Evidencia Martes 27 de Julio (día 02 / Semana 14)

Leonardo Rodenas Escobar

Reflexión:

La clase fue más interesante, aunque rápida. Lo de los hilos es muy útil, aunque no sabría como implementarlo aún. Sin embargo, esta semana ha comenzado mucho mejor y motivante en cuanto a contenidos que la anterior, por lo que el aprendizaje se hace más ameno y disfrutable.

Actividad 01:

Trabajar con hilos de ejecución en Android y hacer un contador

Hilos de ejecución en Android

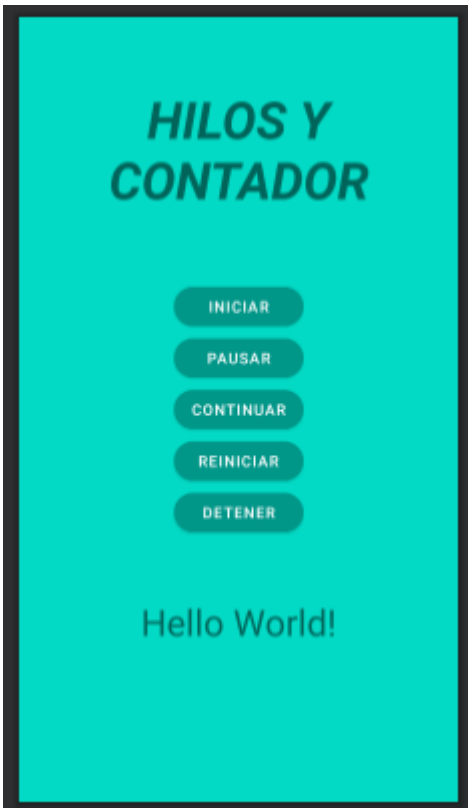
Cuando se lanza una nueva aplicación el sistema crea un nuevo hilo de ejecución (thread) para esta aplicación conocida como hilo principal. Este hilo es muy importante dado que se encarga de atender los eventos de los distintos componentes. Es decir, este hilo ejecuta los métodos onCreate(), onDraw(), onKeyDown(), ... Por esta razón al hilo principal también se le conoce como hilo del interfaz de usuario.

El sistema no crea un hilo independiente cada vez que se crea un nuevo componente. Es decir, todas las actividades y servicios de una aplicación son ejecutados por el hilo principal.
Cuando tu aplicación ha de realizar trabajo intensivo como respuesta a una interacción de usuario, hay que tener cuidado porque es posible que la aplicación no responda de forma adecuada. Por ejemplo, imagina que has de esperar para descargar unos datos de Internet, si lo haces en el hilo de ejecución principal este quedará bloqueado a la espera de que termine la descarga. Por lo tanto, no se podrá redibujar la vista (onDraw()) o atender eventos del usuario (onKeyDown()). Desde el punto de vista del usuario se tendrá la impresión de que la aplicación se ha colgado.

La solución en estos casos es crear un nuevo hilo de ejecución, para que realice este trabajo intensivo. De esta forma no bloqueamos el hilo principal, que puede seguir atendiendo los eventos de usuario. Es decir, cuando estés implementando un método del hilo principal (Empiezan por on...) nunca realices una acción que pueda bloquear este hilo, como cálculos largos o que requieran esperar mucho tiempo. En estos casos hay que crear un nuevo hilo de ejecución y encomendarle esta tarea. En el siguiente apartado describimos cómo hacerlo.
Las herramientas del interfaz de usuario de Android han sido diseñadas para ser ejecutadas desde un único hilo de ejecución, el hilo principal. Por lo tanto no se permite manipular el interfaz de usuario desde otros hilos de ejecución.

> Fuente: UPV - Maser en desarrollo de Aplicaciones móviles

Una vez dejado en claro que son los hilos en Android, se comienza a desarrollar la actividad vista en clases. Para esto se crea una interfaz sencilla como se muestra a continuación:



Y luego sobre la MainActivity se crea la lógica de este contador (la idea es que la labor de realizar el aumento de uno en uno, no se ejecute en el hilo principal, pues pondría muy lenta la app, por eso se ejecuta en un hilo secundario):

```
package com.example.hilosycontador

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import android.widget.TextView
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    var contador = 0
    var estado = true
    var pausa = false

    override fun onCreate(savedInstanceState: Bundle?) {

        var hilo = Hilo(this)

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val tvnumero = findViewById<TextView>(R.id.tvnumero)
        val btniniciar = findViewById<Button>(R.id.btniniciar)
        btniniciar.setOnClickListener {
            hilo.start()
        }
```

```
        val btnpausar = findViewById<Button>(R.id.btnpausar)
        btnpausar.setOnClickListener {
            pausa = true
        }
        val btncontinuar = findViewById<Button>(R.id.btncontinuar)
        btncontinuar.setOnClickListener {
            pausa = false
        }
        val btnreiniciar = findViewById<Button>(R.id.btnreiniciar)
        btnreiniciar.setOnClickListener {
            contador = 0
        }
        val btndetener = findViewById<Button>(R.id.btndetener)
        btndetener.setOnClickListener setOnclikListener@{
            if (hilo.isAlive) {
                estado = false
                return@setOnclikListener
            }
        }
    }

    class Hilo(activity: MainActivity) : Thread() {
        var act = activity
        override fun run() {
            super.run()
            while (act.estado) {
                while (act.pausa == true) {
                    sleep(100)
                }
                sleep(100)
                act.runOnUiThread {
                    act.tvnumero.setText("Hilo: " + act.contador)
                }
                act.contador++
            }
        }
    }
}
```

De esta manera se obtiene el contador, el cual cabe destacar que como no se ha implementado la fucnión de volver a comenzar desde el boton iniciar (no desde reiniciar) podra dar errores a futuro, pero eso es tarea del “future leo” por solucionar

Notar que en el gif de ejemplo, se inicia dando click en iniciar, lo cual comienza el contador (con un delay de 0.1 segundo entre números, eso es lo que en el código aparece como sleep(100), contado en milesimas de segundo), luego la acción es pausada para detener el contador, reiniciada para continuar su avance y finalmente detenida para no continuar contando.

Así concluye mi avance por el día de hoy, muchas gracias 😄 !!!