Juan Menduiña

Trabajo Práctico N° 4

Ejercicio 1.

Crear una aplicación con un TextView centrado en pantalla que contenga un contador que vaya incrementándose automáticamente cada un segundo desde el momento que se abre la aplicación.

(a) *¿Se puede implementar el temporizador utilizando un for/while?*

No se puede implementar el temporizador utilizando un *for/while*, al menos no de forma segura o recomendable en *Android*, ya que se bloquea el hilo principal (*UI thread*), haciendo que la interfaz gráfica se congele y la *app* deje de responder.

(b) *Utilizar la clase Handler para resolver el temporizador. Investigar el método postDelayed.*

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textCounter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="0"
        android:textColor="@color/black"
        android:textSize="48sp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />
</LinearLayout>
```

En *MainActivity.kt*:

```
class MainActivity : AppCompatActivity() {
    private lateinit var textCounter: TextView
    private var counter = 0
    private val handler = Handler(Looper.getMainLooper())
    private val updateCounter = object : Runnable {
        override fun run() {
            counter++
            textCounter.text = counter.toString()
            handler.postDelayed(this, 1000)
```

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

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        textCounter = findViewById(R.id.textCounter)
        handler.post(updateCounter)
    }
    override fun onDestroy() {
        super.onDestroy()
        handler.removeCallbacks(updateCounter)
    }
}
```

Ejercicio 2.

Implementar, en la aplicación anterior, un botón que permita detener/reanudar el contador.

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textCounter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="0"
        android:textColor="@color/black"
        android:textSize="48sp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />
    <Button
        android:id="@+id/btnToggle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Detener"
        android:textColor="@color/white"
        android:textSize="16sp"
        android:textStyle="bold" />
</LinearLayout>
```

En *MainActivity.kt*:

```
class MainActivity : AppCompatActivity() {
    private lateinit var textCounter: TextView
    private lateinit var btnToggle: Button
    private var counter = 0
    private var isRunning = true
    private val handler = Handler(Looper.getMainLooper())
    private val updateCounter = object : Runnable {
        override fun run() {
            if (isRunning) {
                counter++
                textCounter.text = counter.toString()
                handler.postDelayed(this, 1000)
            }
        }
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
```

```
ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        textCounter = findViewById(R.id.textCounter)
        btnToggle = findViewById(R.id.btnToggle)
        handler.post(updateCounter)
        btnToggle.setOnClickListener {
            isRunning = !isRunning
            if (isRunning) {
                btnToggle.text = "Detener"
                handler.post(updateCounter)
            }
            else {
                btnToggle.text = "Reanudar"
                handler.removeCallbacks(updateCounter)
            }
        }
    }
    override fun onDestroy() {
        super.onDestroy()
        handler.removeCallbacks(updateCounter)
    }
}
```

Ejercicio 3.

Implementar un estado intermedio en el contador en donde el número cambie a color rojo 500 milisegundos antes de incrementarse. Una vez que incrementa, debe volver a su color inicial.

(a) *Implementar utilizando dos Handlers.*

En *MainActivity.kt*:

```
class MainActivity : AppCompatActivity() {
    private lateinit var textCounter: TextView
    private lateinit var btnToggle: Button
    private var counter = 0
    private var isRunning = true
    private val colorHandler = Handler(Looper.getMainLooper())
    private val tickHandler = Handler(Looper.getMainLooper())
    private val colorRunnable = Runnable {
        if (isRunning) {
            textCounter.setTextColor(Color.RED)
        }
    }
    private val tickRunnable = object : Runnable {
        override fun run() {
            if (isRunning) {
                counter++
                textCounter.text = counter.toString()
                textCounter.setTextColor(Color.BLACK)
                scheduleNextTick()
            }
        }
    }
    private fun scheduleNextTick() {
        colorHandler.postDelayed(colorRunnable, 500)
        tickHandler.postDelayed(tickRunnable, 1000)
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        textCounter = findViewById(R.id.textCounter)
        btnToggle = findViewById(R.id.btnToggle)
        scheduleNextTick()
        btnToggle.setOnClickListener {
            isRunning = !isRunning
            if (isRunning) {
                btnToggle.text = "Detener"
```

