

## UD 06

**PROGRAMACIÓN MULTIMEDIA Y DISPOSITIVOS MÓVILES 22/23**  
CFGS DAM

### **PRÁCTICA MÉTODOS DE PREFERENCIAS COMPARTIDAS PERSISTENCIA DE DATOS**

Autor: Carlos Espinosa

c.espinosamoreno@edu.gva.es

Fecha: 2022/2023

Licencia Creative Commons

versión 4.0

**Reconocimiento – NoComercial – CompartirIgual (by-nc-sa):** No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

**Índice**

Objetivos de la práctica.....	1
Introducción.....	1
Creación del proyecto.....	1

## 1. Objetivos de la práctica

- Intercambiar valores entre activities
- Conocer la persistencia de datos
- Aprender cómo implementar SharedPreferences en nuestra aplicación de Android usando Kotlin.

## 2. Introducción

SharedPreferences es parte de la API de Android desde el nivel 1 de la API. Es una interfaz que nos permite almacenar/modificar/eliminar datos localmente. Generalmente, se utiliza para almacenar en caché los datos locales del usuario, como los formularios de inicio de sesión. Los datos se almacenan en forma de un par clave-valor. Puede crear varios archivos para almacenar los datos de SharedPreferences .

Veamos algunos métodos importantes para SharedPreferences:

- **getSharedPreferences(String, int)** El método se utiliza para recuperar una instancia de SharedPreferences. Aquí con String semuestra el nombre del archivo SharedPreferences y int se pasa el Contexto.
- El **SharedPreferences.Editor()** se utiliza para editar valores en el archivo SharedPreferences.
- Podemos llamar commit()o apply()para guardar los valores en el archivo SharedPreferences. El commit()guarda los valores inmediatamente, mientras que apply()los guarda de forma asíncrona.

## 3. Creación del proyecto

Vamos a realizar una app que muestre 4 botones para el intercambio de información entre las diferentes casillas de datos.

### Paso 1: Crear un nuevo proyecto

- Haz click en Archivo, luego en Nuevo => Nuevo proyecto.
- Elija Actividad vacía
- Seleccionar idioma como Kotlin
- Seleccione el SDK mínimo.

### Paso 2: Generamos el siguiente Layout

El código para el activity\_main.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <EditText
        android:id="@+id/inUserId"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:hint="User ID"
        android:inputType="number" />

    <EditText
        android:id="@+id/inPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/inUserId"
        android:hint="Password"
        android:inputType="textPassword" />

    <Button
        android:id="@+id/btnSave"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/inPassword"
        android:text="SAVE USER DATA" />

    <Button
        android:id="@+id/btnClear"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_below="@+id/btnSave"
        android:text="CLEAR USER DATA" />

    <Button
        android:id="@+id/btnShow"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_below="@+id/inPassword"
        android:text="SHOW" />

    <Button
```

```
android:id="@+id/btnShowDefault"  
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:layout_alignParentEnd="true"  
android:layout_alignParentRight="true"  
android:layout_below="@+id/btnSave"  
android:text="Show Default" />
```

```
</RelativeLayout>
```

### Paso 3: Generamos el activity

El código para la MainActivity.kt:

```
package...  
import android.content.Context  
import android.content.SharedPreferences  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import android.preference.PreferenceManager  
import android.view.View  
import com.journaldev.androidlysharedpreferences.PreferenceHelper.defaultPreference  
import com.journaldev.androidlysharedpreferences.PreferenceHelper.password  
import com.journaldev.androidlysharedpreferences.PreferenceHelper.userId  
import com.journaldev.androidlysharedpreferences.PreferenceHelper.clearValues  
import com.journaldev.androidlysharedpreferences.PreferenceHelper.customPreference  
  
import kotlinx.android.synthetic.main.activity_main.*  
  
class MainActivity : AppCompatActivity(), View.OnClickListener {  
  
    val CUSTOM_PREF_NAME = "User_data"  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        btnSave.setOnClickListener(this)  
        btnClear.setOnClickListener(this)  
        btnShow.setOnClickListener(this)  
        btnShowDefault.setOnClickListener(this)  
  
    }  
  
    override fun onClick(v: View?) {  
        val prefs = customPreference(this, CUSTOM_PREF_NAME)  
        when (v?.id) {
```

```

        R.id.btnSave -> {
            prefs.password = inPassword.text.toString()
            prefs.userId = inUserId.text.toString().toInt()
        }
        R.id.btnClear -> {
            prefs.clearValues()
        }
        R.id.btnShow -> {
            inUserId.setText(prefs.userId.toString())
            inPassword.setText(prefs.password)
        }
        R.id.btnShowDefault -> {
            val defaultPrefs = defaultPreference(this)
            inUserId.setText(defaultPrefs.userId.toString())
            inPassword.setText(defaultPrefs.password)
        }
    }
}

object PreferenceHelper {

    val USER_ID = "USER_ID"
    val USER_PASSWORD = "PASSWORD"

    fun defaultPreference(context: Context): SharedPreferences =
        PreferenceManager.getDefaultSharedPreferences(context)

    fun customPreference(context: Context, name: String): SharedPreferences =
        context.getSharedPreferences(name, Context.MODE_PRIVATE)

    inline fun SharedPreferences.editMe(operation: (SharedPreferences.Editor) -> Unit) {
        val editMe = edit()
        operation(editMe)
        editMe.apply()
    }

    var SharedPreferences.userId
        get() = getInt(USER_ID, 0)
        set(value) {
            editMe {
                it.putInt(USER_ID, value)
            }
        }
}

```

```

var SharedPreferences.password
    get() = getString(USER_PASSWORD, "")
    set(value) {
        editMe {
            it.putString(USER_PASSWORD, value)
        }
    }

var SharedPreferences.clearValues
    get() = { }
    set(value) {
        editMe {
            it.clear()
        }
    }
}

```

Gracias a Kotlin Android Extensions, no tenemos que usar `findViewById` para cada vista XML. En el código anterior, estamos creando una **clase singleton** usando un **object**. Estamos declarando una función de orden superior en línea `editMe()`, que contiene la llamada lógica para la operación de edición.

Hemos creado propiedades separadas para cada uno de los valores. Usamos las propiedades `get` y `set` de Kotlin para recuperar y configurar los datos en las preferencias compartidas. Kotlin ha reducido la cantidad de código y se ve mucho más limpio. Además, podemos hacerlo más conciso usando otra función de orden superior de Kotlin que se muestra a continuación.

```

fun SharedPreferences.Editor.put(pair: Pair<String, Any>)
    val key = pair.first
    val value = pair.second
    when(value) {
        is String -> putString(key, value)
        is Int -> putInt(key, value)
        is Boolean -> putBoolean(key, value)
        is Long -> putLong(key, value)
        is Float -> putFloat(key, value)
        else -> error("Only primitive types can be stored in SharedPreferences")
    }
}

```

Y hacemos lo siguiente mientras configuramos los valores:

```

var SharedPreferences.password

    get() = getString(USER_PASSWORD, "")
    set(value) {
        editMe {
            it.put(USER_PASSWORD to value)
        }
    }
}

```

El resultado de la app anterior tiene que ser el siguiente:

