Preferencias

Las preferencias son una **forma** conveniente **de almacenar y recuperar la configuración y las opciones del usuario en una aplicación Android**. Aquí hay un listado de posibles usos de preferencias:

1. Configuración de la aplicación:

- Tema de la aplicación: Permitir al usuario elegir entre un tema claro u oscuro.
- Idioma de la aplicación: Permitir al usuario seleccionar el idioma de la interfaz de usuario.
- Tamaño de fuente: Permitir al usuario ajustar el tamaño del texto en la aplicación.
- Notificaciones: Permitir al usuario habilitar o deshabilitar las notificaciones y configurar sus preferencias de notificación.
- Sonidos: Permitir al usuario habilitar o deshabilitar los sonidos de la aplicación y configurar el volumen.
- Vibración: Permitir al usuario habilitar o deshabilitar la vibración para las notificaciones.
- Unidades de medida: Permitir al usuario elegir entre diferentes unidades de medida (por ejemplo, Celsius o Fahrenheit).
- Formato de fecha y hora: Permitir al usuario seleccionar el formato de fecha y hora que se utilizará en la aplicación.

2. Personalización del usuario:

- Nombre de usuario: Almacenar el nombre de usuario del usuario para personalizar la experiencia.
- Avatar: Permitir al usuario seleccionar una imagen de perfil o avatar.
- Preferencias de contenido: Permitir al usuario seleccionar los tipos de contenido que le interesan (por ejemplo, noticias, deportes, tecnología).
- Historial de búsqueda: Almacenar el historial de búsqueda del usuario para facilitar futuras búsquedas.
- Elementos favoritos: Permitir al usuario marcar elementos como favoritos para acceder a ellos fácilmente.

3. Datos de la aplicación:

- Última sesión: Almacenar la fecha y hora de la última sesión del usuario.
- Contador de uso: Registrar el número de veces que se ha utilizado la aplicación.
- Datos de inicio de sesión: Almacenar las credenciales de inicio de sesión del usuario de forma segura.
- Configuración de la cuenta: Almacenar la configuración de la cuenta del usuario, como la dirección de correo electrónico y la contraseña.

4. Otros usos:

- Configuración del juego: Almacenar la configuración del juego, como la dificultad y los controles.
- Configuración de la cámara: Almacenar la configuración de la cámara, como la resolución y el flash.
- Configuración del GPS: Almacenar la configuración del GPS, como la precisión y la frecuencia de actualización.
- Configuración de la red: Almacenar la configuración de la red, como el tipo de conexión y el proxy.

En general, las preferencias son una herramienta versátil que se puede utilizar para almacenar una amplia variedad de información en una aplicación Android. Al utilizar preferencias, puedes crear una experiencia de usuario más personalizada y eficiente.

En primer lugar, vamos a crearnos una nueva aplicación llamada **PreferenciasApp** donde vamos a probar a crearnos una ventana de preferencias.

Tener en cuenta que necesitamos la dependencia *libs.androidx.preference.ktx* en tu build.gradle.

Crearemos un Empty Views Activity, al cual le asociaremos un fragmento embebido dentro de esta activity.

1. **Crea un archivo XML para las preferencias**:

Crea un nuevo archivo XML en la carpeta res/xml de tu proyecto. Por ejemplo, preferences.xml.

Define las preferencias que deseas mostrar en la ventana utilizando elementos como Preference, CheckBoxPreference, EditTextPreference, etc.

Ejemplo:

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android">
    <CheckBoxPreference
        android:key="pref_checkbox"
        android:title="Habilitar notificaciones"
        android:summary="Recibir notificaciones de la aplicación"
        android:defaultValue="true" />
    <EditTextPreference
        android:key="pref_texto"
        android:title="Nombre de usuario"
        android:summary="Introduce tu nombre de usuario"
        android:defaultValue="Usuario" />
</PreferenceScreen>
```

2. Crea fragmento de preferencias:

Para crear una ventana de preferencias en Android, vamos a utilizar el fragmento: **SettingsFragment** y el archivo XML con las preferencias definido en el paso anterior.

El fragmento SettingsFragment extiende de PreferenceFragmentCompat.

En onCreatePreferences(), llama a setPreferencesFromResource() para cargar el archivo XML de preferencias que creaste en el paso anterior.

