CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

UD 05

PROGRAMACIÓN MULTIMEDIA Y DISPOSITIVOS MÓVILES 24/25

CFGS DAM

PRÁCTICA 2
AUTOCOMPLETETEXTVIEW

Autor: Mara Vañó
m.vanoalonso@edu.gva.es
Fecha: 2024/2025

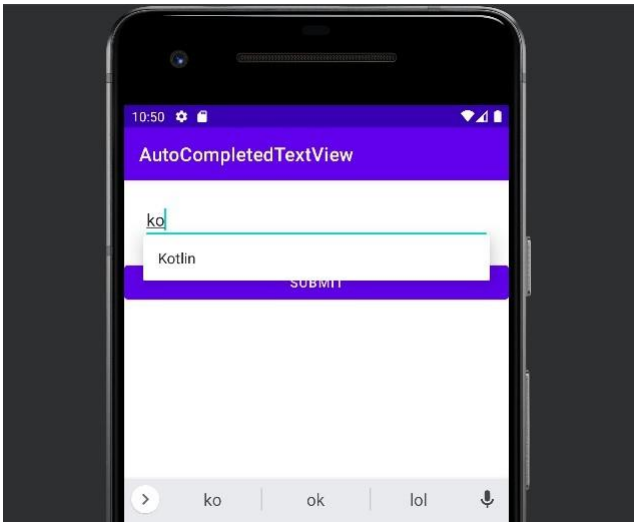
Índice

1. Objetivos de la práctica

- Implementar un AutoCompleteTextView
- Comprender los parámetros del código dentro del activity
- Generar un Layout
- Aprender a generar un listado de elementos en el directorio Values.
- Generar un Toast con el elemento introducido por teclado

2. Introducción

Una AutoCompleteTextView o vista de texto editable muestra sugerencias de finalización automáticamente mientras el usuario escribe. La lista de sugerencias se muestra en un menú desplegable en el que el usuario puede elegir un elemento para reemplazar el contenido del cuadro de edición. El menú desplegable se puede descartar en cualquier momento presionando la tecla Atrás o, si no se selecciona ningún elemento en el menú desplegable, presionando la tecla central enter/dpad.



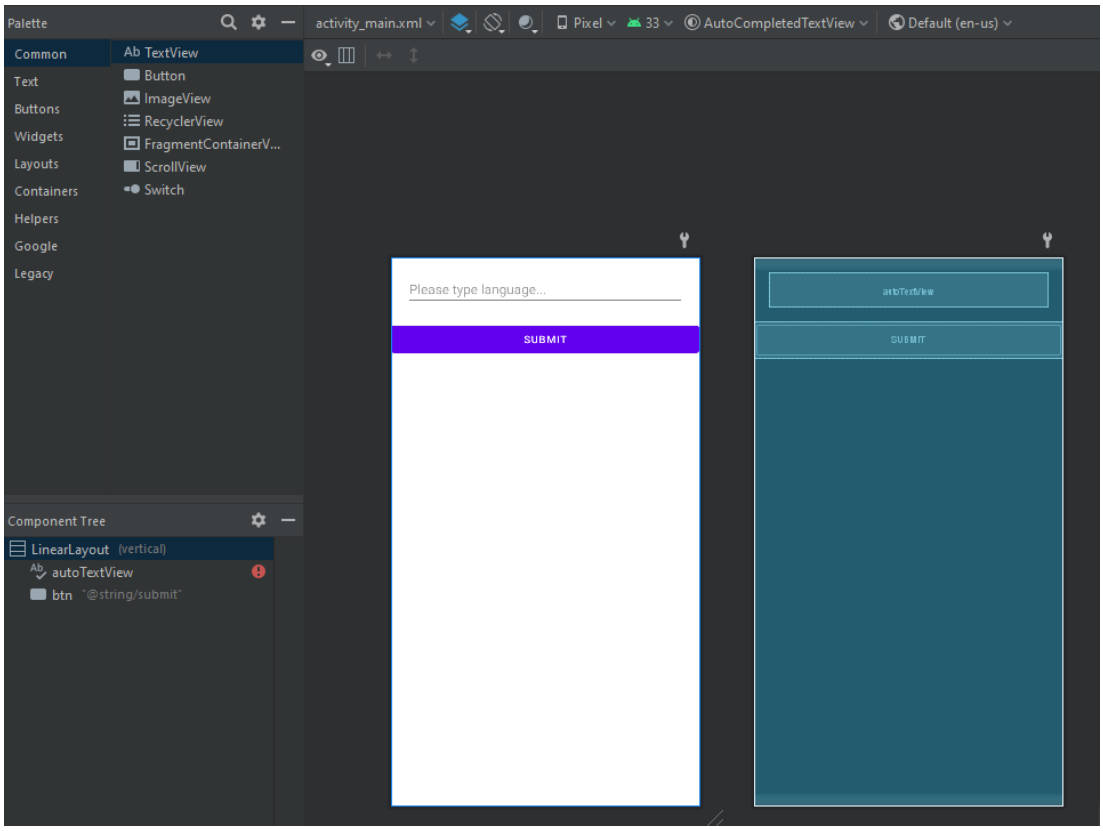
Vamos a realizar un ejemplo donde implementaremos un AutoCompleteTextView para seleccionar un lenguaje de programación.

