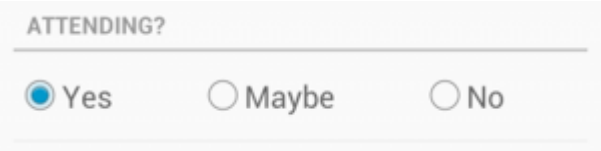
INDICE

1.1. Introducción

- El control **RadioButton** le permite elegir entre varias opciones posibles.
- Es recomendable utilizar este control siempre que desee mostrar al usuario múltiples opciones, una al lado de la otra.
- De lo contrario, se debe utilizar un Spinner.
- En el diseño, los **RadioButtons** están organizados dentro de un **RadioGroup** que controlará que no haya más de una opción marcada.



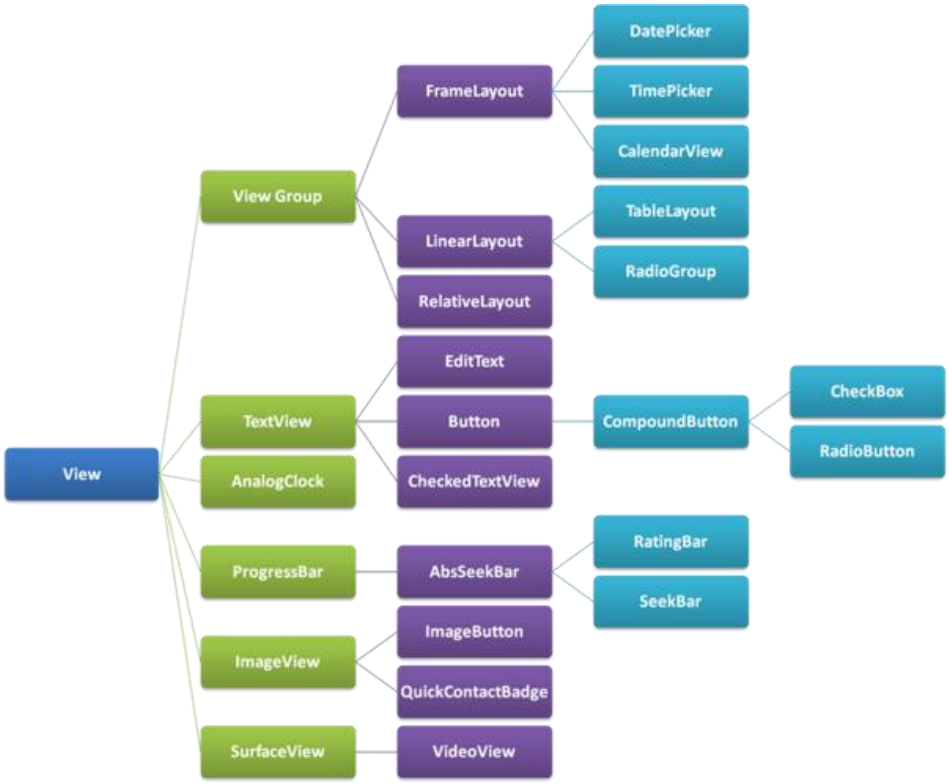
- **RadioGroup** hereda la clase **LinearLayout** con la que tendrá sus propiedades, entre las que se incluyen: **android: orientación**

- **RadioButton** hereda de la clase **CompoundButton** al igual que **CheckBox** :

  Por tanto, funcionan de la misma forma:

  - ✓ Tiene 2 estados (True/False), que podemos comprobar con el método isChecked().
  - ✓ Podemos cambiar su estado mediante programación llamando al método setChecked (boolean)
  - ✓ Podemos gestionar el cambio de estado de un RadioButton con la llamada al método setOnCheckedChangeListener (OnCheckedChangeListener)

- También podemos gestionar eventos de la misma forma que en los botones, con la interfaz OnClickListener o mediante la propiedad de **android:onCLick**.



- Referencias:
  - ✓ Control RadioButton: https://developer.android.com/reference/android/widget/RadioButton.html
  - ✓ Control RadioGroup: https://developer.android.com/reference/android/widget/RadioGroup.html
  - ✓ Introducción al control: https://developer.android.com/guide/topics/ui/controls/radiobutton.html

1.2. Caso práctico

- Partimos que ya tenemos creado el proyecto inicial.

