



Fundamentos de Angular

Contenidos del Curso



1. Introducción a Angular
2. Iniciando con angular: Instalaciones, estructura de proyectos, importar librerías
3. Componentes
4. Data binding
5. Comunicación entre componentes
6. Directivas
7. Servicios
8. Routing

¿Que es Angular?



Angular es un framework de desarrollo, construido sobre TypeScript. Angular incluye:

- ▶ Un Framework basado en componentes para crear aplicaciones web escalables.
- ▶ Una colección de bibliotecas bien integradas que cubren una amplia variedad de características, que incluyen enrutamiento, administración de formularios, comunicación cliente-servidor y más.
- ▶ Un conjunto de herramientas para desarrolladores que le ayudarán a desarrollar, compilar, probar y actualizar su código.

¿Qué es TypeScript?



- ▶ TypeScript es un lenguaje de código abierto que se basa en JavaScript, una de las herramientas más utilizadas del mundo, al agregar definiciones de tipos estáticos.
- ▶ Los tipos proporcionan una forma de describir la forma de un objeto, proporcionando una mejor documentación.
- ▶ Todo el código JavaScript válido también es código TypeScript.

Ventajas de Angular



- ▶ Multiplataforma
- ▶ Velocidad y rendimiento
- ▶ Productividad
- ▶ Arquitectura basada en componentes
- ▶ Data binding
- ▶ Directivas
- ▶ Diseño material
- ▶ Comunidad

Cuando usar Angular

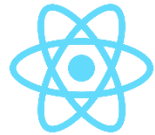


- ▶ Proyectos empresariales o de gran escala
- ▶ Aplicaciones con alto contenido dinámico
- ▶ Aplicaciones web progresivas (PWA)

Frameworks similares



► React



► Vue.js



► Aurelia



► Backbone



Requisitos



- ▶ HTML.
- ▶ CSS, SCSS, SASS, LESS.
- ▶ JavaScript.



Componentes

APP

```
graph TD; APP[APP] --- C1[Componente 1]; APP --- C2[Componente 2]; C1 --- HTML[HTML]; C1 --- CSS[CSS]; C1 --- TS[TypeScript]; C2 --- C3[Componente 3];
```

The diagram illustrates the structure of an application (APP). It is divided into two main components: Componente 1 and Componente 2. Componente 1 is further divided into three sub-components: HTML, CSS, and TypeScript. Componente 2 contains a single sub-component, Componente 3.

Componente 1

HTML

CSS

TypeScript

Componente 2

Componente 3



Data binding

TypeScript
(Logica)

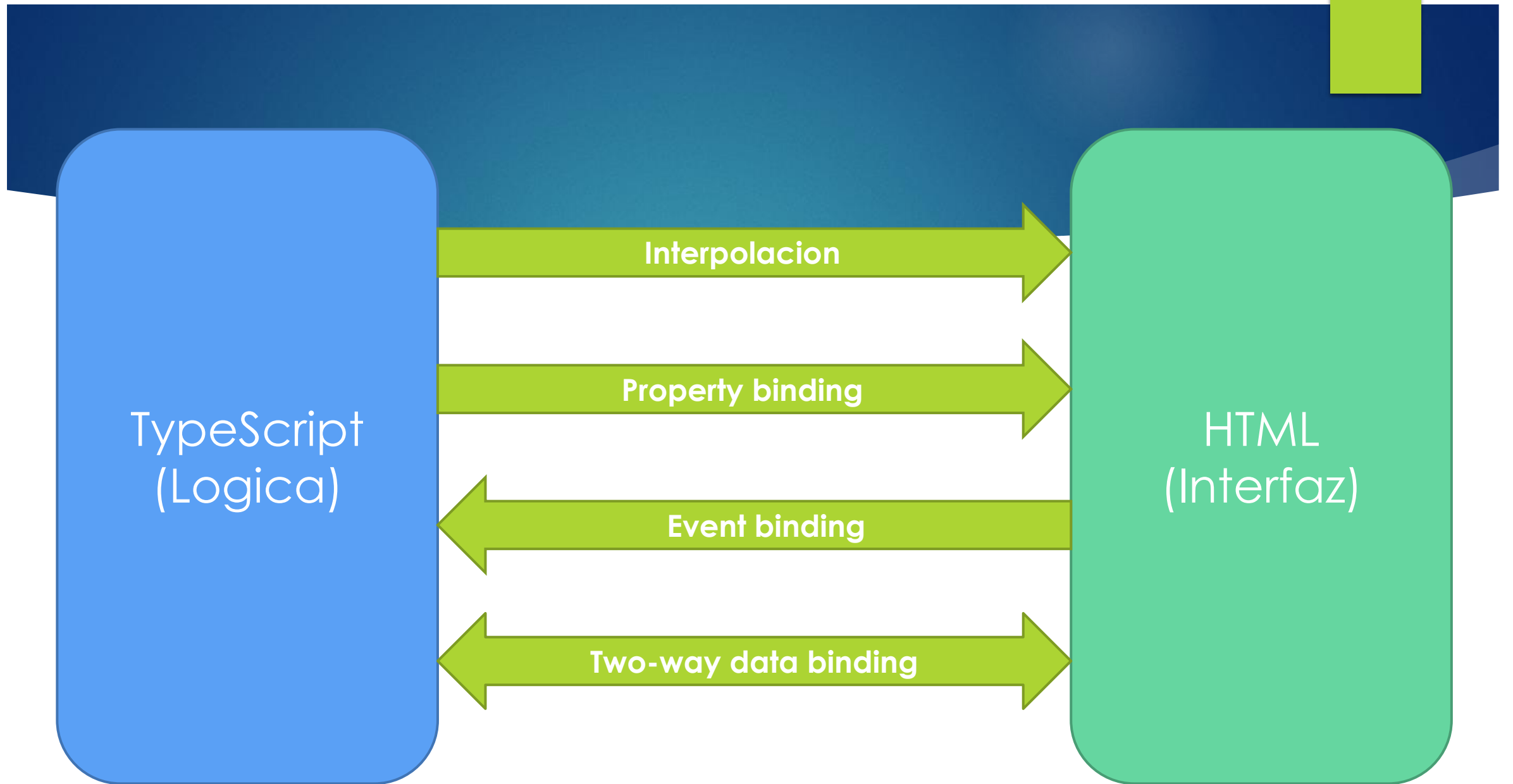
Interpolacion

Property binding

Event binding

Two-way data binding

HTML
(Interfaz)





Comunicación entre componentes

APP

```
graph TD; subgraph APP; direction LR; subgraph C1 [Componente 1]; direction TB; C2[Componente 2]; C1_in["@Input"]; C1_out["@Output"]; C1_in --> C2; C2 --> C1_out; end; C3[Componente 3]; C1 <-->|Servicios| C3; end;
```

The diagram illustrates the architecture of an application (APP). It is divided into three main components: Componente 1, Componente 2, and Componente 3. Componente 1 is a blue rounded rectangle containing Componente 2, which is a green rounded rectangle. A thick green arrow labeled '@Input' points from Componente 1 down to Componente 2, and another thick green arrow labeled '@Output' points from Componente 2 up to Componente 1. Componente 3 is a blue rounded rectangle located to the right of Componente 1. A thick green double-headed arrow labeled 'Servicios' connects Componente 1 and Componente 3, indicating a bidirectional flow of services between them. The entire structure is contained within a larger green rounded rectangle labeled 'APP' at the top.

Componente 1

Componente 3

Servicios

@Input

@Output

Componente 2