Galería de imágenes con IONIC 4

1. El objetivo

En este ejercicio pretendemos desarrollar una aplicación para visualizar imágenes que se encuentran agrupadas en categorías. Para ello sugerimos implementar la siguiente funcionalidad:

- Crear pestañas o *tabs* para cada una de las categorías, agrupando así las imágenes, y permitiendo con un solo click cambiar entre una categoría u otra.
- Utilizando la pantalla táctil de nuestros dispositivos, habilitar el gesto *swipe* para acceder a la imagen anterior o posterior.
- Activar la reproducción automática mediante un botón, permitiendo a su vez pausar la visualización mediante otro botón.
- Permitir ajustar la velocidad en la que se van mostrando las imágenes.

Una posible sugerencia de la interfaz de la aplicación sería la siguiente:

2. Elementos HTML de IONIC 4

2.1. *Slides*

La funcionalidad principal de este ejercicio nos la proporciona IONIC mediante el elemento `<ion-slides></ion-slides>` , donde ya se encuentra encapsulada la

respuesta a los gestos de *swipe* y la reproducción automática, tal como se explica en la documentación.

Por ejemplo, para poder visualizar de manera secuencial las imágenes ***alicante1.jpg***, ***alicante2.jpg*** y ***alicante3.jpg***, ubicadas en la carpeta ***img***, bastaría con insertarlas dentro de elementos `<ion-slide></ion-slide>` , y agruparlas todas con `<ion-slides></ions-slides>` de la siguiente forma:

```
<ion-slides>
  <ion-slide><img src="img/alicante1.jpg" /></ion-slide>
  <ion-slide><img src="img/alicante2.jpg" /></ion-slide>
  <ion-slide><img src="img/alicante3.jpg" /></ion-slide>
</ion-slides>
```

También podríamos haber utilizado el elemento `<ion-img></ion-img>` , que tiene un comportamiento casi idéntico al `<img>` de HTML. La diferencia es que esta última versión de IONIC optimiza especialmente aquellas páginas que tienen una lista muy grande de imágenes, ya que sólo se cargarían cuando fueran a estar visibles.

2.2. *Tabs*

El siguiente elemento clave que utilizaremos es `<ion-tabs></ion-tabs>` , ya que nos permitirá agrupar las fotos por categoría. Nos proporciona la funcionalidad necesaria para mostrar unas secciones de nuestro código HTML mientras se ocultan otras.

Conviene destacar además las numerosas opciones de personalización, tales como el color, que se pueden ajustar especificando el valor deseado en el atributo correspondiente (se puede consultar la documentación para más detalles). Por ejemplo, si queremos que se muestren dos pestañas con el fondo de color rojo, una de ellas etiquetada con el texto *Madrid* y el icono *train*, y la otra con *París* y *airplane*, procederíamos de la siguiente forma:

```
<ion-tabs>
  <ion-tab tab="madrid">
      ...
  </ion-tab>
  <ion-tab tab="paris">
      ...
  </ion-tab>
  <ion-tab-bar slot="bottom" color="danger">
    <ion-tab-button tab="madrid">
      <ion-icon name="train"></ion-icon>
```

```
        <ion-label>Madrid</ion-label>
      </ion-tab-button>
      <ion-tab-button tab="paris">
        <ion-icon name="airplane"></ion-icon>
        <ion-label>París</ion-label>
      </ion-tab-button>
    </ion-tab-bar>
  </ion-tabs>
```

En nuestro ejemplo, donde queremos agrupar las diapositivas con las imágenes en varias categorías, procederíamos de la siguiente forma:

```
<ion-tabs>
  <ion-tab tab="madrid">
    <ion-slides>
      <ion-slide><img src="img/madrid1.jpg" /></ion-slide>
      <ion-slide><img src="img/madrid2.jpg" /></ion-slide>
      <ion-slide><img src="img/madrid3.jpg" /></ion-slide>
    </ion-slides>
  </ion-tab>
  <ion-tab tab="paris">
    <ion-slides>
      <ion-slide><img src="img/paris1.jpg" /></ion-slide>
      <ion-slide><img src="img/paris2.jpg" /></ion-slide>
      <ion-slide><img src="img/paris3.jpg" /></ion-slide>
    </ion-slides>
  </ion-tab>
  <ion-tab-bar slot="bottom">
    <ion-tab-button tab="madrid">
      <ion-icon name="train"></ion-icon>
      <ion-label>Madrid</ion-label>
    </ion-tab-button>
    <ion-tab-button tab="paris">
      <ion-icon name="airplane"></ion-icon>
      <ion-label>París</ion-label>
    </ion-tab-button>
  </ion-tab-bar>
</ion-tabs>
```

Al hacer clic en la categoría *Madrid*, se ocultarían las diapositivas de la categoría *París*, y viceversa.

Además, hacemos uso de los elementos `ion-label` e `ion-icon`, que nos permiten establecer el texto y el icono que aparecen en la pestaña, respectivamente. El elemento `ion-label` acepta un texto libre, mientras que `ion-icon` permite elegir uno de entre todos los posibles iconos que nos proporciona IONIC.

2.3. Botones *play/pause*

Para iniciar y pausar la reproducción automática de las imágenes, insertaremos dos botones formados simplemente por los iconos `play` y `pause` que nos proporciona IONIC:

```
<ion-button onclick="play()"><ion-icon slot="icon-only" name="play">
</ion-icon></ion-button>
<ion-button onclick="pause()"><ion-icon slot="icon-only"
name="pause"></ion-icon></ion-button>
```

Y los insertaremos junto con el título en la parte superior de la pantalla:

```
<ion-header>
  <ion-toolbar>
    <ion-title>Gallery!</ion-title>
    <ion-buttons slot="primary">
      <ion-button onclick="play()"><ion-icon slot="icon-only"
name="play"></ion-icon></ion-button>
      <ion-button onclick="pause()"><ion-icon slot="icon-only"
name="pause"></ion-icon></ion-button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>
```

Un poco más adelante veremos la funcionalidad JavaScript que se ejecutará al pulsar dichos botones.

2.4. Para ajustar la velocidad

Para poder ajustar el tiempo que permanece cada diapositiva en la pantalla, sugerimos utilizar un elemento *range* (`<ion-range></ion-range>`). Por ejemplo, para poder establecer un tiempo que oscile entre 0 y 5 segundos (5000 milisegundos), podríamos utilizar el siguiente código:

```
<ion-range min="0" max="5000">
  <ion-icon slot="start" size="small" name="speedometer"></ion-icon>
  <ion-icon slot="end" name="speedometer"></ion-icon>
</ion-range>
```

Con el objetivo de mejorar el impacto visual del control deslizante, en el ejemplo utilizamos el icono `speedometer` , que será más pequeño a la izquierda (`slot="start" size="small"`) para indicar que las imágenes irán apareciendo lentamente, y más

grande a la derecha (`slot="end"`) para dar a entender que las diapositivas se mostrarán una tras otra más rápidamente.

Sugerimos insertar dicho control también en el elemento `<ion-header></ion-header>` de la aplicación, para que aparezca justo debajo de la barra que contiene el título y los botones:

```
<ion-header>
  <ion-toolbar>
    <ion-title>...</ion-title>
    <ion-buttons>
      ...
    </ion-buttons>
  </ion-toolbar>
  <ion-toolbar>
    <ion-item>
      <ion-range>
        ...
      </ion-range>
    </ion-item>
  </ion-toolbar>
</ion-header>
```

3. El código JavaScript

3.1. Accediendo a cada elemento

Por simplificar un poco el código, utilizaremos tres funciones para acceder a cada uno de los elementos clave (*range*, *tabs* y *slides*):

```
function getDelay() {
    return(document.querySelector('ion-range'));
}

function getTabs() {
    return(document.querySelector('ion-tabs'));
}

function getSlides() {
    return(document.querySelectorAll('ion-slides'));
}
```

3.2. Al pulsar play/pause

Para iniciar la reproducción automática de diapositivas bastará con acceder al elemento `<ion-slides></ion-slides>` correspondiente y ejecutar el método `startAutoplay()`, tal como se indica en la documentación.

