# Día 4

## Conceptos

* Componentes
* Módulos
* Servicios
* Enrutamiento y navegación entre páginas

## Explicación de componentes

* Partir de la aplicación que da solución a los ejercicios propuestos del día anterior
* Mover la página de lista de frutas a una carpeta *pages*
* Crear un componente *fruit-card* que representa una tarjeta de fruta:

*ionic generate component components/fruit-card*

* Crear un módulo desde el cual se exportará el componente (si se declarasen más componentes, también se exportarían desde aquí):

*ionic generate module components*

* En dicho módulo se debe declarar *FruitCardComponent*; debe importar *CommonModule* e *IonicModule*; y debe exportar *FruitCardComponent*
* Este módulo de componentes se debe importar en el módulo de la página de la lista de frutas, porque en ella se usará *FruitCardComponent*
* Probar que el componente funciona correctamente (basta con que contenga un texto estático y se visualice en una página)
* El componente tendrá un parámetro de entrada *fruit* que representa la fruta
* También tendrá un evento de salida *deletePressed* que se emitirá cuando se pulse sobre el botón eliminar

## Explicación de servicios, enrutamiento y navegación entre páginas

* Crear un proyecto base con:

*ionic start explanation-app-2 sidemenu --no-git*

* Dar un vistazo general al Proyecto: cómo funciona una aplicación con sidemenu, cómo se enrutan otras vistas, etc.
* Mover la carpeta de la página *folder* dentro de una carpeta *pages* (en el código sólo implicaría un cambio en el *app-routing.module.ts*).
* En *app.component.ts*, mover el código del *constructor* a *ngOnInit.*
* Crear un servicio *LabelsService* donde se residirá la lógica de gestión de la lista de etiquetas que aparece en el menú:

*ionic generate service services/labels*

* En vez de usar el decorador *Injectable* para que nos lo provea en el módulo raíz, lo proveeremos nosotros explícitamente.
* Crear en el servicio *LabelsService* un método *getLabels* que devuelva la lista de labels que está en *app.component.ts.*
* En el *ngOnInit* de *app.component.ts,* obtener las labels a partir del método *getLabels* de *LabelsService*.
* Crear en el menú una lista más que contendrá un único elemento *Help*, que usaremos para redirigir a una página de ayuda.
* Crear una nueva página *EmailsList* con:

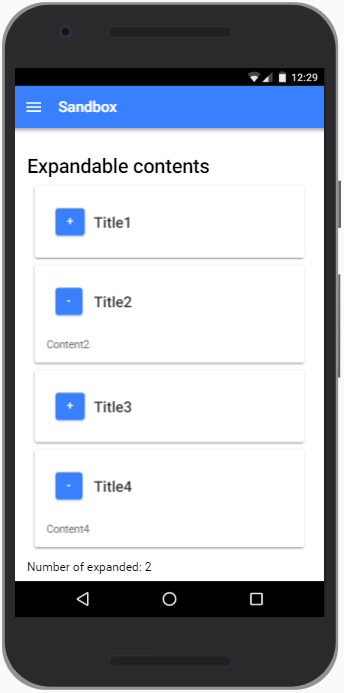
*ionic generate page pages/emails-list*

* Explicar cómo funciona el módulo de *routing* de una página y que podría evitarse incluyendo la lógica de enrutado dentro del propio módulo de la página.
* La idea es que la ruta de Inbox abra la página de *EmailsList* que estaría enrutada como *emails/:folder-id*. Hasta el momento es muy similar que la página folder. De hecho, su controlador, por el momento, es igual cambiando el parámetro.
* Prepararemos el HTML para mostrar una lista de con los títulos de diferentes emails.
* Crear un modelo/interfaz que representará un email. Tendrá un id, un title y un body.
* Crear un servicio que contendrá dos métodos: *getEmails(): Email[]* y *getEmail(id: number): Email*.
* Obtener la lista de emails a través del servicio.
* Crear una página *EmailDetails*, que se enrutaría como *emails/:folder-id/:email-id.*
* Preparar en la cabecera un botón de atrás para que se pueda navegar a la página anterior.
* La navegación en los templates (enlaces) se hace con la directiva *routerLink*, que se puede usar conjuntamente la directiva *routerDirection* para indicar la dirección de la navegación: *root* para hacer que la vista sea la raíz de la pila de navegación, *forward* para apilar (se podría volver a la pantalla anterior). Si no se especifica *routerDirection*, por defecto será forward.
* Con lo que conocemos ahora, resulta más fácil ver cómo funcionan los eventos del ciclo de vida de un componente: *ngOnInit* y *ngOnDestroy.* Implementadlos en la página *EmailDetails* incluyendo un *console.log*.
* Las páginas de Ionic usan *ion-router-outlet*. Este ofrece dos métodos muy útiles del ciclo de vida de las páginas, *ionViewDidEnter* y *ionViewDidLeave*, que indican cuando se entra y se sale de ellas. Implementadlos en la página *EmailDetails* incluyendo un *console.log* y observad el orden en que se ejecutan todos.

## Ejercicio 1

Crear una nueva aplicación con la base de *sidemenu*. Crear una página Sandbox que contendrá un ejemplo de uso de los componentes *expandable-content* y *rating*.

1. Crear el componente *expandable-content* a partir de la maquetación elaborada el día anterior. La página Sandbox tendrá en su cabecera un botón para mostrar el menú principal. Tendría la siguiente apariencia:



La manera en la que se invoca el componente sería:

|  |
| --- |
| <**app-expandable-content title="Title1"**>  Content1 </**expandable-content**>  <**app-expandable-content title="Title2"**>  Content2 </**expandable-content**>  <**aap-expandable-content title="Title3"**>  Content3 </**expandable-content**>  … |

Pista: Pasar datos de entrada al componente como contenido de su tag se llama *transclusión*.

1. Dote al componente de la lógica necesaria para que indique si el contenido está expandido o no. Use esta información para incluir en la página Sandbox el número de contenidos expandibles que están expandidos.

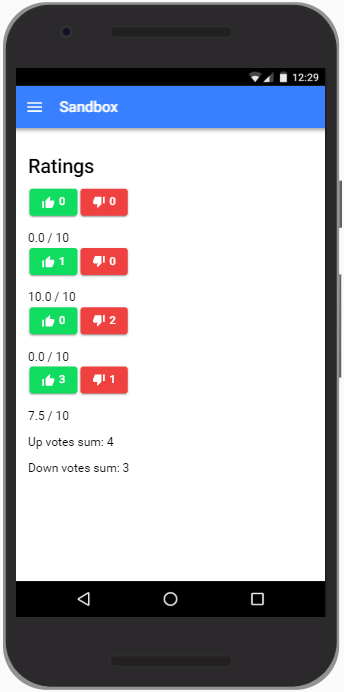
Pista:

|  |
| --- |
| <**app-expandable-content title="Title1"**  **(change)="expandableContentChanged($event)"**>  Content1 </**expandable-content**> |

Siendo *$event* una variable especial de las template que contiene lo que se le pasa como parámetro a la función *emit* del EventEmitter del componente.

## Ejercicio 2

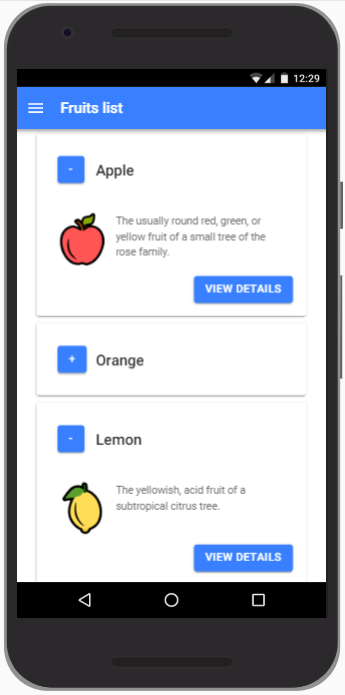
1. Crear el componente *rating* a partir de la maquetación elaborada el día anterior. Añada los componentes de prueba en la página Sandbox detrás de los *expandable-contents*.



1. Dote al componente para que se le puedan pasar como parámetros de entrada (de manera opcional) el número de votos positivos y el número de votos negativos. Si no se le pasa, se consideran cero. Inicialmente, al segundo componente se le pasa 1 voto positivo; al tercero 2 votos negativos; y al cuarto 3 votos positivos y un negativo.
2. Incluya en la parte inferior un contador que indique el número total de votos positivos y el número total de votos negativos de todos los componentes *rating*.

## Ejercicio 3

1. Crear una página de lista de frutas. Las frutas deben ser obtenidas de un servicio. Se reutilizará el componente *expandable-content* para cada uno de los elementos de la lista. La página tendrá en su cabecera un botón para mostrar el menú principal y tendría la siguiente apariencia:



1. Crear una página de detalle de fruta, que se accederá al pulsar sobre el botón View Details de una determinada fruta en la página de la lista. La cabecera de esta pantalla tendrá un botón para poder volver a la página anterior. Tendría la siguiente apariencia:

