RECYCLERVIEWS: Segundo ejemplo RecyclerViews, más completo. Planetas.

En esta app, utilizaremos otra RecyclerView, más completa que en el ejemplo inicial. Aquí ya no hay explicaciones, sino los pasos para ir siguiendo la implementación de esta app. El código está en el GIT.

Consideraciones previas:

En vez de hacer como en el primer ejemplo que pusimos los datos directamente en el **MainActivity**, utilizaremos la siguiente estructura de clases que tendrás que crear en el paquete modelo.

Clase Planeta.

```
package modelo

data class Planet(val name: String, val sizeKm: Int, val distanceAU: Double, val imageName: String = name.lowercase())
```

Y la lista de datos será la siguiente (también en modelo como un Singleton que ya lo vimos):

```
object PlanetData {
    private val planets = mutableListOf(
        Planet("Mercurio", 4879, 0.39, "mercurio"),
        Planet("Venus", 12104, 0.72, "venus"),
        Planet("Tierra", 12756, 1.00, "tierra"),
        Planet("Marte", 6792, 1.52, "marte"),
        Planet("Jupiter", 142984, 5.20, "jupiter"),
        Planet("Saturno", 120536, 9.58, "saturno"),
        Planet("Urano", 51118, 19.22, "urano"),
        Planet("Neptuno", 49528, 30.05, "neptuno")
    )

    fun getPlanets(): MutableList<Planet> {
        return planets
    }
}
```

1. Si quieres cambia un poco el aspecto y **añade** la RecyclerView al activity principal.
2. Crea el archivo xml, para dar aspecto a cada ítem. De momento sin imagen, simplemente para que se muestre el nombre, tamaño y distancia. Yo lo he llamado ítem_planeta.xml.
   (he creado otro ítem_planeta2.xml, que simplemente cambiando la referencia en el Adapter, carga esta manera de hacer los ítems, que es con constraint layout para mejor colocación de los views. Para crear un xml con constraint layout se hace lo siguiente:

   New→XML→Layout XML File
   En **Root Element**, escribe `androidx.constraintlayout.widget.ConstraintLayout`.
3. Añade el adaptador del RecyclerView dentro de un paquete adaptador. Verás que, en este caso, al contrario que en el ejemplo incicial, el ViewHolder se define aquí mismo, de forma interna (inner). Queda más compacto.
4. En este caso nos basaremos en mi proyecto de GIT e rié explicando lo que tiene cada método.
   a. Veréis que ahora se controla si se pulsa otra vez en un elemento y se le deja el color inicial (todo con etiquetas mejor).
   b. Además, se va almacenando el índice del elemento que se pulsa, para así poder mostrarlo en un Toast, o como veremos, gracias a ese índice pasaremos el detalle a otra ventana.
   c. En este caso si se hace pulsación larga, sale un Alert Dialogo indicando si realmente queremos eliminar (en el primer ejemplo para que fuese un código más sencillo, eliminábamos directamente).

    d. He añadido un método, getSelectedItems (para así saber qué elementos han quedado seleccionados al pulsar el botón detalle en el activity principal)
5. He añadido un botón detalle en el activity principal. AL pulsar en él, vamos a un segundo activity, y mostramos el detalle de los planetas seleccionados.
6. He añadido imágenes en el ítem card, que se llama igual que el planeta así se muestra en el RecyclerView. (este punto mejor hacerlo al final, para no perder el tiempo en buscar tanta imagen).