
# #16. 地图应用


# 地图开发

- 随着移动互联网应用的迅速发展，利用智能手机提供的实时地理位置信息服务功能扩展出众多 LBS(Location Based Service) 应用，比如实时定位、导航、搜索周围好友、基于地理位置的信息推荐等。在android开发中地图和定位是很多软件不可或缺的内容，这些特色功能也给人们带来各种生活便利。

- API 会自动处理对 Google 地图服务器的访问、数据下载、地图显示以及对地图手势的响应。您还可以利用 API 调用向基础地图添加标记、多边形和叠层，以及更改特定地图区域的用户视图。这些对象可为地图位置提供附加信息，实现用户与地图的交互。

# Google API 版本

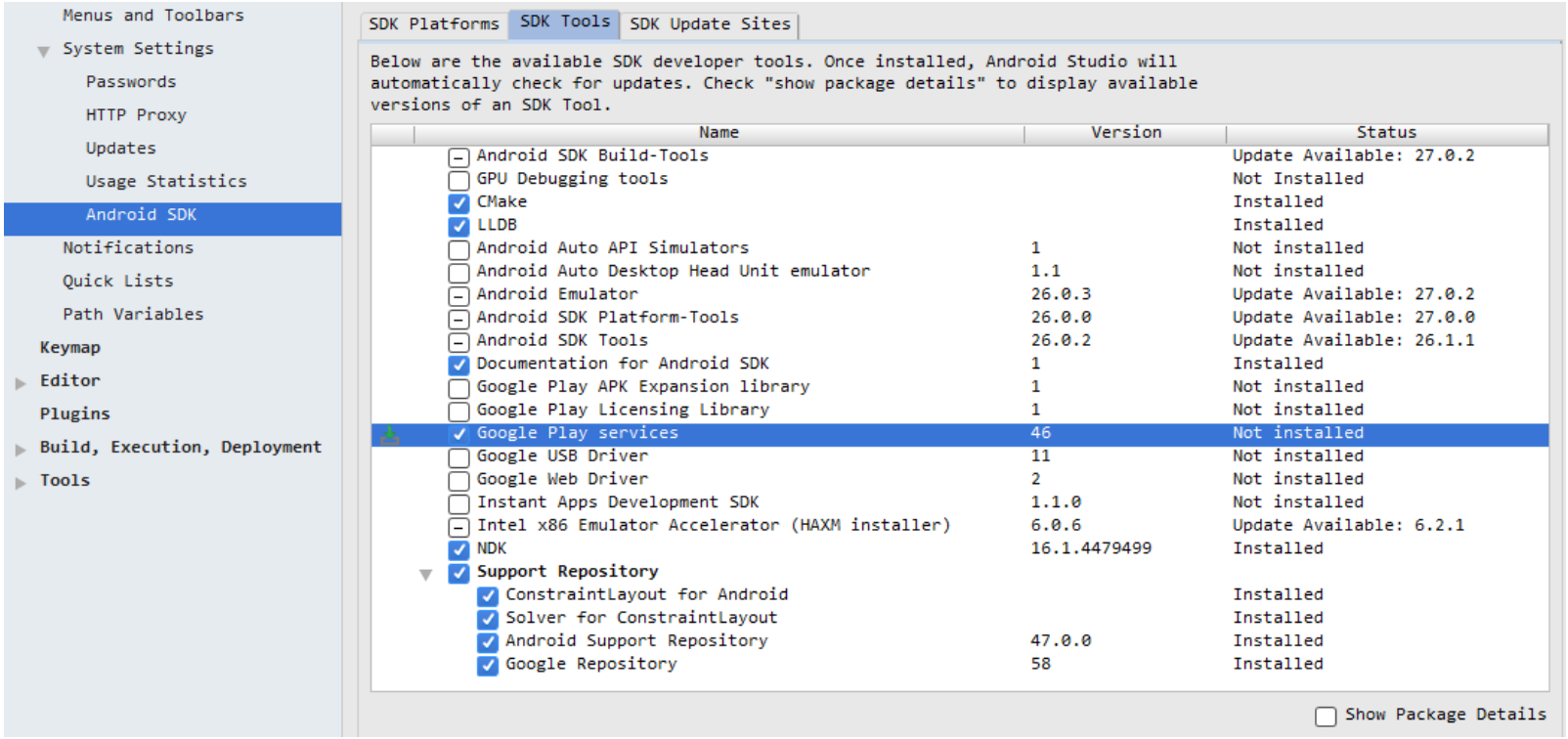
- Google Maps Android API v1已经在2012年12月3号被正式废弃。这意味着开发者无法再使用API V1获取地图密钥。
- Google在2012年12月为Android平台推出了Google Maps Android API v2，使用全新的设计方式，改良地图控件的画图效率，增加3D建筑物，并简化了绘图API。
- 当前Google官方推荐的API是Google Maps API v3，v3是不用密钥的，其是将网页加载到ANDROID的**WEBVIEW控件**中来实现。 v3旨在实现快速载入，尤其是在移动浏览器上。
- V3版本的全称是**Google Maps *Javascript* API v3** ，这时一个基于**javascript**的API，开发者通过webview控件加载地图。V2版本的全称是**Google Maps *Android* API v2** ，它是一个原生Android API，基于**Java**。本讲内容是基于Google Maps *Android* API v2的。

