mobialia

Taller
Mi Primera Aplicación Android

Sobroso Party
Abril 2011

Alberto Alonso Ruibal
alberto.ruibal@mobilia.com
http://www.mobialia.com
T: @mobialia @albertoruibal

Nuestra aplicación: Wikiplaces

Obtendrá los lugares de la Wikipedia cerca de nuestra posición mostrándolos en un mapa y en una lista

Veremos ejemplos de:

- Layouts, ListViews, PreferenceActivity...
- API de geolocalización
- API de Google Maps
- Obtención de datos mediante JSON

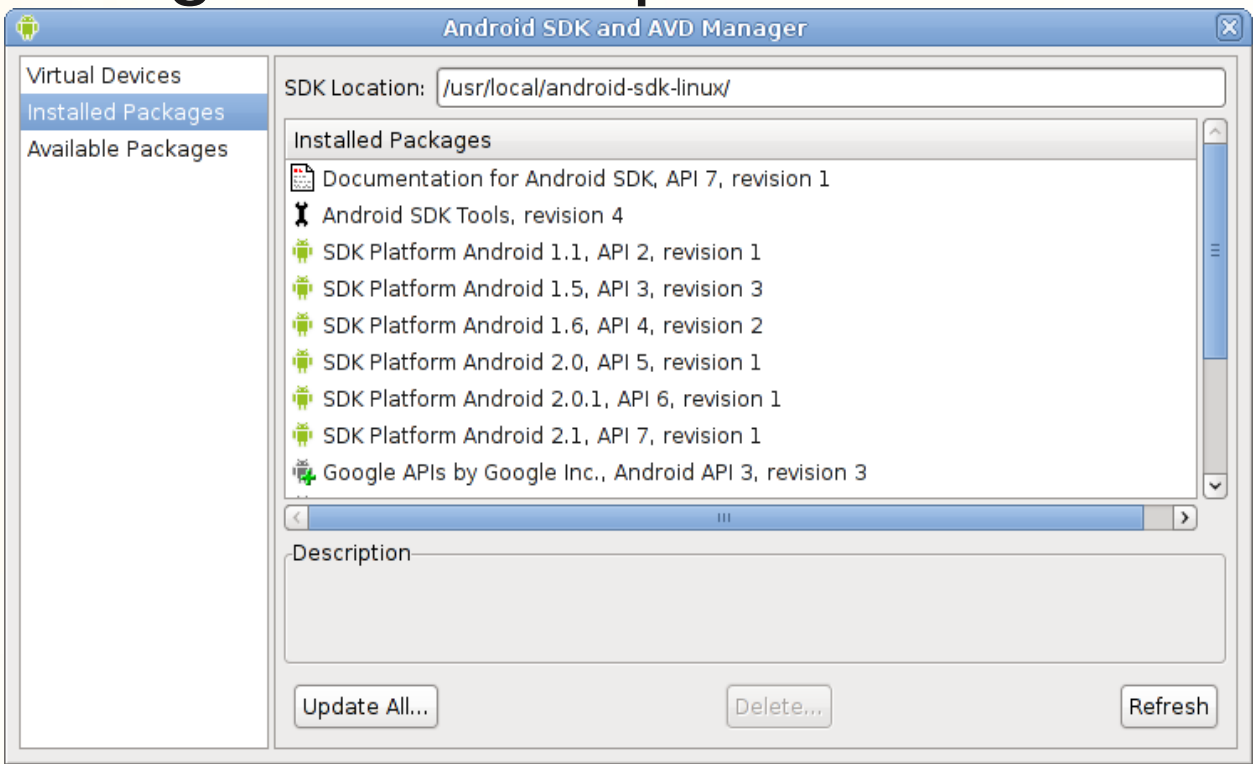
Desarrollar para Android

- Lenguaje **Java**
- También posible en C, hay un NDK
- API muy completa y sencilla
- SDK para Linux/Windows/Mac con emulador
- Eclipse IDE (es la opción oficial, hay otras posibilidades como IntelliJ IDEA)

