**PRACTICA 13: ASTEROIDES**

**Parte VIII: Organizando preferencias**

**Paso 1.** Crea una nueva lista de preferencias <PreferenceScreen> dentro de la lista de preferencias del fichero res/xml/preferencias.xml.

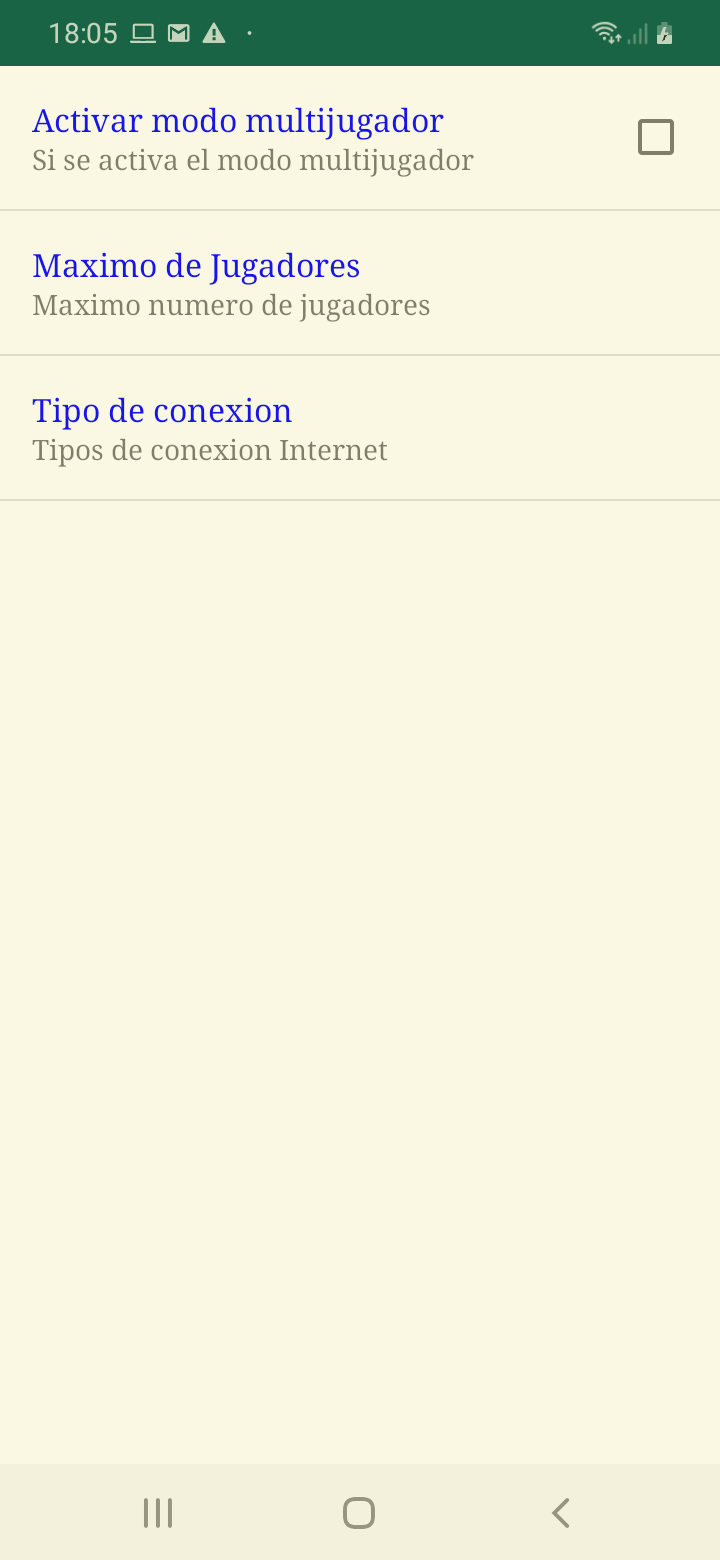
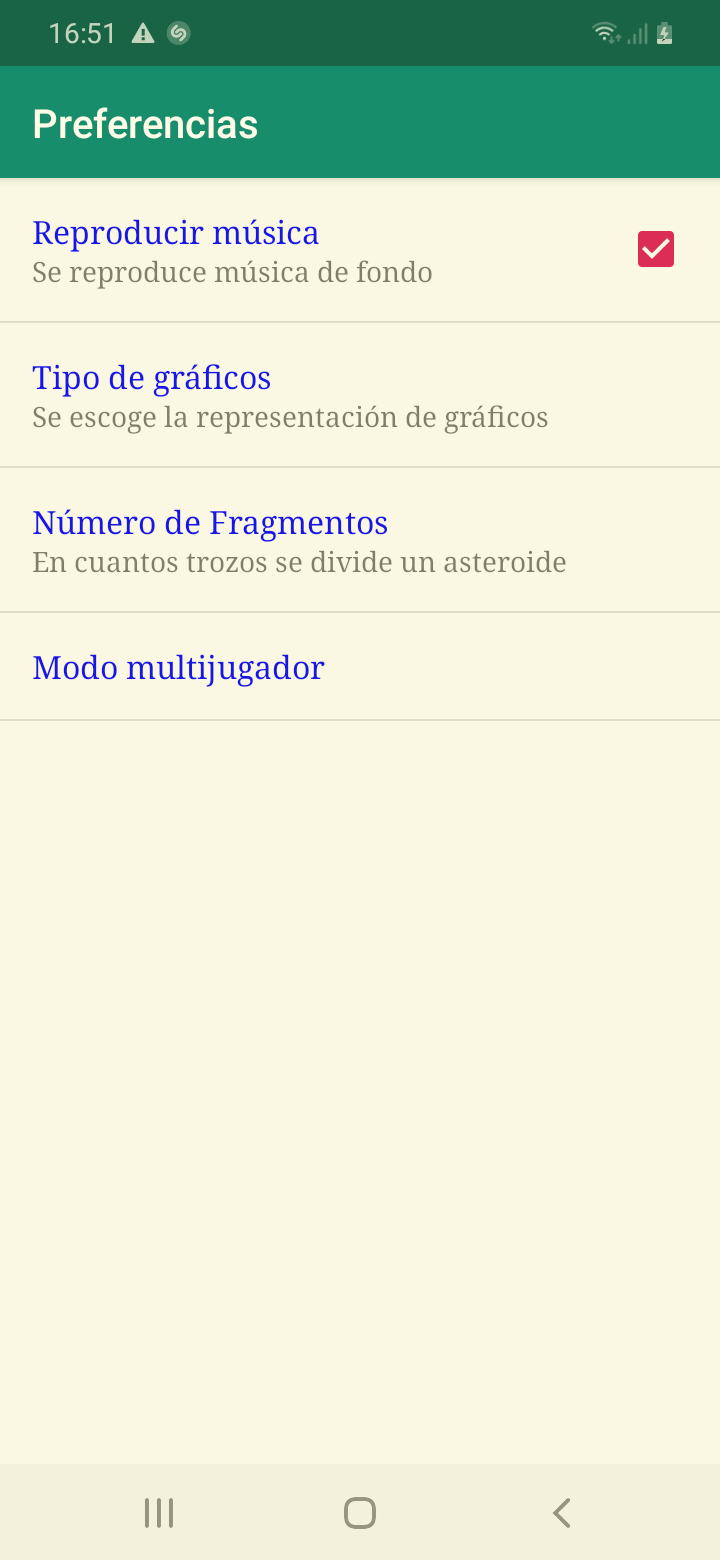


