

{ Angular }

Framework para la creación de Aplicaciones Web y Apps

Formador: Ezequiel Llarena Borges

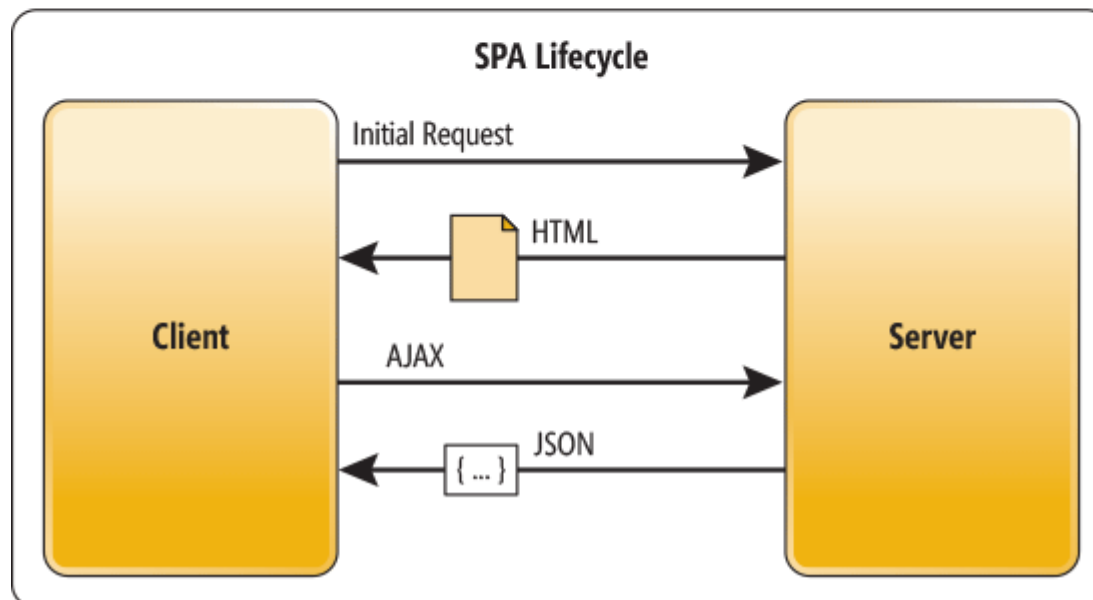
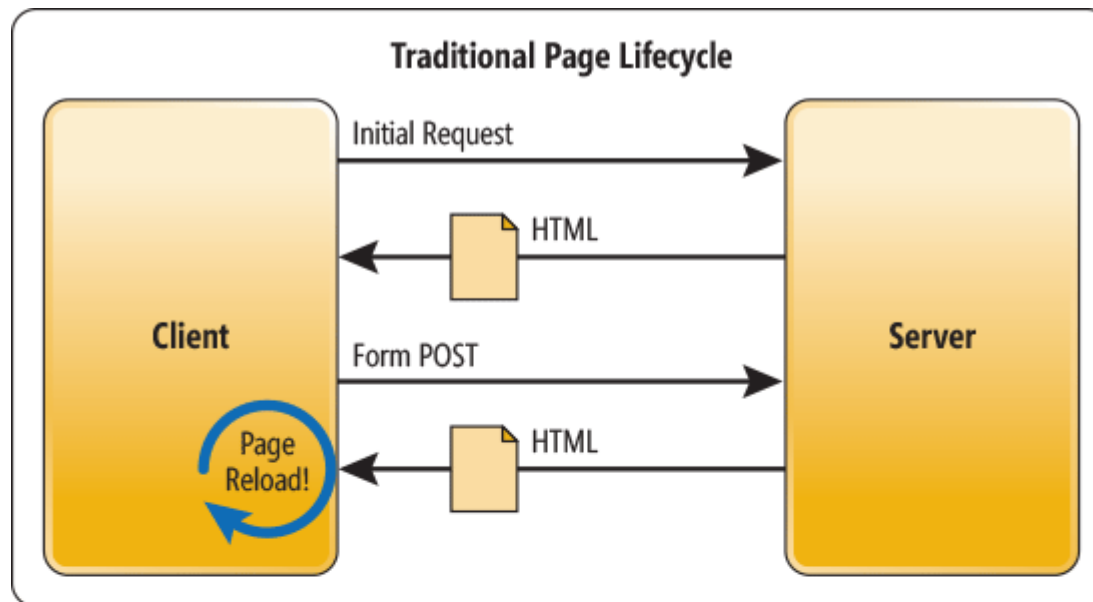
{ Angular }

