01. RecyclerView

La aplicación muestra una lista de películas a través de un RecyclerView.

La aplicación dispone de un *repository* (FilmsRepository) desde el que obtener la información (List<Film>). El *repository* se crea en la *activity* y se le cede al view model en su constructor (MainViewModel). Para ello, en el view model ha sido necesario crear un «factory» (MainViewModel.factory()) utilizado para la instanciación del view model.

El RecyclerView exige un «*adapter*». Este *adapter* debe ser una instancia de `RecyclerView.Adapter`. Su propósito es contener la información a mostrar (List<Film>) y crear las vistas (*views*) con las que visualizar cada elemento de la lista (Film). En la aplicación la clase FilmAdapter representa el adapter del RecyclerView.

La definición de la view en la que mostrarán los ítems en el RecyclerView, se realiza en el recurso de tipo layout identificado por `R.layout.film_item`.

Para acelerar el rendimiento del RecyclerView, el adapter no creará una view por cada item a mostrar. En su lugar dispondrá de un pool de views. Así, cuando el usuario esté realizando scroll en el recyclerview y sea necesario mostrar en pantalla un nuevo item (una nueva película) en lugar de crear un nuevo view se reciclará uno de los que esté disponible en el pool (del mismo modo, cuando al hacer scroll uno de los ítems -película- ya no esté visible en la pantalla, dicho view pasará a estar disponible en el pool) Sólo se creará un view cuando sea necesario mostrar un item y no haya ninguno disponible en el pool.

Para lograr este propósito, el adapter exige la creación de «*view holders*». Un *view holder* no es otra cosa que un contenedor de un view. Este contenedor debe extender de la clase RecyclerView.ViewHolder. En su constructor se le debe indicar el view a través del que, a lo largo del tiempo, se irán mostrando distintos ítems.

En el adapter se debe reescribir tres métodos:

- **`getItemCount()`**: su misión es la de informar del número de ítems que deben ser mostrados a través del recicler view.
- **`onCreateViewHolder()`**: será invocado cuando sea necesario crear un view holder (no haya uno disponible en el pool). Para ello debe crearse un view (una instancia de la view definida en el layout R.layout.film_item) y crear un ViewHolder que lo contenga. Para la instanciación del view se utiliza view binding. Por otra parte, la clase FilmViewHolder representa el view holder encargado de contener las views. A su constructor se le

pasará como parámetro el objeto view binding correspondiente a la vista creada. El valor de la propiedad root de este objeto view binding representa la vista raíz del layout R.layout.film_item (un ConstraintLayout) y será el valor que se le pasará al constructor de ViewHolder (recuérdese que este parámetro debe ser el view a través del que mostrar ítems).

- **`onBindViewHolder()`**: este método será invocado para que en el view de un view holder (uno de los disponibles en el pool y que será recibido como parámetro) se muestre un determinado item (identificado por la posición que ocupa éste dentro de la lista List<Film>). Para mostrar el item en el view del view holder, se invoca al método bind() que se ha definido en la clase FilmViewHolder. En este método, utilizando view binding, se define el valor de los distintos widgets incluidos en el view.

Por otra parte, en el adapter se ha definido el método updateFilms() a través del que se establece la información a mostrar (List<Film>). Para forzar al adaptador a mostrar la información en el recycler view, se invoca a su método notifyDataSetChanged() que notifica que el conjunto de datos ha cambiado.