```
class PreferenciasFragment : PreferenceFragmentCompat() {
    override fun onCreatePreferences(savedInstanceState:
Bundle?, rootKey: String?) {
        setPreferencesFromResource(R.xml.preferences, rootKey)
    }
}
```

3. Añade el fragmento a tu Activity:

En tu Activity, añade el fragmento de preferencias a un contenedor en tu layout. En el archivo layout de la actividad, incluiremos un componente XML llamado <FragmentCointerView>, usando el atributo android:name, de esta manera en la actividad se cargará directamente el fragmento indicado.

Veamos un ejemplo del XML de la actividad:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/mainFragment"
        android:name="com.maestre.preferenciasapp.SettingsFragment"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="1dp"
        android:layout_marginTop="1dp"
        android:layout_marginEnd="1dp"
        android:layout_marginBottom="1dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

4. Recuperar información de la pantalla de preferencias:

Si tenemos una Pantalla de Preferencias, es porque queremos actuar en función de ellas. Lo primero que necesitamos es saber **cómo recuperar información de una pantalla de preferencias.** Puedes utilizar la clase PreferenceManager y el método getDefaultSharedPreferences().

```
val sharedPreferences =
PreferenceManager.getDefaultSharedPreferences(this)
```

Utiliza los métodos getBoolean(), getString(), getInt(), etc., de SharedPreferences para recuperar el valor de una preferencia específica. Debes proporcionar la clave de la preferencia y un valor predeterminado en caso de que la preferencia no esté definida, y actúa en consideración.

```
val notificacionesHabilitadas =
sharedPreferences.getBoolean("pref_checkbox", false)
val nombreUsuario = sharedPreferences.getString("pref_texto",
"Usuario")
```

5. Comprobar si ha cambiado algún valor en la pantalla de preferencias

Para comprobar si en SettingsFragment ha cambiado un valor, haz que implemente la interfaz SharedPreferences.OnSharedPreferenceChangeListener. Esta interfaz te permite registrar un listener que se activará cuando se modifique cualquier preferencia en el SharedPreferences asociado con tu SettingsFragment.

```
class SettingsFragment: PreferenceFragmentCompat(),
SharedPreferences.OnSharedPreferenceChangeListener

{
    //                                                    ...
}
```

En el método onResume() de SettingsFragment, registra el listener utilizando:

```
PreferenceManager.getDefaultSharedPreferences(context).registerO
nSharedPreferenceChangeListener(this).
```

En el método onPause() de SettingsFragment, desregistra el listener utilizando `PreferenceManager.getDefaultSharedPreferences(context).unregiste rOnSharedPreferenceChangeListener(this)`

En el método onSharedPreferenceChanged() debes implementar que acción quieres realizar al recibir notificaciones cuando se cambie una preferencia. Este método recibe la instancia de SharedPreferences y la clave de la preferencia que ha cambiado.

```
override fun onSharedPreferenceChanged(sharedPreferences:
SharedPreferences, key: String) {
    if (key == "pref_checkbox") {
        // La preferencia "pref_checkbox" ha cambiado
        val nuevoValor = sharedPreferences.getBoolean(key,
false)
        // Realizar acciones en consecuencia
    }
}
```

Ejemplo de SettingsFragment completo:

```
package com.maestre.preferenciasapp

import android.content.SharedPreferences
import android.os.Bundle
import android.util.Log
import android.widget.Toast
import androidx.preference.PreferenceFragmentCompat

class SettingsFragment: PreferenceFragmentCompat(),
SharedPreferences.OnSharedPreferenceChangeListener  {

    override fun onCreatePreferences(savedInstanceState: Bundle?, rootKey:
String?) {
        setPreferencesFromResource(R.xml.preferences, rootKey)
    }

    override fun onResume() {
        super.onResume()
preferenceManager.sharedPreferences?.registerOnSharedPreferenceChangeListener
(this)
    }

    override fun onPause() {
        super.onPause()
preferenceManager.sharedPreferences?.unregisterOnSharedPreferenceChangeListen
er(this)
    }

    override fun onSharedPreferenceChanged(sharedPreferences:
SharedPreferences?, key: String?) {
        when (key) {
            "pref_checkbox" -> {
                val isNotificationEnabled =
sharedPreferences?.getBoolean(key, true) ?: true
                //
            }
            "pref_texto" -> {
                val userName = sharedPreferences?.getString(key, "Usuario")
?: "Usuario"
                //

            }
        }
    }

}
```