Introducción, Instalación y Arquitectura de un proyecto Angular 4

Formador: Ezequiel Llarena Borges

Características

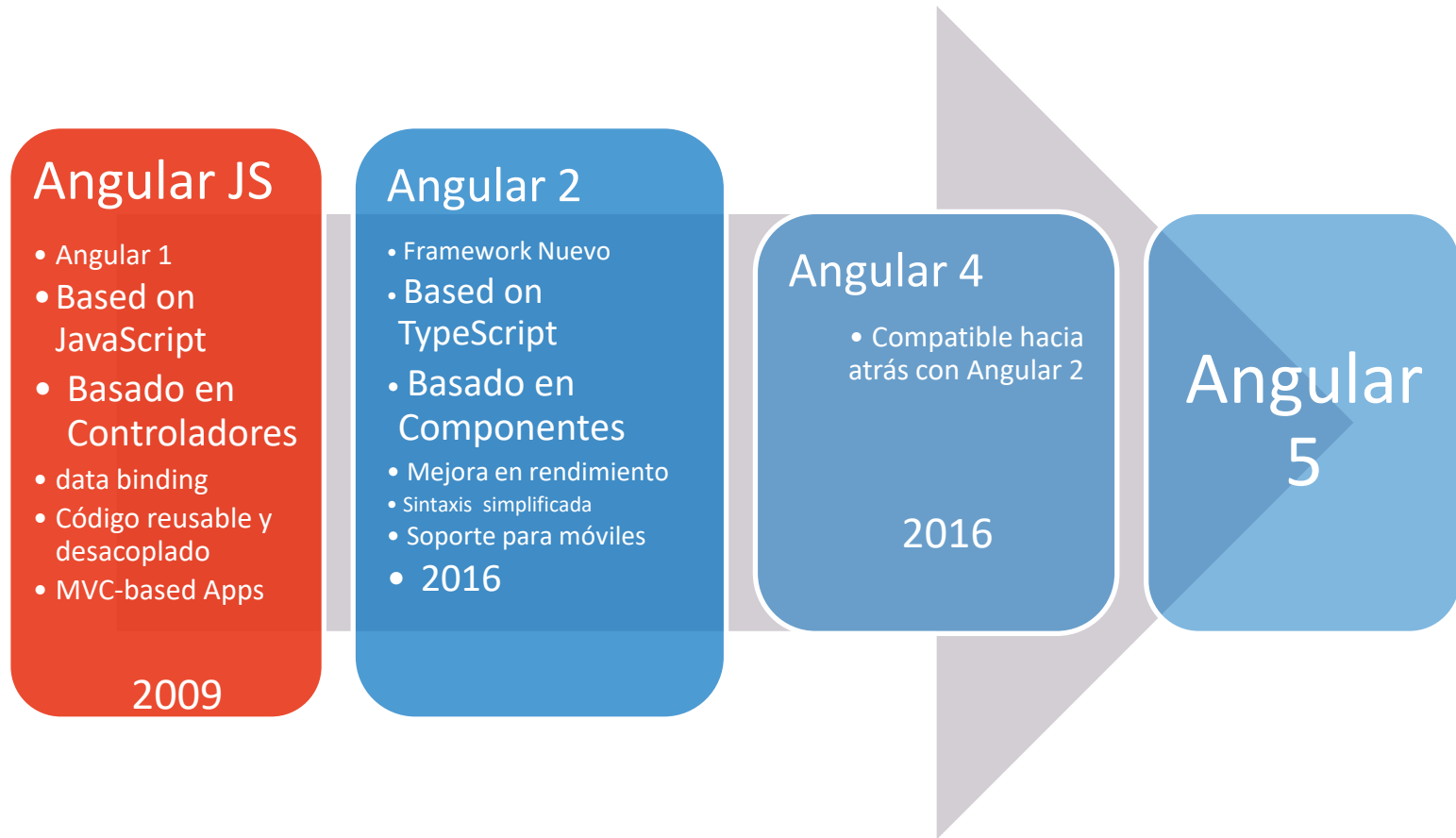
- Framework desarrollado y soportado por **Google**
- Proyectos **SPA** (Single-Page Application): carga de datos **asíncrona**
- Basado en el lenguaje **TypeScript** (desarrollado por Microsoft)
- TypeScript **basado en JavaScript** mejorado (funciones tipado y POO)



