# Día 4

## Conceptos

* Desarrollo de componentes, con sus Inputs y Outputs
* Módulos
* Servicios.

## Explicación

* Partir de la aplicación de la explicación del día anterior (o de la aplicación *explanation-app-base* entregada este mismo día).
* Mover la página de lista de frutas a una carpeta *pages*
* Crear un módulo *ComponentsModule* desde el cual se exportará los componentes comunes que desarrollemos para la aplicación:

*ionic generate module components*

* Crear un componente *fruit-card* que representa una tarjeta de fruta:

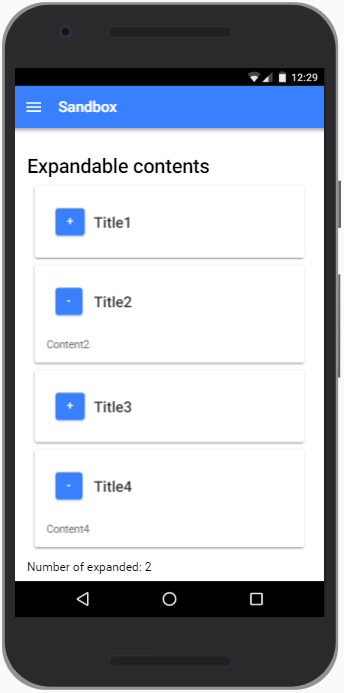
*ionic generate component components/fruit-card*

* En dicho módulo se debe declarar *FruitCardComponent*; debe importar *CommonModule* e *IonicModule*; y debe declarar y 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
* Crear un servicio *FruitsService* que contendría la lógica de gestión de datos relativos a las frutas: dispondrá de un método *getFruitsList*() que devolvería la lista de frutas que ahora mismo está hardcodeada en el controlador de la página de la lista de frutas.
* Declarar el servicio en el nodo *providers* del módulo *FruitsListPageModule* para poder inyectarlo sin problemas en el controlador de la página.
* Usar *FruitsService.getFruitsList*() en el controlador de la página de la lista de frutas.

## Ejercicio 1

Crear una nueva aplicación con 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 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: Disponer del HTML encerrado en el tag del componente se llama *content projection*.

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 el número de contenidos expandibles que están expandidos.

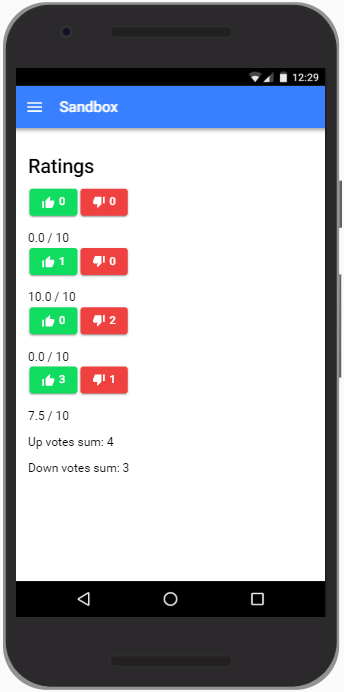
Pista:

|  |
| --- |
| <**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 misma página 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*.