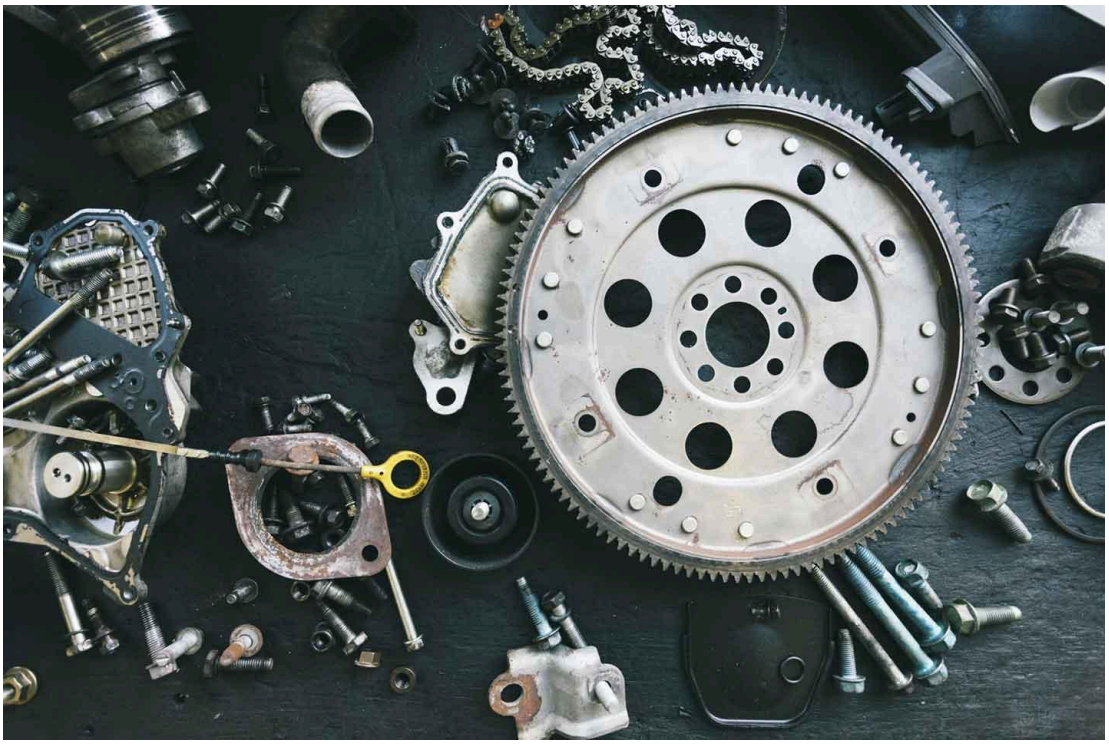
Ionic v8

Search Bar con Checkbox



Progresa
Una formación de futuro

ÍNDICE

Preparación **3**
- Componente 3
- Icono 3
- Json 3

Ejemplo **4**
- Servicio 4
- Lógica 4
  - Explicación 7
- Visualización 8
- Estilos 9
  - Explicación 9

Preparación

Componente

Vamos a ver ahora un ejemplo combinado de barra de búsqueda con checkboxes. En primer lugar vamos a crear la página para trabajar con él. En la **terminal**:

```
ionic g page pages/SearchableSelect --spec=false
```

Icono

```
  addIcons({grid, … , checkMarkCircleOutline, searchOutline});
}
```

Json

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

```
,{
 "nombre": "Searchable Select",
 "ruta": "/searchable-select",
 "icono": "search-outline"
}
```

EJEMPLO

La idea general es crear una página en la que podremos elegir si nuestro select, que haremos a mano, va a ser múltiple o simple y el atributo del objeto que queremos buscar. Una vez elegido al pulsar en el select abriremos un modal donde tendremos todos los valores de la lista de datos, con su correspondiente checkbox, junto con una barra de búsqueda que actuará como filtro.

SERVICIO

Vamos a empezar trabajando con la carga de datos. En nuestro servicio añadiremos una nueva función para cargar datos. En **data.service.ts**:

```
loadUsers() {
 return
this.http.get<any[]>('https://randomuser.me/api/?results=
100&seed=Progresa');
}
```

LÓGICA

Y cargamos la información en un array y procedemos a definir nuestra lógica de carga de elementos, así como de selección de los mismos y controles del modal. En **searchable-select.page.ts**:

```
import {Component, EventEmitter, inject, OnChanges, OnInit, Output,
SimpleChanges} from '@angular/core';
import {CommonModule} from '@angular/common';
import {FormsModule} from '@angular/forms';
import { IonButton, IonButtons, IonCheckbox, IonContent, IonHeader,
IonInput, IonItem, IonLabel, IonModal, IonSearchbar, IonTitle,
IonToolbar } from '@ionic/angular/standalone';
import {SearchbarCustomEvent} from "@ionic/angular";
import {DataService} from "../../services/data.service";
import {HeaderComponent} from
"../../components/header/header.component";

@Component({
 selector: 'app-searchable-select',
```

```
 templateUrl: './searchable-select.page.html',
 styleUrls: ['./searchable-select.page.scss'],
 standalone: true,
    imports:    [IonContent,    IonHeader,    IonTitle,    IonToolbar,
CommonModule,    FormsModule,    IonLabel,    IonCheckbox,    IonItem,
HeaderComponent,    IonInput,    IonModal,    IonButtons,    IonButton,
IonSearchbar]
})
export class SearchableSelectPage implements OnChanges {

 @Output() selectedChanged: EventEmitter<any> = new EventEmitter();
 private dataService: DataService = inject(DataService);
 title = 'Search';
 multiple = false;
 itemTextField = 'name.first';
 users: any[] = [];
 isOpen = false;
 selected: any[] = [];
 filtered: any[] = [];

 constructor() {
   this.loadUsers();
 }

 ngOnChanges(changes: SimpleChanges): void {
   this.filtered = this.users;
 }


 loadUsers() {
   this.dataService.loadUsers().subscribe(
     {
       next: (res: any) => {
         this.users = res.results;
         this.filtered = this.users;
       },
       error: err => {
         console.error(err);
       },
       complete: () => {
         console.log('Users Loaded!');
```

```
        }
      }
    );
  }

  /* Inicio de controles del modal */
  open() {
    this.isOpen = true;
  }

  cancel() {
    this.isOpen = false;
  }

  confirm() {
    this.isOpen = false;
  }

  /* Fin de los controles del modal */

  /* Función para pasar como parámetro una estructura de objeto compleja, en nuestro caso sacar name.first */
  leaf = (obj: any) =>
      this.itemTextField.split('.').reduce((value, el) => value[el], obj);

  itemSelected(myItem: any) {
    /* Si la selección es múltiple y hay varios valores, los borramos y nos quedamos con el seleccionado */
    if (!this.multiple) {
      if (this.selected.length>0) {
        this.selected = [];
        // Ponemos en false todos los elementos
        this.filtered.forEach((item: any) => item.selected = false);
        // Ponemos en true el seleccionado
              this.filtered[this.filtered.findIndex(item => item === myItem)].selected = true;
      }
    }
      /* Ponemos o quitamos los elementos seleccionados del array selected */
```

```
    if (myItem.selected) {
      this.selected.push(myItem);
    } else {
      this.selected.splice(this.selected.indexOf(myItem), 1);
    }
    /* Avisamos del cambio en selected */
    this.selectedChanged.emit(this.selected);
    if (!this.multiple) {
      this.isOpen = false;
    }
  }

  filter(event: SearchbarCustomEvent) {

    if (event.detail.value) {
      if(event.detail.value.length === 0){
        this.filtered = this.users;
      }else {
        const filter = event.detail.value.toLowerCase();

        /* Filtramos por la palabra de búsqueda */
                        this.filtered    =    this.users.filter(item    =>
this.leaf(item).toLowerCase().indexOf(filter) >= 0);
      }
      }
  }

  clear() {
    this.filtered = this.users;
  }
}
```

Explicación

```
@Output() selectedChanged: EventEmitter<any> = new EventEmitter();
```

Es un elemento que nos permitirá lanzar una señal cuando hayamos realizado cambios en nuestra lista de elementos seleccionados y así se ejecute el ngOnChange.

VISUALIZACIÓN

Y por último nos queda la visualización de datos en **searchable-select.page.html**:

```
<app-header titulo="{{title}}"></app-header>

<ion-content>
 <ion-item>
   <ion-label>Multiple</ion-label>
          <ion-checkbox   mode="ios"   slot="end"   color="success"
[(ngModel)]="multiple"></ion-checkbox>

 </ion-item>
 <ion-item>
   <ion-label position="floating">Item text field</ion-label>
   <ion-input [(ngModel)]="itemTextField"></ion-input>
 </ion-item>
 <ion-item (click)="open()">
   <ion-label>Select User</ion-label>

   @if (selected.length) {
     <div>
       @for (item of selected; track $index; let last = $last) {
         <span>
           {{ leaf(item) }}{{ last ? "" : ", " }}
         </span>
       }
     </div>
   } @else {
     Select
   }

   <ion-modal [isOpen]="isOpen">
     <ng-template>
       <ion-header>
         <ion-toolbar color="primary">
           <ion-buttons slot="start">
             <ion-button (click)="cancel()">Cancel</ion-button>
           </ion-buttons>
                       <ion-title  class="ion-text-center">{{  title
}}</ion-title>
```

```
            <ion-buttons slot="end">
              <ion-button (click)="confirm()">Confirm</ion-button>
            </ion-buttons>
          </ion-toolbar>
          <ion-toolbar>
                                  <ion-searchbar    (ionClear)="clear()"
(ionInput)="filter($event)"></ion-searchbar>
          </ion-toolbar>
        </ion-header>
        <ion-content>
          @for (item of filtered; track $index) {
            <ion-item (click)="itemSelected(item)">
                                              <ion-checkbox    slot="start"
[(ngModel)]="item.selected"></ion-checkbox>
              <ion-label>{{ leaf(item) }}</ion-label>
            </ion-item>
          }

        </ion-content>
      </ng-template>
    </ion-modal>
  </ion-item>
</ion-content>
```

Estilos

Y definir un padding a la barra de búsqueda en **searchable-select.page.scss**:

```
ion-searchbar{
 padding: 12px;
}
```

Explicación

```
<div *ngIf="selected.length; else placeholder">
```

Si nuestro array de elementos seleccionados tiene elementos los mostramos, si no entonces mostramos el elemento #placeholder que es una plantilla con la palabra Select.