**Paso 2.** Asígnale al parámetro android:title el valor “Modo multijugador”.

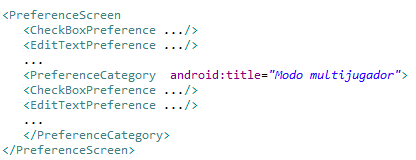
**Paso 3.** Crea tres elementos dentro de esta lista: Activar multijugador, Máximo de jugadores y Tipo de conexión. Para este último han de poder escogerse los valores: Bluetooth, Wi-Fi e Internet.

# 

**Paso 4.** Visualiza el resultado.

****

**Paso 5.** Otra opción para organizar las preferencias consiste en agruparlas por categorías. Con esta opción se visualizarán en la misma pantalla, pero separadas por grupos. Has de seguir el siguiente esquema:



A continuación se representa la forma en la que Android muestra las categorías:

****

**Parte IX: Organizando preferencias**

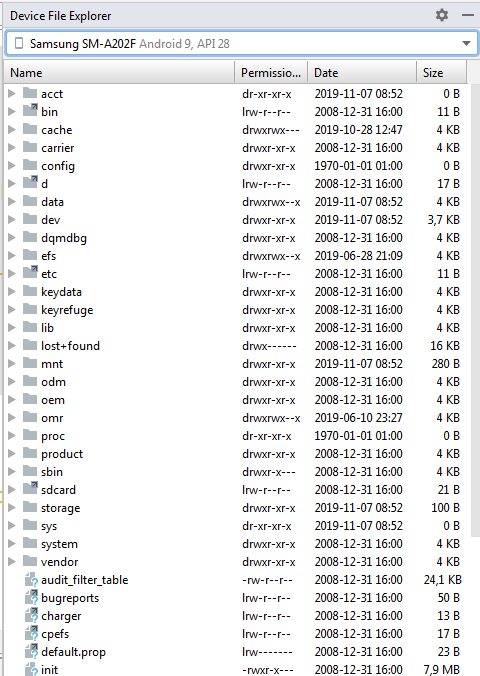
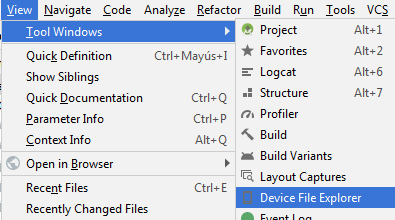
**Paso 1.** Modifica la práctica anterior para que en lugar de mostrar las propiedades en dos pantallas, las muestre en una sola, tal y como se muestra en la imagen anterior.



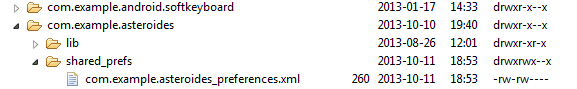
**Parte X: Donde se almacenan las preferencias de usuario**

Veamos donde se han almacenado las preferencias que acabamos de crear:

**Paso 1.** Para navegar por el sistema de ficheros de un dispositivo con **Android Studio**, abre la opción View *> Tools Windows > Device File Explorer.*

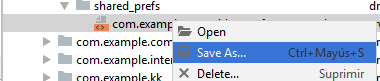


**Paso 2.** Busca el siguiente fichero: /data/data/com.examples.asteroide/shared\_prefs/ org.examples.asteroide\_preferences.xml (la aplicación tiene que estar siendo ejecutada)

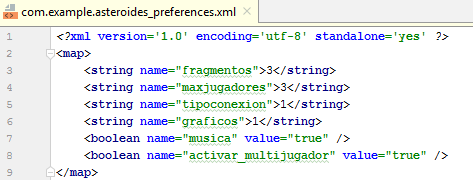


**NOTA:** *En un dispositivo real no tendrás permiso para acceder a estos ficheros (A menos que el dispositivo haya sido rooteado).*

**Paso 3.** Descarga el fichero haciendo click botón derecho “Save As”..



**Paso 4.** Visualiza su contenido. Tiene que ser similar a:



**Parte XI: Accediendo a los valores de las preferencias**

**Paso 1.** Copia la función anterior en la clase Asteroides. Añade el parámetro que se muestra a continuación mostrarPreferencias(View view).

