Nuevos Paradigmas de Interacción (2015-2016)
Grado en Ingeniería Informática
Universidad de Granada

Tutorial appGPSQR

Pedro Antonio Ruiz Cuesta
Ignacio Martín Requena

9 de febrero de 2016

Índice

Índice de figuras

1. Introducción

En este tutorial veremos como podemos desarrollar una aplicación android que lea un código QR de la forma:

LATITUD_37.19716626365634_LONGITUD_-3.6242308062076067

Figura 1.1: Código QR válido

Puesto que estos códigos QR tienen como información:
LATITUD_XXXXXXXXXXX_LONGITUD_XXXXXXXXXXX
Obtendremos las coordenadas para pasarselas al gps y que nos lleve hasta el lugar indicado.

2. Descripción

Esta aplicación constará de dos activities, una que nos permitita reconocer el código QR y otra para la parte del GPS.

3. Desarrollo

Para el desarrollo de nuestra app utilizaremos el IDE Android Studio.

Para la parte de la lectura del código hemos utilizado la libreria Zxing[1] que nos permite hacer uso de la aplicación **Barcode Scan** (si no la tenemos instalada nos dará la posibilidad de hacerlo) la cuál nos ayudará a realizar la tarea.

[1] `https://github.com/zxing/zxing`

3.1. Iniciando el proyecto

Lo primero que vamos a hacer es crear un nuevo proyecto en android studio, elegiremos que nos cree un proecto con Google Maps Activity

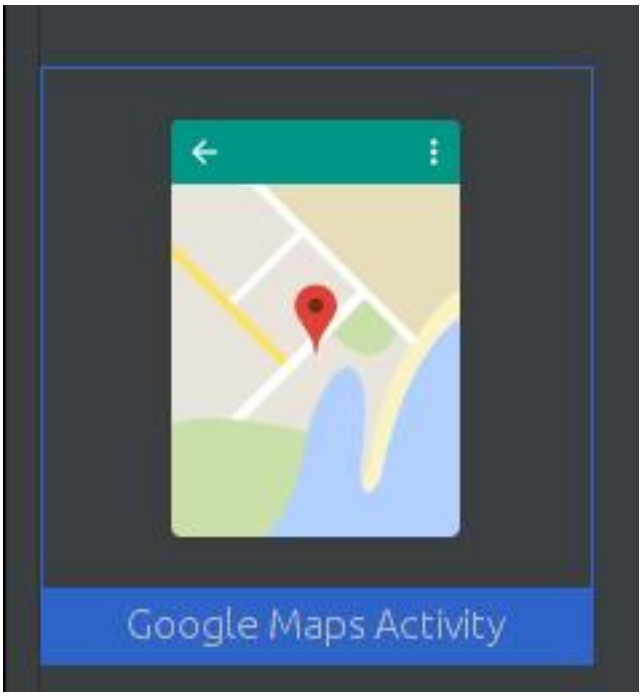

Figura 3.1: Google Maps Activity

Al iniciar este proyecto veremos, en el layout "google_maps_api.xml" unas indicaciones a seguir para que nuestra aplicación pueda acceder al servicio de google maps. Esto sirve para consegir las credenciales necesarias.
El proceso simplemente será registrarnos en la url que aparece y copiar la key que nos dan en el xml que se menciona.

3.2. Lectura del código QR

Si ya hemos importado la librería Zxing que hemos comentado antes, vamos a tener acceso a un nuevo tipo de intents que nos servirán para comenzar a escanear código con Barcode scan, estos son IntentIntegrator. Para lanzar la lectura del código haremos lo siguiente (nsotros hemos asociado esta acción a un botón de la app):

```
1 //Se instancia un objeto de la clase IntentIntegrator
2 IntentIntegrator scanIntegrator = new IntentIntegrator(this);
3 //Se procede con el proceso de escaneo
4 scanIntegrator.initiateScan();
```

Una vez escaneado el código QR debemos recoger los datos de algún sitio, pues esto lo hacemos igual que hacemos con un intent normal, es decir:

```
1     public void onActivityResult(int requestCode, int resultCode,
          Intent data) {
2         IntentResult scanResult = IntentIntegrator.
              parseActivityResult(requestCode, resultCode, data);
3         if ( data != null ) {
4             if(scanResult != null){
5                 String contentData = data.getStringExtra("
                      SCAN_RESULT");
6
7                 String[] datos = contentData.split("_");
8
9                 if(datos[0] != getString(R.string.latitud) ||
                      datos[2] != getString(R.string.longitud)) {
10                    Intent intent = new Intent(this, MapsActivity.
                          class);
11                    intent.putExtra(getString(R.string.latitud),
                          datos[1]);
12                    intent.putExtra(getString(R.string.longitud),
                          datos[3]);
13
14                    startActivity(intent);
15                }else
16                    Toast.makeText(getApplicationContext(),
                          getString(R.string.qr_irreconocible), Toast
                          .LENGTH_LONG).show();
17            }else
18                Toast.makeText(getApplicationContext(),getString(R
                      .string.qr_irreconocible), Toast.LENGTH_LONG).
                      show();
19        }else
20            Toast.makeText(getApplicationContext(), getString(R.
                  string.qr_inexistente), Toast.LENGTH_LONG).show();
21    }
```

Aquí ya hemos incluido algunos ajustes para ocasiones en las que no se haya leido ningún código o se haya leido un código que no tenga la forma adecuada. También se lanza la actividad de google maps con los extras latitud y longitud que tendremos que recoger en la nueva activity.

3.3. Ajustando la navegación

Ya tenemos a dónde queremos ir, solo nos falta comunicárselo a la aplicación yq eu esta haga uso de los sensores necesarios. Como hemos creado el proyecto con un google maps activity, ahora es cuando haremos uso de esa activity que nos han dado creada.
Lo único que tenemos que hacer es recoger del intent la latitud y la longitud que había-

mos leido y pasáserlos a nuestro mapa, esto es tan sencillo como cambiar en el método OnMapReady que nos han dado creado la localización a la que queremos ir (al crearse viene Sidney por defecto). Para ello hacemos:

```
mMap = googleMap;

// Add a marker in Sydney and move the camera
LatLng destino = new LatLng(mLatitud, mLongitud);
mMap.addMarker(new MarkerOptions().position(destino).title(
    getString(R.string.destino)));
mMap.moveCamera(CameraUpdateFactory.newLatLng(destino));
```

4. Librerias externas usadas

Aunque ya se ha comentado antes, debemos hacer incapié en el uso de la librería que hemos usado para el uso del lector de códigos QR, esta librería es Zxing que está disponible para cualquier usuario en `https://github.com/zxing/zxing`