Programación multimedia y dispositivos móviles

UT-2. Controles de entrada. Spinner

https://developer.android.com/guide/

Controles de selección: Listas desplegables (Spinner)

Al igual que en otros frameworks Android dispone de diversos controles que nos permiten seleccionar una opción dentro de una lista de posibilidades. Para esta funcionalidad podremos utilizar listas desplegables (**Spinner**), y listas fijas y tablas (**RecyclerView**).

Antes de empezar con los controles de selección vamos a describir un elemento importante y común a todos ellos, los **adaptadores**.

Adaptadores en Android (adapters)

Un adaptador representa algo así como una interfaz común al modelo de datos que existe por detrás de todos los controles de selección que hemos comentado. Dicho de otra forma, todos los controles de selección accederán a los datos que contienen a través de un adaptador.

Además de proveer de datos a los controles visuales, el adaptador también será responsable de generar a partir de estos datos las vistas específicas que se mostrarán dentro del control de selección. Por ejemplo, si cada elemento de una lista estuviera formado a su vez por una imagen y varias etiquetas, el responsable de generar y establecer el contenido de todos estos «sub-elementos» a partir de los datos será el propio adaptador.

Adaptadores en Android (adapters)

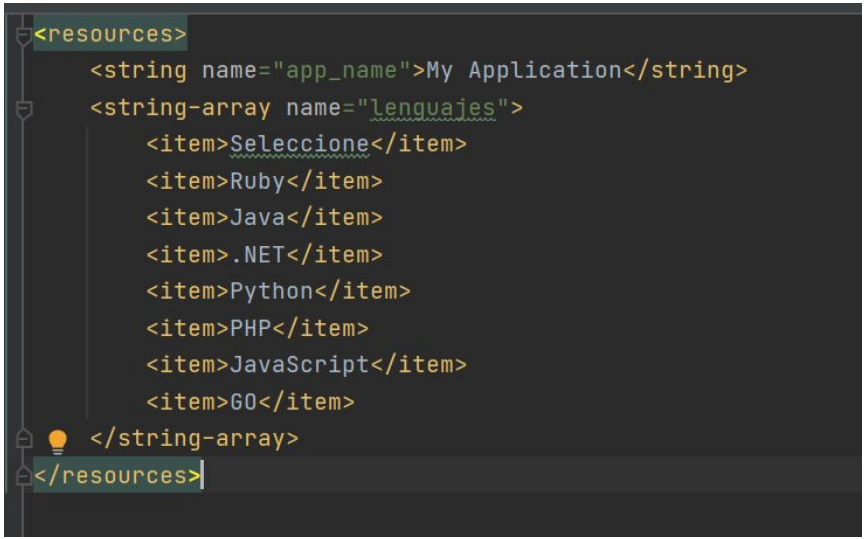
Android proporciona de serie varios tipos de adaptadores sencillos, aunque podemos extender su funcionalidad para adaptarlos a nuestras necesidades. Los más comunes son los siguientes:

- **ArrayAdapter**: Es el más sencillo de todos los adaptadores, y provee de datos a un control de selección a partir de un array de objetos de cualquier tipo.
- **SimpleAdapter**: Se utiliza para mapear datos sobre los diferentes controles definidos en un fichero XML de layout.
- **SimpleCursorAdapter**: Se utiliza para mapear las columnas de un cursor abierto sobre una base de datos sobre los diferentes elementos visuales contenidos en el control de selección.

Por ahora nos vamos a conformar con describir la forma de utilizar los 2 primeros adaptadores, ya que para el tercero necesitamos una conexión a BBDD ..

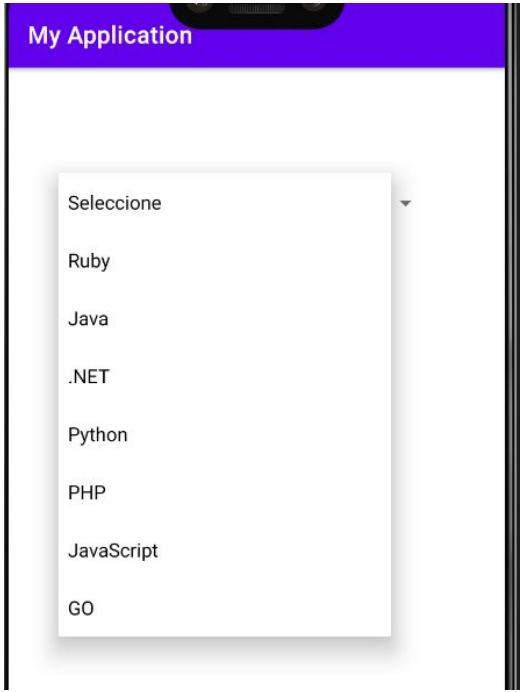
Adaptadores: Introduciendo los datos desde XML