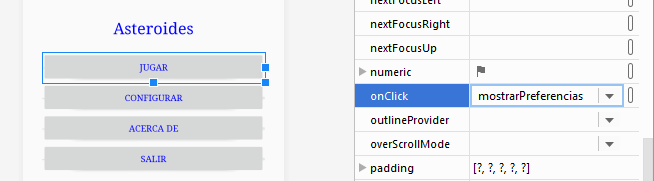
Texto

Descripción generada automáticamente

Texto, Carta

Descripción generada automáticamente

**Paso 2.** Asígnala al atributo onClick del botón Jugar el método anterior.



**Paso 3.** Visualiza también el resto de las preferencias que hayas introducido.

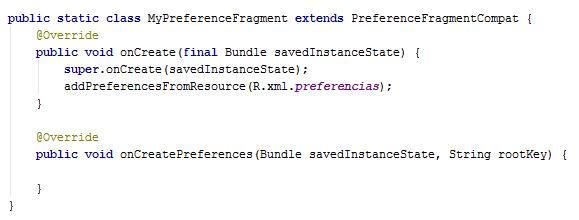


**Parte XII: Verificar valores correctos de una preferencia**

**Paso 1.** Copia el siguiente código al final del método onCreate():

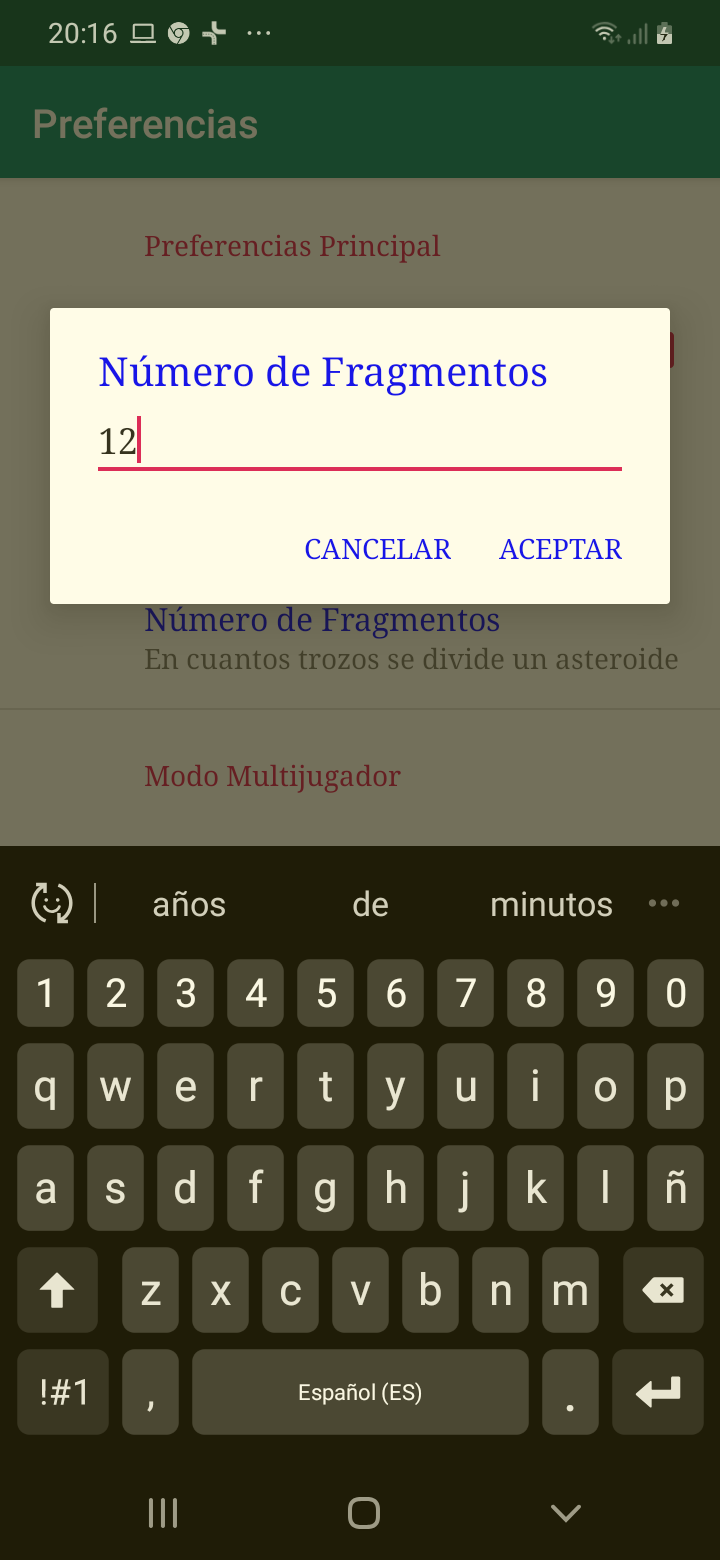
Texto

Descripción generada automáticamente



El código comienza obteniendo una referencia de la preferencia fragmentos, para asignarle un escuchador que será llamado cuando cambie su valor. El escuchador comienza convirtiendo el valor introducido a entero. En caso de producirse un error es porque el usuario no ha introducido un valor adecuado. En este caso, mostramos un mensaje y devolvemos false para que el valor de la preferencia no sea modificado. Si no hay error, tras verificar el rango de valores aceptables, modificamos la explicación de la preferencia para que aparezca el nuevo valor entre paréntesis y devolvemos true para aceptar este valor. Si no está en el rango, mostramos un mensaje indicando el problema y devolvemos false.