Recordar que si estamos empleando las bibliotecas de compatibilidad (por defecto), **vuestra activity derivará de AppCompactActivity y no de Activity.**

En los ejemplos siguientes toda activity deriva de Activity, pero lo normal es que empleéis AppCompactActivity.

Si no lo tenemos creado antes, crear un paquete de nome **UI** como un subpaquete de tu paquete principal.

Dentro do paquete UI, crearemos un nuevo paquete de nombre: **RadioButtons**.

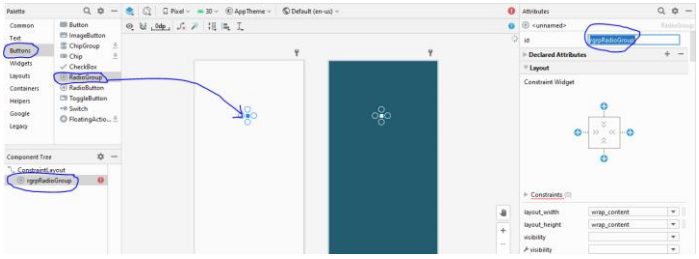
- Dentro del paquete **RadioButtons** crear una nueva 'Empty Activity' de nombre: **UD02_01_RadioButtons** de tipo Launcher.

  Modificar el archivo **AndroidManifiest.xml** y añade un label a la activity como ya vimos en la creación del proyecto base.

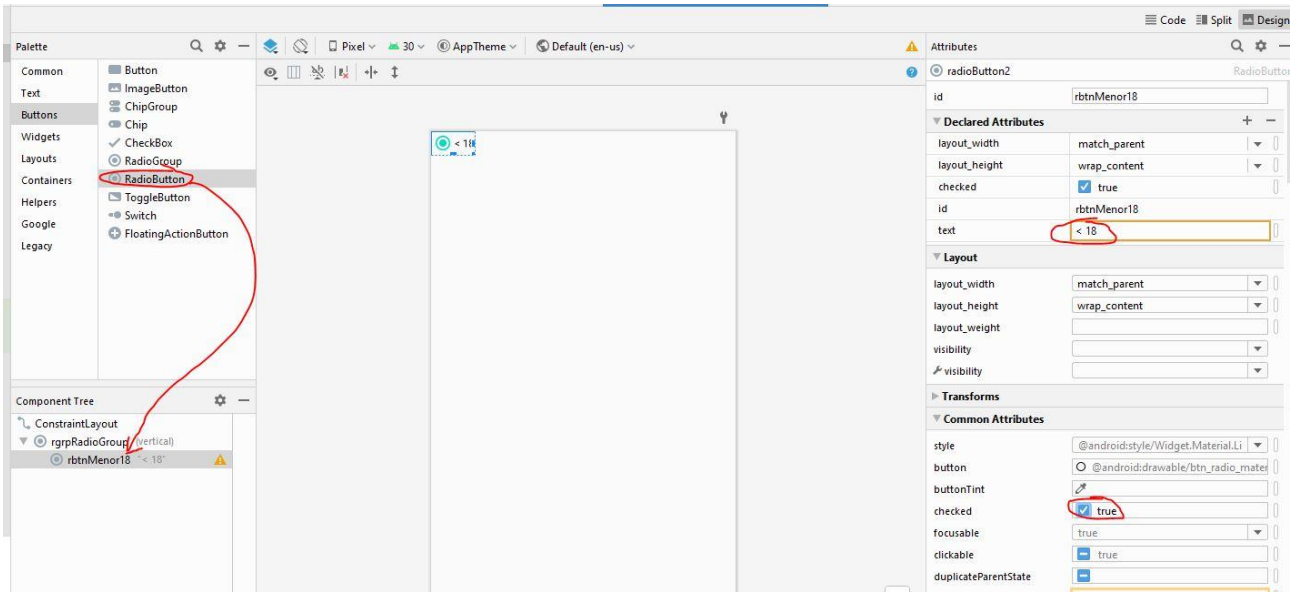
- Vamos a crear una aplicación en la que el usuario indique si es o no mayor de edad y cuál es su deporte favorito.

  Para estos tenemos que crear dos RadioGroups. Cada uno de ellos gestionará el conjunto de RadioButtons que tenga de tal forma que solamente podrá estar activo uno de los dos RadioButtons por grupo.