```
                scheduleNextTick()
            }
            else {
                btnToggle.text = "Reanudar"
                colorHandler.removeCallbacks(colorRunnable)
                tickHandler.removeCallbacks(tickRunnable)
            }
        }
    }
    override fun onDestroy() {
        super.onDestroy()
        colorHandler.removeCallbacks(colorRunnable)
        tickHandler.removeCallbacks(tickRunnable)
    }
}
```

(b) *Implementar utilizando un solo Handler.*

En *MainActivity.kt*:

```
class MainActivity : AppCompatActivity() {
    private lateinit var textCounter: TextView
    private lateinit var btnToggle: Button
    private var counter = 0
    private var isRunning = true
    private var isRedPhase = false
    private val handler = Handler(Looper.getMainLooper())
    private val runnable = object : Runnable {
        override fun run() {
            if (isRunning) {
                if (!isRedPhase) {
                    textCounter.setTextColor(Color.RED)
                    isRedPhase = true
                }
                else {
                    counter++
                    textCounter.text = counter.toString()
                    textCounter.setTextColor(Color.BLACK)
                    isRedPhase = false
                }
                handler.postDelayed(this, 500)
            }
        }
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        textCounter = findViewById(R.id.textCounter)
```

```
        btnToggle = findViewById(R.id.btnToggle)
        handler.post(runnable)
        btnToggle.setOnClickListener {
            isRunning = !isRunning
            if (isRunning) {
                btnToggle.text = "Detener"
                handler.post(runnable)
            }
            else {
                btnToggle.text = "Reanudar"
                handler.removeCallbacks(runnable)
            }
        }
    }
    override fun onDestroy() {
        super.onDestroy()
        handler.removeCallbacks(runnable)
    }
}
```

Ejercicio 4.

Crear una aplicación con dos TextView centrados en la pantalla, "Texto Uno" y "Texto Dos". El primer TextView deberá tener, como tamaño del texto (textSize), "18dp" y el segundo "18sp". Ejecutar en un dispositivo o emulador. Luego, desde la configuración del sistema Android, cambiar el tamaño de la fuente y verificar cómo impacta esto en la aplicación desarrollada.

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textOne"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Texto Uno"
        android:textColor="@color/black"
        android:textSize="18dp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />
    <TextView
        android:id="@+id/textTwo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Texto Dos"
        android:textColor="@color/black"
        android:textSize="18sp"
        android:textStyle="bold" />
</LinearLayout>
```

Al cambiar al tamaño de la fuente, se observa que:

- El *TextView textTwo* con *textSize="18sp"* cambiará de tamaño, ya que *sp* responde a la escala de texto del sistema.
- El *TextView textOne* con *textSize="18dp"* no cambiará de tamaño, ya que *dp* no responde a la escala de texto del sistema.

Esto ocurre porque:

- *sp (scale-independent pixels)* es una unidad que escala con la configuración de tamaño de fuente del usuario, para respetar preferencias de accesibilidad y facilitar lectura.
- *dp (density-independent pixels)* es una unidad fija para elementos gráficos y no considera la escala de fuente configurada.

Ejercicio 5.

En base a la aplicación desarrollada en el Ejercicio 1, agregar soporte para idioma inglés y portugués, usando archivos XML de strings con calificadores de idioma. Verificar que los textos cambien de idioma al cambiar el idioma del sistema operativo desde la configuración.

