27. Preferencias

Android dispone de un mecanismo para almacenar pequeñas cantidades de datos en forma de pares clave/valor: la clave ha de ser de tipo `String` y el valor de uno de los siguientes tipos: `Boolean`, `Integer`, `Long`, `Float`, `String`, o `set<String>` (un conjunto de valores de tipo `String` desde el API 11). El almacenamiento es persistente, es decir, los datos no se pierden aunque se detenga la app o se apague el dispositivo.

27.1 Lectura y escritura de preferencias

Los datos pueden ser privados de la actividad o pueden compartirse entre todas las actividades de una aplicación. En este último caso las preferencias se almacenan por defecto en un archivo, que agrega el sufijo `_preferences.xml` al nombre del paquete. Para el proyecto `CCC27` de esta unidad, el archivo de preferencias es:

```
es.uam.eps.android.CCC27_preferences.xml
```

En cuanto se modifique el valor de una de las preferencias, el archivo se añade a la siguiente carpeta:

```
/data/data/es.uam.eps.android.CCC27/shared_prefs
```

Veamos cómo recuperar el valor de una de estas preferencias sabiendo que su clave es `music`, por ejemplo. La forma más sencilla es utilizar el siguiente método desde cualquier actividad:

```
public Boolean music(){
    Boolean play = false;

    SharedPreferences sharedPreferences =
        PreferenceManager.getDefaultSharedPreferences(this);
    if (sharedPreferences.contains(music))
        play = sharedPreferences.getBoolean(music, false);

    return play;
}
```

Primero conseguimos una instancia de la clase `SharedPreferences` especificando el contexto (`this`). A continuación, siempre que exista la preferencia en el archivo, recuperamos su valor booleano mediante el método `getBoolean()`, al que pasamos la clave que identifica la preferencia como primer argumento, y su valor por omisión como segundo argumento. Más adelante situaremos todas las claves en una única clase para mejorar la organización.

Por otro lado, para modificar el valor de una preferencia compartida utilizaremos un método como el siguiente:

```
public void setMusic (Boolean value) {
     SharedPreferences preferences =
       PreferenceManager.getDefaultSharedPreferences(this);
     SharedPreferences.Editor editor = preferences.edit();
     editor.putBoolean(music, value);
     editor.commit();
}
```

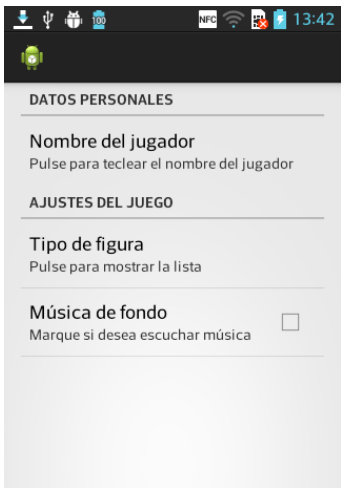
Primero conseguimos una instancia de la clase `SharedPreferences` especificando el contexto (`this`). A continuación creamos un objeto de tipo `Editor` mediante el método `edit()`. Utilizamos el método `putBoolean()` para cambiar el valor de preferencias de tipo `Boolean`. Finalmente, para guardar los cambios en el fichero de preferencias, utilizamos el método `commit()`.

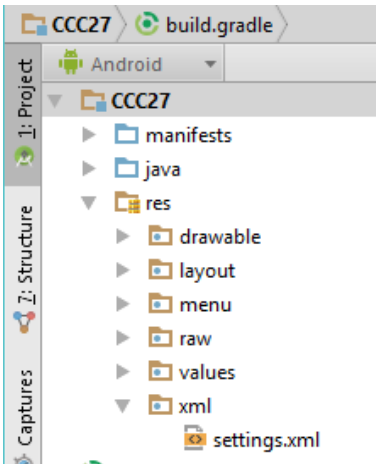
27.2 El menú de preferencias

Veamos ahora cómo solicitar en nuestro proyecto, `CCC27`, el valor de ciertas preferencias (nombre del jugador, tipo de figura inicial y música) mediante una interfaz de usuario especificada en un fichero XML que colocaremos en una subcarpeta de nombre `xml` dentro de la carpeta `res` (estudia la interfaz gráfica a la que da lugar el fichero):

