Tarea 26: La Clase `Widget`

CONCEPTOS

Antecedentes.

El paquete de widgets contiene elementos de interfaz de usuario (en su mayoría visuales) para usarse en la pantalla de aplicaciones.

Un widget es un cuadro gráfico que se posiciona en el escritorio, y en algunos casos también en la pantalla de bloqueo, y permite a las aplicaciones mostrar información, usualmente dinámica, y a los usuarios acceder a funciones específicas de ésta.

`View`: Un View es una clase base para todos los elementos de interfaz de usuario. Por lo tanto, abarca muchas clases y conceptos diferentes, incluyendo `Widgets`, `ViewGroups` y `Layouts`. Hay una raíz View conectada a una instancia de `Window`, que forma la base de la jerarquía View. En general, la palabra View se utiliza para describir elementos de la interfaz en general, o para referirse a las clases abstractas o de base de interfaz de usuario, tales como `ViewGroups`.

`Widget`: Existen varias definiciones para este término, pero la mayoría se refieren a un elemento de interfaz de usuario 'listo para usar', ya sea un `Button`, `ImageView`, `EditText`, etc. Es confuso considerar los widgets como elementos de la IU que son completos (no abstractos) y no son contenedores (como `ViewGroups` (`layouts/listviews`); es también digno de mención que Widget es un nombre de paquete (`android.widget`), donde en los documentos se señala que el paquete de widgets contiene elementos de interfaz de usuario (en su mayoría visuales) para usar en la pantalla de aplicaciones.

`App Widget`: No se debe confundir con un widget como elemento de interfaz de usuario, una `App Widget` es una jerarquía remota `View` que se muestra con mayor frecuencia en la pantalla de inicio del usuario.

Para crear un widget propio, se extienden `View` o una subclase. Para utilizar el widget en la plantilla del XML, hay dos archivos adicionales que se deben crear, que son los siguientes:

- El archivo de implantación de Java. Este es el archivo que implanta el comportamiento del widget. Si se puede crear una instancia del objeto a partir de XML diseño, también se tendrá que codificar un constructor que recupere todos los valores de los atributos del archivo del formato XML.
- El archivo de definición XML. Un archivo XML en res/values/ que define el elemento XML utilizado para instanciar el widget y los atributos que soporta. Otras aplicaciones utilizarán este elemento y los atributos en sus plantillas XML.
- La plantilla XML. Un archivo XML opcional dentro de res/layout/ que describe la plantilla del widget. También se puede hacer esto en el código en el archivo de Java.

DESARROLLO

EJEMPLO 1.

Este ejemplo muestra un widget en la zona de Widgets del dispositivo. El widget se puede desplazar a cualquiera otra región.

Paso 1. Crear un nuevo proyecto Sensores1 en Android Studio. En la carpeta java/com.example.mipaquete, abrir y

```
import android.app.PendingIntent;
import android.appwidget.*;
import android.content.*;
import android.net.Uri;
import android.widget.*;
public class MainActivity extends AppWidgetProvider{
    public void onUpdate(Context c, AppWidgetManager appWidgetManager,int[] appWidgetIds)
{
        for(int i=0; i<appWidgetIds.length; i++){
```

```
            int id = appWidgetIds[i];
            String url = "ESCOM";
            Intent in = new Intent(Intent.ACTION_VIEW);
            in.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            in.setData(Uri.parse(url));
            PendingIntent pi = PendingIntent.getActivity(c, 0, in, 0);
            RemoteViews rv = new RemoteViews(c.getPackageName(),R.layout.activity_main);
            rv.setOnClickPendingIntent(R.id.xbn, pi);
            appWidgetManager.updateAppWidget(id, rv);
            Toast.makeText(c, "Widget agregado", Toast.LENGTH_SHORT).show();
        }
    }
}
```