El SDK de Android

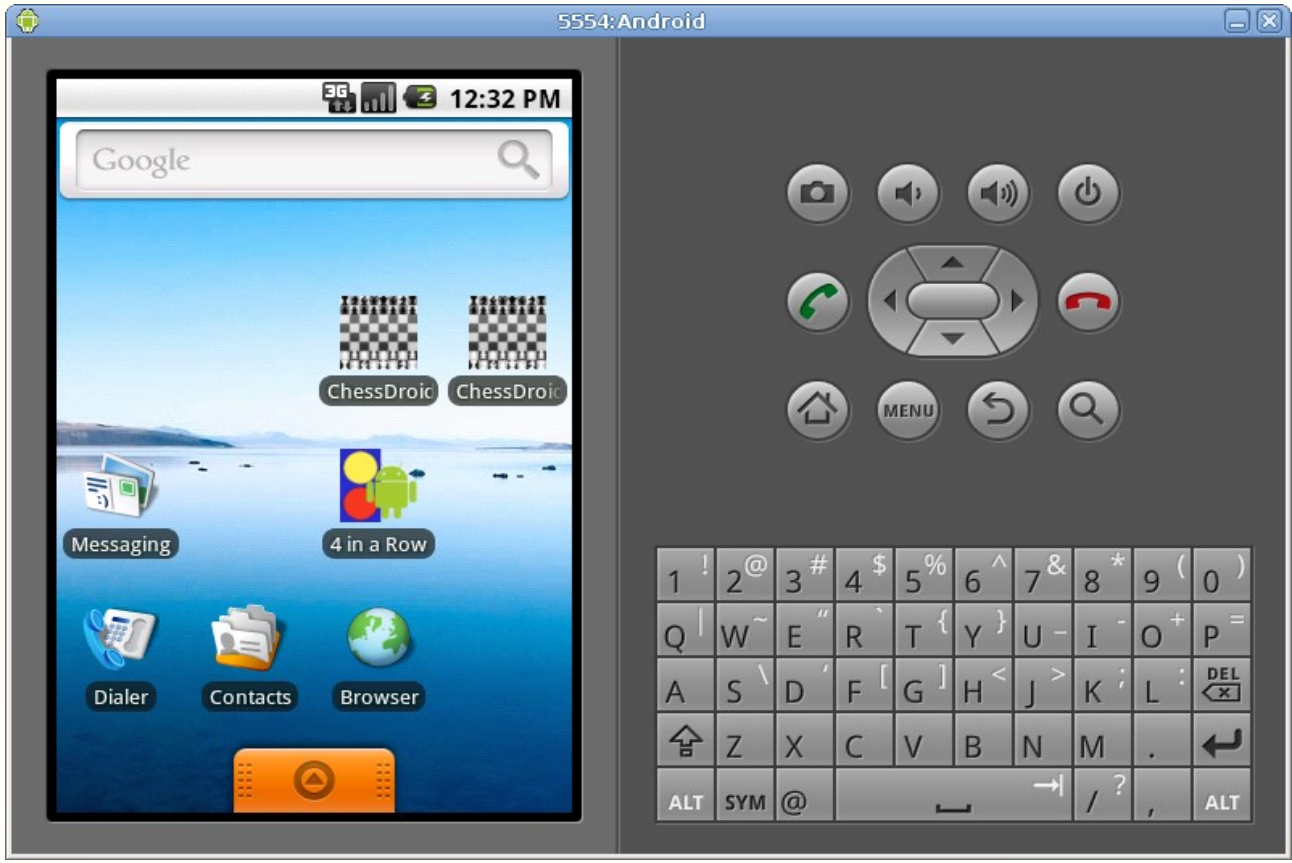
- http://developer.android.com/sdk/
- En el SDK/AVD manager se seleccionan las plataformas necesarias y se descargan
- También permite configurar los dispositivos emulados

(Virtual Devices)



El emulador

- Permite simular distintos dispositivos / tamaños de pantala
- Completamente funcional aunque algo lento



Eclipse IDE

IDE muy versátil para Java, PHP, C++...

http://www.eclipse.org

Instalación del plugin ADT

- Abrir Eclipse
- Seleccionar Help → Software Updates… → Available Software
- En la ventana escoger “Add Site” , e introducir la ubicación:

https://dl-ssl.google.com/android/eclipse/

mobialia

Crear un nuevo proyecto en Eclipse

File->New->Android Project

Cubrimos los detalles con:

Project name: WikiPlaces

Build Target : Google APIs 1.6

Application name: WikiPlaces

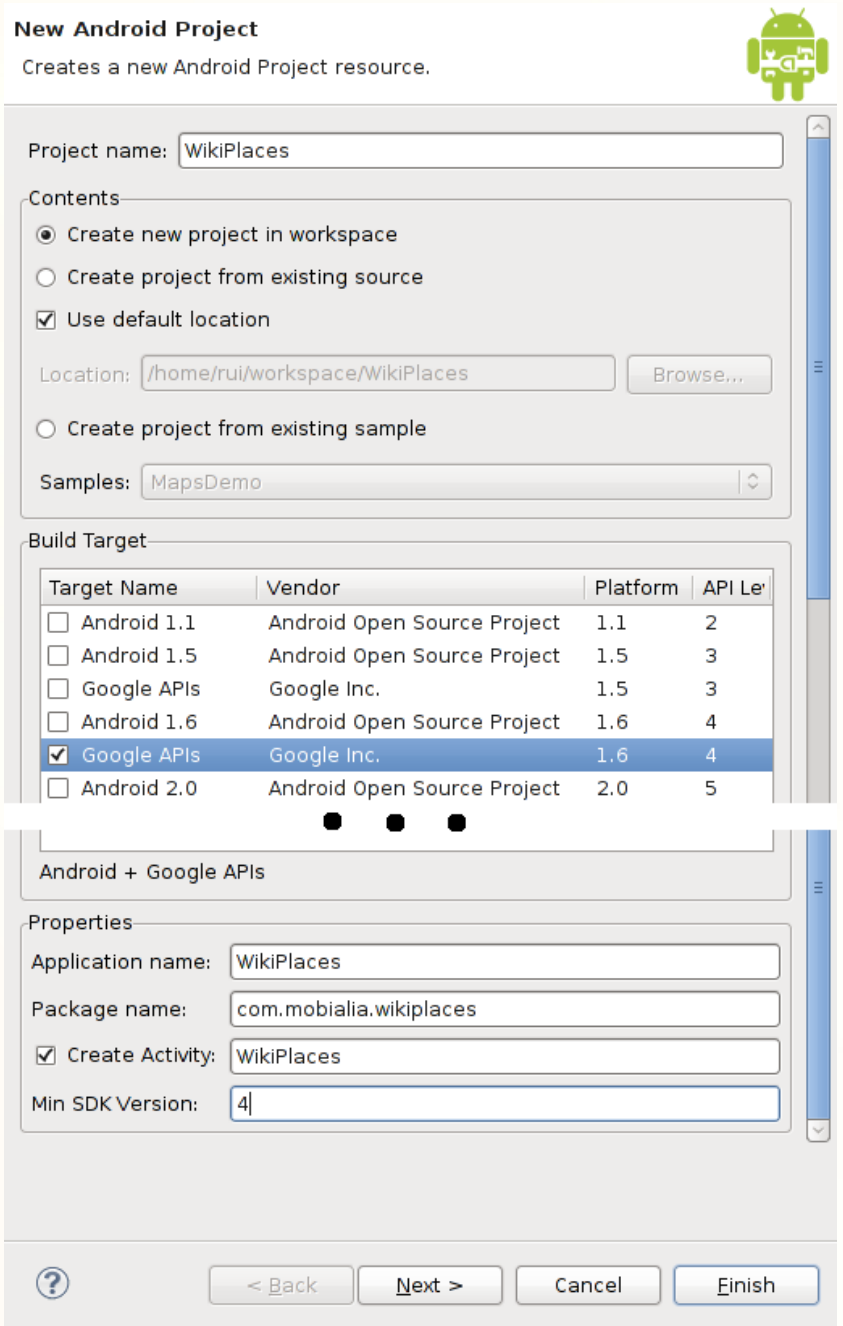
Package name:

com.mobialia.wikiplaces

Create Activity: WikiPlaces

Min SDK Version: 4

Pulsa **Finish**



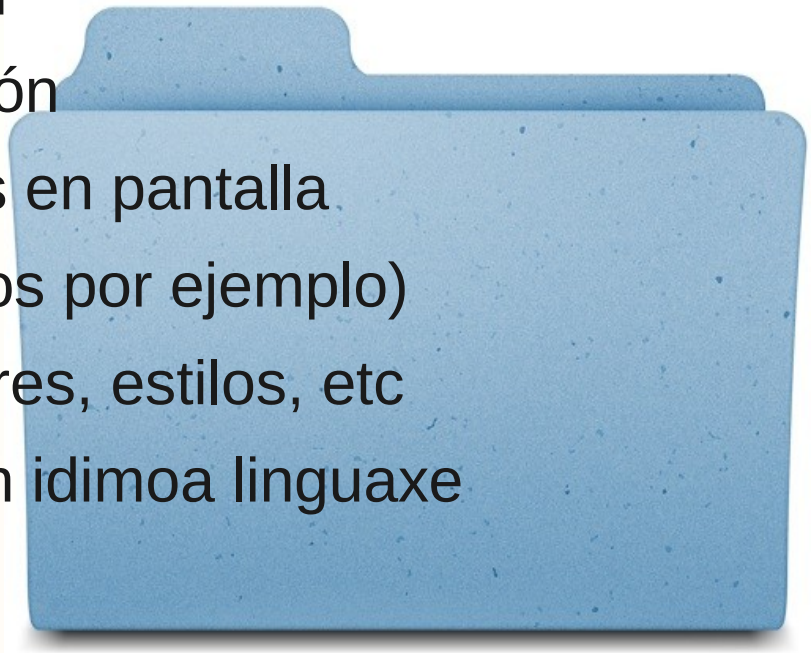
Componentes de una aplicación

Se definen en el AndroidManifest.xml, en el cual también se indican los permisos necesarios, la versión de Android exigida, etc

- **Actividades**
- **Servicios**
- **Intents**: sistema de comunicación entre aplicaciones/actividades

Estructura de directorios

src/	Código fuente
gen/	Archivos generados a partir de los recursos
assets/	Ficheros a los que va a acceder la aplicación
res/	Carpeta de recursos
| drawable-hdpi/	Imágenes alta resolución (dpi)
| drawable-ldpi/	Imágenes baja resolución
| drawable-mdpi/	Imágenes media resolución
| layout/	Disposición de elementos en pantalla
| raw/	Archivos de datos (sonidos por ejemplo)
| values/	Definición de textos, colores, estilos, etc
| values-es/	Cadenas localidadas a un idimoa linguaxe



Primera actividad: Dashboard

Vamos a implementar en la clase WikiPlacesActivity un Dashboard desde el cual accederemos a las distintas secciones de la aplicación:

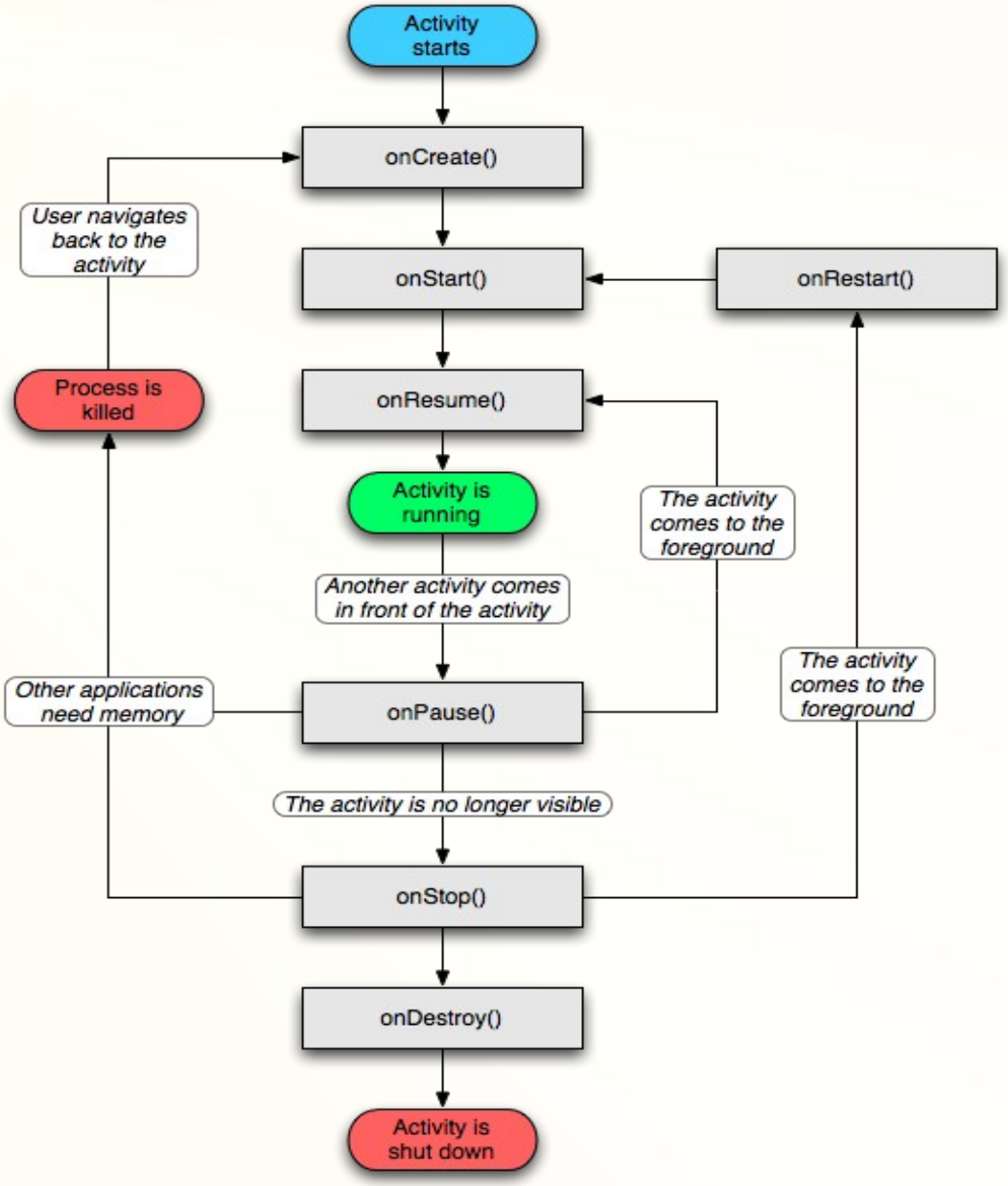
- Ver un mapa
- Ver una lista
- Actualizar los datos
- Acceder a las preferencias

Utilizaremos los patrones Dashboard y ActionBar

http://www.google.com/events/io/2010/sessions/android-ui-design-patterns.html

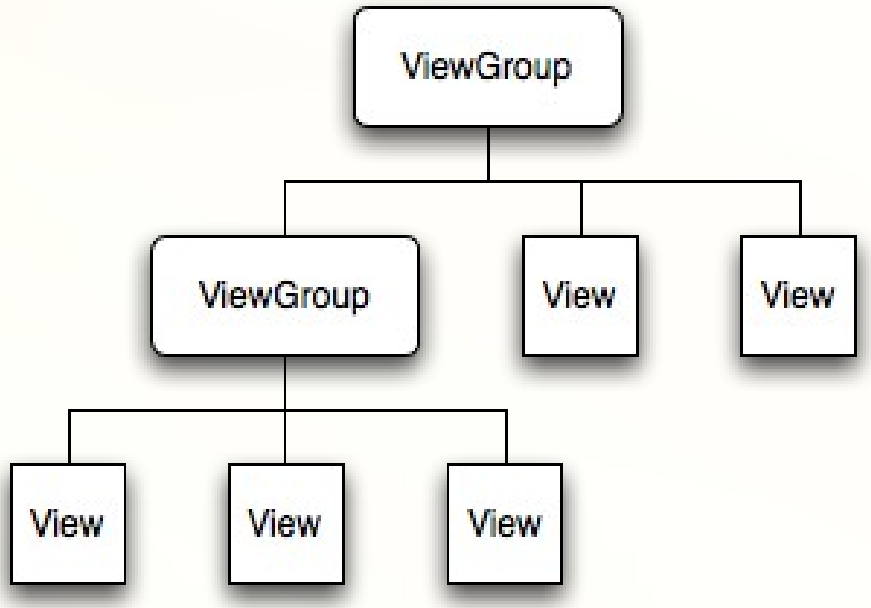
Ciclo de vida de las Actividades

Hay varios métodos dentro de nuestra actividad que son llamados cuando cambia de estado



Definición del Layout

- ViewGroups: LinearLayout, TableLayout, RelativeLayout...
- Views: TextView, ListView, o personalizadas
- Editor WYSIWYG: pero siempre es mejor acudir al XML



Dashboard: Layout

Crearemos un res/layout/dashboard.xml para el cual nos debería llegar con:

- LinearLayout
- Button, ImageButton,Textview

Simplificaremos el layout utilizando:

- values/strings.xml para las cadenas de texto
- values/colors.xml para los colores
- values/styles.xml para los estilos

http://www.alonsoruibal.com/using-styles-on-android-layouts/

Dashboard: Activity

En la clase WikiplacesActivity introduciremos:

```
public void onCreate(Bundle bundle) {
    super.onCreate(icicle);
    // Quitamos la barra de título
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    requestWindowFeature(Window.FEATURE_PROGRESS);
    // Establecemos el contenido
    setContentView(R.layout.dashboard);
}
```

Dashboard: Botones con imágenes

- drawable_hdpi/dashboard_preferences_default.png
- drawable_hdpi/dashboard_preferences_selected.png
- drawable/dashboard_preferences.xml:

```
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_focused="true" android:drawable="@drawable/dashboard_preferences_selected"/>
    <item android:state_pressed="true" android:drawable="@drawable/dashboard_preferences_selected"/>
    <item android:state_focused="false" android:state_pressed="false"
android:drawable="@drawable/dashboard_preferences_default"/>
</selector>
```

- Y en nuestro layout ponemos:

```
 <Button style="@style/DashBoardAction"
  android:drawableTop="@drawable/dashboard_preferences"
  android:text="@string/dashboard_preferences"
  android:onClick="onPreferencesAction"
 />
```

- Al pulsarlo llamará al método onPreferencesAction de nuestra actividad

```
public void onPreferencesAction(View v) {
...
}
```

Lanzando otra actividad: Intents

- Por ejemplo, para lanzar una nueva actividad cuando pulsamos el botón de preferencias:

```
public void onPreferencesAction(View v) {
    Intent intent = new Intent(getApplicationContext(), PreferencesActivity.class);
    startActivity(intent);
}
```

- Creamos la nueva clase de la actividad: también debemos añadirla en el AndroidManifest.xml:

```
<activity android:name="PreferencesActivity"
          android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.EMBED" />
    </intent-filter>
</activity>
```

Actividad para las prefencias

- La clase de la Actividad es tremendamente sencilla:

```
public class PreferencesActivity extends PreferenceActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // Load the preferences from an XML
        addPreferencesFromResource(R.layout.preferences);
    }
}
```

- En layout/preferences.xml definimos los campos que va a tener nuestra pantalla de preferencias con ListPreference, CheckboxPreference, EditTextPrefrence...

Definiendo preferencias en el XML

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
        xmlns:android="http://schemas.android.com/apk/res/android">
         <ListPreference
                android:key="distance"
                android:title="@string/preferences_distance"
                android:summary="@string/preferences_distance_summary"
                android:entries="@array/distances_texts"
                android:entryValues="@array/distances_values"
                android:dialogTitle="@string/preferences_distance"
                android:defaultValue="20"
                android:selectable="true"
                />
        <CheckBoxPreference
                android:key="satellite"
                android:title="@string/preferences_satellite"
                android:summary="@string/preferences_satellite_summary"
                android:defaultValue="false"/>
        <CheckBoxPreference
                android:key="fullscreen"
                android:title="@string/preferences_fullscreen"
                android:summary="@string/preferences_fullscreen_summary"
                android:defaultValue="false"/>
</PreferenceScreen>
```

Obteniendo valores de preferencias

- Ya de nuevo en el onResume de WikiplacesActivity vamos a obtener la preferencia “fullscreen” y actuar en función de su valor:

```
SharedPreferences sharedPref =
    PreferenceManager.getDefaultSharedPreferences(this);
fullscreen = sharedPref.getBoolean("fullscreen", false);

if (fullscreen) {
    getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                         WindowManager.LayoutParams.FLAG_FULLSCREEN);
} else {
    getWindow().setFlags(0,
                         WindowManager.LayoutParams.FLAG_FULLSCREEN);
}
```

Obteniendo la ubicación (I)

- Para hacer que nuestra actividad principal obtenga la ubicación, primero añadimos los permisos en el AndroidManifest.xml

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

- Después implementamos LocationListener

```
.... implements LocationListener

public void onLocationChanged(Location location) {
   Log.d(TAG, “Location Received!!!!”);
}

public void onProviderDisabled(String provider) {}

public void onProviderEnabled(String provider) {}

public void onStatusChanged(String provider, int status, Bundle extras) {}
```

Obteniendo la ubicación (II)

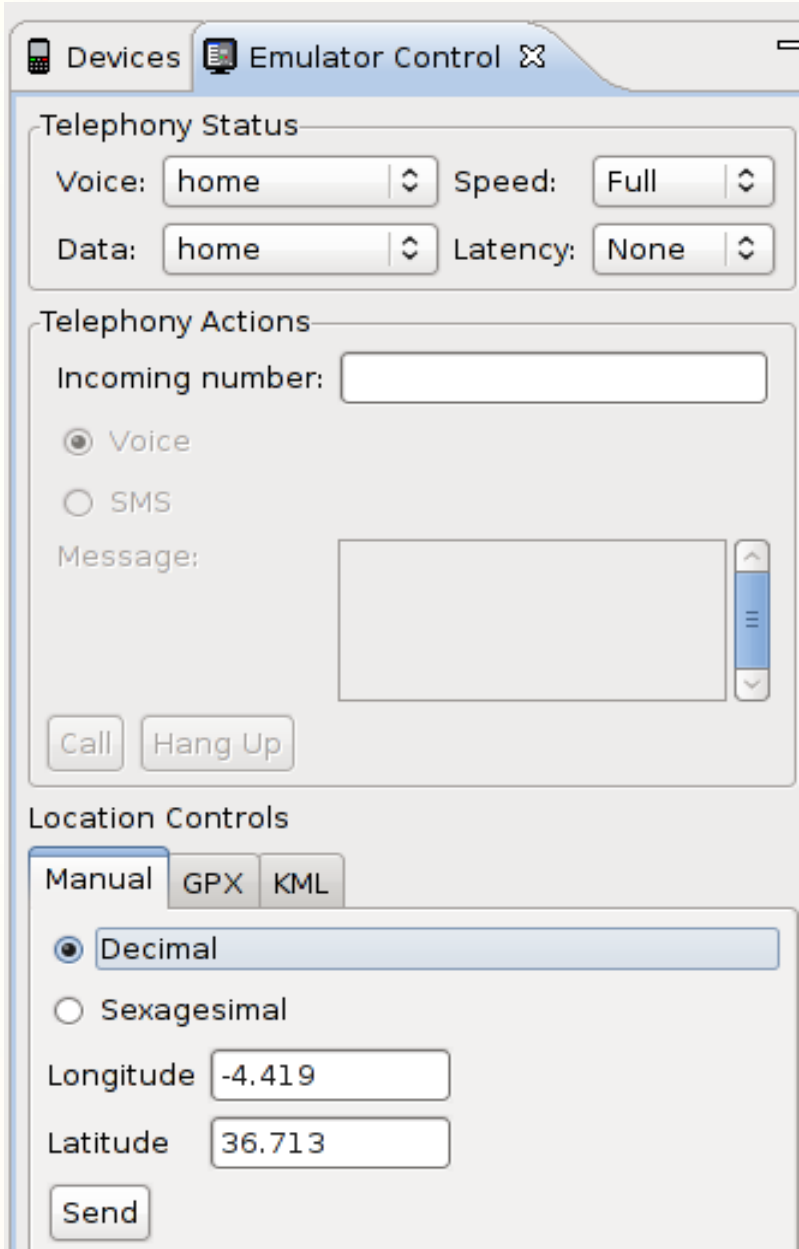
- Ahora solicitamos actualizaciones de la ubicación con dos proveedores de ubicación (sí dos!):

```
manager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);

Criteria criteria = new Criteria();
criteria.setAltitudeRequired(false);
criteria.setBearingRequired(false);
criteria.setCostAllowed(false);
criteria.setPowerRequirement(Criteria.POWER_LOW);

criteria.setAccuracy(Criteria.ACCURACY_FINE);
providerFine = manager.getBestProvider(criteria, true);

criteria.setAccuracy(Criteria.ACCURACY_COARSE);
providerCoarse = manager.getBestProvider(criteria, true);

if (providerCoarse != null) manager.requestLocationUpdates(providerCoarse, 5*60000, 100,
this); // update each 5 minutes
if (providerFine != null) manager.requestLocationUpdates(providerFine, 5*60000, 100, this
); // update each 5 minutes
```

http://www.alonsoruibal.com/using-two-locationproviders-on-android/

Simulando la ubicación en el emulador

- Nuestro ordenador no tiene GPS, pero el emulador nos permite “simular” ubicaciones introduciendo longitud y latitud



Obteniendo los datos a mostrar

- Añadir a la aplicación el permiso para conectarse Internet

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

- Utilizaremos los servicios de GeoNames http://www.geonames.org/
- Primero creamos una cuenta en http://www.geonames.org/login
- Y habilitamos los “Free Web Services” en: http://www.geonames.org/manageaccount
- Para el ejemplo podremos usar la cuenta “mobialia” en lugar de crear una

Servicio JSON de GeoNames

- Acrónimo de JavaScript Object Notation
- Más fácil de utilizar que los servicios XML
- Utilizaremos el WebService: findNearbyWikipediaJSON al que le pasamos, “lat”, “lng”, “radius”, “maxRows” y “username”
- Ejemplo de petición JSON a Geonames:

http://api.geonames.org/findNearbyWikipediaJSON?lat=47&lng=9&username=mobialia

Almacenando los datos recibidos

- Creamos una clase contenedora “Wikiplace” con los campos que nos devuelve el WebService JSON
- Otra clase “WikiplacesData” nos gestionará las peticiones al servidor y almacenará los resultados en memoria, en un ArrayList<Wikiplace>
- Android tiene integrada una librería para procesar JSON:

```
JSONObject base = new JSONObject(dataString);
JSONArray data = base.getJSONArray("geonames");

for (int i = 0; i < data.length(); i++) {
    JSONObject obj = data.getJSONObject(i);
    Wikiplace wikiplace = new Wikiplace();
    wikiplace.setLat(obj.getDouble("lat"));
    ..
```

Mostrando un Mapa: MapActivity

- Creamos una actividad que extienda MapActivity

```
public void onCreate(Bundle icicle) {
    super.onCreate(icicle);
    //..
    setContentView(R.layout.wikiplaces_map);

    mapView = (MapView)findViewById(R.id.mapview);
    mapView.setBuiltInZoomControls(true);
    mapView.getController().setZoom(14);
    //...
}
```

- Y definimos el layout wikiplaces_map.xml, pero antes debemos obtener una API key

Obteniendo una clave para el mapa

- Debemos ir a esta URL

http://code.google.com/intl/gl/android/maps-api-signup.html

- Nos pedirá el “Certificate fingerprint”, que se obtiene así:

```
keytool -list -keystore ~/.android/debug.keystore
```

- Luego ponemos la API key en el tag XML del mapa:

```
<com.google.android.maps.MapView
    android:id="@+id/mapview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:clickable="true"
    android:apiKey="0ty0JlR1lTjC5kB1dCPHOPbNscmPlSKGDA1KQug"
/>
```

Overlays sobre el Mapa

- En primer lugar añadiremos un overlay que muestre nuestra ubicación:

```
List<Overlay> overlays = mapView.getOverlays();
myLocationOverlay = new MyLocationOverlay(this, mapView
);
overlays.add(myLocationOverlay);
```

- El siguiente paso será hacer un overlay personalizado en el que mostremos los datos de Wikiplaces

Overlays personalizados (I)

- Extendemos la clase Overlay:

```
public class WikiplacesOverlay extends Overlay {

@Override
public void draw(Canvas canvas, MapView mapView, boolean shadow) {
    Projection projection = mapView.getProjection();
    // Por cada uno de los lugares obtenemos su geopoint...
    // ..
    // Transformamos el geoPoint en píxeles
    Point point = new Point();
    projection.toPixels(geoPoint, point);
    // Y finalmente pintamos “algo” en esa ubicación
    Paint paint = new Paint();
    paint.setColor(Color.BLACK);
    canvas.drawText(“texto”, point.x, point.y, paint);
}
```

Overlays personalizados (II)

- El Overlay tiene un método que se llama cuando el usuario pulsa en la pantalla

```
@Override
public boolean onTap(GeoPoint point, MapView mapView) {
   /...
}
```

- Para buscar el GeoPoint más cercano nos recorremos los Wikiplaces utilizando un DistanceCalculator

Añadiendo un menú (I)

- Primero creamos el menú interceptando onCreateOptionsMenu

```
public static final int MENU_MY_LOCATION = 1;

@Override

public boolean onCreateOptionsMenu(Menu menu) {

    super.onCreateOptionsMenu(menu);

    menu.add(0, MENU_MY_LOCATION, 0,
R.string.menu_my_location).setIcon(drawable.ic_menu_mylocation);

    return true;

}
```

Añadiendo un menú (II)

- Respondemos a la acción cuando el menú es seleccionado

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    GeoPoint geoPoint = null;
    switch (item.getItemId()) {
        case MENU_MY_LOCATION:
            geoPoint = myLocationOverlay.getMyLocation();
            if (geoPoint != null) mapView.getController().animateTo(geoPoint);
            return true;
    }
    return false;
}
```

Pedir datos al tener la ubicación

- En el WikiplacesActivity:

```
public void onLocationChanged(Location location) {
    geoPoint = location2GeoPoint(location);
    wikiplacesData.setLocationPoint(geoPoint);
    wikiplacesData.getDataFromServer(this);
}
```

- Hay que convertir el location a un Geopoint:

```
private GeoPoint location2GeoPoint(Location location) {
    if (location == null) return null;
    int latitude = (int) (location.getLatitude()*1E6);
    int longitude = (int) (location.getLongitude()*1E6);
    return new GeoPoint(latitude, longitude);
}
```

Mostrar los datos en una lista

- Creamos una nueva actividad WikiplacesListActivity extendiendo ListActivity
- Creamos el layout/wikiplaces_list.xml:

```
<ListView android:id="@id/android:list"
 android:layout_width="fill_parent"
 android:layout_height="fill_parent"
 android:layout_weight="1"
 android:drawSelectorOnTop="false"
 android:background="@color/background"
 android:cacheColorHint="#00000000"
/>
```

- Creamos un adapter personalizado

```
setContentView(R.layout.wikiplaces_list);
adapter = new WikiplacesAdapter(getApplicationContext());
setListAdapter(adapter);
```

Adapter personalizado

- El adapter gestiona los datos a mostrar en la lista

```
public class WikiplacesAdapter extends BaseAdapter {
```

- Definimos un layout para cada elemento de la lista en layout/wikiplaces_adapter.xml
- El método getView devolverá la vista correspondiente a un elemento

```
public View getView(int position, View convertView, ViewGroup parent) {
    LinearLayout ll = (LinearLayout) LayoutInflater.from(mContext
).inflate(R.layout.wikiplaces_adapter, parent, false);
    Wikiplace wikiplace = wikiplacesData.getWikiplace(position);
    TextView title = (TextView) ll.findViewById(R.id.Title);
    title.setText(wikiplace.getTitle());
}
```

Respondiendo al click en la lista

- Cuando se pulsa sobre un elemento de la lista se llama al método onListItemClick de la actividad:

```
@Override
protected void onListItemClick(ListView l, View v, int position,
long id) {
    super.onListItemClick(l, v, position, id);
    Log.d(TAG, “Elemento seleccionado”);
}
```

Pasando parámetros en un intent

- Al intent podemos pasarle un Bundle en los extras:

```
Intent intent = new Intent(this, WikiplacesDetailActivity.class);
Bundle bundle = new Bundle();
bundle.putInt("selectedPlace", position);
bundle.putBoolean("calledFromMap", false);
intent.putExtras(bundle);
startActivity(intent);
```

- Y en el onCreate de la actividad llamada:

```
Bundle extras = getIntent().getExtras();
selectedPlace = extras.getInt("selectedPlace");
calledFromMap = extras.getBoolean("calledFromMap");
```

Actividad con el detalle

- Crearemos una actividad WikiplacesDetailActivity en la cual mostraremos los detalles para un lugar seleccionado
- Desde esta actividad añadiremos botones para lanzar un navegador web con la URL de wikipedia o Google Maps Navigation con las instrucciones de navegación

