Este texto forma parte de "El Gran libro de Android" de la editorial Marcombo, de Jesús Tomás

Más información en www.androidcurso.com

11.4 Añadiendo publicidad en tu aplicación con AdMob.

Hay muchas maneras de monetizar una aplicación. Una de las opciones más aceptadas por los usuarios es la de poner publicidad en las aplicaciones, ya que no requiere que el usuario pague nada directamente. Para esto, Android no impone ningún tipo de restricción, se pueden realizar acuerdos con cualquier compañía de publicidad. Algunas de las plataformas que te permiten insertar anuncios en tus aplicaciones son: AdMob (Google), iAd (Apple), AdHub (Samsung), AirPush, Millennial Media, MobFox y GeenApp (para promocionar aplicaciones).

En este apartado veremos cómo añadir publicidad de la compañía ***AdMob*** (ADvertising on MOBile") que es propiedad de Google. Esta compañía ofrece soluciones para añadir anuncios en aplicaciones desarrolladas en Android, iOS, webOS, Flash Lite, Windows Phone y cualquier navegador para móvil. El anuncio que se visualiza en cada momento es seleccionado por la compañía según el tipo de aplicación, situación geográfica y el usuario en concreto que la está utilizando. Su funcionamiento es muy similar al que ofrece Google a través de su compañía *AdSense*, con el que podemos recibir ingresos insertando anuncios en nuestros sitios Web.

Vídeo[Tutorial]: *AdMob: cómo obtener ingresos de su aplicación*

Ejercicio paso a paso: *Una primera aplicación con anuncios*

1. Crea un nuevo proyecto con los siguientes datos:

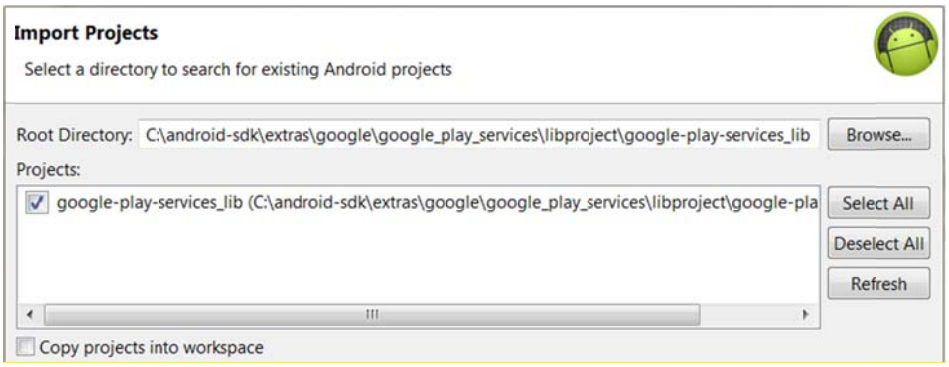
```
Application Name: Anuncios
Package name: com.example.anuncios
Minimum Required SDK: API 9 (2.3)
```

NOTA: *Se requiere una versión de Android 2.3 o superior para poder utilizar Google Play Services.*

2. Abre *Android SDK Manager* y asegúrate que el paquete *Google Play Services* está instalado:

1. A continuación vamos a importar al workspace el proyecto con las librerías de *Google Play Services.* Puedes saltarte el siguiente punto si ya lo has importado para proyectos anteriores (Puedes verificarlo mirando si en el *Pakage Explorer* aparece el proyecto *google-play-services_lib*).

2. Utilizando el entorno **Eclipse**, entra a *File* > *Import* > *Android* > *Existing Android Code Into Workspace*. Pulsa en *Browse…*, y ve a la carpeta donde está instalado *Android SDK Manager*. Desde ésta, selecciona la carpeta */android-sdk/extras/google/google_play_services/libprojects/google-play-services_lib*. Debería aparecer seleccionado un proyecto llamado *google-play-services_lib*. Pulsa en *Finish*.



   Utilizando el entorno **Android Studio**, abre la ruta donde está instalado Android Studio. En Windows puede ser `C:\Users\<Usuario>\AppData\Local\Android\android-studio`. Desde esta carpeta, accede a la ruta `sdk\extras\google\google_play_services\libprojects` y copia la carpeta `google-play-services_lib` a la carpeta de proyectos de *Android Studio* (por defecto esta carpeta se llama `AndroidStudioProjects`, y está dentro de la carpeta de usuario). Esto es solo por comodidad, puede estar en cualquier otro sitio. Copia a la misma carpeta el archivo `android-support-v13.jar` que encontrarás en `sdk\extras\android\support\v13`.