**Paso 2.** Ejecuta el proyecto y verifica que funciona correctamente.

**Paso 3.** **Mostrar el valor de una preferencia**

En el ejercicio anterior cuando se modifica el número de fragmentos se muestra entre paréntesis el nuevo valor introducido. El funcionamiento no es del todo correcto, cuando entramos por primera vez o cuando se cambia el teléfono de vertical a horizontal este valor no se muestra. Añade el código necesario para que este valor aparezca siempre.

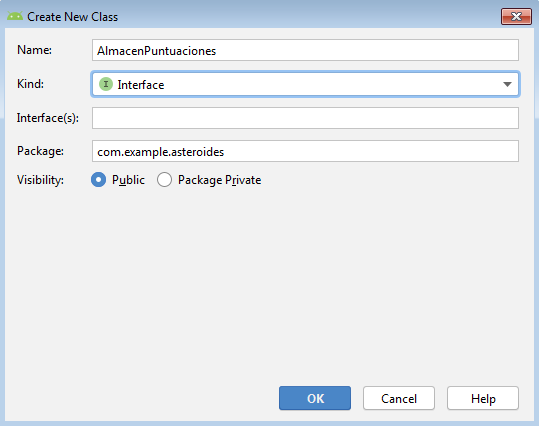
**Parte XIII: El interfaz AlmacenPuntuaciones**

**Paso 1.** Abre la aplicación Asteroides.

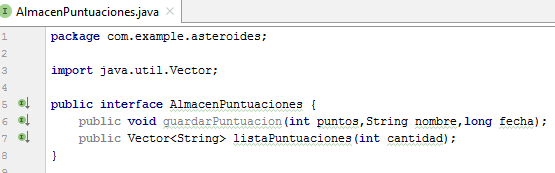
**Paso 2.** Pulsa con el botón derecho sobre la carpeta de código (com.example.asteroides) y selecciona New > Java Class.



**Paso 3.** En el campo *Name:* introduce AlmacenPuntuaciones, en el campo Kind introduce Interface y pulsa *Ok*.



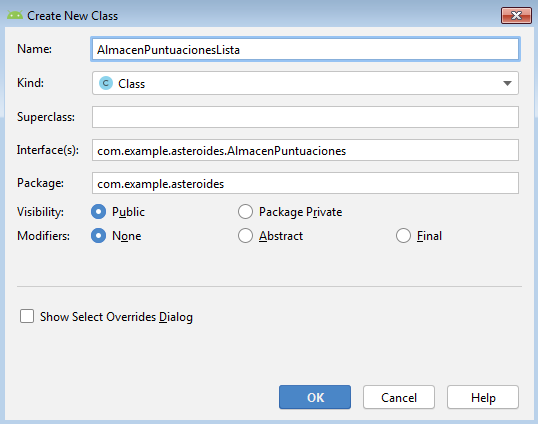
**Paso 4.** Introduce el código que se muestra a continuación:

****

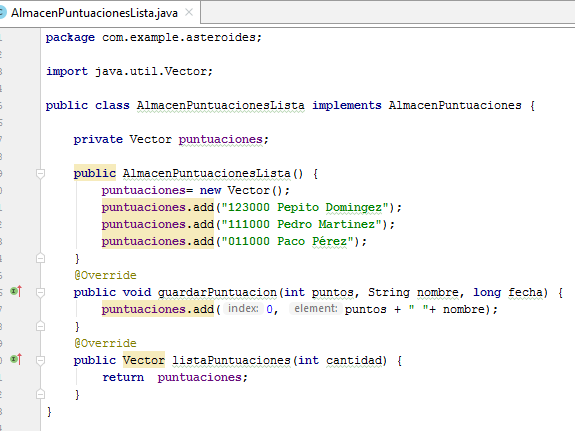
**Nota sobre Java**: La interfaz es una clase abstracta pura, es decir una clase donde se indican los métodos pero no se implementa ninguno (en este caso se dice que los métodos son abstractos). Permite al programador de la clase establecer la estructura de esta (nombres de métodos, sus parámetros y tipos que retorna, pero no el código de cada método). Una interfaz también puede contener constantes, es decir campos de tipo static y final.