# 学习目标

- Hello Map
- 地图属性
- 用户与地图交互
- 位置信息 MyLocation
- 地图标记 Marker
- 添加形状

# Hello Map

- Google Maps Android API 包含在Google Play Service SDK中。首先，需要确保在SDK Manager中已下SDK Platform + Google APIs

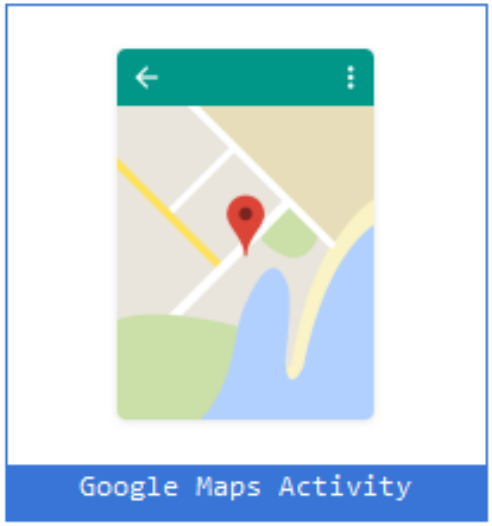


# Hello Map

- 在 Android/sdk/extras/google/google_play_services目录中，包含了Google服务相关的文档和实例代码。

# Hello Map

- 下面以一个简单的例子说明如何显示Google地图。
- **步骤1. 新建Google Maps项目。**
- 按照平常新建项目的步骤新建Google Maps项目，在“Add an activity to Mobile”对话框中选择 **Google Maps Activity**。



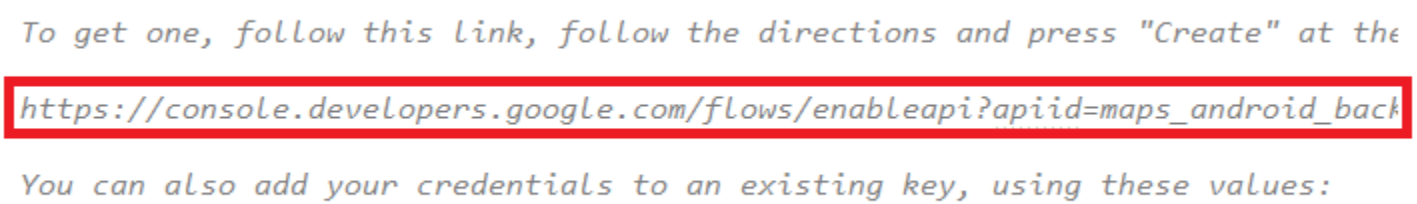

构建完成后，Android Studio 会在编辑器中打开 google_maps_api.xml 文件和 MapsActivity.java 文件

# Hello Map

**步骤2：申请地图密钥**

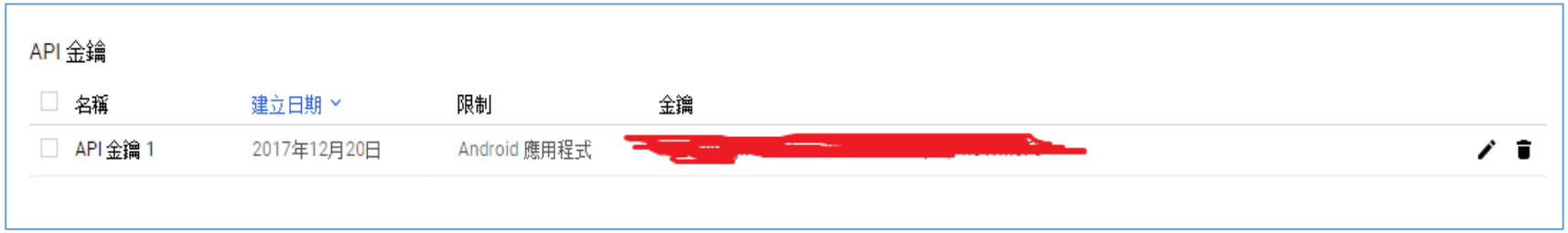
- 应用需要使用 API 密钥来访问 Google 地图服务器。密钥类型是 **Android 应用专用密钥**，这意味着该密钥只能为当前这个应用使用。

- `google_maps_api.xml` 文件包含有关在您尝试运行应用前获取 Google Maps API 密钥的说明。将文件中提供的链接复制到浏览器中，根据界面提示快速获取API密钥。



# Hello Map

按照网页上的提示操作，获取到应用的API 密钥，将其复制到google_maps_api.xml文件的相应位置。（获取密钥需要一个Google账号，因此需要提前申请Google账号）




```
<string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">
    [redacted]
</string>
```

# Hello Map

**步骤3：**检查Android Studio自动生成的相关配置文件

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="[redacted]" />
```

在gradle中，已经自动添加了对Google Play Service 的依赖

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:26.+'
    compile 'com.google.android.gms:play-services-maps:11.0.4'
    testCompile 'junit:junit:4.12'
}
```

# Hello Map

**步骤4：声明权限**

- 在manifest中定义了应用需要获取位置信息的权限

```
<!--
     The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
     Google Maps Android API v2, but you must specify either coarse or fine
     location permissions for the 'MyLocation' functionality.
-->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

- 如果目标平台是**Android 6.0(API 23) 以前**的版本，还需要申明外部存储器读写权限

```
<!-- External_STORAGE permission are optional for Android6.0 conwards. -->
<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE"
    android:maxSdkVersion="22"/>

<uses-permission
    android:name="android.permission.READ_EXTERNAL_STORAGE"
    android:maxSdkVersion="22"/>
```

- 地图应用需要使用网络，但是网络权限已经默认声明了，因此不需要在manifest中显式声明。

# Hello Map

配置好相关文件后，如何显示Google地图呢？Google使用地图对象时的关键类是 **GoogleMap** 类。 GoogleMap 在应用内为地图对象建模。

GoogleMap 自动处理下列操作：

- 连接到 Google 地图服务
- 下载地图图块。
- 在设备屏幕上显示图块。
- 显示如平移和缩放等各类控件。
- 通过移动和缩放地图响应平移和缩放手势

# Hello Map

地图将由 **MapFragment** 对象或 **MapView** 对象表示。

**MapFragment**

- MapFragment 是 Android Fragment 类的一个子类，用于在 Android Fragment 中放置地图。MapFragment 对象充当地图容器，并提供对 GoogleMap 对象的访问权。

- 与 View 不同，Fragment 表示的是 Activity 中的一种行为或用户界面的某一部分。 可以将多个 Fragment 组合在一个 Activity 中来构建多窗格 UI，以及在多个 Activity 中重复使用某个 Fragment。

**MapView**

- MapView 是 AndroidView 类的一个子类， 用于在View 中放置地图。MapView 与 MapFragment 很相似，它也充当地图容器，通过 GoogleMap 对象公开核心地图功能。

# Hello Map

**步骤5：在HelloMap中显示地图**

- 为了显示地图，首先需要向将处理地图的 Activity 添加 **Fragment** 对象。
- 最简单的实现方式是，向 Activity 的布局文件添加 <fragment> 元素。在该元素中，将 android:name属性设置为 "com.google.android.gms.maps.MapFragment"。 此操作会自动将 MapFragment 附加到 Activity。