Funcionamiento

- **Componentes** (controlador en Angular JS) que representan la vista/presentación de la webapp
- Componentes soportados por **Clases**
- Utiliza **TypeScript** para lógica de las aplicaciones
- Sintaxis y desarrollo incorpora mejoras y características de **ECMAScript 6** (estandarización de JavaScript)
- Angular **transpila** el código TypeScript directamente a JavaScript ya que ECMAScript 6 aún no es soportado por los navegadores web
- Permite **trabajar en tiempo real** sobre la aplicación (podemos levantarla mediante un servidor disponible con **Node JS**)

Angular JS vs Angular



TypeScript

- 85% JavaScript normal
- Nuevas características de EcmaScript 6 (ES6)
- Tipado fuerte
- Orientación a Objetos mejorada
- Lenguaje interpretado (se transpila a JS)
- Extensión de los archivos .ts
- Superset de JS desarrollado por MicroSoft

API Angular

<https://angular.io/api>

Herramientas de desarrollo

- **Node JS** (Entorno de ejecución para JavaScript)
- **NPM** (Gestor paquetes; *el Maven de Angular*)
- **Angular CLI** (Agiliza y facilita instalación de Angular + Inicio proyecto Angular)
- **Sublime Text / Visual Studio Code** (Editor de código / IDE)
- **Angular** (Framework)

Node JS

- Librería y Entorno de ejecución del lado del **servidor**
- Basado en **eventos** → Programación **Asíncrona**
- Compila y ejecuta JavaScript
- Utiliza motor V8 desarrollado por Google
- Open source (código abierto)

Pasos

1. Instalar **Node JS**
2. Instalar Angular con *quickstart* de su repositorio GitHub
3. “Hola Mundo” de Angular
4. Crear proyecto con **Angular CLI**
5. Componentes: **AppComponent**
6. Creación de componentes
7. Trabajar con múltiples componentes (**Binding** y **Selector Tag**)

Comandos básicos Angular CLI

- `npm -v / npm --version`
- `node -v / node --version`
- `tsc -v`
- `ng new <nombre_proyecto>` Crear aplicación nueva
- `ng serve [--port <puerto>|--open]` Lanzar aplicación en el servidor