Las diferentes clases que definamos para almacenar puntuaciones han de implementar esta interfaz. Como ves tiene dos métodos. El primero para guardar la puntuación de una partida, con los parámetros puntuación obtenida, nombre del jugador y fecha de la partida. La segunda es para obtener una lista de puntuaciones previamente almacenadas. El parámetro cantidad indica el número máximo de puntuaciones que ha de devolver.

**Paso 5.** Veamos a continuación una clase que utiliza esta interfaz. Para ello crea en el proyecto la clase AlmacenPuntuacionesLista.

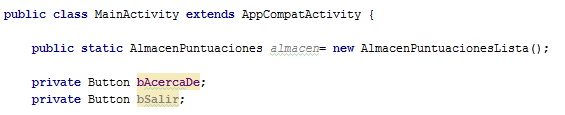
****

**Paso 6.** Introduce el siguiente código:



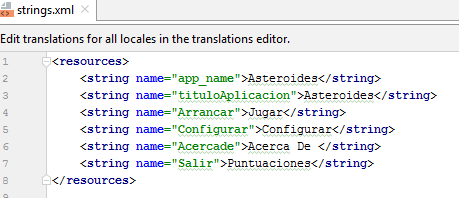
Esta clase almacena la lista de puntuaciones en un vector de String. Tiene el inconveniente de que al tratarse de una variable local, cada vez que se cierre la aplicación se perderán las puntuaciones. El constructor inicializa el array e introduce tres valores. La idea es que aunque todavía no esté programado el juego y no podamos jugar, tengamos ya algunas puntuaciones para poder representar una lista. El método guardarPuntuacion() se limita a insertar en la primera posición del array un String con los puntos y el nombre. La fecha no es almacenada. El método listaPuntuaciones() devuelve el vector de String entero, sin tener en cuenta el parámetro cantidad que debería limitar el número de Strings devueltos.

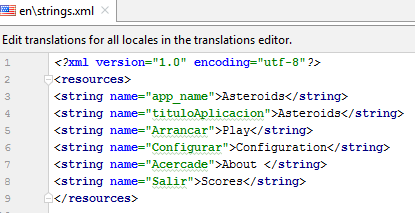
**Paso 7.** En la actividad MainActivity tendrás que declarar una variable para almacenar las puntuaciones:



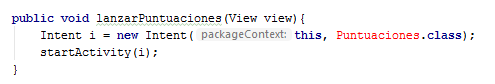
**Nota sobre Java:** El modificador **static** permite compartir el valor de una variable entre todos los objetos de la clase. Es decir, aunque se creen varios objetos, solo existirá una única variable almacen compartida por todos los objetos. El modificador **public** permite acceder a la variable desde fuera de la clase. Por lo tanto, no será necesario crear métodos getters y setters. Para acceder a esta variable no tendremos más que escribir el nombre de la clase seguida de un punto y el nombre de la variable. Es decir MainActivity.almacen.

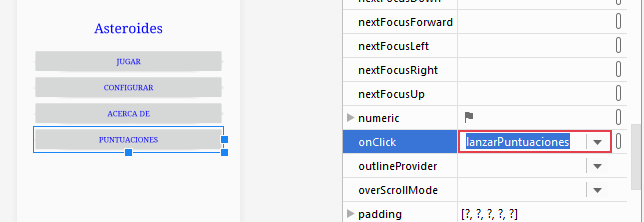
**Paso 8.** Para que los jugadores puedan ver las últimas puntuaciones obtenidas, modifica el cuarto botón del layout activity\_main.xml para que en lugar del texto “Salir” se visualice “Puntuaciones”. Para ello modifica los ficheros res/values/strings. También sería interesante que cambiaras el fichero res/values-en/strings.





**Paso 9.** Modifica el escuchador asociado al cuarto botón para que llame al método lanzarPuntuaciones:





**Paso 10.** De momento no te permitirá ejecutar la aplicación. Hasta que en el siguiente apartado no creemos la actividad Puntuaciones no será posible.