```
<fragment xmlns:android="http://schemas.android.com/apk/res/and
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="lwk.hellomap.MapsActivity" />
```

# Hello Map

• 应用内使用地图，需要实现 **OnMapReadyCallback** 接口，并在 MapFragment 对象或 MapView对象上设置回调实例，这通过使用 **getMapAsync()** 设置 。

```
public class MapsActivity extends FragmentActivity implements OnMapReadyCallback
```

```
SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
        .findFragmentById(R.id.map);
mapFragment.getMapAsync(this);
```

• 使用 **onMapReady(GoogleMap)** 回调方法获取 GoogleMap 对象的句柄。 这个回调将在地图做好使用准备时触发。

```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Add a marker in Sydney and move the camera
    LatLng sydney = new LatLng(-34, 151);
    mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"))
    mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
}
```

# Hello Map

- 完成上述步骤后，就可以直接运行程序，在界面上显示出地图。
- 真机需要安装google play 商店,google play 服务作为支撑。（推荐使用 go谷歌安装器，无须root，一键安装google play 商店和服务）
- 如果使用模拟器运行程序，需要的模拟器映像具有的 Google API平台应基于**Android 4.2.2 或更高版本。** 此外，将模拟器设置为**x86 目标 AVD**。这可以改善您的模拟器使用体验。


# 地图属性

- Maps API 允许配置地图的初始状态，可以指定以下内容：
  - 摄像头位置，包括：位置、缩放比例、方位和倾斜角度。
  - 地图类型
  - 缩放按钮和/或指南针是否出现在屏幕上
  - 用户在操纵摄像头时可使用的手势

# 地图属性

- 如果已将地图添加到 Activity 的布局文件，可以通过 XML 配置地图的初始状态；如果是以编程方式添加地图，则可以编程方式进行配置。

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="lwk.hellomap.MapsActivity"
    map:cameraBearing="112.5"
    map:cameraTargetLat="-33.796923"
    map:cameraTargetLng="150.922433"
    map:cameraTilt="30"
    map:cameraZoom="13"
    map:uiCompass="false"
    map:uiRotateGestures="true"
    map:uiScrollGestures="false"
    map:uiTiltGestures="true"
    map:uiZoomControls="false"
    map:uiZoomGestures="true"
    />
```


- 如果是以编程方式添加地图，则可以编程方式进行配置。

```
//设置地图属性
GoogleMapOptions options = new GoogleMapOptions();
options.mapType(GoogleMap.MAP_TYPE_SATELLITE)
        .compassEnabled(false)
        .rotateGesturesEnabled(false)
        .tiltGesturesEnabled(false);
```

# 地图属性

- 在Hello Map中设置摄像头的位置和缩放大小

```
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Add a marker in Sydney and move the camera
      LatLng sydney = new LatLng(-34, 151);
      mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
      mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));

    //设置摄像头的位置和缩放
    LatLng cali = new LatLng(3.4383,-76.5161);
    CameraPosition cameraPosition = CameraPosition.builder()
            .target(cali)       //目的地
            .zoom(10)           //缩放级别
            .build();
    googleMap.moveCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));
}
```