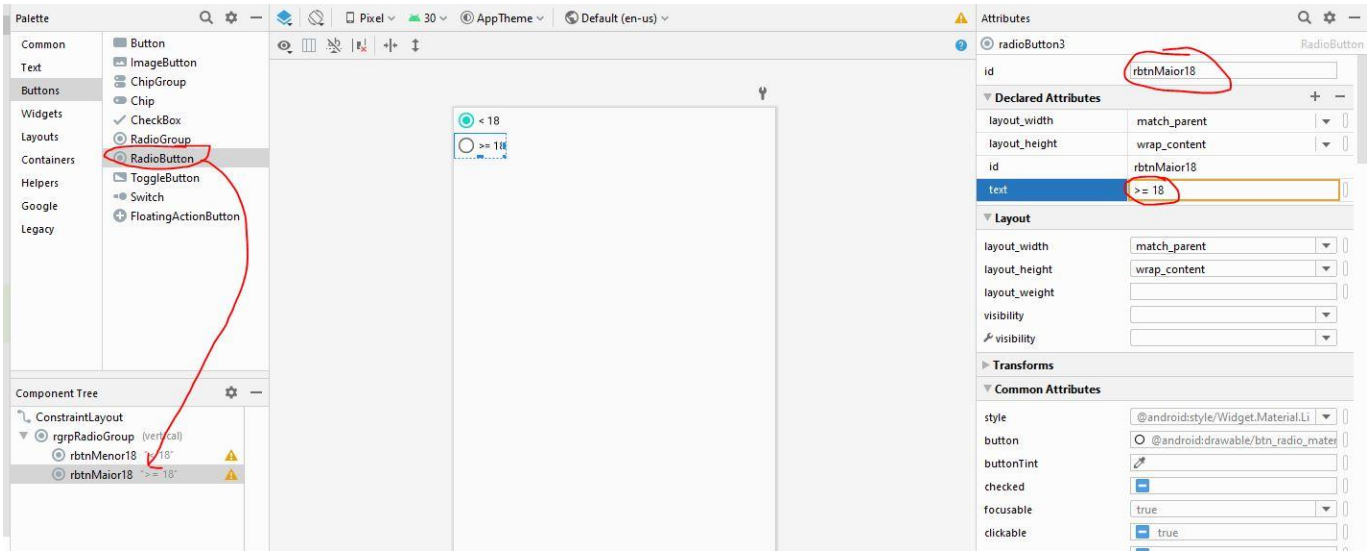
RadioButton



Arrastramos un RadioGroup y lo anclamos. Fijarse que o RadioGroup tiene como layout un LinearLayout y por tanto podemos cambiar la propiedad Orientation para hacer que los RadioButton se sitúen en horizontal o vertical.