3. Enunciado

Realiza un app que incluya un AutoCompleteTextView donde puedas elegir entre varios lenguajes de programación. Haz un toast con el texto introducido.

4. Layout

Para implementar un AutoCompleteTextView, lo arrastramos desde el apartado Common→ TextView→ AutoCompletedTextView. Para hacer este ejercico más estruturado hemos transformado la vista en un LinerLayout y hemos añadido un botón para confirmar la selección.



El layout xml quedaría de la siguiente manera:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/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"
    android:orientation="vertical"
    tools:context=".MainActivity" >
    <AutoCompleteTextView
    android:id="@+id/autoTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp"
    android:hint="@string/hint"/>

    <Button
        android:id="@+id/btn"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/submit" />

</LinearLayout>
```

5. Activity

En el activity debemos realizar los siguientes pasos:

1. Importar el widget AutoCompleteTextView, Button y Toast al inicio del código.
2. Definir las constantes autotestView, languages, adapter.
3. Utilizar la instrucción findViewById para definir el valor de cada variable dentro del estado en el que estemos en este ejemple es en el onCreate.
4. Utilizar getStringArray para cargar la lista de idiomas que guardaremos en el strings.xml
5. Genera un toast en el botón con el enteredText que hemos introducido por teclado.

```
package com.carlos.autocompletedtextview

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.ArrayAdapter
import android.widget.AutoCompleteTextView
import android.widget.Button
import android.widget.Toast

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val autotestView = findViewById<AutoCompleteTextView>(R.id.autoTextView)
        // Get the array of languages
        val languages = resources.getStringArray(R.array.Languages)
        // Create adapter and add in AutoCompleteTextView
        val adapter = ArrayAdapter(this,android.R.layout.simple_list_item_1, languages)
        autotestView.setAdapter(adapter)

        val button = findViewById<Button>(R.id.btn)
        if (button != null)
        {
            button ?.setOnClickListener(View.OnClickListener { val enteredText = getString(R.string.submitted_lang) + " "
+ autotestView.getText()
                Toast.makeText(this, enteredText, Toast.LENGTH_SHORT).show()
            })
        }

    }
}
```

6. Recursos (Strings.xml)

Para evitar que el código del activity quede demasiado largo cuando tengamos muchas opciones dentro de nuestro AutoCompleteTextView se puede generar un **Array** con todos los elementos en el Xml de recursos de tipo String dentro de la carpeta **Values**. En dicha carpeta encontraremos también diferentes recursos como colores, fuentes...

```
<resources>
    <string name="app_name">My Application</string>
    <string name="hint">Please type language...</string>
    <string name="submit">Submit</string>
    <string name="submitted_lang">Submitted language:</string>

    <string-array name="Languages">
        <item>Java</item>
        <item>Kotlin</item>
        <item>Swift</item>
        <item>Python</item>
        <item>Scala</item>
        <item>Perl</item>
        <item>Javascript</item>
        <item>Jquery</item>
    </string-array>
</resources>
```

7. Documentación

https://developer.android.com/reference/android/widget/AutoCompleteTextView