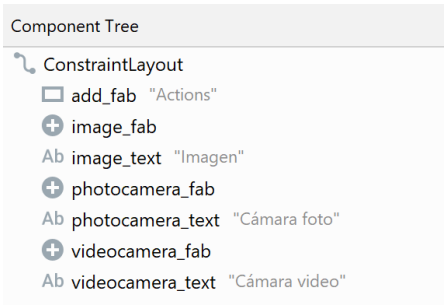
Ejercicio resuelto FAB menu

Descargar estos apuntes

Cuando pulsamos un `FAB` podemos aplicar transiciones al mismo. Es posible que nos sirva para mostrar un menú que contiene un conjunto de acciones (de tres a seis según las guías de diseño) o una `Toobar` o bien una actividad nueva.

Vamos a desarrollar un menú de acciones vinculado al `FAB` que tendrá el siguiente aspecto una vez pulsado el botón, además aparecerá un texto asociado al `FAB` y cada una de sus acciones:

Veamos los componentes necesarios para la interfaz:



El archivo xml correspondiente a esta interfaz sería:

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.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=".MainActivity">

    <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
        android:id="@+id/add_fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="end"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="16dp"
        android:text="Actions"
        app:icon="@drawable/baseline_add_black_24dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/image_fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="24dp"
        app:fabSize="normal"
        app:layout_constraintBottom_toTopOf="@id/add_fab"
        app:layout_constraintEnd_toEndOf="@id/add_fab"
        app:srcCompat="@drawable/outline_image_24" />

    <TextView
        android:id="@+id/image_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:text="Imagen"
        app:layout_constraintBottom_toBottomOf="@id/image_fab"
        app:layout_constraintEnd_toStartOf="@id/image_fab"
        app:layout_constraintTop_toTopOf="@id/image_fab" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/photocamera_fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="24dp"
        app:fabSize="normal"
        app:layout_constraintBottom_toTopOf="@id/image_fab"
        app:layout_constraintEnd_toEndOf="@id/image_fab"
        app:layout_constraintStart_toStartOf="@id/image_fab"
        app:srcCompat="@drawable/outline_photo_camera_24" />
```

```
        <TextView
            android:id="@+id/photocamera_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="8dp"
            android:text="Cámara foto"
            app:layout_constraintBottom_toBottomOf="@id/photocamera_fab"
            app:layout_constraintEnd_toStartOf="@id/photocamera_fab"
            app:layout_constraintTop_toTopOf="@id/photocamera_fab" />

        <com.google.android.material.floatingactionbutton.FloatingActionButton
            android:id="@+id/videocamera_fab"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginBottom="24dp"
            app:fabSize="normal"
            app:layout_constraintBottom_toTopOf="@id/photocamera_fab"
            app:layout_constraintEnd_toEndOf="@id/photocamera_fab"
            app:layout_constraintStart_toStartOf="@id/photocamera_fab"
            app:srcCompat="@drawable/outline_video_camera_front_24" />

        <TextView
            android:id="@+id/videocamera_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="8dp"
            android:text="Cámara video"
            app:layout_constraintBottom_toBottomOf="@id/videocamera_fab"
            app:layout_constraintEnd_toStartOf="@id/videocamera_fab"
            app:layout_constraintTop_toTopOf="@id/videocamera_fab" />
    </androidx.constraintlayout.widget.ConstraintLayout>
```

📌 Aclaraciones:

- **Línea 2**: el elemnro contenedor principal será un `ConstraintLayout`. Recordar que las vistas incluidas en el mismo deberán llevar al menos dos anclajes que determinen su posición.
- **Líneas 9-19**: para poder implemaentar esta funcionalidad utilizamos como elemnto principal un `ExtendedFloatingButton`. En la línea 16 especificamos el texto que se visualizará cuando el `FAB` esté pulsado. El resto de propiedades ya están explicadas.
- **Líneas 21-29**: aquí definimos el primer `FAB` de nuestro menú de acciones, las propiedades son también fácilmente entendibles. El anclaje del mismo se define en función del `FAB` principal.
- **Líneas 31-39**: aquí definimos el texto asociado a la acción. Su posición se define en referencia a la posición al `FAB` al que está asociado.
- Esta estructura de `FAB` y `TextView` se repite para todas las acciones que queramos incorporar.

Solamente nos falta analizar el código de nuestra aplicación, donde iremos visualizando los elementos de la vista y las acciones de cada uno de los botones.

```
class MainActivity : AppCompatActivity() {
    var isAllFabsVisible: Boolean? = null
    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        val view = binding.root
        setContentView(view)
        binding.imageFab.visibility= View.GONE
        binding.photocameraFab.visibility= View.GONE
        binding.videocameraFab.visibility= View.GONE
        binding.imageText.visibility= View.GONE
        binding.photocameraText.visibility= View.GONE
        binding.videocameraText.visibility= View.GONE
        isAllFabsVisible = false

        binding.addFab.shrink()

        binding.addFab.setOnClickListener(){
            if (isAllFabsVisible==false){
                isAllFabsVisible=true
                binding.imageFab.show()
                binding.photocameraFab.show()
                binding.videocameraFab.show()
                binding.imageText.visibility= View.VISIBLE
                binding.photocameraText.visibility= View.VISIBLE
                binding.videocameraText.visibility= View.VISIBLE
                binding.addFab.extend()
            }
            else{
                isAllFabsVisible=false
                binding.imageFab.hide()
                binding.photocameraFab.hide()
                binding.videocameraFab.hide()
                binding.imageText.visibility= View.GONE
                binding.photocameraText.visibility= View.GONE
                binding.videocameraText.visibility= View.GONE
                binding.addFab.shrink()
            }
        }

        binding.imageFab.setOnClickListener(){
            Toast.makeText(applicationContext,"Pulsaste el botón IMAGEN",Toast.LENGTH_LON
        }

        binding.photocameraFab.setOnClickListener(){
            Toast.makeText(applicationContext,"Pulsaste el botón CAMARA",Toast.LENGTH_LON
        }

        binding.videocameraFab.setOnClickListener(){
```

```
                Toast.makeText(applicationContext,"Pulsaste el botón VIDEO",Toast.LENGTH_LONG
            }
        }
    }
```

📌 Aclaraciones:

- **Línea 2**: utilizamos la variable `isAllFabsVisible` para saber si está extendido o mno el `FAB` principal, en principio (línea 16) se inicia a `false` indicando que está el menú contraído.
- **Líneas 10-15**: las vistas asociadas al menú de acciones están no visibles, para ello utilizamos el atributo `visibility` de cada una de ellas y le asociamos el valor `View.GONE` que hace que la vista sea invisible sin que la vista ocupe espacio en el diseño. `View.INVISIBLE` hace que la vista sea invisible pero ocupando espacio.
- **Líneas 18** con el método `shrink()` se desactiva el mode extendiso del `FAB` ocultando el texto que tiene asociado cuando se expanda.
- **Líneas 20-41**: aquí gestionamos el `click` sobre el botón principal. Si el menú de acciones estaba invisible, entonces hacemos visibles las vistas con `View.VISIBLE` y hacemos que aparezca el texto asociado al `FAB` inicial con el método `extend()` . Por el contrario si el `FAB` principal estaba pulsado, aplicamos la lógica inversa a lo explicado anteriormente. El método `hide()` oculta la vista con animación. Similar a `show()` que la muestra con animación.
- **Líneas 43-45** y siguientes, donde se implementa el `click` de cada una de las acciones del menú. Simplemente visualizamos un `Toast` indicando la acción pulsada.