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textCounter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="0"
        android:textColor="@color/black"
        android:textSize="48sp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />
    <Button
        android:id="@+id/btnToggle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btn_stop"
        android:textColor="@color/white"
        android:textSize="16sp"
        android:textStyle="bold" />
</LinearLayout>
```

En *MainActivity.kt*:

```
class MainActivity : AppCompatActivity() {
    private lateinit var textCounter: TextView
    private lateinit var btnToggle: Button
    private var counter = 0
    private var isRunning = true
    private var isRedPhase = false
    private val handler = Handler(Looper.getMainLooper())
    private val runnable = object : Runnable {
        override fun run() {
            if (isRunning) {
                if (!isRedPhase) {
                    textCounter.setTextColor(Color.RED)
                    isRedPhase = true
                }
                else {
                    counter++
```

```
                    textCounter.text = counter.toString()
                    textCounter.setTextColor(Color.BLACK)
                    isRedPhase = false
                }
                handler.postDelayed(this, 500)
            }
        }
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        textCounter = findViewById(R.id.textCounter)
        btnToggle = findViewById(R.id.btnToggle)
        handler.post(runnable)
        btnToggle.setOnClickListener {
            isRunning = !isRunning
            if (isRunning) {
                btnToggle.text = getString(R.string.btn_stop)
                handler.post(runnable)
            }
            else {
                btnToggle.text = getString(R.string.btn_start)
                handler.removeCallbacks(runnable)
            }
        }
    }
    override fun onDestroy() {
        super.onDestroy()
        handler.removeCallbacks(runnable)
    }
}
```

En *strings.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">TP4_E5aE6</string>
    <string name="btn_stop">Stop</string>
    <string name="btn_start">Resume</string>
</resources>
```

En *strings.xml (es)*:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">TP4_E5aE6</string>
    <string name="btn_stop">Detener</string>
    <string name="btn_start">Reanudar</string>
</resources>
```

En *strings.xml (pt)*:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">TP4_E5aE6</string>
    <string name="btn_stop">Prender</string>
    <string name="btn_start">Retomar</string>
</resources>
```

Ejercicio 6.

Agregar a la aplicación anterior una imagen como recurso drawable llamado "bandera" que se muestre debajo de los textos. Usando calificadores de idiomas para la carpeta drawable, esta bandera deberá ser la de Argentina, Brasil o Reino Unido.

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textCounter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="0"
        android:textColor="@color/black"
        android:textSize="48sp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />
    <Button
        android:id="@+id/btnToggle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btn_stop"
        android:textColor="@color/white"
        android:textSize="16sp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />
    <ImageView
        android:id="@+id/imgFlag"
        android:layout_width="240dp"
        android:layout_height="160dp"
        android:src="@drawable/bandera" />
</LinearLayout>
```

Ejercicio 7.

Cambiar la apariencia de la galería de imágenes del Ejercicio 3 de la Práctica 3, para que use un fondo oscuro y letras blancas. Aplicar un estilo con estas características, usando el atributo "theme" en la activity dentro del Manifest.

En *colors.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

En *themes.xml*:

```
<resources>
    <style name="Base.Theme.TP4_E7aE8"
parent="Theme.AppCompat.NoActionBar">
        <item name="android:windowBackground">@color/black</item>
        <item name="android:textColor">@color/white</item>
        <item name="android:textStyle">bold</item>
    </style>
    <style name="Theme.TP4_E7aE8" parent="Base.Theme.TP4_E7aE8" />
</resources>
```

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:theme="@style/Theme.TP4_E7aE8"
    tools:context=".MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <TextView
            android:id="@+id/title"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Práctica 3"
            android:textSize="20sp"
            android:background="#800000"
            android:gravity="center_vertical"
            android:padding="16dp"
            android:layout_marginBottom="6dp" />
        <TextView
            android:id="@+id/txtViewImage"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Facultad de Informática"
```

```
            android:textSize="30sp"
            android:gravity="center"
            android:padding="16dp"
            android:layout_marginBottom="6dp" />
        <ImageView
            android:id="@+id/imgViewFacultad"
            android:layout_width="match_parent"
            android:layout_height="250dp"
            android:scaleType="centerCrop"
            android:layout_marginBottom="12dp" />
        <Button
            android:id="@+id/btnSiguiente"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="SIGUIENTE"
            android:textSize="20sp"
            android:background="#888888"
            android:layout_marginStart="12dp"
            android:layout_marginEnd="12dp" />
    </LinearLayout>
</ScrollView>
```

Ejercicio 8.

Investigar cómo aplicar un tema a la activity programáticamente desde el código Kotlin. Declarar dos estilos en XML, uno llamado "día", con colores claros, y otro "noche", con colores oscuros. Los colores deberán estar declarados en el archivo de recursos de colores XML. Según la hora actual al iniciarse la aplicación, deberá mostrar uno u otro estilo.

Nota: El método setTheme() debe ejecutarse antes del método setContentView() en el onCreate() de la Activity. De este modo, se aplican los cambios de estilo antes de cargar la vista.

En *colors.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

En *themes.xml*:

```
<resources>
    <style name="dia" parent="Theme.AppCompat.NoActionBar">
        <item name="android:windowBackground">@color/white</item>
        <item name="android:textColor">@color/black</item>
        <item name="android:textStyle">bold</item>
    </style>
    <style name="noche" parent="Theme.AppCompat.NoActionBar">
        <item name="android:windowBackground">@color/black</item>
        <item name="android:textColor">@color/white</item>
        <item name="android:textStyle">bold</item>
    </style>
    <style name="Theme.TP4_E7aE8" parent="Base.Theme.TP4_E7aE8" />
</resources>
```

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:theme="@style/Theme.TP4_E7aE8"
    tools:context=".MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <TextView
            android:id="@+id/title"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
            android:text="Práctica 3"
            android:textColor="?android:textColorPrimary"
            android:textSize="20sp"
            android:background="#800000"
            android:gravity="center_vertical"
            android:padding="16dp"
            android:layout_marginBottom="6dp" />
        <TextView
            android:id="@+id/txtViewImage"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Facultad de Informática"
            android:textSize="30sp"
            android:gravity="center"
            android:padding="16dp"
            android:layout_marginBottom="6dp" />
        <ImageView
            android:id="@+id/imgViewFacultad"
            android:layout_width="match_parent"
            android:layout_height="250dp"
            android:scaleType="centerCrop"
            android:layout_marginBottom="12dp" />
        <Button
            android:id="@+id/btnSiguiente"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="SIGUIENTE"
            android:textSize="20sp"
            android:background="#888888"
            android:layout_marginStart="12dp"
            android:layout_marginEnd="12dp" />
    </LinearLayout>
</ScrollView>
```

En *MainActivity.kt*:

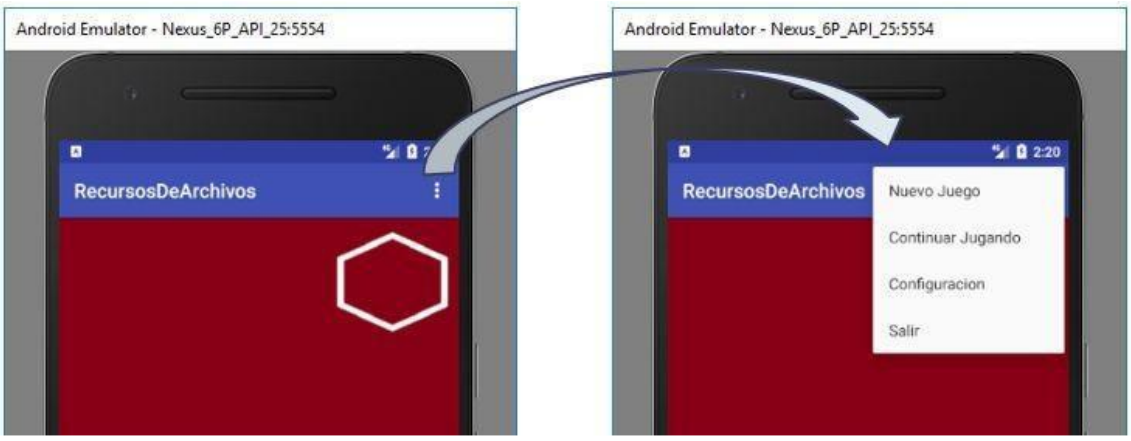
```
class MainActivity : AppCompatActivity() {
    private val imagenes = arrayOf(
        R.drawable.facultad_informatica_1,
        R.drawable.facultad_informatica_2,
        R.drawable.facultad_informatica_3
    )
    override fun onCreate(savedInstanceState: Bundle?) {
        val hour = Calendar.getInstance().get(Calendar.HOUR_OF_DAY) +
1
        if (hour in 10..22) {
            setTheme(R.style.dia)
        }
        else {
            setTheme(R.style.noche)
        }
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
```

```
            insets
        }
        val imgViewFacultad =
findViewById<ImageView?>(R.id.imgViewFacultad)
        val btnSiguiente = findViewById<Button?>(R.id.btnSiguiente)
        var index = 0
        imgViewFacultad?.setImageResource(imagenes[index])
        btnSiguiente?.setOnClickListener {
            index = (index + 1) % imagenes.size
            imgViewFacultad?.setImageResource(imagenes[index])
        }
    }
}
```

Ejercicio 9.

Agregar un menú principal, como el visto en la teoría, a la aplicación de galería de imágenes. Deberá contar con dos opciones, una "Siguiente", que adelanta una imagen, y la otra "Anterior", que vuelve a la imagen anterior.



En *menu_ppal.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<menu
    xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menuSiguiente"
        android:title="Siguiente" />
    <item
        android:id="@+id/menuAnterior"
        android:title="Anterior" />
</menu>
```

En *activity_main.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:theme="@style/Theme.TP4_E9"
    tools:context=".MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:background="#800000"
            android:gravity="center_vertical"
            android:padding="8dp"
```

```
            android:layout_marginBottom="6dp">
            <TextView
                android:id="@+id/title"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="Práctica 3"
                android:textSize="20sp" />
            <androidx.appcompat.widget.Toolbar
                android:id="@+id/toolbar"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1" />
        </LinearLayout>
        <TextView
            android:id="@+id/txtViewImage"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Facultad de Informática"
            android:textSize="30sp"
            android:gravity="center"
            android:padding="16dp"
            android:layout_marginBottom="6dp" />
        <ImageView
            android:id="@+id/imgViewFacultad"
            android:layout_width="match_parent"
            android:layout_height="250dp"
            android:scaleType="centerCrop" />
    </LinearLayout>
</ScrollView>
```

En *MainActivity.xml*:

```
class MainActivity : AppCompatActivity() {
    private val imagenes = arrayOf(
        R.drawable.facultad_informatica_1,
        R.drawable.facultad_informatica_2,
        R.drawable.facultad_informatica_3
    )
    private lateinit var imgViewFacultad: ImageView
    private var index = 0
    override fun onCreate(savedInstanceState: Bundle?) {
        val hour = Calendar.getInstance().get(Calendar.HOUR_OF_DAY)
        if (hour in 10..22) {
            setTheme(R.style.Theme_Day)
        }
        else {
            setTheme(R.style.Theme_Night)
        }
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
```

```
        val toolbar = findViewById<Toolbar?>(R.id.toolbar)
        setSupportActionBar(toolbar)
        supportActionBar?.setDisplayShowTitleEnabled(false)
        imgViewFacultad = findViewById(R.id.imgViewFacultad)
        mostrarImagenActual()
        val menuHost: MenuHost = this
        menuHost.addMenuProvider(object : MenuProvider {
            override fun onCreateMenu(menu: Menu, menuInflater:
MenuInflater) {
                menuInflater.inflate(R.menu.menu_ppal, menu)
            }
            override fun onMenuItemSelected(item: MenuItem): Boolean {
                return when (item.itemId) {
                    R.id.menuSiguiente -> {
                        mostrarImagenSiguiente()
                        true
                    }
                    R.id.menuAnterior -> {
                        mostrarImagenAnterior()
                        true
                    }
                    else -> false
                }
            }
        }, this)
    }
    private fun mostrarImagenActual() {
        imgViewFacultad.setImageResource(imagenes[index])
    }
    private fun mostrarImagenSiguiente() {
        index = (index + 1) % imagenes.size
        mostrarImagenActual()
    }
    private fun mostrarImagenAnterior() {
        index = if (index - 1 < 0) imagenes.size - 1 else index - 1
        mostrarImagenActual()
    }
}
```