# 地图属性

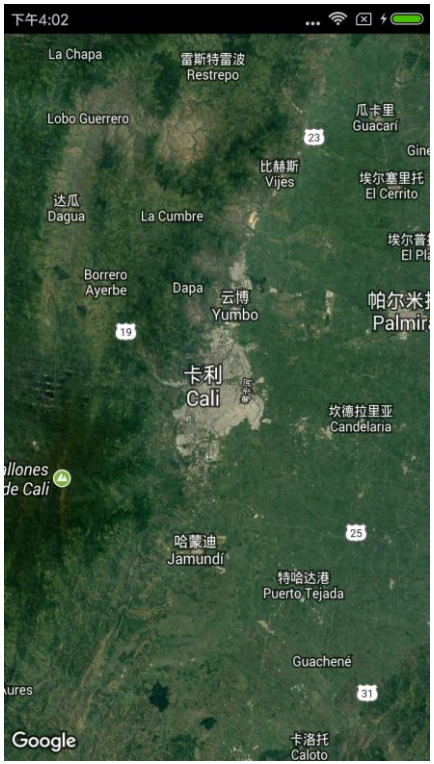
Google Maps Android API 提供了**四种地图**，**以及不显示任何地图的选项**：

- **Normal**：典型道路地图。显示道路、一些人造景观以及河流等重要的自然景观。 此外，还会显示道路和景观标签。
- **Hybrid**：添加了道路地图的卫星照片数据。 此外，还会显示道路 和景观标签。
- **Satellite**：卫星照片数据。 不显示道路和景观标签。
- **Terrain**：地形数据。 地图包含颜色、轮廓线和标签以及 透视阴影。此外，还会显示一些道路和标签。
- **None**：无图块。 地图将渲染为空网格，不加载任何图块。

# 地图属性

- 以下图像显示的是同一位置 normal 地图、hybrid 地图和 terrain 地图的比较


normal




hybrid


terrain

- 如需设置地图类型，可以调用 GoogleMap 对象的 **setMapType()** 方法，传递 GoogleMap 中定义的其中一个类型常量。或者直接在布局文件中设置。

# 用户与地图交互

Google地图还支持用户使用手势与地图进行交互。比如放大、缩小、移动、水平旋转、倾斜地图等。

zoom:

- *Double tap* to zoom
- *Press and Tap* to zoom out
- *Press and drag* to zoom in / out
- *Spread* to bring
- *Pinch* to zoom out

Displacement:

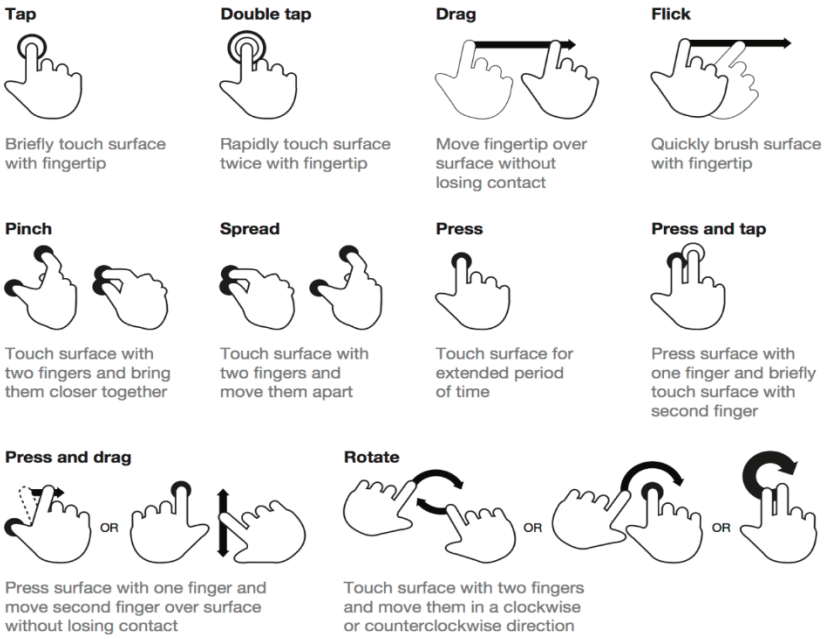
- *Drag* to continuously scroll the map omnidirectionally
- *Flick* to move in small portions omnidirectionally map