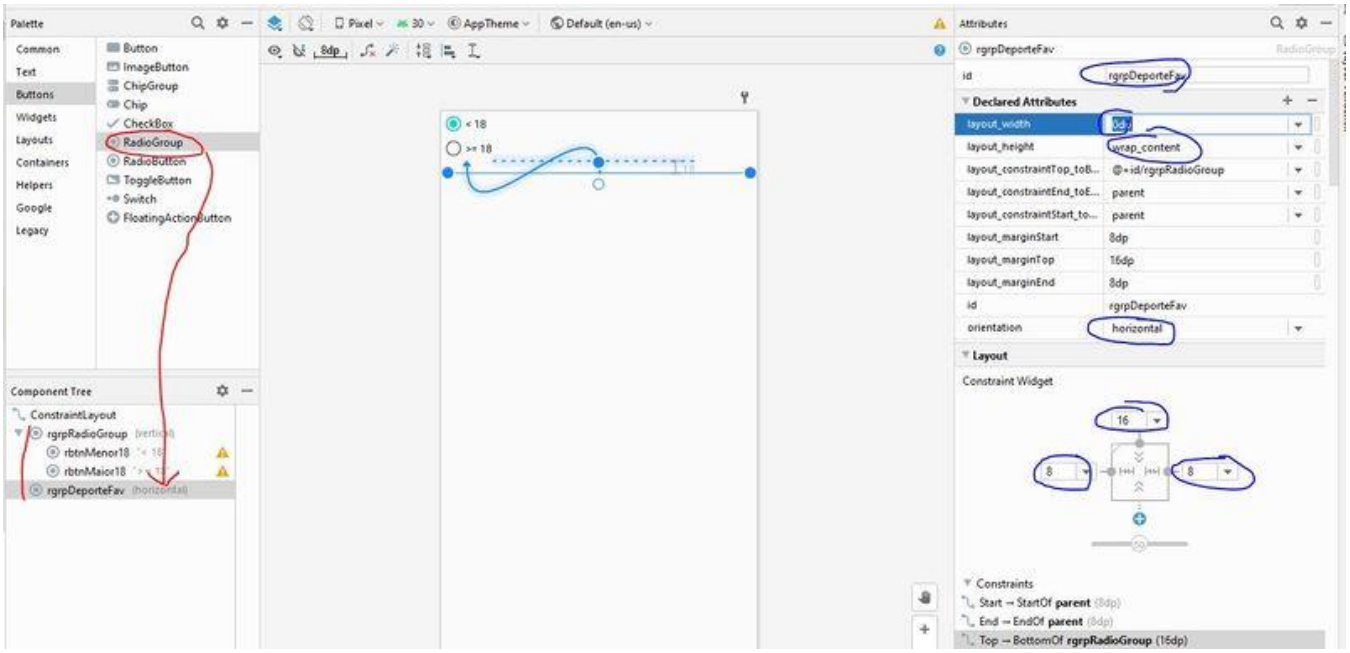


Arrastramos ahora un RadioButton al interior del RadioGroup. Tiene que quedar dentro de la jerarquía Views (en la ventana Component Tree). Cambiamos el texto y lo checkeamos por defecto y también le damos un *id*. Nota: En la imagen aparece el texto '**< 18**'. Para escribirlo tenéis que poner en la propiedad Text: <18

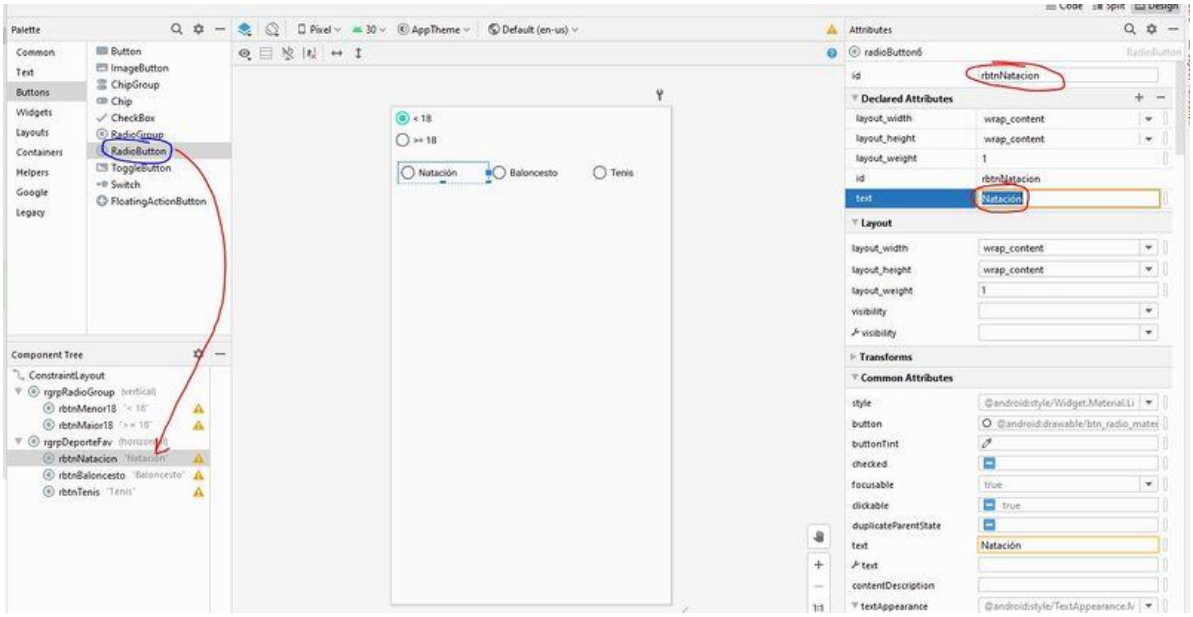


Arrastramos el siguiente. Cambiamos el id y el texto.

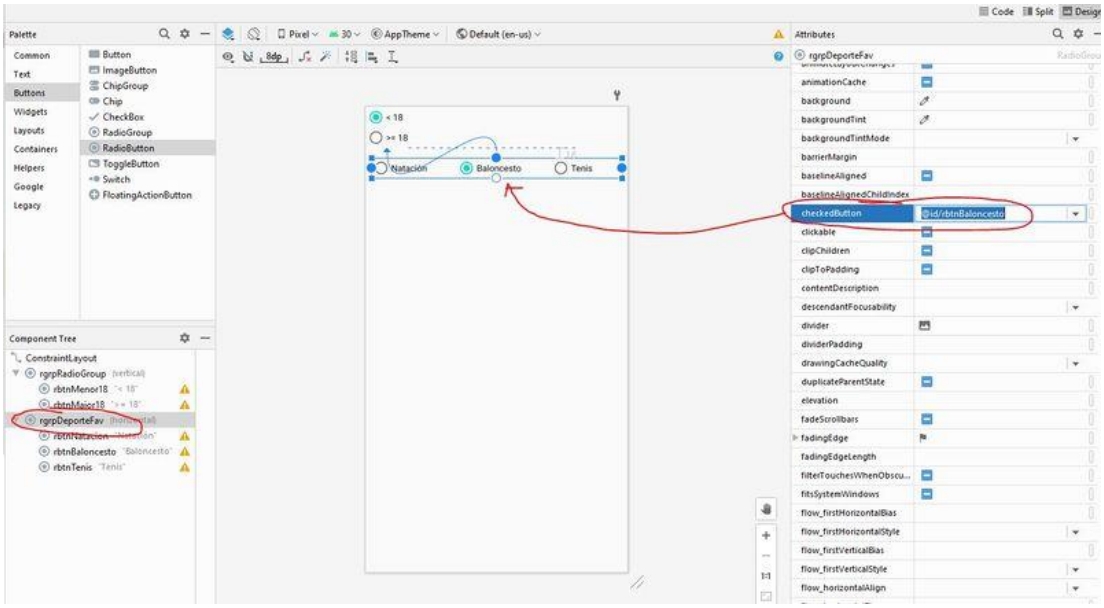
RadioGroup



Arrastramos el segundo RadioGroup y lo anclamos según nuestras preferencias. Fijarse que o RadioGroup tiene que estar a la misma altura que el otro Radiogroup en la ventana ComponentTree. Cambiamos su id y orientación (tenéis que buscar la propiedad por la lista desde abajo). En la imagen aparece arriba porque el valor fue cambiado.



Arrastramos los tres RadioButtons dentro del RadioGroup y cambiamos el id y el texto de cada uno de ellos.



Ahora, a diferencia del radioGroup anterior, vamos a seleccionar el RadioButton por defecto cambiando la propiedad checkedButton del RadioGroup.

XML del Layout

- Observar como **RadioGroup** es un layout más de tipo Linear.
- Observar como la primera opción está activada por defecto.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".UI.RadioButtons.UD02_01_RadioButtons">

    <RadioGroup
        android:id="@+id/rgrpRadioGroup"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <RadioButton
            android:id="@+id/rbtnMenor18"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="&lt; 18" />

        <RadioButton
            android:id="@+id/rbtnMaior18"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text=">= 18" />

    </RadioGroup>

    <RadioGroup
        android:id="@+id/rgrpDeporteFav"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:checkedButton="@id/rbtnBaloncesto"
        android:orientation="horizontal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/rgrpRadioGroup">

        <RadioButton
            android:id="@+id/rbtnNatacion"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Natación" />

        <RadioButton
            android:id="@+id/rbtnBaloncesto"
            android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Baloncesto" />

        <RadioButton
            android:id="@+id/rbtnTenis"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Tenis" />
    </RadioGroup>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Código Java