Puesto que tendremos varias pestañas (una por cada categoría) primero deberemos acceder al *tab* que se encuentre seleccionado (`getTabs().getSelected()`), que será devuelto en una promesa. Utilizaremos luego simplemente el atributo `id` para acceder a la categoría activa, iniciando (`s.startAutoplay()`) o parando (`s.stopAutoplay()`) la reproducción según el botón que se haya pulsado (*play* o *pause*):

```
function play() {
    getTabs().getSelected().then(function(tab) {
        document.getElementById(tab).startAutoplay();
    });
}

function pause() {
    getSlides().forEach(function(s) {
        s.stopAutoplay();
    });
}
```

3.3. Al ajustar la velocidad de reproducción

Cada vez que movamos el control deslizante, se generará un evento `ionChange`, que nos indicará que tenemos que cambiar la velocidad con la que se muestran las diapositivas:

```
getDelay().addEventListener("ionChange", init);

...

function init() {
    getSlides().forEach(function(s) {
        s.options = {
            width: window.innerWidth,
            autoplay: {
                delay: 5000 - getDelay().value
            }
        };
    });
}
```

Puesto que la posición izquierda del control deslizante indicará menos velocidad (tiempo de espera mayor), y la posición derecha mayor velocidad (tiempo de espera menor), bastará con hacer una resta del tiempo máximo de visualización (5000 milisegundos) para calcular el tiempo (`delay`) que deberá permanecer cada imagen en pantalla:

```
autoplay: {
    delay: 5000 - getDelay().value
}
```

Inicializamos también la anchura de cada diapositiva indicando que ocupan toda la ventana:

```
s.options = {
    width: window.innerWidth,
    ...
};
```

3.4. Al cambiar de pestaña

Cada vez que cambiemos de pestaña, pararemos la reproducción:

```
getTabs().addEventListener("ionTabsWillChange", pause);

...

function pause() {
    getSlides().forEach(function(s) {
        s.stopAutoplay();
    });
}
```

3.5. Velocidad de reproducción inicial

Utilizaremos los eventos `onload` y `resize` para inicializar la velocidad de reproducción de las diapositivas que hemos especificado en el control deslizante, así como la anchura de cada una para que ocupen toda la pantalla:

```
<body onload="onLoad()">
```

```
2.
3.  ...
4.
5.  function onLoad() {
6.      window.addEventListener('resize', init);
7.      ...
8.      init();
9.  }
10.
11. ...
12.
13. function init() {
14.     getSlides().forEach(function(s) {
15.         ...
16.     });
17. }
```