Nos vamos al archivo res/values/string.xml para crear una entrada con las opciones.



Adaptadores: Introduciendo los datos desde XML

Lo siguiente es indicar en el spinner que usa esta entrada para mostrar las opciones, esto lo hacemos con la características “**entries**” (android:entries="@array/lenguajes" se puede poner bien directamente en el xml de la Activity o bien en la variable del editor gráfico)



Adaptadores: introduciendo los datos por programa

Para introducir en el Spinner datos por programa debemos usar la clase ArrayAdapter. Esta clase tiene varios constructores y vamos a utilizar uno en concreto.

El primer paso es dar tener un array con los datos a introducir. Ahora mismo va a ser un array estático, pero piensa que estos datos pueden salir de una BD o estar condicionados a elecciones que haga el usuario en pantalla.

```
val paises = arrayOf<String>("España","Francia", "Italia")
```

Adaptadores: introduciendo los datos por programa

Veamos cómo crear un adaptador de tipo ArrayAdapter para trabajar con un array genérico de kotlin:

```
var miAdapter =  ArrayAdapter( context: this, android.R.layout.simple_spinner_item, paises )
```

- This: es el contexto, que normalmente será simplemente una referencia a la actividad donde se crea el adaptador. OJO, más adelante no será tan directo conseguir el contexto de la actividad.
- El ID del layout sobre el que se mostrarán los datos del control. En este caso le pasamos el ID de un layout predefinido en Android (android.R.layout.simple_spinner_item), formado únicamente por un control TextView. Podría ser un Layout personalizado, pero requiere más conocimiento.
- El array que contiene los datos a mostrar.

Adaptadores: introduciendo los datos por programa

Veamos cómo crear un adaptador de tipo ArrayAdapter para trabajar con un array genérico de kotlin:

- This: es el contexto, que normalmente será simplemente una referencia a la actividad donde se crea el adaptador. OJO, más adelante no será tan directo conseguir el contexto de la actividad.
- El ID del layout sobre el que se mostrarán los datos del control. En este caso le pasamos el ID de un layout predefinido en Android (android.R.layout.simple_spinner_item), formado únicamente por un control TextView. Podría ser un Layout personalizado, pero requiere más conocimiento.
- El array que contiene los datos a mostrar.

Adaptadores: introduciendo los datos por programa

Para introducir en el Spinner datos por programa debemos usar la clase ArrayAdapter. Esta clase tiene varios constructores y vamos a utilizar uno en concreto.

El primer paso es dar tener un array con los datos a introducir. Ahora mismo va a ser un array estático, pero piensa que estos datos pueden salir de una BD o estar condicionados a elecciones que haga el usuario en pantalla.

```
val paises = arrayOf<String>("España","Francia", "Italia")
```

Adaptadores: introduciendo los datos por programa

Luego, debes llamar a **setDropDownViewResource**() del adaptador para especificar el diseño que debe usar el adaptador a fin de mostrar la lista de elecciones del control. Nuevamente cogemos uno predefinido por Android.

```
miAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
```

Después, o antes, debemos a castear en un objeto Spinner que definamos (el Spinner que queremos rellenar con este array). A nuestro spinner le asignaremos el adaptador a la propiedad “adapter”.

```
miSpinner.adapter = miAdapter
```

