Adaptadores y Listas - II

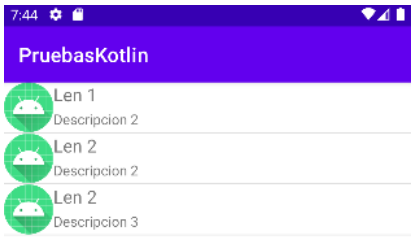
Adapters para [Kotlin]

Utilizar los adapters y listas en Kotlin es más sencillo que en Java. En el caso más básico, mostrar un listado de Strings, basta con hacer lo siguiente:

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    var myList = findViewById<ListView>(R.id.myList)
    var valores = arrayOf ("Elemento 1", "Elemento 2", "Elemento 3")

    val adapter = ArrayAdapter ( context: this, android.R.layout.simple_list_item_1, valores)
    myList.adapter = adapter

}
```

El parámetro **android.R.layout.simple_list_item_1** no es un *layout* creado por nosotros. Le está indicando la ID del *layout* que contiene un TextView para mostrar el contenido de *valores*. Existen varios ya definidos, y podríamos haber creado uno nuevo, incluirlo en nuestro proyecto y añadirlo por aquí (como en Java.

Mi adapter personalizado para [Kotlin]



De nuevo, el ejemplo de arriba sólo sirve para mostrar texto simple. En Kotlin también podemos diseñar nuestro propio adapter para que en cada ítem de la lista se muestre lo que nos interese. En este caso, vamos a hacer que se muestre un icono y dos textos por cada ítem de la lista. Nuestro ejemplo mostrará una lista de lenguajes de programación. Comenzaremos creando la clase **Lenguajes** que contiene la información de dichos lenguajes.

```
class Lenguajes (var nombre : String, var descripcion : String, var icono : Int)
```

¿Se ve raro? Recuerda: en Kotlin puedes escribir el constructor en la cabecera de la clase y pasarle parámetros de golpe. Si no dices nada más, el compilador asume que las variables son atributos de clase, que automáticamente tienen los métodos get/set, y que lo que quieres hacer es un constructor que cargue dichos parámetros.

Lo siguiente es especificar el **LenguajesArrayAdapter** y el *layout* **item_language**. Como ves, el código no es muy diferente al de Java. Hacemos referencia a los widgets del *layout* y lso cargamos con los valores correspondientes del

```
class LenguajesArrayAdapter (
    context: Context?,
    resource: Int,
    objects: List <Lenguajes>?
): ArrayAdapter<Lenguajes>(context!!, resource, objects!!) {

    override fun getView (position: Int, convertView: View?, parent: ViewGroup): View {
        val view : View = convertView?: LayoutInflater.from(this.context)
            .inflate(R.layout.item_language, parent, attachToRoot: false)

        val name = view.findViewById(R.id.nombre) as TextView
        val desc = view.findViewById(R.id.descripcion) as TextView
        val img = view.findViewById(R.id.icono) as ImageView

        getItem(position)?.let{ it: Lenguajes
            name.text = it.nombre
            desc.text = it.descripcion
            img.setImageResource(it.icono)
        }

        return view
    }
}
```

Finalmente, desde el *activity*:

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    val myList = findViewById<ListView>(R.id.myList)
    val lenguajes = mutableListOf<Lenguajes>()

    lenguajes.add (Lenguajes( nombre: "Len 1",  descripcion: "Descripcion 2", R.mipmap.ic_launcher))
    lenguajes.add (Lenguajes( nombre: "Len 2",  descripcion: "Descripcion 2", R.mipmap.ic_launcher))
    lenguajes.add (Lenguajes( nombre: "Len 2",  descripcion: "Descripcion 3", R.mipmap.ic_launcher))

    val adapter = LenguajesArrayAdapter ( context: this, R.layout.item_language, lenguajes)
    myList.adapter = adapter

}
```

Práctica 24

[Kotlin] Crea una app que muestre en un ListView una serie de objetos Usuario. Cada objeto Usuario tiene un nombre, un apellido y una edad. Crea un adapter que muestre los tres elementos. Utiliza un **ListView**.

Práctica 25

[Kotlin] Crea una app que muestre en un ListView una serie de objetos Usuario. Cada objeto Usuario tiene un nombre, un apellido y una edad. Crea un adapter que muestre los tres elementos. Utiliza un **RecyclerView**.

Práctica 26

[Kotlin] Al ejemplo de los *Lenguajes* le falta el fichero **item_language**. Completa la app.

Práctica 27

[Kotlin] Crea una app que muestre en un ListView una serie de objetos Usuario. Cada objeto Usuario tiene un nombre, un apellido y una edad. Crea un adapter que muestre los tres elementos, pero en lugar de la edad, muestra un icono si es mayor de 18 y otro diferente si no lo es. Pulsar el icono hará que éste cambie. (Acuérdate de refrescar el ListView)