

Programación multimedia y dispositivos móviles

Actividad 2.5.1

Francisco José García Cutillas | 2FPGS_DAM

Índice

Ejercicio 1	3
-------------------	---

Ejercicio 1

Cambiando el diseño del RecyclerView

A partir del ejemplo anterior, se pide rediseñar el layout para el CardView para que, por un lado, ocupe todo el área de la pantalla y, por otro lado, el desplazamiento por el RecyclerView se realice de forma horizontal en lugar de vertical. Gráficamente, se pide lo que se observa en la imagen derecha:



Consejos e indicaciones:

Crea una copia del diseño de la tarjeta y modifícala para el nuevo diseño. Así puedes alternar fácilmente entre los dos.

Puedes adaptar fácilmente el diseño modificando algunas de las constraints.

Consulta la documentación para ver cómo realizar el desplazamiento en horizontal en lugar de hacerlo en vertical.

11:42

Parques



Parque Felicidad

Parque especial para disfrutar en familia

695333653

www.felicidad.es

OpeningTime

ClosingTime

10:30

21:30

Activities

☒ Sports


☐ Mascotas

☒ ChildrenPark

☐ Bar

Save

Parques



Parque Animal

Parque especial para mascotas

695999332

www.animal.es

OpeningTime

ClosingTime

10:30

21:30

Activities

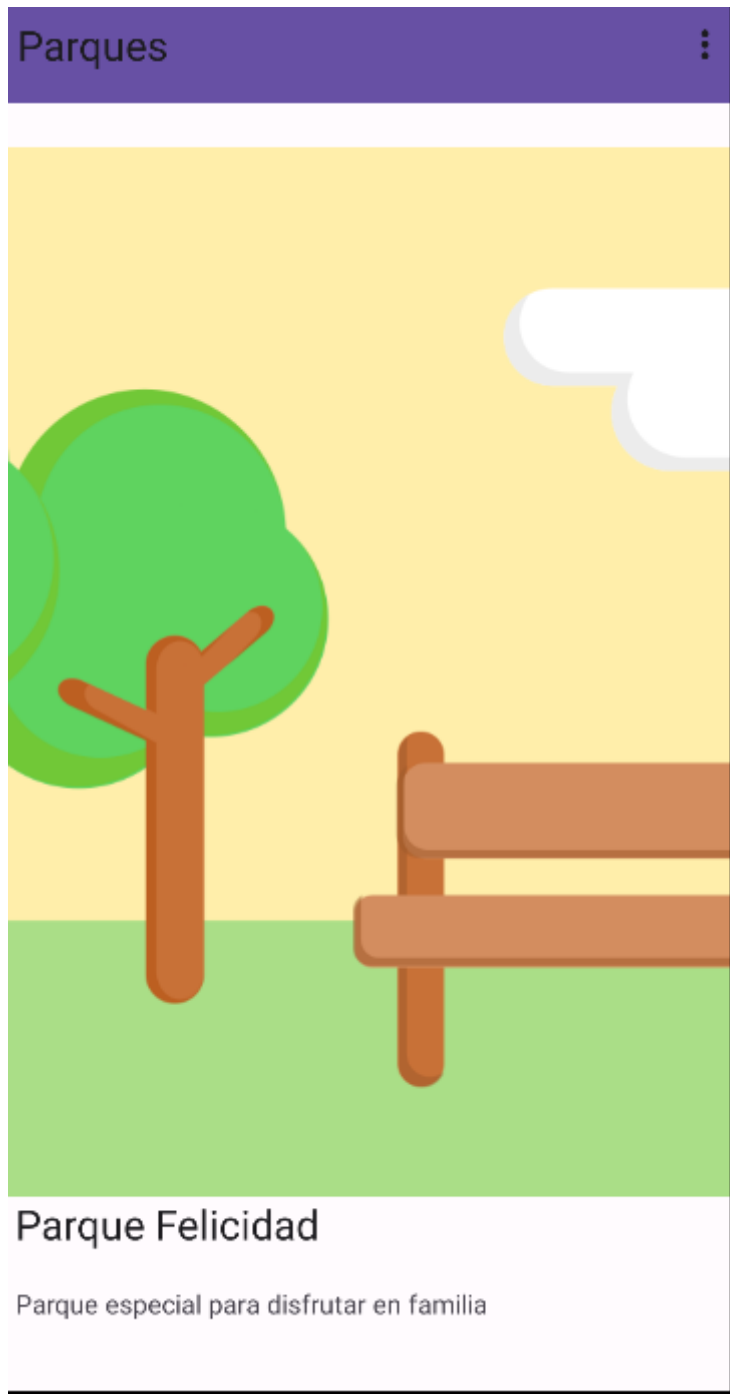
☒ Sports

☒ Mascotas

☐ ChildrenPark

☒ Bar

Save





```
AdapterPark.kt x
1 package com.example.act2_1
2
3 import ...
4
5
6
7 class AdapterPark: RecyclerView.Adapter<RecyclerView.ViewHolder>() {
8     override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): RecyclerView.ViewHolder {
9
10         val inflater = LayoutInflater.from(parent.context)
11         val vista = inflater.inflate(R.layout.parque_card, parent, attachToRoot: false)
12         return ViewHolderPark(vista)
13     }
14
15
16     override fun getItemCount(): Int {
17
18         return Parks.parks.size
19     }
20
21
22     override fun onBindViewHolder(holder: RecyclerView.ViewHolder, position: Int) {
23
24         (holder as ViewHolderPark).bind(Parks.parks[position])
25     }
26
27
28
29 }
```



```

1 package com.example.act2_1
2
3 import ...
4
15 class EditParkActivity : AppCompatActivity() {
16
17     private lateinit var editTextName: EditText
18     private lateinit var editTextDescription: EditText
19     private lateinit var editTextPhone: EditText
20     private lateinit var editTextWebsite: EditText
21     private lateinit var spinnerOpening: Spinner
22     private lateinit var spinnerClosing: Spinner
23     private lateinit var checkBoxSports: CheckBox
24     private lateinit var checkBoxChildrenPark: CheckBox
25     private lateinit var checkBoxBar: CheckBox
26     private lateinit var checkBoxMascotas: CheckBox
27     private lateinit var buttonSave: Button
28     private lateinit var binding: EditParkActivityBinding
29
30     override fun onCreate(savedInstanceState: Bundle?) {
31         super.onCreate(savedInstanceState)
32         binding = EditParkActivityBinding.inflate(layoutInflater)
33         val view = binding.root
34         setContentView(view)
35         setSupportActionBar(findViewById(R.id.materialToolbar))
36
37         val intent = Intent(packageContext, this, EditParkActivity::class.java)
38         intent.addFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT)
39         startActivity(intent)
40
41         spinner()
42
43         editTextName = binding.Name
44         editTextDescription = binding.Description
45         editTextPhone = binding.Phone
46         editTextWebsite = binding.webSite
47         spinnerOpening = binding.SpinnerOpeningTime
48         spinnerClosing = binding.SpinnerClosingTime
49         checkBoxSports = binding.cbSport
50         checkBoxChildrenPark = binding.cbChildren
51         checkBoxBar = binding.cbBar
52         checkBoxMascotas = binding.cbMascotas
53         buttonSave = binding.btSave

```

```
85 buttonSave.setOnClickListener { it: View!
86
87     val name = editTextName.text.toString()
88     val description = editTextDescription.text.toString()
89     val phone = editTextPhone.text.toString()
90     val website = editTextWebsite.text.toString()
91     val opening = spinnerOpening.selectedItem.toString()
92     val closing = spinnerClosing.selectedItem.toString()
93     var sports = ""
94     var childrenPark = ""
95     var bar = ""
96     var mascotas = ""
97
98     if (checkBoxSports.isChecked){
99         sports = "Con instalaciones deportivas"
100     } else {
101         sports = "Sin instalaciones deportivas"
102     }
103
104     if (checkBoxChildrenPark.isChecked){
105         childrenPark = "Con instalaciones para niños"
106     } else {
107         childrenPark = "Sin instalaciones para niños"
108     }
109
110     if (checkBoxBar.isChecked){
111         bar = "Con zona de restauración"
112     } else {
113         bar = "Sin zona de restauración"
114     }
115 }
```

```

98         if (checkBoxMascotas.isChecked){
99             mascotas = "Con zona para mascotas"
100         } else {
101             mascotas = "Sin zona para mascotas"
102         }
103     }
104     mostrarDialogoConfirmacion(name, description, phone, website, opening, closing, sports, childrenPark,
105                                bar, mascotas)
106 }
107
108 fun spinner(){
109     //Spinner del layout
110     spinnerClosing = binding.SpinnerClosingTime
111     val adaptador = ArrayAdapter.createFromResource( context this, R.array.horas, android.R.layout.simple_dropdown_item_1line)
112     spinnerClosing.adapter = adaptador
113 }
114
115 fun mostrarDialogoConfirmacion(nombre :String, descripcion : String, telefono : String, web : String,
116                                horaApertura :String, horaCierre :String, instDep : String,
117                                instWin :String, rest :String, masc :String){
118     var dialogoConfirmacion = AlertDialog.Builder( context this)
119     dialogoConfirmacion.setTitle("Confirmación")
120     dialogoConfirmacion.setMessage(mensajeMostrar(nombre, descripcion, telefono, web, horaApertura, horaCierre,
121                                                    instDep, instWin, rest, masc))
122     dialogoConfirmacion.setPositiveButton(android.R.string.ok, DialogInterface.OnClickListener {dialog, which ->
123         toast( mensajeToast: "Se han guardado los datos")
124         Parks.add(crearParque())
125     })
126     dialogoConfirmacion.setNegativeButton(android.R.string.cancel, DialogInterface.OnClickListener {dialog, which ->
127         toast( mensajeToast: "Acción cancelada")
128     })
129 }

```

```

130 val dialogo = dialogoConfirmacion.create()
131 dialogo.show()
132 }
133
134 fun toast(mensajeToast: String){
135     Toast.makeText(applicationContext, mensajeToast, Toast.LENGTH_SHORT).show()
136 }
137
138 fun mensajeMostrar( nombre :String, descripcion : String, telefono : String, web : String,
139                    horaApertura :String, horaCierre :String, instDep : String,
140                    instWin :String, rest :String, masc :String) :String{
141     return ("Nombre: " + nombre + "\n" +
142            "Descripción: " + descripcion + "\n" +
143            "Teléfono: " + telefono + "\n" +
144            "Web: " + web + "\n" +
145            "Hora de apertura: " + horaApertura + "\n" +
146            "Hora de cierre: " + horaCierre + "\n" +
147            "InstDep: " + instDep + "\n" +
148            "InstWin: " + instWin + "\n" +
149            "rest: " + rest + "\n" +
150            "masc: " + masc + "\n" +
151            "¿Data quieres que se guarden los datos?")
152 }
153
154 fun crearParque(): Park{
155     val newPark = Park(binding.Name.text.toString(), binding.Description.text.toString(), binding.Phone.text.toString(), binding.website.text.toString(), binding.textviewOpening.text.toString(), binding.textviewClose.text.toString(),
156                        binding.cbSport.isChecked, binding.cbChildren.isChecked, binding.cbBar.isChecked, binding.cbMascotas.isChecked)
157     return newPark
158 }

```

```

1 package com.example.act2_1
2
3 import ...
4
5 class MainActivity : AppCompatActivity() {
6
7     private lateinit var binding : ActivityMainBinding
8
9     override fun onCreate(savedInstanceState: Bundle?) {
10         super.onCreate(savedInstanceState)
11         binding = ActivityMainBinding.inflate(layoutInflater)
12         val view = binding.root
13         setContentView(view)
14         setSupportActionBar(findViewById(R.id.material_toolbar))
15
16         val intent = Intent(packageContext, MainActivity::class.java)
17         intent.addFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT)
18         startActivity(intent)
19
20         binding.ParksRecyclerView.layoutManager = LinearLayoutManager(context, LinearLayoutManager.HORIZONTAL, false)
21
22         binding.ParksRecyclerView.setHasFixedSize(true)
23
24         binding.ParksRecyclerView.adapter = AdapterPark()
25     }
26 }

```

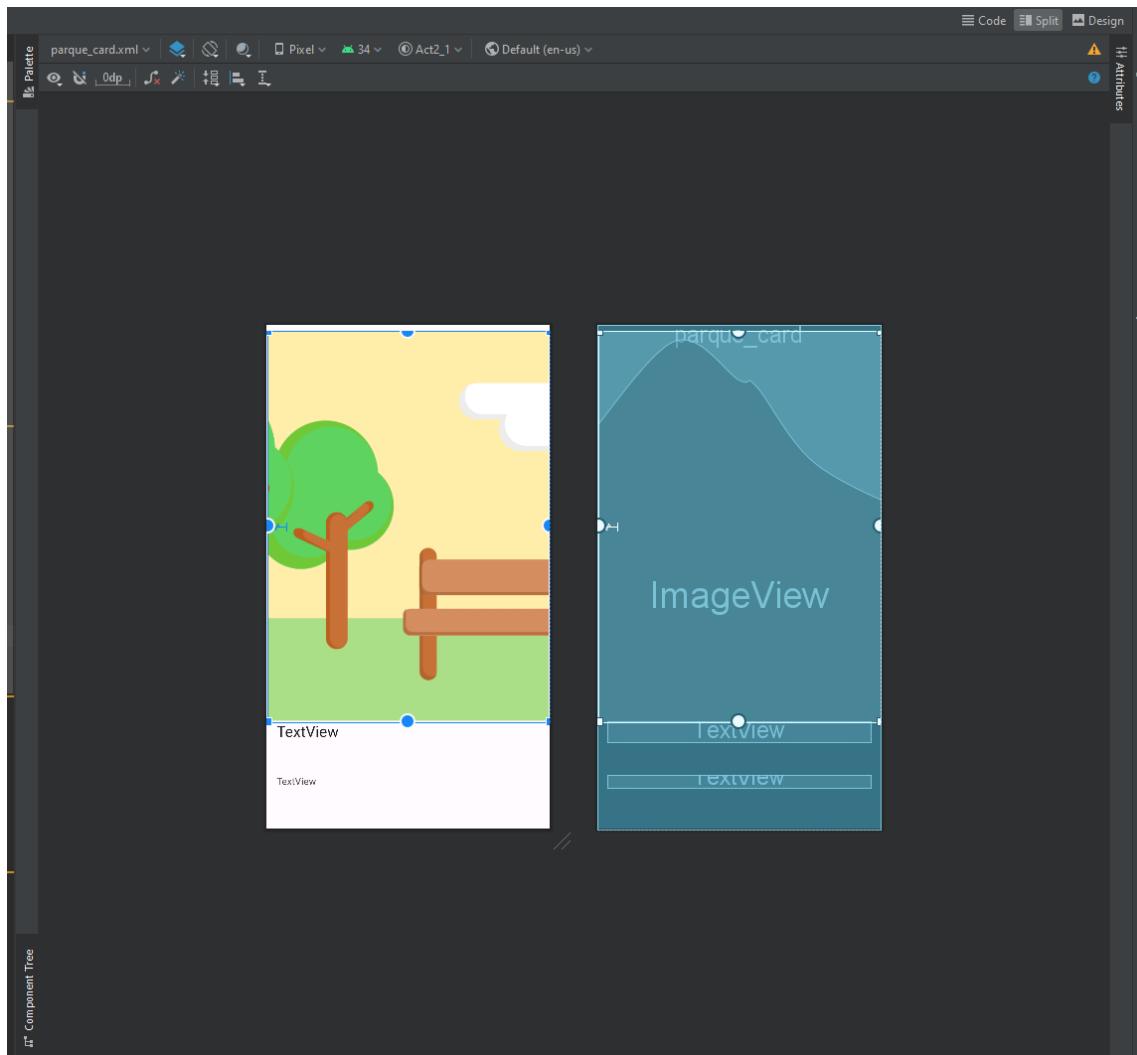
```

1 package com.example.act2_1
2
3 data class Park (
4     var name: String,
5     var parkDesc: String?,
6     var phone: String?,
7     var website: String?,
8     var opening: String?,
9     var closing: String?,
10    var sports: Boolean?,
11    var children: Boolean?,
12    var bar: Boolean?,
13    var mascotas: Boolean?
14 )