Adaptadores: introduciendo los datos por programa

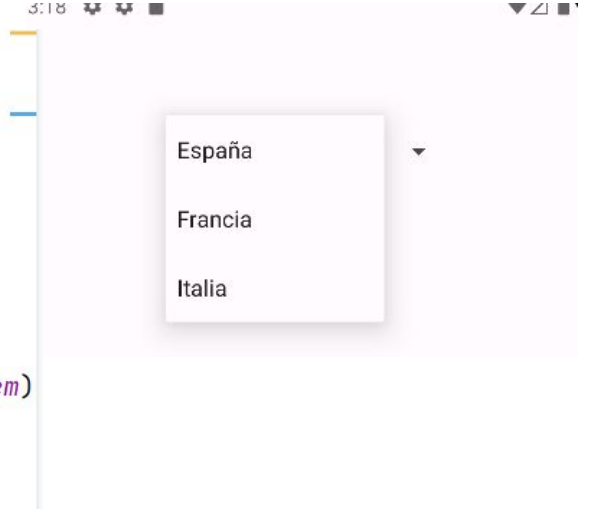
```
miSpinner = findViewById(R.id.spinner1)

val paises = arrayOf<String>("España","Francia", "Italia")

var miAdapter =  ArrayAdapter( context: this,
    android.R.layout.simple_spinner_item, paises )

miAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)

miSpinner.adapter = miAdapter
```



Adaptadores: introduciendo los datos por programa

También se puede crear el desplegable introduciendo datos por programa cuando los datos son un array que tenemos en el fichero de strings.xml utilizando el método **createFromResource().**

No es complicado, pero en este punto no le vamos a dedicar más tiempo ya que si tenemos un array fijo utilizaremos la propiedad entries del spinner.

Controles de selección: Listas desplegables (Spinner)

- **Spinner**: son listas desplegables. Funcionan de forma similar a cualquier control de este tipo, el usuario selecciona la lista, se muestra una especie de lista emergente al usuario con todas las opciones disponibles y al seleccionarse una de ellas ésta queda fijada en el control.

  Poco vamos a comentar de aquí ya que lo que nos interesan realmente son los datos a mostrar. En cualquier caso, las opciones para personalizar el aspecto visual del control (fondo, color y tamaño de fuente, …) son las mismas ya comentadas para los controles básicos.

  En cuanto a los eventos lanzados por el control Spinner, el más comúnmente utilizado será el generado al seleccionarse una opción de la lista desplegable. Para capturar este evento se procederá de forma similar a lo ya visto para otros controles anteriormente, asignándole su controlador mediante la propiedad **onItemSelectedListener**

Controles de selección: Listas desplegables (Spinner)

A diferencia de ocasiones anteriores, para este evento se definen dos métodos, el primero de ellos (**onItemSelected**) que será llamado cada vez que se seleccione una opción en la lista desplegable, y el segundo (**onNothingSelected**) que se llamará cuando no haya ninguna opción seleccionada. Para recuperar el dato seleccionado utilizamos el método **getSelectedItem**() del parámetro AdapterView que recibimos en el evento.

También podemos utilizar las propiedades **selectedItem** para ver la opción seleccionada o **selectedItemPosition** si lo que queremos en saber la posición del array elegida (por ejemplo tenemos dos arrays, uno más descriptivo para que el usuario elija una opción pero luego eso en realidad en código necesitamos un dato que está en un 2º array oculto al usuario pero en el que las posiciones se corresponden).

https://developer.android.com/guide/topics/ui/controls/spinner

Spinner: ejemplo de uso

En la actividad extendemos de la interfaz **AdapterView.OnItemSelectedListener**

```
class MainActivity : AppCompatActivity(), View.OnLongClickListener,
    AdapterView.OnItemSelectedListener
```

Al extender la interfaz debo implementar los métodos abstractos, que en este caso son 2: Cuando se seleccione un item, pintaré el valor en un textview. En caso de no haber nada seleccionado no haré nada.

Spinner: ejemplo de uso

```
override fun onItemSelected(p0: AdapterView<*>?, p1: View?, p2: Int, p3: Long) {
    val texto = miSpinner.selectedItem.toString()
    miTexto.text = texto
}

override fun onNothingSelected(p0: AdapterView<*>?) {
    TODO( reason: "Not yet implemented")
}
```

Spinner: ejemplo de uso

Por último en el evento onCreate implementamos el evento setOnItemSelectedListener como parte de la actividad (tal y como hemos hecho con otros eventos).

```
miSpinner.setOnItemSelectedListener(this);
```