revisarán los conceptos básicos para poder ejecutar una aplicación con Angular y al final de esta guía seremos capaces de hacer lo siguiente:

- Usar las directivas de Angular para mostrar y ocultar elementos.
- Crear componentes en Angular para trabajar con los datos.
- Usar *one-way data binding* para mostrar datos.
- Añadir campos editables para actualizar un modelo con *two-way data binding*.
- Enlazar métodos de los componentes para usar eventos.
- Mostrar datos con *pipes*.
- Crear servicios para recuperar datos del servidor.
- Usar *routing* para navegar por las diferentes vistas y sus componentes.
- Crear una aplicación multimodular.
- Crear formularios.
- Dar estilo a nuestra aplicación

1.1.- ¿Qué es una página Single-Page application (SPA)?

Se trata de una aplicación web o sitio web donde solamente se carga el contenido de dicha web en la primera petición al servidor: HTML, CSS y Javascript. Con esto se consigue dar una experiencia más fluida a los usuarios como una aplicación de escritorio.

Los retos a los que se enfrenta una aplicación SPA son:

- El enrutado: saber qué contenido mostrar dependiendo de la URL.
- Interfaz de usuario dinámica: posibilidad de actualizar la página cuando recibamos nuevos datos del servidor de manera fluida y rápida sin que dé la sensación de que se ha cargado una nueva página.
- Acceso a datos ya sean en loca, localStorage y sessionStorage, o peticiones a servidor.

1.2.- Pero.. ¿porqué Angular?

Angular es un *framework* desarrollado en *Typescript* de código abierto que construye aplicaciones en HTML y JavaScript. Fue desarrollado por Google. Este *framework* se utilizó para superar los obstáculos encontrados al trabajar con aplicaciones *Single Page*. El lanzamiento inicial de este *framework* fue en octubre de 2010.

Principales características de Angular:

- Velocidad y rendimiento.

    - El código de la app se puede convertir en código optimizado con las herramientas que Angular posee, ejemplo Ivy.
    - Se puede ejecutar bajo cualquier servidor que renderice una web, el más común y usado es node.js
    - División del código para que la web solo cargue el código de la vista que se esté visualizando en ese momento.
- Productividad.
    - Creación de componentes, módulos o servicios de forma rápida.
    - Angular CLI. Herramienta de línea de comandos que permite crear proyectos nuevos, elementos y realizar test.
    - IDEs. Integración con la mayoría de editores populares.

1.3.- Requisitos previos

1.3.1.- Typescript

Typescript es un lenguaje de programación libre y de código abierto desarrollado por Microsoft. Extiende la sintaxis de Javascript, esencialmente añade tipos estáticos y objetos basados en clases, esto quiere decir que pasamos de un lenguaje abierto a todo a un lenguaje más tipado y orientado a objetos.

Pero al tratarse de un superconjunto de Javascript podremos hacer funcionar cualquier código de Javascript en Typescript. Esto quiere decir que podremos crear proyectos al más puro estilo de Javascript o crear proyectos más tipados y orientados a objetos.

Tipos básicos

String	Number	Array
Tuple (similar al Array)	Enum	Any
Void	Boolean	Never (valores que nunca se producen)

String and Number

```
let color: string = "blue";
color = 'red';
```

```
let fullName: string = `Bob Bobbington`;
let age: number = 37;
let sentence: string = `Hello, my name is ${ fullName }.

I'll be ${ age + 1 } years old next month.`;
```

```
let sentence: string = "Hello, my name is " + fullName + ".\n\n" +
    "I'll be " + (age + 1) + " years old next month.";
```

Array

```
let list: number[] = [1, 2, 3];
```

```
let list: Array<number> = [1, 2, 3];
```

Tuple

```
// Declare a tuple type
let x: [string, number];
// Initialize it
x = ["hello", 10]; // OK
// Initialize it incorrectly
x = [10, "hello"]; // Error
```

```
console.log(x[0].substring(1)); // OK
console.log(x[1].substring(1)); // Error, 'number' does not have 'substring'
```

```
x[3] = "world"; // Error, Property '3' does not exist on type '[string, number]'.

console.log(x[5].toString()); // Error, Property '5' does not exist on type '[string,
number]'.
```

Any

```
let notSure: any = 4;
notSure = "maybe a string instead";
notSure = false; // okay, definitely a boolean
```

```
let notSure: any = 4;
notSure.ifItExists(); // okay, ifItExists might exist at runtime
notSure.toFixed(); // okay, toFixed exists (but the compiler doesn't check)

let prettySure: Object = 4;
prettySure.toFixed(); // Error: Property 'toFixed' doesn't exist on type 'Object'.
```

```
let list: any[] = [1, true, "free"];

list[1] = 100;
```

Clases

```
class Greeter {
    greeting: string;
    constructor(message: string) {
        this.greeting = message;
    }
    greet() {
        return "Hello, " + this.greeting;
    }
}

let greeter = new Greeter("world");
```

```
class Animal {
    move(distanceInMeters: number = 0) {
        console.log(`Animal moved ${distanceInMeters}m.`);
    }
}

class Dog extends Animal {
    bark() {
        console.log('Woof! Woof!');
    }
}

const dog = new Dog();
dog.bark();
dog.move(10);
dog.bark();
```

Funciones

```
// Named function
function add(x, y) {
    return x + y;
}

// Anonymous function
let myAdd = function(x, y) { return x + y; };
```

```
function add(x: number, y: number): number {
    return x + y;
}

let myAdd = function(x: number, y: number): number { return x + y; };
```

Parámetros opcionales

```
function buildName(firstName: string, lastName?: string) {
    if (lastName)
        return firstName + " " + lastName;
    else
        return firstName;
}

let result1 = buildName("Bob");                  // works correctly now
let result2 = buildName("Bob", "Adams", "Sr."); // error, too many parameters
let result3 = buildName("Bob", "Adams");         // ah, just right
```

1.3.2.- Instalar Node.js y Angular 8

Para poder trabajar con Angular 8 debemos tener instalado:

- Node.js en versiones superiores a la 10.9.0. Se aconseja usar el gestor nvm, node version manager, para instalar la versión de node.js que necesitemos.
- Npm, node package manager, que se instala con la versión de Node.js

Instalar Angular CLI

```
npm install -g @angular/cli@8
```

Angular CLI

Angular CLI es una interfaz de línea de comandos que nos permitirá inicializar, crear, desarrollar una aplicación de Angular así como realizar pruebas y el despliegue de nuestra aplicación.

Los comandos más usados son:

add	Añade soporte para una librería externa	build	Compila nuestra aplicación para su uso en un servidor
config	Añade o modifica los valores de configuración de Angular, también se pueden ver en el angular.json	generate	Añade o modifica archivos
help		new	Crea un nuevo proyecto
serve	Construye y ejecuta la aplicación a nuestro servidor local	test	
update			

Crear un proyecto nuevo

```
ng new cursoAngular8
```

En la instalación de nuestro nuevo proyecto nos irá marcando las diferentes características que queremos añadir o no.

1.3.3.- Ejecutar una aplicación

```
ng serve
```

El comando anterior ejecutará el servidor y permitirá construir y ejecutar las aplicaciones en Angular y reconstruirlas cuando hagamos cambios para que se reflejen.

El servidor se lanzará bajo la dirección http://localhost:4200/ por defecto pero podremos cambiar este valor añadiendo al comando lo siguiente:

```
ng serve --port 4210
```