```

```
Parks.kt x
1 package com.example.act2_1
2
3 object Parks {
4
5     var parks: ArrayList<Park>
6
7     init {
8         parks = ArrayList<Park>()
9     }
10
11     fun add(park: Park){
12
13         parks.add(park)
14
15     }
16
17 }
```

```
ViewHolderPark.kt x
1 package com.example.act2_1
2
3 import ...
4
5
6
7 class ViewHolderPark(itemView: View) : RecyclerView.ViewHolder(itemView){
8
9     val parkName = itemView.findViewById(R.id.parkName) as TextView
10    val parkDesc = itemView.findViewById(R.id.parkDesc) as TextView
11
12    fun bind(park : Park){
13
14        parkName.text = park.name
15        parkDesc.text = park.parkDesc
16
17    }
18
19 }
```



```

1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent">
7
8      <androidx.cardview.widget.CardView
9          android:id="@+id/parque_card"
10         android:layout_width="0dp"
11         android:layout_height="wrap_content"
12         android:scrollbarStyle="outsideInset"
13         app:cardUseCompatPadding="true"
14         app:contentPadding="0dp"
15         app:layout_constraintEnd_toEndOf="parent"
16         app:layout_constraintStart_toStartOf="parent"
17         app:layout_constraintTop_toTopOf="parent" />
18
19         <androidx.constraintlayout.widget.ConstraintLayout
20             android:layout_width="match_parent"
21             android:layout_height="match_parent"
22             android:layout_marginTop="8dp"
23             app:layout_constraintEnd_toEndOf="parent"
24             app:layout_constraintHorizontal_bias="1.0"
25             app:layout_constraintStart_toStartOf="parent"
26             app:layout_constraintTop_toTopOf="parent">
27
28                 <ImageView
29                     android:id="@+id/parkImageView"
30                     android:layout_width="409dp"
31                     android:layout_height="567dp"
32                     android:layout_marginStart="15dp"
33                     android:layout_marginEnd="15dp"
34                     android:scaleType="centerCrop"
35                     app:layout_constraintBottom_toTopOf="@+id/parkName"
36                     app:layout_constraintEnd_toEndOf="parent"
37                     app:layout_constraintStart_toStartOf="parent"
38                     app:layout_constraintTop_toTopOf="parent"
39                     app:layout_constraintVertical_bias="0.035"
40                     app:srcCompat="@drawable/appimg" />
41

```

```

42         <TextView
43             android:id="@+id/parkName"
44             android:layout_width="0dp"
45             android:layout_height="wrap_content"
46             android:layout_marginStart="15dp"
47             android:layout_marginEnd="15dp"
48             android:text="@string/parkName"
49             android:textAppearance="@style/TextAppearance.AppCompat.Large"
50             app:layout_constraintEnd_toEndOf="parent"
51             app:layout_constraintHorizontal_bias="0.0"
52             app:layout_constraintStart_toStartOf="parent"
53             app:layout_constraintTop_toBottomOf="@+id/parkImageView" />
54
55         <TextView
56             android:id="@+id/parkDesc"
57             android:layout_width="0dp"
58             android:layout_height="wrap_content"
59             android:layout_marginStart="15dp"
60             android:layout_marginEnd="15dp"
61             android:text="@string/parkDesc"
62             app:layout_constraintBottom_toBottomOf="parent"
63             app:layout_constraintEnd_toEndOf="parent"
64             app:layout_constraintHorizontal_bias="0.0"
65             app:layout_constraintStart_toStartOf="parent"
66             app:layout_constraintTop_toBottomOf="@+id/parkName"
67             app:layout_constraintVertical_bias="0.448" />
68     </androidx.constraintlayout.widget.ConstraintLayout>
69 </androidx.constraintlayout.widget.ConstraintLayout>

```