{ Angular }

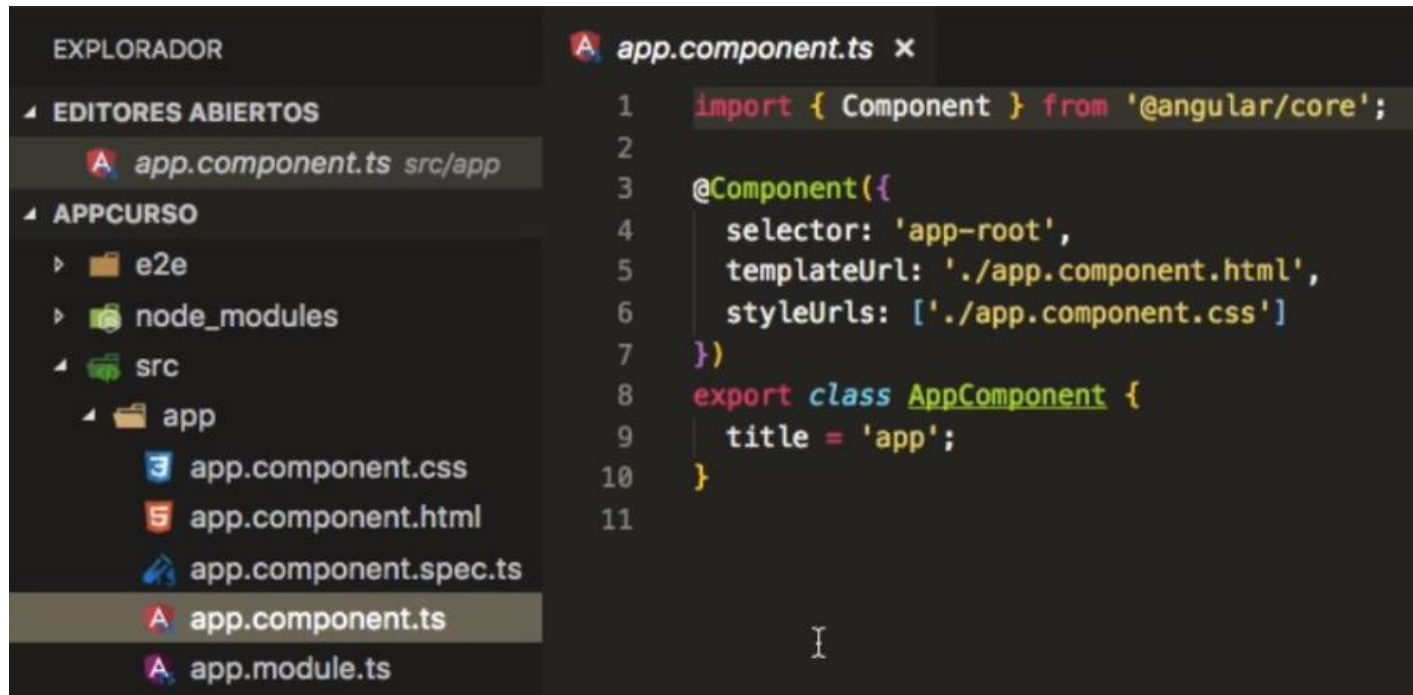
Componentes

Formador: Ezequiel Llarena Borges

Componente

Conjunto de archivos que realizan una determinada función en la aplicación
(nombre_comp.component.ts)

- .html** (Plantilla del componente)
- .ts** (Lógica de negocio del componente)
- .css** (Estilos del componente)
- .spec.ts** (Testing)



```
1  import { Component } from '@angular/core';
2
3  @Component({
4    selector: 'app-root',
5    templateUrl: './app.component.html',
6    styleUrls: ['./app.component.css']
7  })
8  export class AppComponent {
9    title = 'app';
10 }
11
```

Componente

```
app.component.ts x
1  import { Component } from '@angular/core';
2
3  @Component({
4    selector: 'app-root',
5    templateUrl: './app.component.html',
6    styleUrls: ['./app.component.css']
7  })
8  export class AppComponent {
9    title = 'app';
10 }
```

Importa la clase **Component** del paquete de Angular instalados en la carpeta **node_modules**:

Importaciones = Aplicaciones ligeras

Comandos Angular CLI

ng generate component <nuevo-componente>

- Crear un nuevo componente

ng g c <nuevo_comp> --spec false

- Versión simplificada anterior
- Omite archivo de pruebas `app.component.spec.ts`