Inclination:

- *Drag x 2* fingers vertically. Up tilts up 90 ° and down declines to 0 °

Rotation:

- *Rotate* to rotate the map for or against clockwise.



# 用户与地图交互

可以在代码中通过UiSettings类设置用户手势控制的相关属性，其包括如下几个函数。

- setZoomGesturesEnabled(boolean) 设置是否可以放大/缩小地图
- setScrollGesturesEnabled(boolean): 设置是否可以滑动地图
- setTiltGesturesEnabled(boolean): 设置是否可以倾斜地图
- setRotateGesturesEnabled(boolean) 设置是否可以旋转地图
- setAllGesturesEnabled(boolean) 设置是否可以使用所有手势控制

```
//设置手势控制
UiSettings uiSettings = googleMap.getUiSettings();
uiSettings.setAllGesturesEnabled(false);              //设置所有手势控制不可用
```

# 位置信息 My Location

• Google Maps API 提供给 Android 设备的位置数据包括设备的当前位置（结合使用多种技术确定）、移动的方向和方式，以及设备移动范围是否覆盖预定义的地理边界（或称地理围栏）。

• Google Map API 通过**My Location** 层提供了一种在地图上显示设备位置的简单方式。它并不提供数据。

• 在启用 My Location 层之前，必须确保具备所需的**运行时位置权限**。即在程序运行时动态检查和申请位置权限。

```
//my location
if(ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
        == PackageManager.PERMISSION_GRANTED){
    googleMap.setMyLocationEnabled(true);
}else{
    //show error
}
googleMap.getUiSettings().setZoomControlsEnabled(true);
```

# 位置信息 My Location

- 在HelloMap中利用支持库检查权限，然后启用 My Location 层
- 启用了My Location 层时，**My Location 按钮**会出现在地图的右上角。 当用户点击该按钮时，摄像头将设备的当前位置显示为地图的中心。


# 位置信息 My Location

• 我们还可以使用GPS传感器或者Google Play services Location API 获取当前位置。

```
@Override
public void onLocationChanged(Location location) {
    latLng = new LatLng(location.getLatitude(),location.getLongitude());
    showLocation(latLng);
}
```

• 获取Google Play services Location API 返回的Location对象，然后将其转化为**LatLng对象**才能够显示在地图上。此外，还需要将摄像头移动到**LatLng对象**所在位置才能正确显示.

Google Play services Location API 使用说明：https://developer.android.com/training/location/index.html

