Swift

REALIZACIÓN DE BÚSQUEDAS Y FILTRADOS

Alfonso López Ruiz
alr00048@red.ujaen.es

FUNDAMENTOS TEÓRICOS

Los fundamentos teóricos necesarios para la correcta comprensión de este minitutorial son los siguientes:

- Swift:
  - **Constantes (let):** variables inmutables.
  - **Arrays**: conjunto lineal de elementos accesibles a través de índices.
    Ejemplo: meal = ["Caprese Salad", "Chicken and Potatoes"]

    Utilizaremos especialmente los métodos **remove** (actualizaciones y borrados) e **index**(of:) para identificar el índice de una comida en el vector principal de comidas. También utilizaremos el atributo **count** para conocer el tamaño de un vector.
  - **Funciones**: bloques de código con un propósito muy concreto. Nos interesa especialmente conocer que estas funciones pueden ser declaradas como privadas con la palabra **private** antes de la función.
    También nos interesan los **valores por defecto de los argumentos** de la función.
  - **Variables opcionales (Optionals):** variables que pueden contener un valor o no (nil). Ejemplo: var newIndexPath: IndexPath?
  - **Obtener el valor de una variable opcional (Unwrapping):** sea newIndexPath una variable opcional, accederemos a su valor con el carácter ! Ejemplo: newIndexPath!
  - **Unwrapping mediante Optional Binding (en una estructura if):** de esta forma podemos acceder al interior de una estructura if si la variable tiene un valor distinto de nulo. Ejemplo:

    ```
    if let index = newIndexPath { /* TO DO, index no nulo. */}
    ```
  - **Funciones como parámetro de otras funciones**: lo utilizaremos para filtrar la comida después de una búsqueda. Nosotros además especificaremos una función lambda.
  - **Funciones lambda (Closure):** nos permite crear funciones locales que se pueden utilizar como argumentos. La sintaxis es la siguiente:

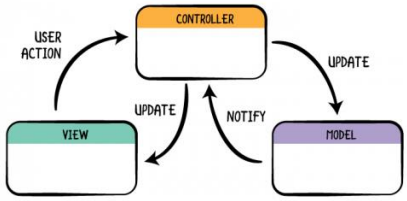
    ```
    { ( parameters ) -> return type in
        statements
    }
    ```
  - **Protocolo**: Define un modelo mediante propiedades y métodos necesarios para realizar una tarea concreta. Es similar al concepto de interfaz.
    Ejemplo:

    ```
    protocol UISearchResultsUpdating {
                                    ...
        func updateSearchResults(for searchController: UISearchController) {

        }
                                    ...
    }
    ```

- **Controlador**: Es un objeto que actúa como intermediario entre una o más vistas y uno o más modelos.

  Conduce los cambios en el modelo hacia la vista y viceversa. Pueden realizar tareas de configuración y coordinación, asi como manejar el ciclo de vida de otros objetos.



- **Table view**: lista dinámica de elementos en forma de tabla. Los métodos que vamos a necesitar son los siguientes:
  - Número de comidas por sección: en función de si estamos filtrando o no, este número variará.
  - Celda en una fila concreta: en función también de si hay una búsqueda activa o no habrá que coger la comida de una fuente u otra. Hay que devolver una celda lista para poder ser visualizada en la tabla, con todos los atributos que creamos convenientes.
- **Unwind segue**: es una navegación hacia atrás, de tal forma que en lugar de crear una nueva vista lo que se hace es volver a una vista anterior.

  En nuestro caso será necesario para añadir y actualizar comidas, de tal forma que la tabla recibirá una comida añadida o actualizada que se extraerá en el método unwindToMeaList (es donde llama la vista de añadir y actualizar comidas para hacer efectivo el segue unwind).

- Bars:
  - **Search bar**: acepta texto como entrada que puede utilizarse en una búsqueda. Nosotros utilizaremos estos componentes:
    - **Campo de texto para búsqueda.**
    - **Botón para limpiado.**
    - **Título descriptivo.**
  - **Scope bar**: define el ámbito de la búsqueda y se combina con la barra de búsqueda. En esta barra aparecen categorías claramente definidas, que en nuestro caso serán Nombre y Valoración.

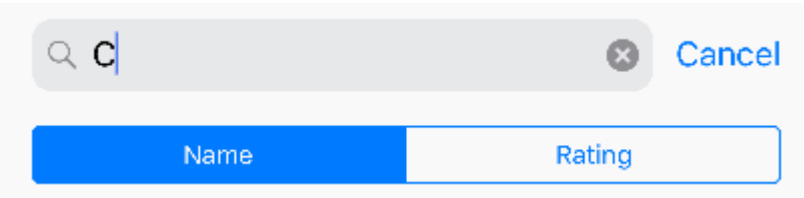


Figura 1. La barra de búsqueda y los campos que implementaremos.