3. En el proyecto *Anuncios vamos a importar esta librería.* Utilizando **Eclipse**, haz clic con el botón derecho y selecciona *Properties* > *Android*. En la sección de sección de librerías, pulsa en el botón *Add…* y selecciona el proyecto que acabas de importar.

   Utilizando el entorno **Android Studio**, con el proyecto abierto, ves a *File* > *Project Structure...* y selecciona en la izquierda *Modules*. Hacer clic en el botón de mas (+) de arriba y seleccionar *Import Module*. En la ventana que aparece, selecciona la carpeta `google-play-services_lib` y pulsa en aceptar. En las ventanas siguientes, acepta todas las opciones por defecto. Otra vez en la ventana donde hemos seleccionado *Modules*, asegúrate de que nuestro proyecto sigue seleccionado, elige la pestaña *Dependences* y pulsa en el botón mas (+) que aparece a la derecha. Pulsa en *Library* y selecciona la librería que acabamos de añadir.

   Con esto, ya deberían estar las librerías adecuadas para trabajar con *Google AdMob*.

4. Hay dos formas de añadir un anuncio, por código o por XML. Lo más sencillo es añadirlo por XML. Para ello, copia el siguiente elemento dentro del `RelativeLayout` de *activity_main.xml*:

```
<com.google.android.gms.ads.AdView
    android:id="@+id/adView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    ads:adSize="BANNER"
    ads:adUnitId="ADMOB_PUBLISHER_ID"/>
```

NOTA: *En el atributo `adUnitId` hemos puesto ADMOB_PUBLISHER_ID. Ahí va el ID que nos proporcionará AdMob. Al final del tutorial conseguiremos este ID.*

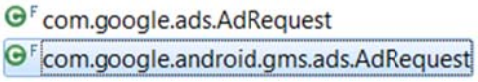
5. Como estamos utilizando atributos que no son de Android (los del espacio de nombres `ads`), tendremos que añadir la siguiente línea al elemento principal (el `RelativeLayout`):

```
<RelativeLayout
    xmlns:ads="http://schemas.android.com/apk/res-auto"
    xmlns:android=…
```

6. Ahora, en la clase `MainActivity`, indicaremos al API de *AdMob* que nos muestre un anuncio. Para ello, copia dentro del método `onCreate()` las siguientes líneas:

```
AdView adView = (AdView) findViewById(R.id.adView);
AdRequest adRequest = new AdRequest.Builder().build();
adView.loadAd(adRequest);
```

Pulsa Shift-Ctrl-O para incorporar los imports automáticamente. La clase AdRequest se encuentra definida en dos paquetes diferentes. Utiliza el que se indica a continuación:



7. En *AndroidManifest.xml*, hay que pedir los permisos de *"INTERNET"* y *"ACCESS_NETWORK_STATE"* dentro de la etiqueta `<manifest>`:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

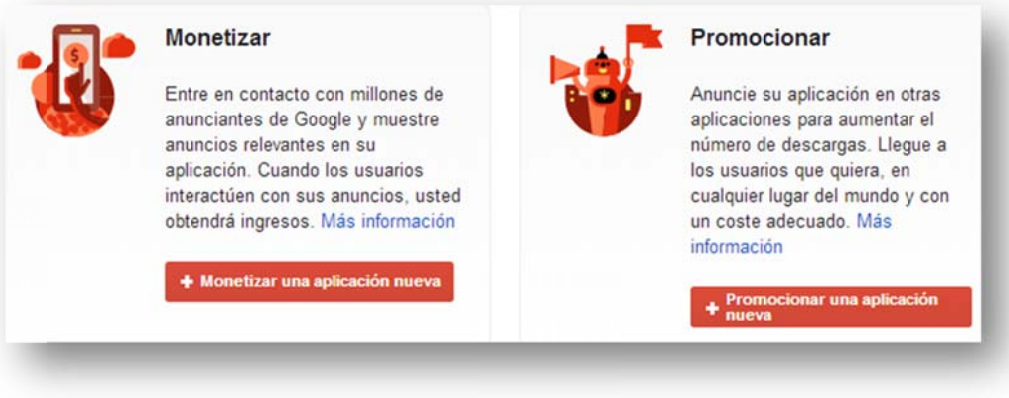
8. También en *AndroidManifest.xml*, dentro de la etiqueta `<application>`, añade las siguientes etiquetas:

```
<meta-data
    android:name="com.google.android.gms.version"
    android:value="@integer/google_play_services_version" />
<activity
    android:name="com.google.android.gms.ads.AdActivity"
    android:configChanges="keyboard|keyboardHidden|orientation|
                    screenLayout|uiMode|screenSize|smallestScreenSize" />
```