{ Angular }

Data Binding

Formador: Ezequiel Llarena Borges

Data Binding

One Way Binding

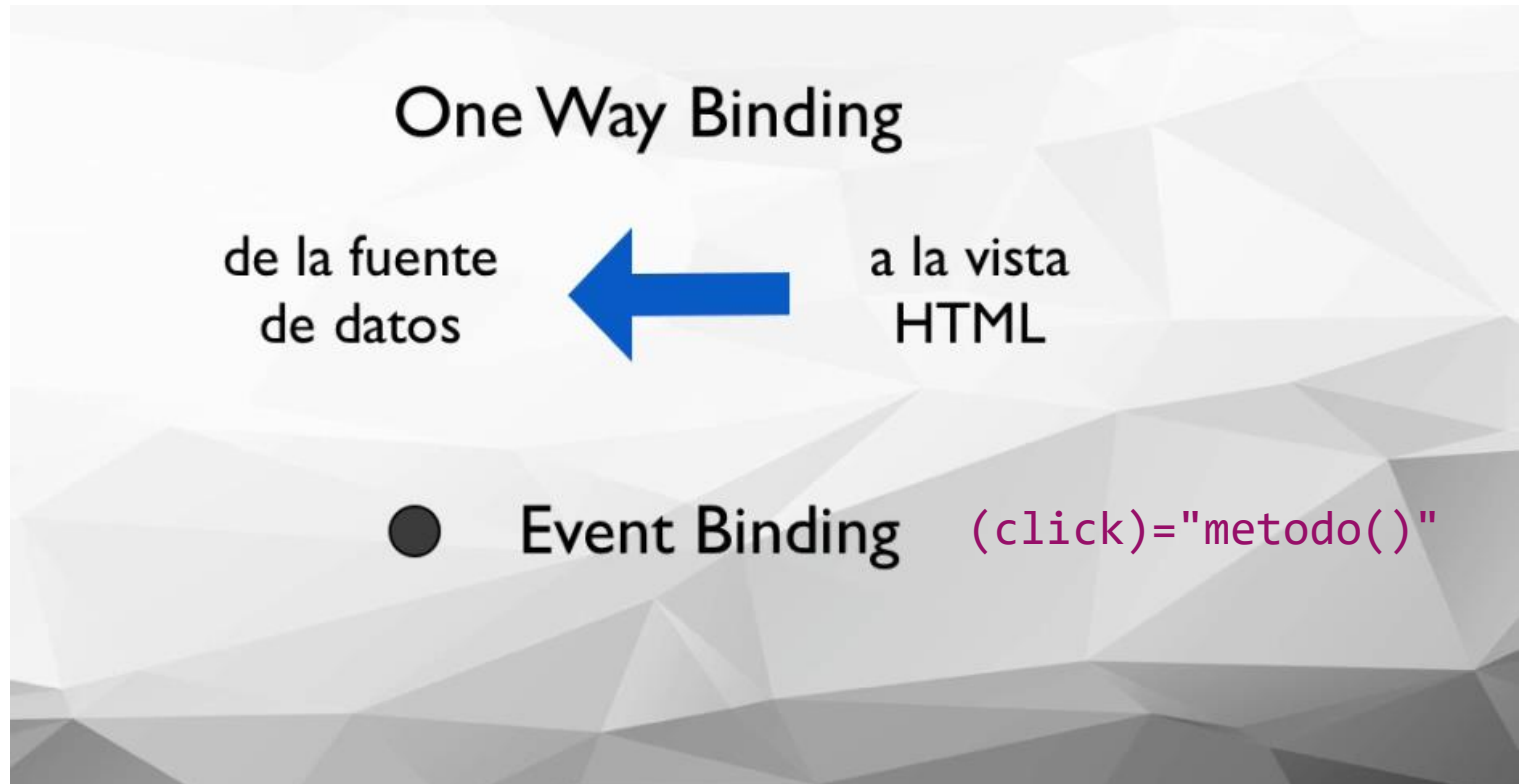
de la fuente
de datos



a la vista
HTML

- Interpolación `{{ data }}`
- Property Binding `[placeholder]="data"`

Data Binding



Data Binding

Two Way Binding

de la fuente
de datos



a la vista
HTML

- Two Way Binding `[(ngModel)]="data"`



Interpolación de strings vs Binding a Propiedad

```
<p>  </p>
```



```
<p> <img [src]="urlImagen"> </p>
```

```
<ul>
```

```
<li [class]="valorClass">item 1</li>
```

```
<li class="{{valorClass}}">item 2</li>
```

```
</ul>
```



```
<p>{{texto}}</p>
```

```
<p [innerHTML]="texto"></p>
```



```
<button [disabled]="activo">Clic aquí</button>
```



```
<button disabled="{{activo}}">Clic aquí</button>
```



{ Angular }

Directivas

Formador: Ezequiel Llarena Borges

Directivas

Clases Angular con código para crear, formatear e interaccionar con elementos HTML con el DOM.

Tipos:

- **Componentes** (@Component)
- **Estructurales** (modifican el DOM del elemento HTML en el que se encuentran)
- **Atributos** (funcionan como un atributo HTML, similar al *property binding*)

Directivas Estructurales y de Atributos

`*ngIf`

`*ngIf + else`

`[ngStyle]`

`[ngClass]`

`*ngFor`

`[ngSwitch]`

{ Angular }

Formularios

Formador: Ezequiel Llarena Borges

Técnicas de control de formularios

- **Template Driven**
 - Lógica de captura de datos + Validación desde HTML
- **Reactive**
 - Gestión del Formulario desde TypeScript
 - Mayor control de la gestión de formularios
 - En tiempo real se van actualizando los valores de los campos del formulario
 - Clases: **FormControl**, **FormGroup**, **FormBuilder**

Validación HTML

- Por defecto, Angular elimina la validación HTML5
- **NgForm** tiene su propio sistema de validación mediante **Estados**:
 - Dirty
 - Pristine
 - Touched
 - Valid
 - Invalid