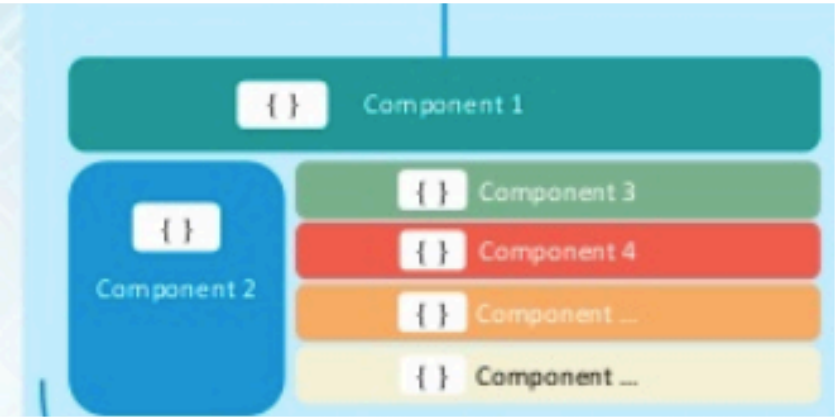
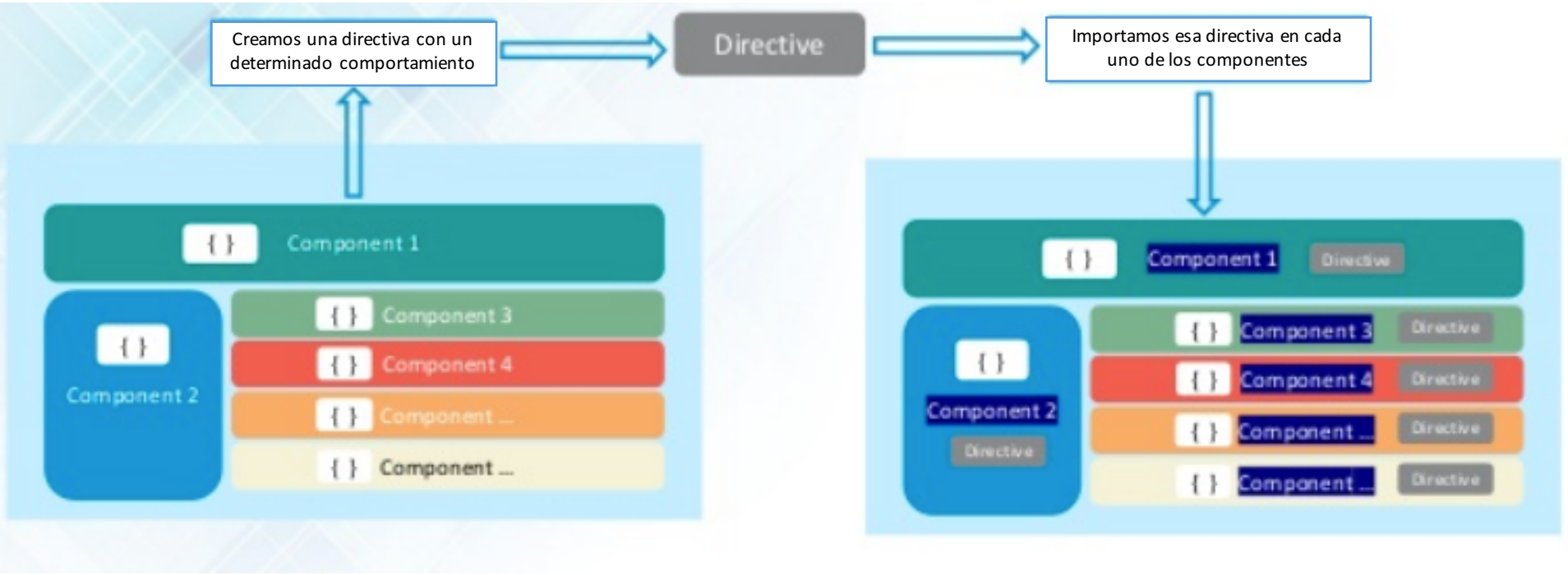
ANGULAR DIRECTIVAS

POR QUÉ NECESITAMOS DIRECTIVAS



POR QUÉ NECESITAMOS DIRECTIVAS



- Las directivas son un mecanismo que nos permite:
  - Separar el código "de funcionamiento" del código de vista
- Si desarrollamos las directivas en modo standalone, es decir, autocontenidas, podremos reutilizarlas en todo el proyecto

QUÉ ES UNA DIRECTIVA?

- Una directiva es un mecanismo que **extiende** HTML con nuevos **atributos**
  - A diferencia de un **componente**, que **crea nuevas etiquetas** HTML, una **directiva extiende componentes** HTML que ya existen
- Una directiva es el único mecanismo de Angular que tiene acceso al DOM
  - Por tanto, es el único punto en el cual puedo manipular el DOM directamente
- Una directiva siempre aparecerá en un TAG de HTML
- Angular ya provee de unas cuantas directivas built-in, que ya hemos estado usando…

```
<div *ngFor="let a of _huellas" style="margin-top:100px">
  <p>{{a.nombre}}</p>
  <p>{{a.agua}}</p>
  <p>{{a.electricidad}}</p>
  <p>{{a.gas}}</p>
</div>
```

CÓMO PUEDO CREAR UNA DIRECTIVA?

- Podemos ver las directivas como componentes sin HTML asociado
  - Y algunas vitaminas :P
- Crear una directiva en Angular es tan sencillo como introducir el siguiente código:

```
import { Directive, ElementRef, Renderer } from '@angular/core';

// Directive decorator
@Directive({ selector: '[myHidden]' })
// Directive class
export class HiddenDirective {
    constructor(el: ElementRef, renderer: Renderer) {
      // Use renderer to render the element with styles
        renderer.setElementStyle(el.nativeElement, 'display', 'none');
    }
}
```

TIPOS DE DIRECTIVAS

- Componentes
  - Directivas con template HTML :P
- Directivas estructurales (Structural Directive)
  - Directivas que añaden o eliminan elementos al DOM
  - Por ejemplo, *ngFor añade elementos al DOM en función de sus parámetros de entrada
  - *ngIf renderiza su contenido si la condición se evalúa a true, de lo contrario no renderiza nada
- Directivas de atributo (Attribute Directive)
  - Cambian el aspecto de un elemento
  - Cambian el comportamiento de un elemento

```
<app-huella [huella]="it"></app-huella>
```

```
[(ngModel)]="usuarioIntroducido"
```

DIRECTIVA ESTRUCTURAL NG-SWITCH

```
 EnterColor:
 <input type = "text" ng-model="Color"/>
<div ng-switch on = "Color">
    <span ng-switch-when="red" style="color:{{Color}}"> I am very RED
    </span>
    <span ng-switch-when="green" style="color:{{Color}}"> I am very GREEN
    </span>
    <span ng-switch-default  style="color:{{Color}}"> I am default
    </span>
</div>
```

```
import { Directive, ElementRef, HostListener } from '@angular/core';

@Directive({
    selector: '[myHighlight]'
})
export class MyHighlightDirective {
    defaultBgColor: string = 'none';

    @HostListener('mouseenter')
    onMouseenter() {
        this.elRef.nativeElement.style.backgroundColor = 'red';
    }

    @HostListener('mouseleave')
    onMouseleave() {
        this.elRef.nativeElement.style.backgroundColor = this.defaultBgColor;
    }

    constructor(private elRef: ElementRef) {
        this.defaultBgColor = elRef.nativeElement.style.backgroundColor;
    }
}
```

Nos permite acceder al DOM

Nos permite suscribirnos a eventos del DOM del elemento

Con nativeElement accedemos al componente, o etiqueta HTML que "hostea" a la directiva

Tenemos que inyectar el servicio para poder utilizarlo, de lo contrario será undefined

DECLARACIÓN DE DIRECTIVAS EN EL MÓDULO

- Como con casi todo en Angular, que yo cree una directiva no significa que la pueda usar en todas partes
- Tengo que declararla en el Módulo principal de mi aplicación, o en el módulo en el que quiera utilizarla
- Como una directiva es "un componente sin HTML que puede acceder al DOM", se declaran igual que un componente, en el apartado de declarations del módulo
- En la arquitectura que vimos, las directivas se ubicarían idealmente en el SHareModule, por lo que un ejemplo de declaración sería el siguiente

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

import { HiddenDirective } from './hidden.directive';

@NgModule({
    declarations: [
        HiddenDirective
    ],
    exports: [
        HiddenDirective
    ]
})
export class SharedModule{}
```

¡OJO! Si lo declaro en un módulo, y no lo exporto, esa directiva solo podrá usarse en ese módulo, hay que exportarlo para que puedan utilizarlo el resto de módulos

NO NOS LIBRAMOS DE LOS INPUT

```
import { Directive, ElementRef, Input, Renderer } from '@angular/core';

@Directive({ selector: '[myHidden]' })
export class HiddenDirective {

    constructor(public el: ElementRef, public renderer: Renderer) {}

    @Input() myHidden: boolean;

    ngOnInit(){
        // Use renderer to render the emelemt with styles
        console.log(this.myHidden)
        if(this.myHidden) {
            console.log('hide');
            this.renderer.setElementStyle(this.el.nativeElement, 'display', 'none');
        }
    }
}
```

```
<h1>Welcome</h1>
<h1 [myHidden]="val">Hidden Welcome</h1>
```

CÓMO CREAR UNA DIRECTIVA ESTRUCTURAL

```
import { Directive, Input, TemplateRef, ViewContainerRef } from '@angular/core';

@Directive({ selector: '[myIf]' })
export class IfDirective {
  constructor(
    private templateRef: TemplateRef<any>,
    private viewContainer: ViewContainerRef
    ) { }

  @Input() set myIf(shouldAdd: boolean) {
    if (shouldAdd) {
      // If condition is true add template to DOM
      this.viewContainer.createEmbeddedView(this.templateRef);
    } else {
     // Else remove template from DOM
      this.viewContainer.clear();
    }
  }

}
```

Nos permite acceder a la template HTML completa

Nos da una referencia al contenedor de la directiva (al componente o HTML de esa directiva)

Si shouldAdd es true, le decimos al template HTML que contenga la directiva que cree una vista embebida del componente que contiene esa directiva

De lo contrario, elimina el componente que contiene esa directiva de la plantilla. Tenemos que eliminarlo, porque el usuario si que lo ha añadido...

```
class ViewContainerRef {
  get element: ElementRef
  get injector: Injector
  get parentInjector: Injector
  clear(): void
  get(index: number): ViewRef|null
  get length: number
  createEmbeddedView<C>(templateRef: TemplateRef<C>, context?: C,
index?: number): EmbeddedViewRef<C>
  createComponent<C>(componentFactory: ComponentFactory<C>, index?:
number, injector?: Injector, projectableNodes?: any[][], ngModule?:
NgModuleRef<any>): ComponentRef<C>
  insert(viewRef: ViewRef, index?: number): ViewRef
  move(viewRef: ViewRef, currentIndex: number): ViewRef
  indexOf(viewRef: ViewRef): number
  remove(index?: number): void
  detach(index?: number): ViewRef|null
}
```

Esto es todo amigos

MUCHAS GRACIAS