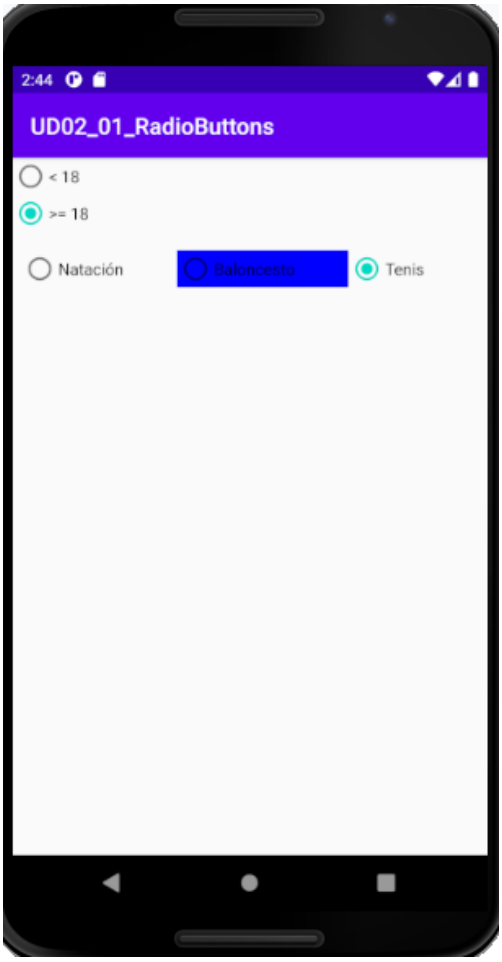
```
public class UD02_01_RadioButtons extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_u_d02_01__radio_buttons);
        RadioButton rbtnMenor18 = findViewById(R.id.rbtnMenor18);   // Obtenemos una referencia
al RadioButton
        Toast.makeText(getApplicationContext(),"Estado del rbtnMenor18:" +
String.valueOf(rbtnMenor18.isChecked()), Toast.LENGTH_SHORT).show();
        // Con el método isChecked() sabemos si está seleccionado

        ((RadioButton)findViewById(R.id.rbtnMaior18)).setChecked(true); // Mediante programa-
ción podemos cambiar el estado de un RadioButton.
                                                                        // Con setChe-
cked(boolean) cambiamos el estado.

        RadioGroup rgroupDeportes = findViewById(R.id.rgrpDeporteFav);    // Obtemos la refe-
rencia a un RadioGroup
        int idRadioButtonSel = rgroupDeportes.getCheckedRadioButtonId();  // getCheckedRadio-
ButtonId() devuelve el ID del RadioButton seleccionado en el RadioGroup.
        if (idRadioButtonSel == R.id.rbtnBaloncesto){                     // Podemos comparar
dicho id cos íd´s de la clase R.
            RadioButton rbtnBaloncesto = findViewById(idRadioButtonSel);  // A través de ese id
podemos obtener una referencia al View (o RadioButton)
            Toast.makeText(getApplicationContext(),"Está seleccionado o RadioButton de balon-
cesto", Toast.LENGTH_SHORT).show();
            rbtnBaloncesto.setBackgroundColor(Color.BLUE);                //  Podemos cambiar
cualquier propiedad de ese RadioButton
        }

        ((RadioGroup)findViewById(R.id.rgrpDeporteFav)).check(R.id.rbtnTenis);
        // Mediante programación podemos cambiar el RadioButton seleccionado a través del Ra-
dioGroup como hicimos graficamente.
    }
}
```



1.3. Métodos mínimos a conocer en el manejo de los RadioButton

- Referenciar a un RadioButton - RadioGroup con el método findViewById.
- Recuperar el contenido del texto asociado a un RadioButton.
- Cambiar el contenido del texto, pudiendo emplear recursos guardados en /res/values.
- Añadir nuevo texto a uno existente.
- Modificar propiedades básicas, como color, tamaño, visibilidad,...
- Obtener el radiobutton seleccionado.
- Seleccionar un radiobutton concreto.

```
// Ejemplo para obtener el id del radiobutton seleccionado. Podríamos ir comprobando el isChecked de cada uno, pero de esta forma es más rápido
        RadioGroup rg = findViewById(R.id.rgrpGrupoBotons);
        int idRadioButonSel = rg.getCheckedRadioButtonId();   // Obtenemos el id del RadioButton seleccionado
        RadioButton rbtnSel = findViewById(idRadioButonSel);  // Obtenemos la referencia al View a través del id anterior.


        RadioButton rb = findViewById(R.id.rbtnIdaBuscar);
        if (rb.isChecked()) {    // con isChecked() podemos saber si está seleccionado
                                 // Código a ejecutar en el caso de que o radiobutton esté seleccionado.
        }

        rb.setCheked(true);      // Seleccionamos por código un radiobutton concreto.

        rb.check(R.id.rbtnID);   // Seleccionamos un radiobutton concreto a través del radiogroup.
```