Lanzar el navegador web

- Respondemos a la pulsación del botón “web”
- Creamos un intent “ACTION_VIEW” pasándole la URL que queremos mostrar

```
public void onWebAction(View v)  {
    String uri = "http://" +
            wikiplacesData.getWikiplace(selectedPlace).getWikipediaUrl();
    Intent myIntent = new Intent(android.content.Intent.ACTION_VIEW,
                                        Uri.parse(uri));
    startActivity(myIntent);
}
```

Lanzando Google Maps Navigation

- En el layout del detalle creamos un botón con android:onClick="onNavigateAction"
- En el Activity, creamos el método asociado que lanzará el intent de Google Maps Navigation

```
public void onNavigateAction(View v)  {
  GeoPoint point = gasStationData.getWikiplace(selectedPlace).getGeoPoint();

  String uri = "google.navigation:q="
    +Double.valueOf(point.getLatitudeE6()/1000000+","
    +Double.valueOf(point.getLongitudeE6())/1000000;

  Intent myIntent = new Intent(android.content.Intent.ACTION_VIEW, Uri.parse(uri));
  startActivity(myIntent);
}
```

Monetizar las aplicaciones

Hay tres opciones:

- Vender directamente en Android Market
- Insertar publicidad en las aplicaciones: **Admob**
- Desarrollar por encargo

Insertando anuncios de AdMob (I)

- Crear una cuenta en http://www.admob.com
- Crear la aplicación en “Sites & Apps” y obtener el Publisher ID
- 1) Descargar la última versión del admob-sdk-android.jar, situarlo en /lib y añadirlo al Build Path del proyectyo
- 2) Añadir en AndroidManifest.xml el ADMOB_PUBLISHER_ID:

```
...
<meta-data android:value="a14b82b10806662" android:name="ADMOB_PUBLISHER_ID" />
<meta-data android:value="true" android:name="ADMOB_ALLOW_LOCATION_FOR_ADS" />
</application>
```

Insertando anuncios de AdMob (II)

- 3) Crear el values/attrs.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<declare-styleable name="com.admob.android.ads.AdView">
 <attr name="backgroundColor" format="color" />
 <attr name="primaryTextColor" format="color" />
 <attr name="secondaryTextColor" format="color" />
 <attr name="keywords" format="string" />
 <attr name="refreshInterval" format="integer" />
</declare-styleable>
</resources>
```

- 4) Añadimos el AdView en el layout donde mostaremos el anuncio:

```
<com.admob.android.ads.AdView
 android:layout_height="wrap_content"
 android:layout_width="wrap_content"
 app:backgroundColor="@color/actionbar_background"
 app:primaryTextColor="#ffffff"
 app:secondaryTextColor="#ffffff"
 app:keywords="wikipedia, places"
/>
```

Otras consideraciones sobre AdMob

- Podemos poner anuncios sin geolocalización, pero obtenemos mejores ratios si la habilitamos
- “Debemos” especificar cuáles son nuestros terminales de prueba en el onCreate():

```
AdManager.setTestDevices( new String[]
{ "122E56EF9F911CBBA412F67B094A786A" } );
```

- Interesante posibilidad de introducir “House Ads”

Publicar en Android Market

Crear una cuenta de desarrollador:
http://market.android.com/publish

Antes de publicarla es necesario exportar un fichero APK firmado con un certificado definitivo

Vemos ejemplo práctico...

Exportar APK firmado

En el menú contextual del proyecto:

Android Tools-> Export Signed Application package

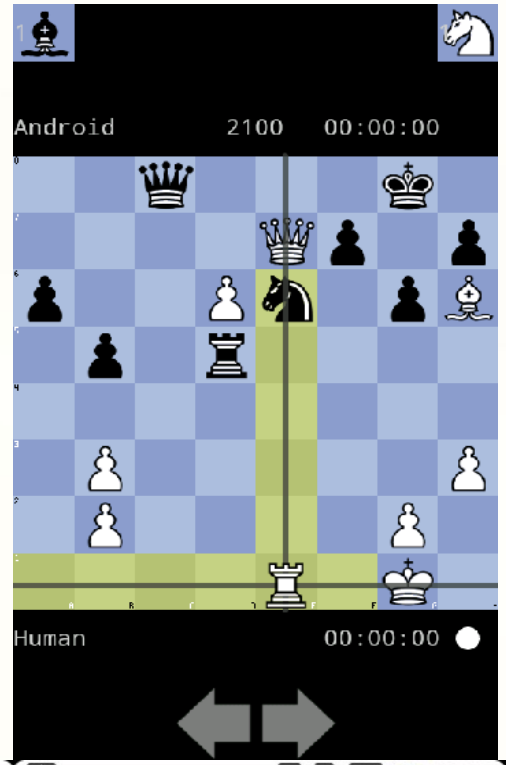
- Muy importante el nombre del paquete
- El certificado se genera desde Eclipse
- Obligatorio antes de publicar unha aplicación en el Market
- Las actualizaciones sucesivas de la aplicación deben ser firmadas con el mismo certificado
- Mucho cuidado con perder el certificado!!!

Mobialia

Start-up desarrollando exclusivamente aplicaciones Android

- Mobialia Chess
- Connect-4
- Gasolineras España
- ...

http://www.mobialia.com



Consultas, preguntas...

Gracias por vuesta atención

Alberto Alonso Ruibal
alberto.ruibal@mobialia.com
http://www.mobialia.com
T: @mobialia @albertoruibal