**SHARED PREFERENCES**

Sin duda el manejo de información es una de las principales características de cualquier aplicación, ya que son una mina de oro para diversas empresas que se dedican a proveer productos y servicios, si bien la información por si sola realmente no dice mucho, pero cuando se le selecciona y discrimina de forma conveniente, se puede convertir en la mejor aliada de las empresas. Es por eso que las compañías gastan millones en la adquisición de información, ya sea por medios propios o comprándola a terceros. Las aplicación móviles, en la actualidad recaban millones de datos de diversos usuarios, estos datos pueden ser almacenados de forma local, en servidores particulares y/o en la nube. Los datos pueden ser almacenados en archivos y/o en bases de datos.

Así como en las aplicaciones web se puede almacenar información fuera de línea sin la necesidad de una base de datos, con el propósito de recuperar información al momento de abrirlas nuevamente, las aplicaciones móviles también pueden hacerlo de forma similar utilizando SharedPreferences.

//RECUPERAR INFO SIN NECESIDAD DE BASE DE DATOS

1. Leer y escribir datos con el framework proveído por la interfaz ISharedPreferences. Los datos que se pueden almacenar y recuperar aun cuando la aplicación haya sido terminada son los tipos de datos primitivos bool, float, int, long, y string, estos se almacenan en pares del tipo llave-valor.
2. Crear un nuevo proyecto del tipo Aplicación de Android (Xamarin)
3. Llamarlo UsingSharedPreferences
4. Aplicación vacía
5. Minimun Android Version (4.4)
6. Abrir el archivo activity\_main.axml y sustituir el código que contiene, por este otro:

<?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">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Usuario: "/>

<EditText

android:layout\_width="300px"

android:layout\_height="75px"

android:id="@+id/etUsuario"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Correo: "/>

<EditText

android:layout\_width="300px"

android:layout\_height="75px"

android:id="@+id/etCorreo"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Contraseña: "/>

<EditText

android:layout\_width="300px"

android:layout\_height="75px"/>

</LinearLayout>

</LinearLayout>

1. Crear un objeto de tipo ISharedPreferences e instanciarlo por medio del método GetSharedPreferences(), quien recibe dos parámetros, el primero es el nombre de las preferencias (nombre del archivo donde se guardan las preferencias) y el segundo es el tipo del archivo, que en este caso es privado, para proteger la información. Para ello agregar el siguiente código dentro del método OnCreate() de MainActivity, después del método SetContentView():

...

Android.Content.ISharedPreferences preferences = GetSharedPreferences("PreferenciasUsuario", Android.Content.FileCreationMode.Private);

...

1. Una vez que tenemos la referencia a las preferencias, podemos leer valores en la forma llave-valor y utilizarlos a nuestro antojo, siempre y cuando el par llave-valor exista (Es recomendable leer valores de preferencias cuando se crea la aplicación en el método OnCreate() y escribirlos cuando la aplicación es detenida). Para ello agregamos después del código del paso 6 lo siguiente:

...

string preferenciaUsuario = preferences.GetString("Usuario", "Tu nombre de usuario");

string preferenciaCorreo = preferences.GetString("correo", "Tu correo electónico");

EditText usuario = FindViewById<EditText>(Resource.Id.etUsuario);

usuario.Text = preferenciaUsuario;

EditText correo = FindViewById<EditText>(Resource.Id.etCorreo);

correo.Text = preferenciaCorreo;

...

La primer vez que corramos la aplicación, no se encontraran las preferencias Usuario ni Correo, por lo que se visualizaran en las vistas, “Tu nombre de usuario” y “Tu correo electrónico” respectivamente. La próxima vez que se ejecute la aplicación, se mostraran los últimos valores introducidos por el usuario.

1. Para escribir pares de valores en la forma llave-valor (es recomendable hacerlo cuando se detiene la aplicación), requerimos primeramente obtener una referencia de las preferencias, tal y como lo hicimos en el paso 5, lo siguiente es crear un objeto de tipo ISharedPreferencesEditor por medio del método Edit() del objeto preferencias, luego escribir las preferencias y finalmente guardar los cambios. Todo lo anterior lo hacemos sobrescribiendo el método OnStop() y para ello necesitamos agregar el siguiente código a MainActivity:

...

protected override void OnStop()

{

base.OnStop();

Android.Content.ISharedPreferences preferences = GetSharedPreferences("PreferenciasUsuario", Android.Content.FileCreationMode.Private);

Android.Content.ISharedPreferencesEditor editor = preferences.Edit();

EditText usuario = FindViewById<EditText>(Resource.Id.etUsuario);

string preferenciaUsuario = usuario.Text;

EditText correo = FindViewById<EditText>(Resource.Id.etCorreo);

string preferenciaCorreo = correo.Text;

editor.PutString("Usuario", preferenciaUsuario);

editor.PutString("Correo", preferenciaCorreo);

editor.Commit();

}

...

1. Ya vimos como guardar y restaurar preferencias en una aplicación, el ejemplo desarrollado muestra como guardar y recuperar datos básicos para loguearse a un servicio ofrecido por una aplicación, ahora bien, hay ocasiones en las que se desea cambiar los ajustes de alguna aplicación y que estos se preserven aun después de cerrarla, si queremos que esto suceda, podemos usar PreferenceFragment o PreferenceFragmentCompat, ya que permiten crear interfaces de usuario que contienen vistas, las cuales automáticamente almacenan y restauran las preferencias compartidas de usuario. Siguiendo con el proyecto actual, lo completaremos, de tal forma que implemente un fragment para ajustes que hererde de PreferenceFragmentCompat para que se preserven los ajustes del usuario.
2. Crear un archivo xml llamado preferencias.xml, el cual contendrá dos preferencias (se le pueden agregar las que uno quiera, los nombres de las vistas sonigual a los convencionales, únicamente se les agrega al final la palabra Preference.), este archivo contiene la representación de la pantalla que mostrara las vistas de las preferencias de usuario de la aplicación. El archivo debe estar contenido en una carpeta llamada xml que a su vez es una subcarpeta de Resources.
3. Agregar el siguiente código al archivo preferencias.xml:

...

<android.support.v7.preference.PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android">

<android.support.v7.preference.EditTextPreference

android:defaultValue="#fff111"

android:dialogTitle="Background color"

android:key="BackGroundColor"

android:summary="Tu background color."

android:title="Background color"/>

<android.support.v7.preference.EditTextPreference

android:defaultValue="desconocido"

android:dialogTitle="Nombre de usuario"

android:key="KeyNombreUsuario"

android:summary="Tu nombre de usuario"

android:title="Nombre de usuario"/>

</android.support.v7.preference.PreferenceScreen>

...

1. Agregar una nueva clase al proyecto, llamada AjustesFragment, que herede de Android.Support.V7.Preferences.PreferenceFragmentCompat .
2. Instalar desde el administrador de paquetes de NuGet la librería Xamarin.Android.Support.v7.Preference.
3. Dentro de la clase AjustesFragment, agregar el siguiente método:

...

public override void OnCreatePreferences(Bundle bundle, string root)

{

AddPreferencesFromResource(Resource.Xml.preferencias);

}

...

1. En el método OnCreate() de MainActivity, después de correo.Text…, agregamos el siguiente código para obtener las preferencias de la aplicación:

...

var preferencias = Android.Support.V7.Preferences.PreferenceManager.GetDefaultSharedPreferences(this);

string backGroundColor = preferencias.GetString("BackGroundColor", "#fff111");

string keyNombreUsuario = preferencias.GetString("KeyNombreUsuario", "desconocido");

...

1. Pero aún nos falta una vista, que es la que contiene al fragment AjustesFragment, para ello abrimos el archivo activity\_main.axml y le agregamos el siguiente código después del último LinearLayout:

...

<fragment

android:name="usingsharedpreferences.AjustesFragment"

android:id="@+id/ajustesFragment"

android:layout\_weight="1"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

...

1. Guardar todo, compilar e implementar en emulador o dispositivo.

**Nota:** Si bien el uso de PreferenceFragment o PreferenceFragmentCompat es para preservar preferencias compartidas de usuario, en realidad se puede utilizar para preservar cualquier preferencia con un valor de tipo primitivo. El ejemplo aquí mostrado sobre PreferenceFragmentCompat es solo para efectos demostrativos, lo recomendable si se quieren preservar preferencias exclusivas de la actividad, es utilizar GetPreferences().

**Actividad complementaria**

Nota: Esta actividad es la décima que impacta en la calificación para el criterio de evaluación “Prácticas varias”.

Con el método GetSharedPreferences() obtenemos los valores de las preferencias compartidas por diversas actividades y/o aplicaciones, y con el método GetPreferences() obtenemos las preferencias que son exclusivas de la actividad.

1. Complementar esta práctica creando otra aplicación móvil que comparta las preferencias de la actividad aquí desarrollada y que también utilice 3 preferencias exclusivas.
2. Una de las preferencias exclusivas debe ser KeyBackGround, el valor que debe guardar es el color del background del LinearLayout principal, cuyo valor al ser recuperado servirá para que el background del LinearLayout principal tome ese color.