4. En resumen...

4.1. El fichero *index.html*

```
1.  <!doctype html>
2.  <html lang="en">
3.  <head>
4.    <meta charset="utf-8">
5.    <meta name="viewport" content="viewport-fit=cover, width=device-
      width, initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0,
      user-scalable=no">
6.
7.    <script type="module"
      src="https://cdn.jsdelivr.net/npm/@ionic/core@4.9.1/dist/ionic/ionic
      .esm.js"></script>
8.    <script nomodule
      src="https://cdn.jsdelivr.net/npm/@ionic/core@4.9.1/dist/ionic/ionic
      .js"></script>
9.    <link rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/@ionic/core@4.9.1/css/ionic.bundl
      e.css"/>
10.
11.   <script src="gallery.js"></script>
12.   <title>Gallery!</title>
13. </head>
14. <body onload="onLoad()">
15.   <ion-app>
16.     <ion-header>
17.       <ion-toolbar>
18.         <ion-title>Gallery!</ion-title>
```

```
19.           <ion-buttons slot="primary">
20.             <ion-button onclick="play()"><ion-icon slot="icon-only"
    name="play"></ion-icon></ion-button>
21.             <ion-button onclick="pause()"><ion-icon slot="icon-only"
    name="pause"></ion-icon></ion-button>
22.           </ion-buttons>
23.         </ion-toolbar>
24.         <ion-toolbar>
25.           <ion-item>
26.             <ion-range min="0" max="5000" value="4000">
27.               <ion-icon slot="start" size="small" name="speedometer">
    </ion-icon>
28.               <ion-icon slot="end" name="speedometer"></ion-icon>
29.             </ion-range>
30.           </ion-item>
31.         </ion-toolbar>
32.       </ion-header>
33.       <ion-content>
34.         <ion-tabs>
35.           <ion-tab tab="alicante">
36.             <ion-slides id="alicante">
37.               <ion-slide><img src="img/alicante1.jpg" /></ion-slide>
38.               <ion-slide><img src="img/alicante2.jpg" /></ion-slide>
39.               <ion-slide><img src="img/alicante3.jpg" /></ion-slide>
40.             </ion-slides>
41.           </ion-tab>
42.           <ion-tab tab="madrid">
43.             <ion-slides id="madrid">
44.               <ion-slide><img src="img/madrid1.jpg" /></ion-slide>
45.               <ion-slide><img src="img/madrid2.jpg" /></ion-slide>
46.               <ion-slide><img src="img/madrid3.jpg" /></ion-slide>
47.             </ion-slides>
48.           </ion-tab>
49.           <ion-tab tab="paris">
50.             <ion-slides id="paris">
51.               <ion-slide><img src="img/paris1.jpg" /></ion-slide>
52.               <ion-slide><img src="img/paris2.jpg" /></ion-slide>
53.               <ion-slide><img src="img/paris3.jpg" /></ion-slide>
54.             </ion-slides>
55.           </ion-tab>
56.           <ion-tab-bar slot="bottom">
57.             <ion-tab-button tab="alicante">
58.               <ion-icon name="walk"></ion-icon>
59.               <ion-label>Alicante</ion-label>
60.             </ion-tab-button>
61.             <ion-tab-button tab="madrid">
62.               <ion-icon name="train"></ion-icon>
63.               <ion-label>Madrid</ion-label>
64.             </ion-tab-button>
65.             <ion-tab-button tab="paris">
66.               <ion-icon name="airplane"></ion-icon>
67.               <ion-label>París</ion-label>
68.             </ion-tab-button>
```

```
        </ion-tab-bar>
      </ion-tabs>
    </ion-content>
  </ion-app>
</body>
</html>
```

4.2. El fichero *gallery.js*

```
function onLoad() {
    getDelay().addEventListener("ionChange", init);
    window.addEventListener('resize', init);
    getTabs().addEventListener("ionTabsWillChange", pause);
    init();
}

function getDelay() {
    return(document.querySelector('ion-range'));
}

function getTabs() {
    return(document.querySelector('ion-tabs'));
}

function getSlides() {
    return(document.querySelectorAll('ion-slides'));
}

function init() {
    getSlides().forEach(function(s) {
        s.options = {
            width: window.innerWidth,
            autoplay: {
                delay: 5000 - getDelay().value
            }
        };
    });
}

function play() {
    getTabs().getSelected().then(function(tab) {
        document.getElementById(tab).startAutoplay();
    });
}

function pause() {
    getSlides().forEach(function(s) {
        s.stopAutoplay();
    });
```

```
41.     }
```

5. El resultado

Puedes hacer clic aquí para observar el aspecto que tiene la galería de imágenes y probar la funcionalidad resultante utilizando el código especificado.