22. Intenciones

Una aplicación puede contener cero o más actividades. Cuando una aplicación posee más de una actividad, es bastante frecuente arrancar una de ellas desde otra. Esto se consigue mediante una intención (*Intent* en inglés).

Vamos a utilizar una intención para iniciar nuestro juego desde una pantalla inicial que muestra una imagen. Hasta ahora todas nuestras aplicaciones tenían una única actividad. La aplicación de esta unidad posee dos: `Initial` y `MainActivity`, que deben especificarse en el fichero manifiesto:

`AndroidManifest.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="es.uam.eps.android.CCC22"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="19" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".Initial"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainActivity">
            <intent-filter>
                <action android:name="es.uam.eps.android.CCC22.MAINACTIVITY" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Veamos lo que significan los atributos del elemento `activity`:

- El atributo `android:name` especifica el nombre de la clase, que extiende actividad. Se puede indicar el nombre completo de la clase como, por ejemplo, `es.uam.eps.android.CCC22.Initial`, o como alternativa más sencilla, si se empieza por punto, podemos poner solo el nombre de la clase (`.Initial`, por ejemplo).
- El atributo `android:label` es el texto que se muestra en la barra de acción de la pantalla cuando la actividad está visible. En caso de que no se especifique, como en el caso de `MainActivity`, se mostrará la etiqueta especificada en el elemento `application`.

Dentro del elemento `activity` se encuentra el elemento `intent-filter` que especifica los tipos de intenciones a los que responde la actividad. El elemento `intent-filter` contiene:

- Al menos un elemento `action`. El atributo `android:name` indica a qué acciones responde esta actividad. En el caso de `Initial` este atributo toma el valor `android.intent.action.MAIN`, que identifica a esta actividad como el punto de entrada para la ejecución de nuestra aplicación. En el caso de `MainActivity` el valor es `es.uam.eps.android.CCC22.MAINACTIVITY`, que se trata de una acción definida por el programador. El nombre de dominio invertido reduce la probabilidad de colisión con otros nombres.
- Elementos `category` que contienen información adicional sobre el tipo de actividad. En nuestro caso, el nombre de la categoría del filtro es `android.intent.category.DEFAULT`. Este valor permite que otras actividades puedan arrancar a `MainActivity` con el método `startActivity()`. El valor `android.intent.category.LAUNCHER` indica que se añadirá un icono en el menú de aplicaciones del sistema.

A continuación mostramos los ficheros `Initial.java` e `initial.xml`, correspondientes a la nueva actividad de nuestro juego:

`/src/Initial.java`

```
package es.uam.eps.android.CCC22;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.MotionEvent;

public class Initial extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.initial);
    }

    public boolean onTouchEvent(MotionEvent event) {
        if (event.getAction() == MotionEvent.ACTION_DOWN) {
            startActivity(new Intent("es.uam.eps.android.CCC22.MAINACTIVITY"));
        }
        return true;
    }
}
```

El método `onTouchEvent()` de la actividad se ejecuta cuando el usuario toca la pantalla y ninguna otra vista gestiona el evento. Concretamente, tras comprobar que el código de acción es `ACTION_DOWN`, se instancia un objeto de la clase `Intent` al que se le pasa como argumento el nombre del filtro de la actividad que deseamos invocar: `MainActivity`. El objeto de tipo `Intent` se pasa como argumento al método `startActivity()`, el cual finalmente invoca la actividad `MainActivity`. Se debe devolver `true` cuando el evento fue gestionado y `false` en caso contrario.

Si la actividad que se invoca se encuentra en el mismo paquete que la actividad invocadora, se puede utilizar esta otra llamada a `startActivity()`:

```
startActivity(new Intent(this, MainActivity.class));
```

En este caso no es necesario incluir el elemento `intent-filter` en el fichero de manifiesto. El archivo `initial.xml` utiliza un `ImageView` para mostrar una imagen que, si el usuario toca, arranca la actividad `MainActivity` como hemos visto en el fichero `Initial.java`:

`/res/layout/initial.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <ImageView
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       android:src="@drawable/initial"
       android:scaleType="fitXY"
       android:contentDescription="@string/initialImage" />
</LinearLayout>
```

El archivo `initial.png` se ha colocado en la carpeta `drawable-mdpi` del proyecto. La pantalla que se observa al arrancar la aplicación es la siguiente:

Al pulsar sobre esta pantalla se arrancará la actividad `MainActivity`, que mostrará el tablero que ya conocemos de unidades anteriores.