

**UNIVERSIDAD ACADEMICA DE LA REGION NORTE DE GUERRERO UT**

**Docente; ING; Abel Jerónimo Vargas**

**Alumno:**

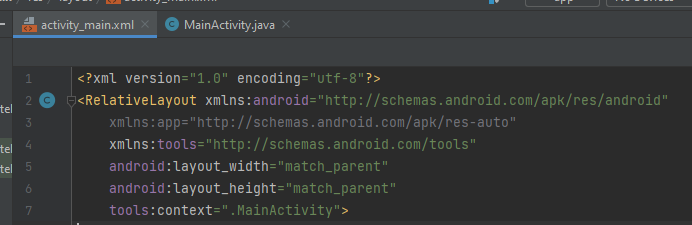
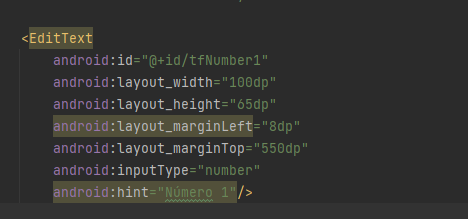
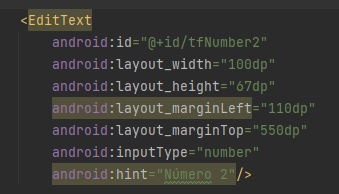
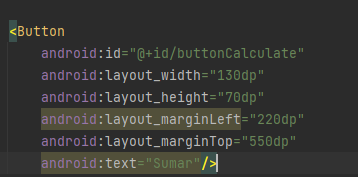
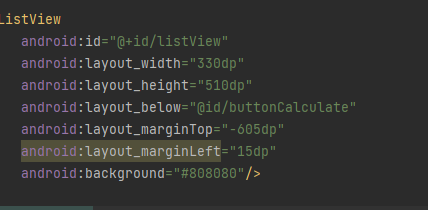
* **Roberto Chauteco Bello**
* **Documentación Act.8 Componentes de ListViews**

**Chilapa de Álvarez Guerrero, OCTUBRE 2023**

# **ARCHIVO MAIN.XML\_ESTRUCTURA**

Ilustración ARCHIVO MAIN.XML

Principalmente se crea en Diseño de la interfaz de usuario en el (archivo XML), en esta parte en el archivo XML es donde definimos la estructura de la Interfaz del usuario.

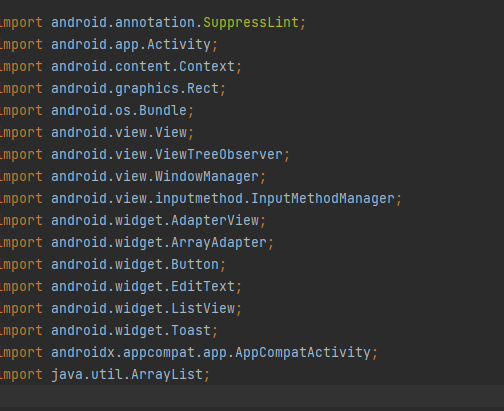
* Además, definimos un RelativeLayout como el diseño principal, que permite posicionar elementos de la interfaz de usuario de forma relativa.
* Agregamos un EditText para que el usuario ingrese el "Número 1", tiene una altura y ancho específicos, con un margen izquierdo y superior, y acepta solo números como se visualiza en la foto.
* Agregamos un EditText para que el usuario ingrese el "Número 2", tiene una altura y ancho específicos, con un margen izquierdo y superior, y acepta solo números como se visualiza en la foto.
* Agregamos un botón con un identificador único (buttonCalculate) para realizar la suma y tiene un texto de etiqueta "Sumar" y se posiciona a la derecha de los cuadros de texto.
* Por último, agregamos un ListView con un identificador único (listView) para mostrar los resultados de las sumas, es decir se posiciona debajo del botón de calcular y tiene un fondo de color gris oscuro.

# **ARCHIVO MAINACTIVITY.JAVA**

Ilustración ARCHIVO MAIN.XML

La aplicación contiene una lista simple de elementos texto que pueden agregarse a través de un campo de entrada y un botón, Cuando se hace clic en un elemento de la lista, se muestra un mensaje de Toast, Además, el código contiene e incluye su funcionalidad para controlar la aparición y desaparición del teclado virtual en la interfaz de usuario.

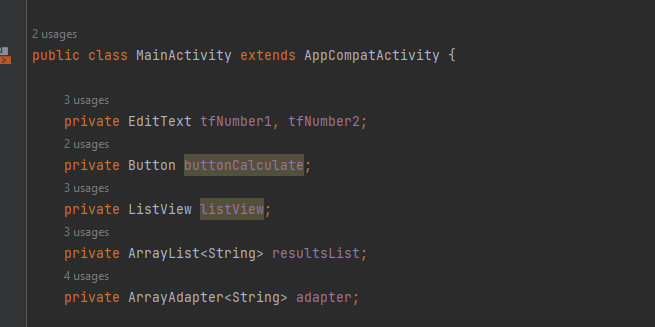
* Paquetes y Importaciones es decir este bloque de código comienza con la declaración del paquete y la importación de clases y bibliotecas necesarias para el desarrollo de la aplicación, también algunas de las importaciones son muy importantes y son claves para clases relacionadas con elementos de la interfaz de usuario y la gestión de vistas.
* La Clase Principal: es MainActivity y extiende AppCompatActivity, esto requiere que es la actividad principal de la aplicación, que generalmente es la interfaz de usuario principal.
* Y las Variables y Objeto del código, se declaran variables y objetos que se utilizan para diferentes propósitos.
* A continuación, se muestran los paquetes de importación.



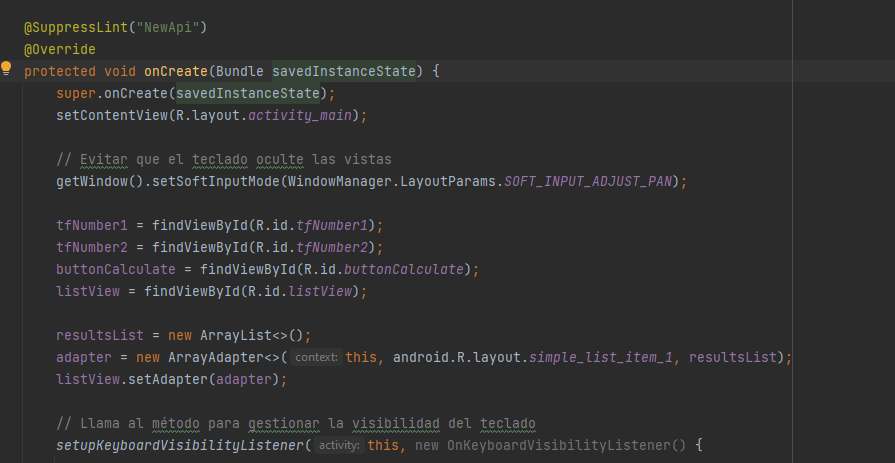
Creamos la clase principal y visualizamos lo que es la Actividad principal (Java), que lleva por nombre.

public class MainActivity extends AppCompatActivity {

* Estos objetos Editex son campos de entrada de texto en los que los usuarios pueden introducir números. Por sus nombres, parece que se espera que ingresen dos números.
* Declaran los atributos de las variables para los elementos de la interfaz de usuario y para manejar la lista de resultados como se visualizan en la foto.



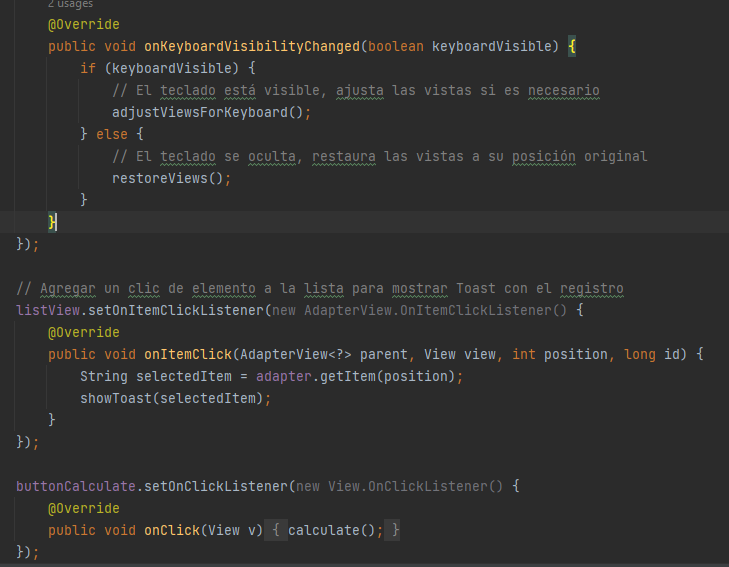
* Posteriormente creamos el método de OnCreate Se ejecuta cuando cuando se crea una nueva activity por primera vez, además llama al método onCreate de la clase base, que realiza las operaciones básicas necesarias para crear la actividad.
* setContentView(R.layout.activity\_main);: Establece el contenido de la actividad utilizando el diseño definido en el archivo XML llamado activity\_main, este archivo generalmente define la disposición y los elementos visuales de la interfaz de usuario.
* getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT\_INPUT\_ADJUST\_PAN);: Configura el modo de entrada suave (soft input, como el teclado) para ajustar la ventana para que no oculte las vistas.
* SOFT\_INPUT\_ADJUST\_PAN ajusta la ventana para que la vista activa no quede oculta por el teclado.
* Mandamos a traer cada uno de los elementos tfNumber1 = findViewById(R.id.tfNumber1);,
* tfNumber2 = findViewById(R.id.tfNumber2);,
* buttonCalculate = findViewById(R.id.buttonCalculate);,
* listView = findViewById(R.id.listView);:
* Busca y asigna las referencias de las vistas definidas en el archivo de diseño XML a las variables correspondientes. tfNumber1 y tfNumber2 son EditText (cajas de texto), buttonCalculate es un botón, y listView es una vista de lista.
* En el método onCreate se encarga de configurar la interfaz de usuario de la actividad, inicializar elementos, vincular un adaptador al ListView y asegurarse de que el teclado virtual no oculte las vistas cuando está activo.
* Su funcionamiento específico de la actividad dependerá de que se agregue en la aplicación, como el manejo de eventos del botón buttonCalculate y la actualización de resultsList.
* Como lo es Mandar a llamada al onCreate(savedInstanceState), este es el método onCreate y es necesario para inicializar correctamente la actividad.



# **Estructura Componentes**

Ilustración Estructura Componentes

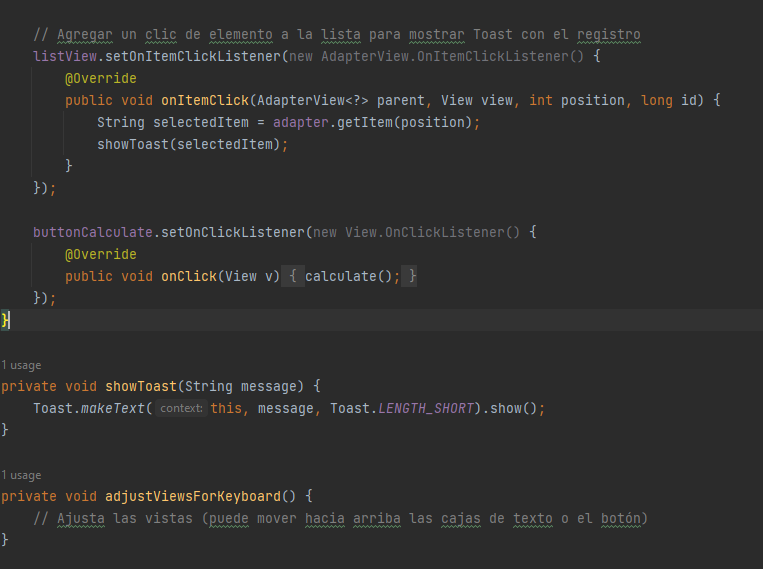
* En esta sección de la visibilidad del teclado virtual en la actividad actual va dependiendo de si el teclado se muestra u oculta, se ejecutarán acciones específicas para ajustar o restaurar las vistas de la interfaz de usuario, garantizando una experiencia de usuario adecuada cuando se utiliza el teclado virtual.
* Es decir setupKeyboardVisibilityListener(this, new OnKeyboardVisibilityListener() {...});: y esta línea de código llama a la función setupKeyboardVisibilityListener pasando dos argumentos:
* This este se refiere al contexto actual que se refiere a la actividad en la que se encuentra, el que contexto se utiliza para acceder a recursos y realizar acciones dentro de la actividad.



# **Agregación del Listiew**

Ilustración Agregación del Listiew

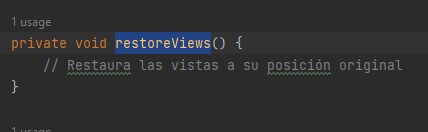
* En elementos de la lista (ListView) para mostrar mensajes de Toast cuando se hace clic en un elemento. También configura un clic en el botón "Calcular" para realizar algún tipo de cálculo o procesamiento, de la función calcúlate no se han proporcionado en el código.
* También el listView.setOnItemClickListener - Clic en elementos del ListView, es decir que Cuando se establece un clic en elementos del ListView, se crea un escuchador de clic utilizando setOnItemClickListener.
* El buttonCalculate.setOnClickListener, Clic en el botón "Calcular, que Cuando se establece un clic en el botón "Calcular" (buttonCalculate), se crea un escuchador de clic utilizando setOnClickListener.
* El showToast(String message), Este método se encarga de mostrar un mensaje de Toast (Toast) en la pantalla, recibe un mensaje como argumento y lo muestra en forma de mensaje emergente en la parte inferior de la pantalla utilizando Toast.makeText.



# **Restore\_Views**

Ilustración Restore\_Views

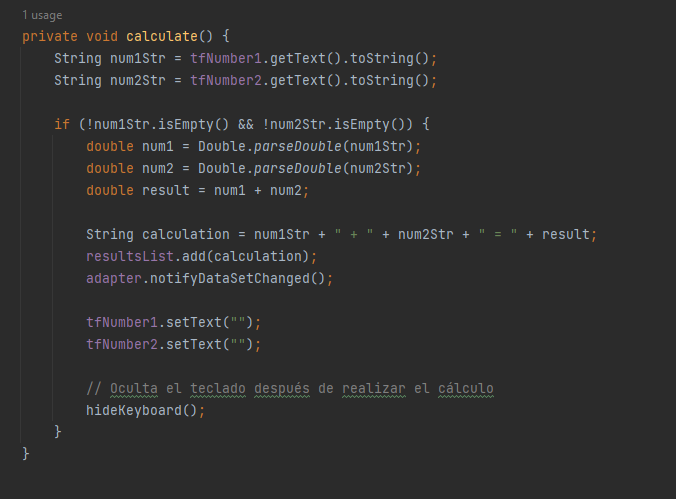
En este campo esto restaura las vistas a su posición original es decir una vez terminado su proceso esto vuelve a su estado inicio.Carga el calcúlate



# **Botton Calcúlate**

Ilustración Botton Calcúlate

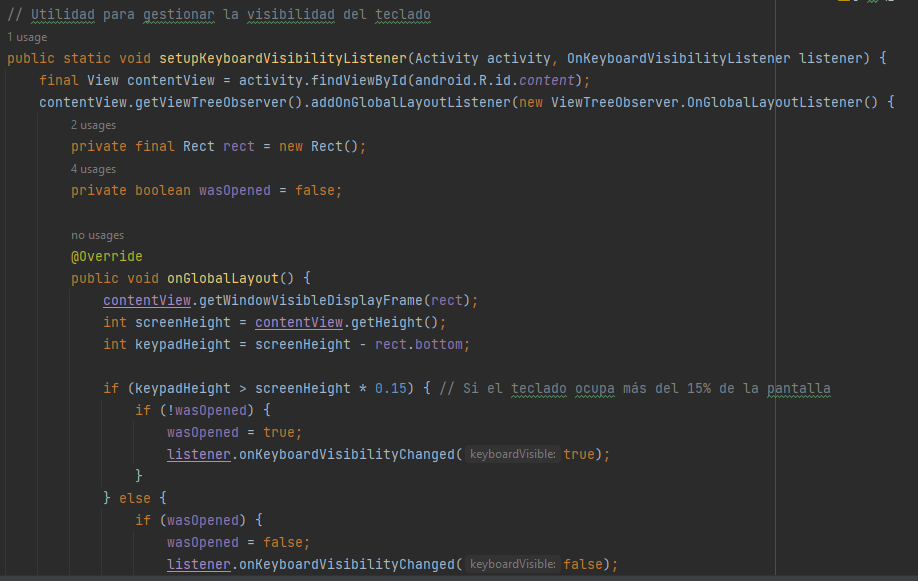
Esta función calcúlate() se encarga de obtener valores ingresados por el usuario, realizar un cálculo simple, agregar el resultado a una lista de resultados, actualizar la vista del ListView, borrar los campos de entrada y ocultar el teclado virtual.

* Procedimiento.
* Se de los valores de los campos de entrada de texto (tfNumber1 y tfNumber2), los valores ingresados por el usuario en los campos de entrada de texto tfNumber1 y tfNumber2 se obtienen y almacenan en las cadenas num1Str y num2Str, respectivamente, es decir estos valores se obtienen como cadenas de texto utilizando tfNumber1.getText().toString() y tfNumber2.getText().toString().

# **Gestion y Validacion**

Ilustración Gestion y Validacion

* Esta utilidad detecta la visibilidad del teclado virtual en una actividad de Android y notifica a un listener cuando se produce un cambio en la visibilidad del teclado, para realizar ajustes en la interfaz de usuario cuando el teclado se muestra , oculta.
* Procedimiento
* Esta línea de código cumple con la función para ser llamado desde cualquier actividad de Androi, toma dos argumentos setupKeyboardVisibilityListener(Activity activity, OnKeyboardVisibilityListener listener)::
* activity: La actividad actual a la que se aplica la gestión de la visibilidad del teclado.
* listener: crea un objete que implementa la interfaz OnKeyboardVisibilityListener, lo que permite a la actividad que llama ser notificada cuando cambia la visibilidad del teclado.



# **ESTRUCTURA FINAL DE LA APLICACION**

Ilustración ESTRUCTURA FINAL DE LA APLICACIÓN.

Finalmente asi se visualiza una vez terminada la aplicación funcionable asi el usuario calculamos dos números y nos calcula el resultado final y lo más importante es que una vez termina la operación podemos calcular de nuevo otras cantidades.

