Clase 3

Data binding y directivas

Interpolación (Componente a plantilla)

- Vincula los datos del componente a la plantilla.
- Unidireccional **one-way binding**, lo que significa que los cambios en los datos del componente se reflejan en la vista, pero no al revés.
- Se utilizan llaves dobles `{{ }}`

```
export class HeaderComponent {
  companyName = 'Dev Senior'
  userName = 'Admin'
}
```

```
<header class="dashboard-header">
  <div class="logo-section">
    <h1>
      <strong>{{ companyName }} Dashboard</strong>
    </h1>
  </div>
  <div class="header-stats">
    <span class="user">Bienvenido, {{ userName }}</span>
  </div>
</header>
```

Property Binding (Componente a plantilla)

- Permite asignar una propiedad de un elemento HTML (como `src`, `href` o `disabled`) al valor de una propiedad del componente.
- También es una vinculación unidireccional **one-way binding** del componente a la plantilla.
- La propiedad del elemento se encierra entre corchetes `[ ]`.

```ts
export class TeamMemberCardComponent {
  member = {
    name: 'Carlos Rodríguez'
    avatar: 'https://ui-avatars.com/api/?name=Carlos+Rodriguez'
  }
  isFavoriteBtnDisabled = true
}
```

```html
<div class="member-card">
  <div class="card-header">
    <img [src]="member.avatar" [alt]="member.name" class="avatar" />
    <div class="basic-info">
      <h3>{{ member.name }}</h3>
    </div>
    <button class="favorite-btn" [disable]="isFavoriteBtnDisabled">⭐</button>
  </div>
</div>
```

Event Binding (Plantilla a componente)

- Detecta las acciones del usuario en la plantilla (como clics, movimientos del ratón o pulsaciones de teclas) y, como respuesta, llama a un método del componente.
- Esta vinculación es unidireccional **one-way binding** entre la plantilla y el componente
- El nombre del evento se escribe entre paréntesis `( )`

```
export class TeamMemberCardComponent {
  onToggleFavorite(event: Event) {
    event.stopPropagation(); // Prevent card click
    console.log('Clic en marcar/desmarcar favorito');
  }
}
```

```
<div class="member-card">
  <div class="card-header">
    <div class="card-actions">
      <button (click)="onToggleFavorite($event)">
        ★
      </button>
    </div>
  </div>
</div>
```

ngModel

- El enlace bidireccional **two-way binding** es una sincronización continua de datos entre el componente y la plantilla.
- Combina el enlace de propiedades `[ ]` y el enlace de eventos `( )` en una única notación conocida como "banana in a box" `[( )]`.
- e usa con mayor frecuencia con entradas de formulario.

```
export class TeamMemberCardComponent {
  searchTerm: string = '';

  if (this.searchTerm.trim()) {
    filtered = filtered.filter(m =>
      m.name.toLowerCase().includes(this.searchTerm.toLowerCase())
    );
  }
}
```

```
<div class="search-container">
  <input
    type="text"
    placeholder="Buscar por nombre..."
    [(ngModel)]="searchTerm"
  >
</div>
```

Directivas de componentes

- Este es el tipo de directiva más común.
- Cada componente Angular es técnicamente una directiva con una plantilla.

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

@Component({
  selector: 'app-skill-badge',
  template: `
    <span class="skill-badge">
      {{ skill }}
    </span>
  `,
})
export class SkillBadgeComponent {
  skill = '';
}
```

```ts
<app-skill-badge></app-skill-badge>
```

Directivas estructurales

- Se encargan de moldear o remodelar la estructura del DOM, generalmente añadiendo, eliminando o manipulando elementos.
- Se reconocen fácilmente por el prefijo de asterisco (`*`).
- `*ngIf` : Agrega o elimina condicionalmente un elemento del DOM.
- `*ngFor` : Repite un nodo para cada elemento de una lista.
- `*ngSwitch` : Funciona como una declaración `switch` de JavaScript, mostrando un elemento de varias opciones posibles en función de una condición.

*ngIf

Agrega o elimina condicionalmente un elemento del DOM.

```
<div class="members-grid" *ngIf="teamMembers.length > 0; else noTeamMembers">
  <app-team-member-card>
  </app-team-member-card>
</div>
<ng-template #noTeamMembers>No se encontraron miembros registrados</ng-template>
```

```
export class AppComponent {
  teamMembers: TeamMember[] = [];
}
```

*ngFor

```
<div class="skills-container">
  <app-skill-badge
    *ngFor="let skill of member.skills; let i = index"
    [skill]="skill" />
</div>
```

```
export class TeamMemberCardComponent {
   member: TeamMember = {
    skills: ['Angular', 'TypeScript', 'Node.js', 'MongoDB', 'Docker']
   };
}
```

*ngSwitch

```
<span class="user" [ngSwitch]="getTimeOfDay()">
  <span *ngSwitchCase="'morning'">Buenos días, Admin</span>
  <span *ngSwitchCase="'afternoon'">Buenas tardes, Admin</span>
  <span *ngSwitchDefault>Buenas noches, Admin</span>
</span>
```

```
export class HeaderComponent {
   getTimeOfDay(): string {
    const hour = this.today.getHours();
    if (hour < 12) return 'morning';
    if (hour < 18) return 'afternoon';
    return 'evening';
  }
}
```

Directivas de atributos

- Modifican la apariencia o el comportamiento de un elemento, componente u otra directiva.
- No modifican la estructura del DOM.
- `ngClass`: Agrega o elimina un conjunto de clases CSS dinámicamente.
- `ngStyle`: Aplica un conjunto de estilos en línea de forma dinámica.

ngClass

```
<span class="skill-badge" [ngClass]="skillClasses">
  {{ skill }}
</span>
```

```
export class SkillBadgeComponent {
  skill = '';
  get skillClasses() {
    return {
      'angular-skill': this.skill.toLowerCase().includes('angular')
    };
  }
}
```

ngStyle

```
<span class="status-dot" [ngStyle]="getStatusStyles()"> </span>
```

```
export class AvailabilityIndicatorComponent {
  getStatusStyles() {
    const colors = {
      disponible: '#4caf50',
      ocupado: '#f44336',
    };
    return {
      'background-color': colors[this.status],
    };
  }
}
```