Paso 2. En la carpeta res/layout, abrir y modificar el archivo activity_main.xml con el siguiente código:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
android:paddingBottom="@dimen/activity_vertical_margin"
tools:context=".MainActivity"
android:transitionGroup="true">
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="ESCOM IPN"
    android:id="@+id/xtv"
    android:layout_centerHorizontal="true"
    android:textColor="#ff3412ff"
    android:textSize="35dp" />
<Button
    android:id="@+id/xbn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Widget"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="61dp"
    android:layout_below="@+id/xtv" />
</RelativeLayout>
```

Paso 3. En la carpeta res, crear la carpeta xml. En la carpeta res/xml, crear el archivo miwidget.xml para agregar el siguiente código:

```
<?xml version="1.0" encoding="utf-8"?>
<appwidget-provider
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:minWidth="146dp"
    android:updatePeriodMillis="0"
    android:minHeight="146dp"
    android:initialLayout="@layout/activity_main">
</appwidget-provider>
```

Paso 4. En la carpeta res/values, abrir y modificar el archivo strings.xml con el siguiente código:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Mi Widget</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
</resources>
```

Paso 5. En la carpeta res/values, abrir y modificar el archivo dimens.xml con el siguiente código:

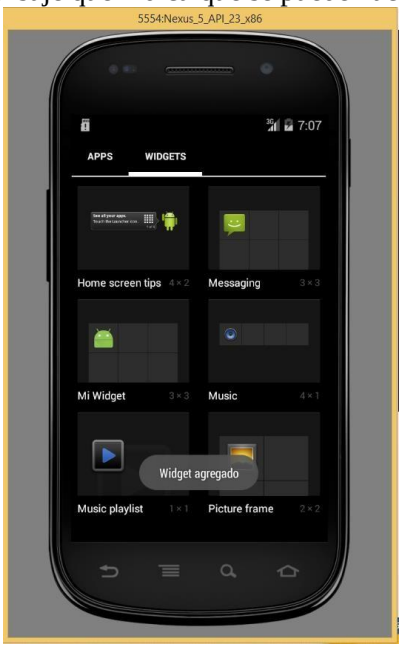
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
    <dimen name="fab_margin">16dp</dimen>
    <dimen name="widget_margin">8dp</dimen>
</resources>
```

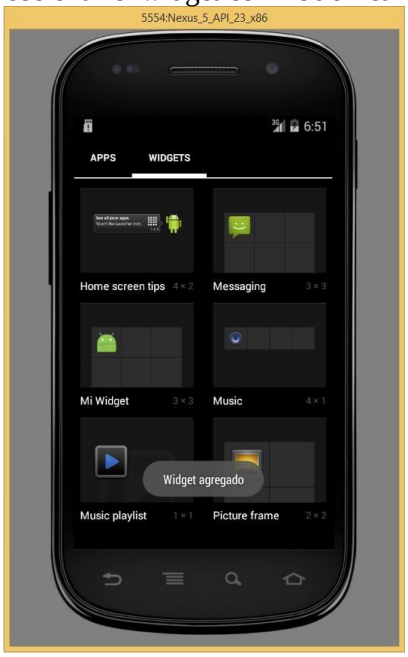
Paso 6. Por último, SÓLAMENTE si es necesario, incluir en el archivo AndroidManifest.xml, la siguiente línea:

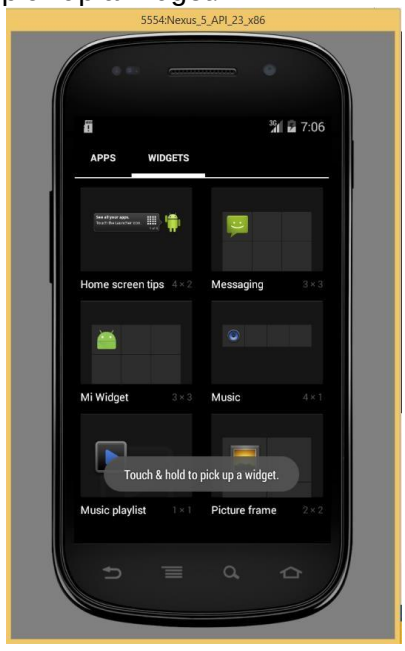
```
<meta-data android:name="android.appwidget.provider"
    android:resource="@xml/miwidget"></meta-data>
```

En la cual se hace referencia al recurso **`miwidget.xml`** agregado y que configura al widget.

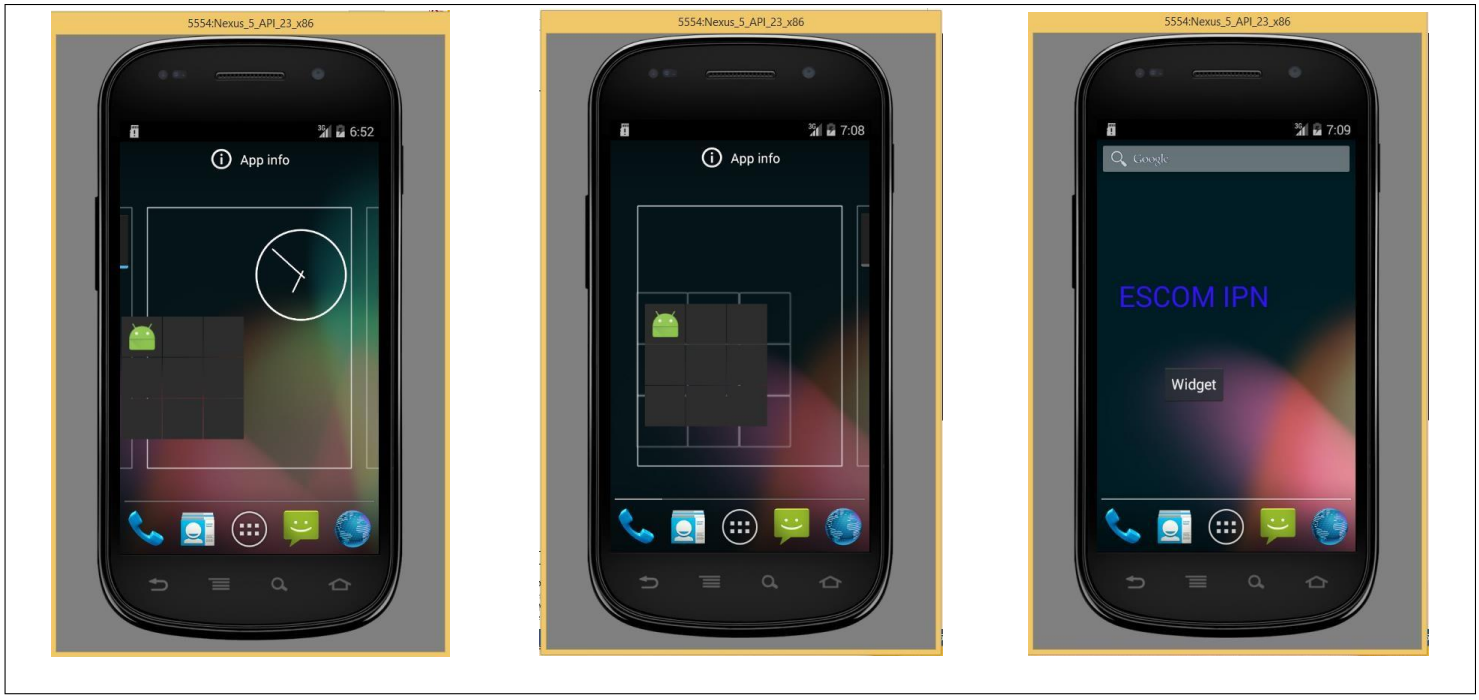
Digitar en la pestaña superior WIDGETS del dispositivo para encontrar el widget creado. Al seleccionarlo, se muestra un `Toast` con el mensaje Widget agregado, que indica que es agregado a la zona de widgets; también se muestra otro mensaje que indica qué se puede hacer al seleccionar el widget con Touch & holg to pick up a widget.







Para cambiar de ubicación al widget, se digita sosteniendo sobre él y se le arrastra, para posteriormente liberarlo en la zona deseada. Una vez ubicado, se muestra la imagen actualizada del widget.



NOTA: Guardar las imágenes de la aplicación y generar el reporte en un documento con la sintaxis AlumnoTarea26Grupo.pdf y enviarlo al sitio indicado por el profesor.