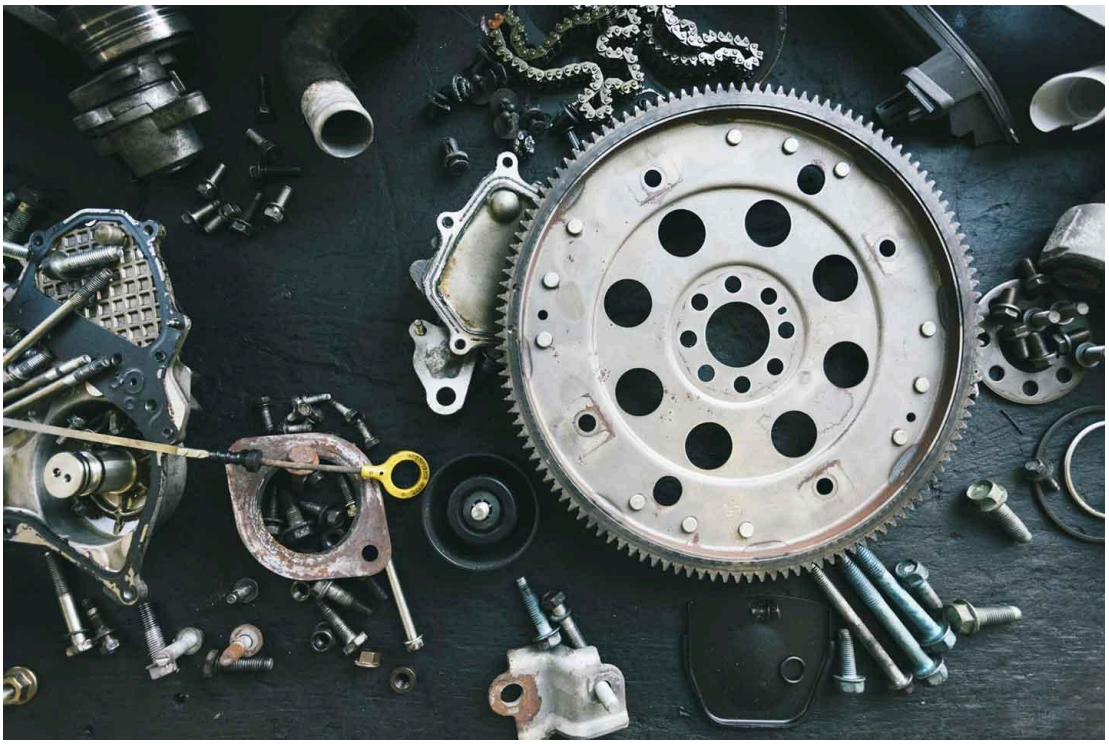
Ionic v8

Accordion



Progresa
Una formación de futuro

ÍNDICE

Introducción

Los accordions nos proporcionan secciones colapsables para reducir el espacio vertical a la vez que nos proporciona una forma de organizar y agrupar la información.

La documentación del Accordion está en la página:
https://ionicframework.com/docs/api/accordion

Preparación

Componente

Para empezar a trabajar con el accordion vamos a necesitar una página donde usarlo. Creamos la página en la **terminal**:

ionic g page pages/Accordion --spec=false

Icono

Añadimos en nuestro **app.component.ts** el icono que queremos usar para el menú en la lista de iconos(addIcons).

```
addIcons({grid, logoAppleAppstore, americanFootball, beaker, star, trash, shareSocial, caretForwardCircle, close, card, pin, car, logoFacebook,logoTwitter,logoTiktok,logoInstagram,logoYoutube,logoGithub,add, albums, ellipsisHorizontalOutline, list, heart, call, infinite, infiniteOutline, phonePortrait, cog, settings, person, arrowDownCircleOutline, layersOutline});
```

JSON

Añadimos en nuestro **menu.json** el nuevo elemento que queremos que aparezca tanto en el menú como en la página de inicio.

```
,{
 "nombre": "Accordion",
 "ruta": "/accordion",
 "icono": "layers-outline"
}
```

EJEMPLO

VISUALIZACIÓN

En primer lugar vamos a definir el accordion en **accordion.page.html**:

```
<app-header titulo="Accordion"></app-header>

<ion-content class="ion-padding">
 <ion-accordion-group (ionChange)="accordionGroupChange($event)">
   <ion-accordion value="first">
     <ion-item slot="header" color="light">
       <ion-label>First Accordion</ion-label>
     </ion-item>
     <div class="ion-padding" slot="content">First Content</div>
   </ion-accordion>
   <ion-accordion value="second">
     <ion-item slot="header" color="light">
       <ion-label>Second Accordion</ion-label>
     </ion-item>
     <div class="ion-padding" slot="content">Second Content</div>
   </ion-accordion>
```

```
    <ion-accordion value="third">
      <ion-item slot="header" color="light">
        <ion-label>Third Accordion</ion-label>
      </ion-item>
      <div class="ion-padding" slot="content">Third Content</div>
    </ion-accordion>
  </ion-accordion-group>

  <p #listenerOut></p>

  <ion-grid>
    <ion-row>
      <ion-col size="8">
        <ion-accordion-group (ionChange)="optionAccordion($event)">
              <ion-accordion value="A" toggleIcon="caret-down-circle"
toggleIconSlot="start">
            <ion-item slot="header" color="danger">
              <ion-label>Accordion A</ion-label>
            </ion-item>
            <div class="ion-padding" slot="content">Content A</div>
          </ion-accordion>
              <ion-accordion value="B" toggleIcon="caret-down-circle"
toggleIconSlot="start">
            <ion-item slot="header" color="danger">
              <ion-label>Accordion B</ion-label>
            </ion-item>
            <div class="ion-padding" slot="content">Content B</div>
          </ion-accordion>
              <ion-accordion value="C" toggleIcon="caret-down-circle"
toggleIconSlot="start">
            <ion-item slot="header" color="danger">
              <ion-label>Accordion C</ion-label>
            </ion-item>
            <div class="ion-padding" slot="content">Content C</div>
          </ion-accordion>
        </ion-accordion-group>
      </ion-col>
     <ion-col size="4"><h2 class="ion-text-center">{{option}}</h2></ion-col>
    </ion-row>
  </ion-grid>
</ion-content>
```

Y añadimos el icono en **app.component.ts**:

```
addIcons({grid,  … , layersOutline, caretDownCircle});
```

Lógica

Y una vez que tenemos el html, definimos el **accordion.page.ts**:

```
import {Component, ElementRef, ViewChild} from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { IonAccordion, IonAccordionGroup, IonCol, IonContent,
IonGrid, IonHeader, IonItem, IonLabel, IonRow, IonTitle, IonToolbar
} from '@ionic/angular/standalone';
import {HeaderComponent} from
"../../components/header/header.component";

@Component({
 selector: 'app-accordion',
 templateUrl: './accordion.page.html',
 styleUrls: ['./accordion.page.scss'],
 standalone: true,
   imports: [IonContent, IonHeader, IonTitle, IonToolbar,
CommonModule, FormsModule, HeaderComponent, IonAccordionGroup,
IonAccordion, IonItem, IonLabel, IonGrid, IonRow, IonCol]
})
export class AccordionPage {
 /* Buscamos en el html el elemento identificado con #listenerOut
para poder usarlo aquí con la variable listenerOut */
  @ViewChild('listenerOut', { static: true }) listenerOut!:
ElementRef;
 values: string[] = ['first', 'second', 'third'];
 option!: string;
 constructor() { }

 accordionGroupChange = (ev: any) => {
   const nativeEl = this.listenerOut.nativeElement;
```

```
      /* Comprobamos que al iniciar el componente el elemento
#listenerOut sea vacío */
    if (!nativeEl) { return; }
    /* Buscamos el elemento seleccionado
     y lo sacamos de la lista de elementos colapsados y los guardamos
en una constante */
     const collapsedItems = this.values.filter((value) => value !==
ev.detail.value);
    /* Guardamos el elemento seleccionado en una constante */
    const selectedValue = ev.detail.value;
    /* Mostramos el resultado de elementos colapsados y expandidos */
    nativeEl.innerText = `
            Expanded: ${selectedValue === undefined ? 'None' :
ev.detail.value}
     Collapsed: ${collapsedItems.join(', ')}
   `;
  };

  optionAccordion(ev: any) {
    this.option = ev.detail.value;
  }
}
```

Explicación

`<ion-accordion-group (ionChange)="accordionGroupChange($event)">`

Lanzará el evento cuando cambie de estado el accordion.

`<ion-accordion value="first">`

Asignamos un valor al accordion para poder interactuar con él.

`<p #listenerOut></p>`

Asignamos un identificador al párrafo para poder acceder a él desde nuestro archivo de Typescript.

`@ViewChild('listenerOut', { static: true }) listenerOut: ElementRef;`

Localizamos el identificador **#listenerOut** en el html para poder utilizarlo dentro de nuestro ts.