2019

Ejercicios

PROYECTO 1

IAGO

1- Describir los siguientes tipos de layout en Android, sus propiedades, uso recomendado, y como se disponen los componentes en su interior. Incluir ejemplos tanto de código como de casos de uso:

ConstraintLayout: que le permite posicionar y dimensionar widgets de manera flexible compuesto por:

- Posicionamiento relativo
- Márgenes
- Posicionamiento de centrado
- Posicionamiento circular
- Comportamiento de visibilidad
- Restricciones de dimensión
- Cadenas
- Objetos de ayuda virtual
- Optimizador

Ejemplo:

```
<LinearLayout  xmlns:android="http://...
android:layout_height="match_parent"
android:layout_width="match_parent"
 android:orientation ="vertical">
</LinearLayout>
```

RelativeLayout: permite situar los elementos en cualquiera de los cuatro lados del contenedor e ir añadiendo nuevos elementos pegados a estos.

Ejemplo:

```
<RelativeLayout
xmlns:android="http://schemas...
android:layout_height="match_parent"
android:layout_width="match_parent">
<TextView
android:id="@+id/TextView01"
android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:layout_alignParentBottom="true"
android:text="Un texto cualquiera"/>
</RelativeLayout>
```

***LinearLayout*:** es uno de los Layout más utilizado. Distribuye los elementos uno detrás de otro, bien de forma horizontal o vertical.

```
<LinearLayout  xmlns:android="http://...
android:layout_height="match_parent"
android:layout_width="match_parent"
android:orientation ="vertical">
<Button
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Un botón"/>
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Un texto cualquiera"/>
</LinearLayout>
```

FrameLayout: posiciona las vistas usando todo el contenedor, sin distribuirlas espacialmente. Este Layout suele utilizarse cuando queremos que varias vistas ocupen un mismo lugar. Podemos hacer que solo una sea visible, o superponerlas.Para modificar la visibilidad de un elemento utilizaremos la propiedad visibility.

Ejemplo:

```
<FrameLayout xmlns:android="http://schemas...android:
layout_height="mach_parent"
android:layout_width="match_parent">
<Button
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Un botón"
android:visibility="invisible"/>
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Un texto cualquiera"
android:visibility="invisible"/>
</FrameLayout>
```

TableLayout: distribuye los elementos de forma tabular. Se utiliza la etiqueta <TableRow> cada vez que queremos insertar una nueva línea.

Ejemplo:

```
<TableLayout xmlns:android=”http://...
android:layout_height="match_parent"
android:layout_width="match_parent">
<TableRow>
<AnalogClock
android:layout_width="wrap_content"
android:layout_height="wrap_content"/>
<CheckBox
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Un checkBox"/>
</TableRow>
<TableRow>
<Button
android:layout_width="wrap_content"
android:layout_height="wrap_content"
```

```
android:text="Un botón"/>
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Un texto cualquiera"/>
</TableRow>
</TableLayout>
```

***GridLayout**:* se utiliza igualmente para distribuir los diferentes elementos de la interfaz de forma tabular, distribuidos en filas y columnas.

Ejemplo:

```
<GridLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:rowCount="2"
android:columnCount="3"
android:orientation="horizontal" >
<TextView android:text="Celda 1.1" />
<TextView android:text="Celda 1.2" />
<TextView android:text="Celda 1.3" />
<TextView android:text="Celda 2.1" />
<TextView android:text="Celda 2.2" />
<TextView android:text="Celda 2.3" />
<TextView android:text="Celda 3.1"
android:layout_columnSpan="2" />
<TextView android:text="Celda 3.2" />
</GridLayout>
```

2-Describir cada uno de los directorios, ficheros y subdirectorios de la carpeta "app" de un proyecto Android de Android Studio, Recopilar información de esta página, de Android Developers y/o otras Webs.

Carpeta manifests: en ella se encuentra el archivo AndroidManifest.xml el cual describe la aplicación Android. Se define su nombre, paquete, icono, estilos, etc. Se indican las activitys de la aplicación.

Carpeta java*:* contiene el código fuente de la aplicación como por ejemplo el MainActivity, que este a su vez se encuentra en una subcarpeta llamama ui, además también existe un fichero .java el cual contiene los métodos que resolverán determinados problemas, este se encuentra en la carpeta llamada core.

Carpeta res**:** esta carpeta contiene distintos recursos que utilizará la aplicación android.

Carpeta drawable*:* En esta carpeta se almacenan las imágenes que será utilizadas en el programa.

Carpeta mipmap: Aquí se almacenará el icono de la aplicación.

Carpeta layout*:* Contiene los ficheros XML que nos permitirán modelar la interfaz de usuario.

Carpeta menu*:* Ficheros XML con los menús de cada activity.

Carpeta values: También utilizaremos ficheros XML para indicar valores usados en la aplicación, de esta manera podremos cambiarlos desde estos ficheros sin necesidad de ir al código fuente.

Carpeta anim*:* Contiene ficheros XML con animaciones de vistas para nuestro interfaz gráfico.

Carpeta animator*:* Contiene ficheros XML con animaciones de propiedades.xml: Otros ficheros XML requeridos por la aplicación.

Carpeta raw*:* Ficheros adicionales que no se encuentran en formato XML.