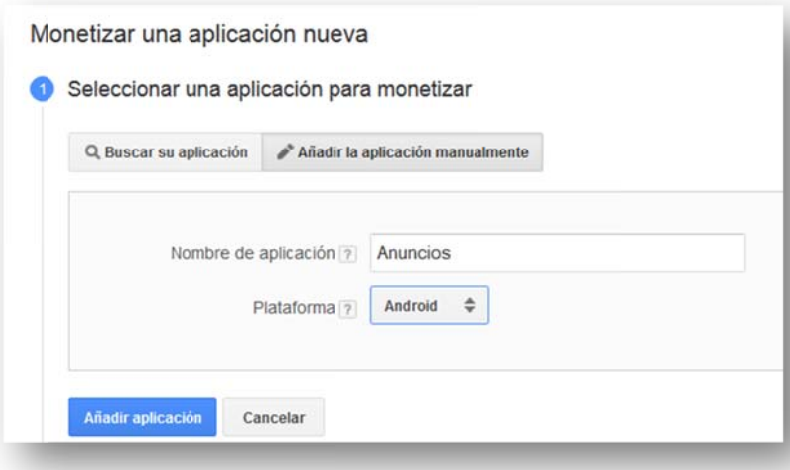
Siempre que utilices los servicios de Google Play es imprescindible que añadas la etiqueta `<meta-data>` que se muestra para definir la versión. Además, se ha declarado una nueva actividad que será usada para visualizar el anuncio cuando el usuario pulse sobre su banner.

9. Solo nos queda crear una cuenta en *AdMob*. Para ello entra en la página www.google.com/ads/admob/ y pulsa en *Sign Up with AdMob* para crear la cuenta.

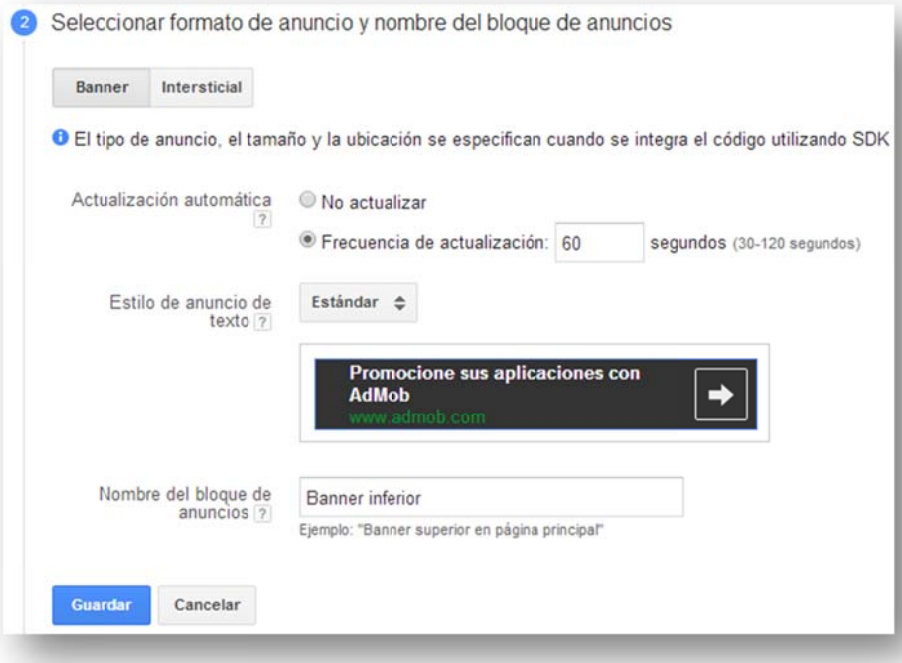
10. Introduce todos los datos solicitados. Una vez registrada la cuenta se presentaran dos opciones: *Monetizar* (para poner anuncios en tus aplicaciones) o *Promocionar* (para anunciar tus aplicaciones). Haz clic en la primera opción.