`/res/xml/settings.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android">
    <PreferenceCategory android:title="Datos personales">
        <EditTextPreference
            android:key="playerName"
            android:title="Nombre del jugadro"
            android:summary="Pulse para teclear el nombre del jugador" />
    </PreferenceCategory>
    <PreferenceCategory android:title="Ajustes del juego">
        <ListPreference
            android:key="figure"
            android:title="Tipo de figura"
            android:summary="Pulse para mostrar la lista"
            android:entries="@array/figures"
            android:entryValues="@array/figureCodes"
            android:dialogTitle="Elija la figura inicial"
            android:defaultValue="c00111000011100111111111101111111111001110000011100"/>
        <CheckBoxPreference
            android:key="music"
            android:title="Música de fondo"
            android:summary="Marque si desea escuchar música"
            android:defaultValue="false"/>
    </PreferenceCategory>
</PreferenceScreen>
```





El elemento `PreferenceScreen` del fichero `settings.xml` corresponde al menú de preferencias dentro del cual se especifican otros elementos como `ListPreference`, `CheckBoxPreference` y `EditTextPreference`. Todos estos elementos se corresponden con vistas normales de Android a las que hemos añadido el sufijo `Preference`.

El atributo `android:key` es la clave de la preferencia, que utilizaremos en el código Java para recuperar su valor. El significado de `android:title` y `android:summary` queda claro después de observar la interfaz gráfica de la página anterior.

Dentro del elemento `ListPreference`, los atributos `android:entries` y `android:entry_values` especifican recursos correspondientes a las cadenas y códigos, respectivamente, de las distintas figuras iniciales. Estos recursos se especifican en el fichero `arrays.xml`:

`/res/values/arrays.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="figures">
        <item>Completo</item>
        <item>Circulo</item>
        <item>Anillo</item>
    </string-array>
    <string-array name="figureCodes">
        <item>c00111000011100111111111101111111110011100011100</item>
        <item>c00000000011100011111001101100111110001110000000</item>
        <item>c00000000011100011011001000100110110001110000000</item>
    </string-array>
</resources>
```

Estos valores se necesitan para construir la lista que aparece cuando se pulsa la preferencia titulada “Tipo de figura”. La cabecera del diálogo que surge contiene la cadena especificada en el atributo `android:dialogTitle` del elemento de tipo `ListPreference`:

En el código del proyecto, las cadenas del fichero `settings.xml`, como "Tipo de figura" se sustituyen por recursos (`@string/listPrefTitle`) que facilitarán más adelante la internacionalización:

`/res/values/strings.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">CCC27</string>
    <string name="action_settings">Ajustes</string>
    <string name="gameOverTitle">Fin del juego</string>
    <string name="gameOverMessage">¿Quiere jugar otra partida?</string>
    <string name="initialImage">Imagen inicial del juego</string>
    <string name="aboutText">Acerca de</string>
    <string name="sendMessageText">Enviar mensaje</string>
    <string name="aboutMessage">Cha cha cha es un juego solitario
        en el que hemos de comer una a una las fichas
        saltando por encima de ellas, solo en vertical u
        horizontal, hacia la casilla adyacente que se
        encuentre vacía. El objetivo es dejar
        la última ficha en el centro del tablero, el
        lugar que comenzó libre.</string>
    <string name="firstCategoryPrefTitle">Datos personales</string>
    <string name="editTextPrefTitle">Nombre del jugador</string>
    <string name="editTextPrefSummary">Pulse para teclear el nombre del jugador</string>
    <string name="secondCategoryPrefTitle">Ajustes del juego</string>
    <string name="listPrefTitle">Tipo de figura</string>
    <string name="listPrefSummary">Pulse para mostrar la lista</string>
    <string name="checBoxPrefTitle">Música de fondo</string>
    <string name="checBoxPrefSummary">Marque si desea escuchar música</string>
    <string name="listPrefDialogTitle">Elija la figura inicial</string>
    <string name="preferencesText">Ajustes</string>
</resources>
```

Finalmente, el atributo `android:defaultValue` indica cuál es el valor por omision. Sin embargo no es suficiente con indicar este valor en el fichero settings.xml. También debemos incluir la siguiente llamada en el método onCreate de la primera actividad que se arranca en nuestro proyecto, en esta caso en Initial.java:

```
public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main_void);

