ANDROID BÁSICO

1. Elementos básicos

Iniciar otra Activity:

```
val intent = Intent(this, SegundaActivity::class.java)
startActivity(intent)
```

Pasar datos (Activity A -> B):

```
// ENVIAR
val intent = Intent(this, SegundaActivity::class.java)
intent.putExtra("CLAVE_NOMBRE", "Maria")
startActivity(intent)

// RECIBIR (En onCreate de la Activity B)
val nombre = intent.getStringExtra("CLAVE_NOMBRE")
```

RadioGroup (Verificar selección):

```
val radioGroup = findViewById<RadioGroup>(R.id.rgOpciones)
if (radioGroup.checkedRadioButtonId != -1) {
    val selectedId = radioGroup.checkedRadioButtonId
    val radioButton = findViewById<RadioButton>(selectedId)
    val texto = radioButton.text.toString()
}
```

2. FRAGMENTS. *IMPORTANTE: Usar `view.findViewById` dentro de `onCreateView`.*

```
class MiFragment : Fragment() {
    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // 1. INFLAR (false al final es obligatorio)
        val view = inflater.inflate(R.layout.fragment_ejemplo,
container, false)

        // 2. BUSCAR VISTAS
        val boton = view.findViewById<Button>(R.id.btnFragment)
        val texto = view.findViewById<TextView>(R.id.tvFragment)

        // 3. LÓGICA
        boton.setOnClickListener { texto.text = "Hola" }

        return view
    }
}
```

3. RECYCLERVIEW (Adapter)

```
class MiAdapter(private val lista: List<String>) :
RecyclerView.Adapter<MiAdapter.MiViewHolder>() {

    // A. VIEWHOLDER
    class MiViewHolder(itemView: View) :
RecyclerView.ViewHolder(itemView) {
        val titulo: TextView =
itemView.findViewById(R.id.tvItemTitulo)
    }

    // B. ON CREATE (Inflar diseño de fila)
    override fun onCreateViewHolder(parent: ViewGroup, viewType:
Int): MiViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.item_lista, parent, false)
        return MiViewHolder(view)
    }

    // C. ON BIND (Poner datos)
    override fun onBindViewHolder(holder: MiViewHolder,
position: Int) {
        holder.titulo.text = lista[position]
    }

    // D. GET ITEM COUNT
    override fun getItemCount() = lista.size
}
```

RECYCLERVIEW CONFIGURACIÓN:

Sin esto, la lista no se ve. Poner después de `findViewById`.

Opción A: Si estás en una ACTIVITY

```
val recycler = findViewById<RecyclerView>(R.id.miRecycler)
recycler.layoutManager = LinearLayoutManager(this) // Usamos
'this'
recycler.adapter = MiAdapter(listaDeDatos)
```

Opción B: Si estás en un FRAGMENT

```
val recycler = view.findViewById<RecyclerView>(R.id.miRecycler)
recycler.layoutManager = LinearLayoutManager(requireContext())
// Usamos 'requireContext()'

recycler.adapter = MiAdapter(listaDeDatos)
```

4. VIEWPAGER2 + TABLAYOUT (Pestañas)

Paso 1: El Adapter del ViewPager (Copia la clase) *Gestiona qué Fragment sale en cada pestaña.*

```
class ViewPagerAdapter(activity: FragmentActivity) :
FragmentStateAdapter(activity) {
    override fun getItemCount(): Int = 2 // Número total de
pestañas

    override fun createFragment(position: Int): Fragment {
        return when (position) {
            0 -> FragmentUno() // Tu primer fragment
            1 -> FragmentDos() // Tu segundo fragment
            else -> FragmentUno()
        }}
```

Paso 2: Conectar en la Activity (onCreate) *Une el ViewPager con las pestañas (Tabs).*

```
val viewPager = findViewById<ViewPager2>(R.id.viewPager)
val tabLayout = findViewById<TabLayout>(R.id.tabLayout)

// 1. Asignar el adapter
val adapter = ViewPagerAdapter(this)
viewPager.adapter = adapter

// 2. Vincular texto de pestañas (TabLayoutMediator)
TabLayoutMediator(tabLayout, viewPager) { tab, position ->
    when (position) {
        0 -> tab.text = "INICIO"
        1 -> tab.text = "PERFIL"
    }
}.attach()
```

EL SPINNER

El Spinner funciona igual que el RecyclerView: necesita un **Adapter**, pero es más fácil porque usas uno que ya viene hecho (`ArrayAdapter`).

// 1. Datos

```
val dias = listOf("Viernes", "Sábado", "Domingo")
hay que bindear spinner = findviewbyid
```

// 2. Adapter (Contexto, Diseño por defecto de Android, Lista)

```
val adapter = ArrayAdapter(requireContext(),
android.R.layout.simple_spinner_item, dias)
```

// 3. Diseño del desplegable (Opcional pero recomendado)

```
adapter.setDropDownViewResource(android.R.layout.simple_spinner_
dropdown_item)
```

// 4. Asignar

```
spinner.adapter = adapter
```

TOOLBAR (Menú Superior Personalizado)

PASO 1: ELIMINAR LA BARRA POR DEFECTO (¡Vital!) Si no haces esto, la app se cerrará sola al iniciar (Crash) porque intentará cargar dos barras a la vez.

Ve a `res/values/themes.xml` (y `themes.xml (night)` si existe) y cambia el `parent`:

```
<style name="Theme.MiApp"
parent="Theme.MaterialComponents.DayNight.NoActionBar">
    </style>
```

PASO 2: XML (Activity) En el layout de tu Activity `activity_main.xml`), añade la Toolbar arriba del todo.

```
<androidx.appcompat.widget.Toolbar
    android:id="@+id/miToolbar"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:background="@color/purple_500"
    android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
    app:layout_constraintTop_toTopOf="parent"
    app:title="Mi App"
    app:titleTextColor="@color/white" />
```

PASO 3: CREAR EL MENÚ (Los botones)

1. Clic derecho en carpeta `res` -> **New** -> **Android Resource Directory**.
2. En Resource type elige: **menu**.
3. Dentro de la carpeta `menu` creada -> Clic derecho -> **Menu Resource File** -> Llámalo `menu_toolbar.xml`.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/action_settings"
        android:icon="@android:drawable/ic_menu_preferences"
        android:title="Ajustes"
```

```
        app:showAsAction="always" />
        </menu>
```

PASO 4: KOTLIN (`MainActivity.kt`) Configuramos para que detecte el clic en "Ajustes".

```
class MainActivity : AppCompatActivity() {

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

        // 1. Vincular
        val toolbar =
findViewById<androidx.appcompat.widget.Toolbar>(R.id.miToolbar)
        setSupportActionBar(toolbar)
    }

    // 2. Cargar el menú
    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.menu_toolbar, menu)
        return true
    }

    // 3. Detectar clic
    override fun onOptionsItemSelected(item: MenuItem): Boolean
{
        if (item.itemId == R.id.action_settings) {
            // TU CÓDIGO AQUÍ (Ej: Ir a otra Activity o mostrar
Toast)
            Toast.makeText(this, "Abriendo Ajustes...",
Toast.LENGTH_SHORT).show()
            return true
        }
```

```
            return super.onOptionsItemSelected(item)
        }
    }
```