11. En el siguiente paso, selecciona la opción *Añadir aplicación manualmente*. Introduce como nombre de la aplicación: *Anuncios* y selecciona Android como plataforma. Después haz clic en *Añadir aplicación*.



12. A continuación, selecciona como formato de anuncio: *Banner,* el periodo de actualización y estilo que desees e introduce como nombre del bloque de anuncios: *Banner inferior.* Pulsa en *Guardar*.



13. Por último, se mostrará el ID del bloque de anuncios que acabas de configurar. Copia este ID en el portapapeles y pulsa el botón *Listo*.

14. Abre e layout *activity_main.xml* del proyecto y pega dentro del atributo adUnitId el ID del anuncio:

```
<com.google.android.gms.ads.AdView
    …
    ads:adUnitId="ca-app-pub-2307875346715292/8646625924" />
```

15. Ejecuta la aplicación en un dispositivo real. El anuncio ha de aparecer en la parte inferior de la pantalla. La primera vez puede tardar más tiempo en procesar la petición del anuncio.

    Nota: *En caso de utilizar un emulador deberás instalar antes* Google Play Services. *Este proceso resulta bastante complejo*.

Ejercicio paso a paso: *Escuchador de eventos sobre vista de anuncios*

Podemos hacer un seguimiento de los eventos relacionados con los anuncios añadiendo un escuchador de la clase `AdListener` a la vista del anuncio. Así podemos registrar cuando se carga un anuncio, cuando es abierto, si se produce algún error, etc.

1. Añade la siguiente línea al final del método `onCreate()`:

```
adView.setAdListener(new miAdListener());
```

2. Añade el siguiente código al final de la clase, antes de la última `}`:

```
class miAdListener extends AdListener {

 @Override    public void onAdLoaded() {
       Toast.makeText(MainActivity.this, "onAdLoaded",

       Toast.LENGTH_SHORT).show();
 }

 @Override public void onAdOpened() {
       Toast.makeText(MainActivity.this, "onAdOpened",

       Toast.LENGTH_SHORT).show();
 }

 @Override    public void onAdLeftApplication() {
       Toast.makeText(MainActivity.this, "onAdLeftApplication",

       Toast.LENGTH_SHORT).show();
 }

 @Override    public void onAdClosed() {
       Toast.makeText(MainActivity.this, "onAdClosed",

       Toast.LENGTH_SHORT).show();
 }

 @Override    public void onAdFailedToLoad(int error) {
       Toast.makeText(MainActivity.this, "onAdFailedToLoad: " + error,

       Toast.LENGTH_SHORT).show();
 }
}
```

Hemos creado una nueva clase que extiende `AdListener` para que actue como escuchador de los eventos asociados a la visualización de un anuncio. El método `onAdLoaded()` será llamado cada vez que se cargue un nuevo anuncio. Ha de aparecer un Toast con el primer anuncio y cada vez que es reemplazado (En nuestra configuración cada 60 seg). El método `onAdOpened()` es llamado cuando el usuario pulsa sobre el banner y se abre la nueva actividad para el anuncio. El método `onAdLeftApplication()` es llamado cuando se sale de muestra aplicación (suele ocurrir tras el evento anterior). El método `onAdClosed()` es llamado cuando el usuario cierra el anuncio para volver a nuestra

aplicación. Finalmente el método `onAdFailedToLoad()` es llamado cuando se produce algún error.

3. Ejecuta la aplicación y observa los eventos que van ocurriendo. Para ello prueba a hacer clic en un anuncio para que se abra otra ventana y después vuelve a la aplicación.

Enlaces de interés: *Para aprender más sobre Google AdMob:*.

- https://developers.google.com/mobile-ads-sdk/docs/?hl=es
- https://developers.google.com/mobile-ads-sdk/docs/admob/fundamentals?hl=es
- https://developers.google.com/mobile-ads-sdk/docs/admob/intermediate?hl=es