        //Cargamos las preferencias por omisión
        PreferenceManager.setDefaultValues(this, R.xml.settings, false);
}
```

La llamada a `setDefaultValues` carga los valores por defecto de las preferencias. El parámetro false indica que esta carga solo se debe hacer la primera vez que se llama a la función.

27.3 La clase PreferenceFragment

Una vez especificada la interfaz en el fichero `settings.xml`, Android recomienda extender la clase `PreferenceFragment` para gestionar las preferencias. El fichero java del fragmento es muy sencillo pues simplemente invoca al método `addPreferencesFromResource()` desde su callback `onCreate()`:

`/src/CCCPreferenceFragment.java`

```
package es.uam.eps.android.CCC27;

import android.os.Bundle;
import android.preference.PreferenceFragment;

public class CCCPreferenceFragment extends PreferenceFragment{

    public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.settings);
    }
}
```

Vamos a utilizar una transacción para construir y mostrar el fragmento dentro del menú de opciones de la actividad `MainActivity`. Primero actualicemos la interfaz del menú de opciones para incluir a las preferencias:

`/res/menu/ccc_menu.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<menu
    xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:title="@string/aboutText"
        android:id="@+id/menuAbout"
        android:icon="@android:drawable/ic_menu_info_details" />
    <item
        android:title="@string/sendMessageText"
        android:id="@+id/sendMessage"
        android:icon="@android:drawable/ic_dialog_email" />
    <item
        android:title="@string/preferencesText"
        android:id="@+id/preferences"
        android:icon="@android:drawable/ic_menu_preferences" />
</menu>
```

La transacción se ejecuta en el método `onCreate()` de una nueva actividad llamada `CCCPreference` que, además, contiene las constantes con las claves de cada una de las preferencias:

`/src/CCCPreference.java`

```
package es.uam.eps.android.CCC27;

import android.app.Activity;
import android.app.FragmentManager;
import android.app.FragmentTransaction;
import android.os.Bundle;

public class CCCPreference extends Activity {
    public final static String  PLAY_MUSIC_KEY        = "music";
    public final static boolean PLAY_MUSIC_DEFAULT    = true;
    public final static String  PLAYER_KEY            = "playerName";
    public final static String  PLAYER_DEFAULT        = "unspecified";
    public final static String  FIGURE_KEY            = "figure";
    public final static String  FIGURE_DEFAULT        =
                                  "c0011100001110011111111110111111111100111000011100";

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.main_void);

        FragmentManager fragmentManager = getFragmentManager();
        FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
        CCCPreferenceFragment fragment = new CCCPreferenceFragment();
        fragmentTransaction.replace(android.R.id.content, fragment);
        fragmentTransaction.commit();
    }
}
```

Esta nueva actividad se arranca en el caso correspondiente del menú de opciones de la clase `MainActivity`:

`/src/MainActivity.java`

```
...
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.menuAbout:
             startActivity(new Intent(this, About.class));
             return true;
        case R.id.sendMessage:
             Intent intent = new Intent(android.content.Intent.ACTION_SEND);
             intent.setType("text/plain");
             intent.putExtra(Intent.EXTRA_SUBJECT, "CHA CHA CHA");
             intent.putExtra(Intent.EXTRA_TEXT,
               "Hola ..., he llegado a ... puntos en cha cha cha ...");
             startActivity(intent);
             return true;
        case R.id.preferences:
             startActivity(new Intent(this, CCCPreference.class));
             return true;
    }
    return super.onOptionsItemSelected(item);
} ...
```

Ahora solo nos queda utilizar la información recogida por el menú de preferencias para alterar el curso del juego. En este proyecto, para no alargarnos demasiado, vamos a utilizar tan solo la preferencia ligada a la música. Haremos que la música se active solo si así lo solicita el jugador. Esto queda reflejado en el método `onResume()` de `MainActivity`:

`/src/MainActivity.java`

```
...
protected void onResume(){
    super.onResume();
    Boolean play = false;

    SharedPreferences sharedPreferences =
        PreferenceManager.getDefaultSharedPreferences(this);
    if (sharedPreferences.contains(CCCPreference.PLAY_MUSIC_KEY))
        play = sharedPreferences.getBoolean(CCCPreference.PLAY_MUSIC_KEY,
                                            CCCPreference.PLAY_MUSIC_DEFAULT);
    if (play == true)
        Music.play(this, R.raw.funkandblues);
} ...
```

Este es el resultado final: