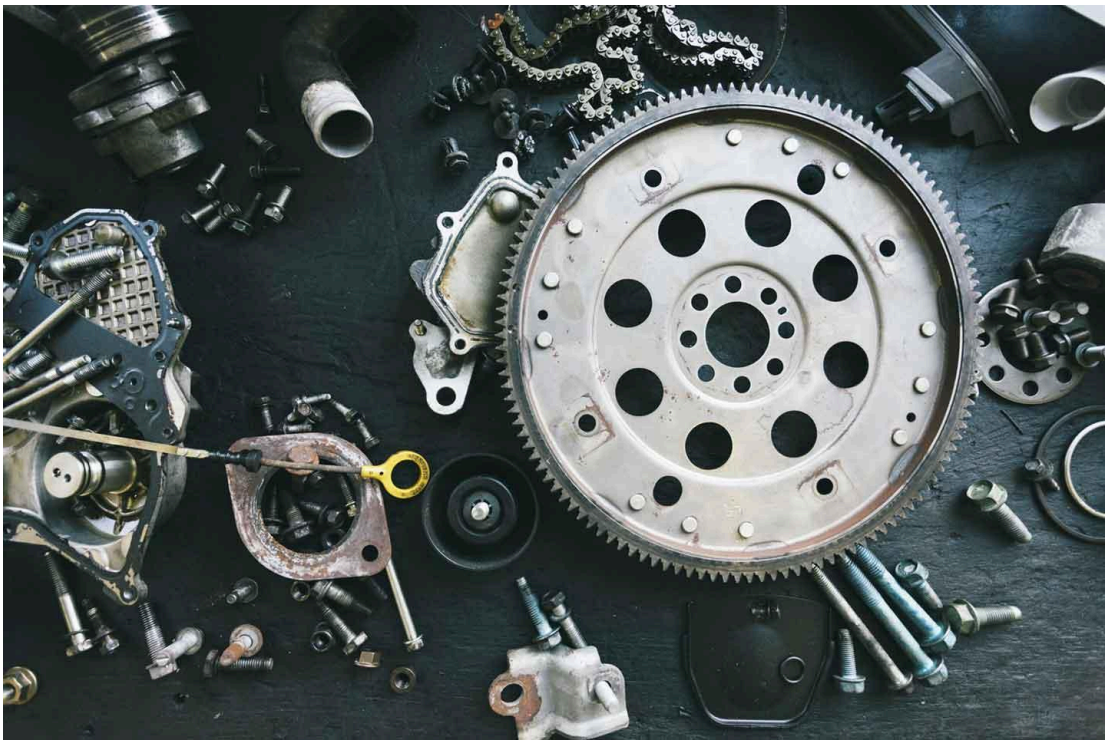
Ionic v8

Refresher y Skeleton



Progresa
Una formación de futuro

ÍNDICE

Introducción

Refresher nos proporciona la funcionalidad de recargar la página pulsando y desplazando el cursor. Se utiliza comúnmente para actualizar la información de una lista de elementos. Esta información se actualizará con el evento del refresher y una vez hayamos terminado tenemos que usar la función complete() del refresher.

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

Skeleton nos proporciona una estructura de precarga de datos, como un placeholder visual.

La documentación del skeleton está en la página:
https://ionicframework.com/docs/api/skeleton-text

Preparación

Componente

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

ionic g page pages/Refresher-skeleton --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});
```

Json

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

```
,{
 "nombre": "Refresher y Skeleton",
 "ruta": "/refresher-skeleton",
 "icono": "arrow-down-circle-outline"
}
```

Ejemplo

Visualización

Vamos a darle contenido a nuestra página. En **refresher.page.html**:

```
<app-header titulo="Refresher y Skeleton"></app-header>

<ion-content class="ion-padding">
 <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
   <ion-refresher-content></ion-refresher-content>
 </ion-refresher>
```

```
@if (items.length === 0){
  <h4 class="ion-text-center">     Arrastre hacia abajo para recargar</h4>
} @else if (items.length > 0 && skeleton){
  <ion-list>
    @for (i of items; track ind;let ind =$index){
    <ion-item *ngFor="let i of items;let ind=index;">
      Item {{ ind + 1 }}
    </ion-item>
    }
  </ion-list>
} @else if(items.length > 0 && !skeleton){
  <ion-list>
    @for (i of items; track ind;let ind =$index){
      <ion-item>
        <ion-skeleton-text [animated]="true"></ion-skeleton-text>
      </ion-item>
    }
  </ion-list>
}
</ion-content>
```

Lógica

Y en **refresher.page.ts** definimos la lógica del controlador, donde creamos la función para simular la carga de datos y controlamos el booleano del skeleton:

```
import { Component, OnInit } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { IonContent, IonHeader, IonItem, IonList, IonRefresher,
 IonRefresherContent, IonSkeletonText, IonTitle, IonToolbar
} from '@ionic/angular/standalone';
import {HeaderComponent} from
"../../components/header/header.component";

@Component({
 selector: 'app-refresher-skeleton',
 templateUrl: './refresher-skeleton.page.html',
 styleUrls: ['./refresher-skeleton.page.scss'],
 standalone: true,
```

```
 imports: [IonContent, IonHeader, IonTitle, IonToolbar, CommonModule,
FormsModule,   HeaderComponent,   IonRefresher,   IonRefresherContent,
IonList, IonItem, IonSkeletonText]
})
export class RefresherSkeletonPage {
 items: any[] = [];
 skeleton = false;
 constructor() { }

 doRefresh(event: any) {
   setTimeout(() => {
     this.items.push(...Array(10));
     this.skeleton = true;
     event.target.complete();
   }, 1500);
   this.skeleton = false;
 }
}
```

Explicación

```
<ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
```

Es la función que detecta el movimiento vertical de recarga de la página.

```
 <h4 class="ion-text-center" *ngIf="items.length === 0">
```

Si la longitud del array es 0 saco el mensaje de que desplace hacia abajo para recargar.

```
<ion-list *ngIf="items.length > 0 && skeleton">
```

Si la longitud del array es mayor que 0 y skeleton es verdadero muestra la lista.

```
<ion-list *ngIf="items.length > 0 && !skeleton">
```

Si la longitud del array es mayor que 0 y skeleton es falso mostramos el skeleton.

```
setTimeout(() => {
```

Función síncrona, al final se define el tiempo que tardará.

```
event.target.complete();
```

Llamada que finaliza la carga de elementos.

ANIMACIÓN POR CSS

Vamos a añadir unas animaciones para dar fluidez a la carga de elementos. Para ello usaremos unas clases de css que podemos encontrar en:

https://raw.githubusercontent.com/daneden/animate.css/master/animate.css

CSS

Vamos a añadir en **global.scss** estas 3:Animated, Animated.Fast y FadeIn:

```
.animate__animated {
 -webkit-animation-duration: 1s;
 animation-duration: 1s;
 -webkit-animation-fill-mode: both;
 animation-fill-mode: both;
}

.animate__animated.animate__fast {
 -webkit-animation-duration: 800ms;
 animation-duration: 800ms;
}

/* Fading entrances  */
@-webkit-keyframes fadeIn {
 from {
   opacity: 0;
 }
 to {
   opacity: 1;
 }
}

@keyframes fadeIn {
 from {
   opacity: 0;
 }
 to {
   opacity: 1;
 }
}
```

```
.animate__fadeIn {
 -webkit-animation-name: fadeIn;
 animation-name: fadeIn;
}
```

HTML

Y por último usamos estas clases en nuestro **refresher.page.html**:

```
<app-header titulo="Refresher y Skeleton"></app-header>

<ion-content class="ion-padding">
 <ion-refresher slot="fixed" (ionRefresh)="doRefresh($event)">
   <ion-refresher-content></ion-refresher-content>
 </ion-refresher>
 @if (items.length === 0){
   <h4 class="ion-text-center animate__animated animate__fadeIn">
     Arrastre hacia abajo para recargar</h4>
 } @else if (items.length > 0 && skeleton){
   <ion-list class="animate__animated animate__fadeIn">
     @for (i of items; track ind;let ind =$index){
     <ion-item *ngFor="let i of items;let ind=index;">
       Item {{ ind + 1 }}
     </ion-item>
     }
   </ion-list>
 } @else if(items.length > 0 && !skeleton){
   <ion-list class="animate__animated animate__fadeIn">
     @for (i of items; track ind;let ind =$index){
       <ion-item>
         <ion-skeleton-text [animated]="true"></ion-skeleton-text>
       </ion-item>
     }
   </ion-list>
 }
</ion-content>
```