# 位置信息 My Location

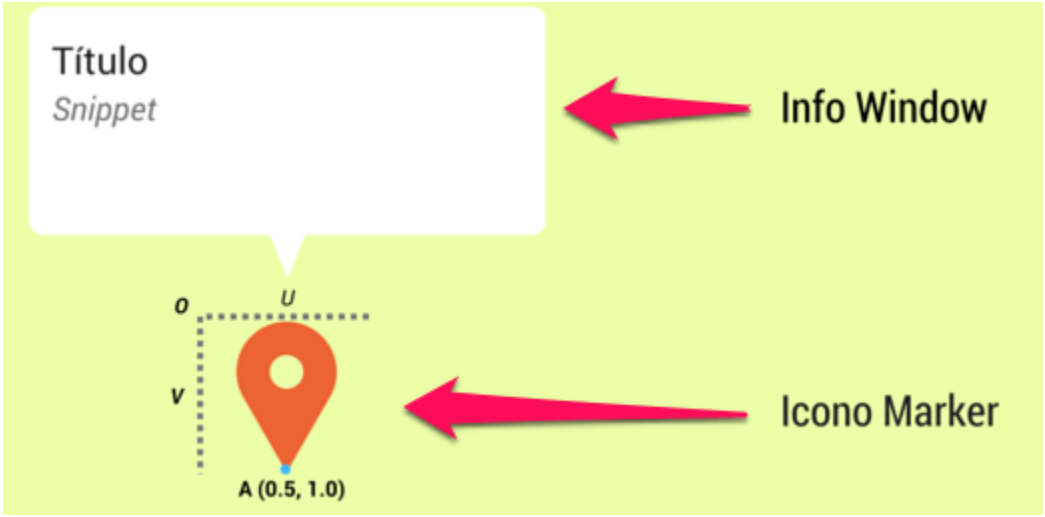
- 在HelloMap中使用GPS传感器获取当前位置，使用Toast显示经纬度并将摄像头移动到当前位置。

```
@Override
public void onLocationChanged(Location location) {
    latLng = new LatLng(location.getLatitude(),location.getLongitude());
    showLocation(latLng);
}
```

```
private void showLocation(LatLng latlng){
    if(latlng != null){
        Toast.makeText(MapsActivity.this, "经度：" + latlng.longitude + '\n' + "纬度："
        + latlng.latitude, Toast.LENGTH_LONG).show();
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latlng));
    }else{
        Toast.makeText(MapsActivity.this,"无法获得位置",Toast.LENGTH_SHORT).show();
    }
}
```

# 地图标记 Marker

- **标记**用于标识地图上的单个位置。标记包括位置、标记图标、信息窗口、标题、详细信息等内容。



# 地图标记 Marker

标记是 **Marker** 类型的对象，通过 GoogleMap.addMarker(markerOptions) 方法向地图添加。

```
// Add a marker in Sydney and move the camera
// 标记
LatLng sydney = new LatLng(-34, 151);
mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
```

# 地图标记 Marker

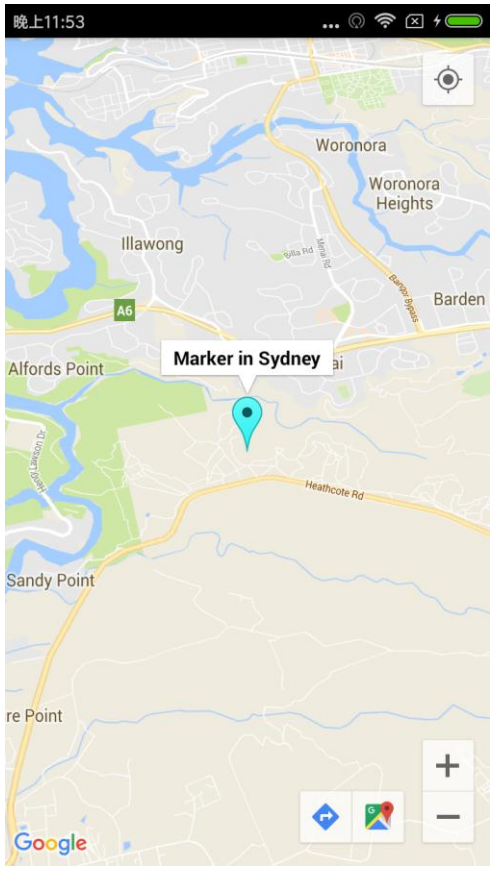
开发者可以根据程序需求，自定义标记的相关属性。

- 标记的颜色
- 标记的透明度
- 标记的图像（图标）
- 将标记平面化
- 旋转标记

# 地图标记 Marker

改变标记的颜色需要使用defaultMarker（ ）函数，传入的参数就是标记的颜色。其包括如下已定义的颜色。

```
LatLng sydney = new LatLng(-34, 151);
mMap.addMarker(new MarkerOptions().position(sydney)
        .title("Marker in Sydney"))
        //设置标记颜色
        .setIcon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_CYAN));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
```

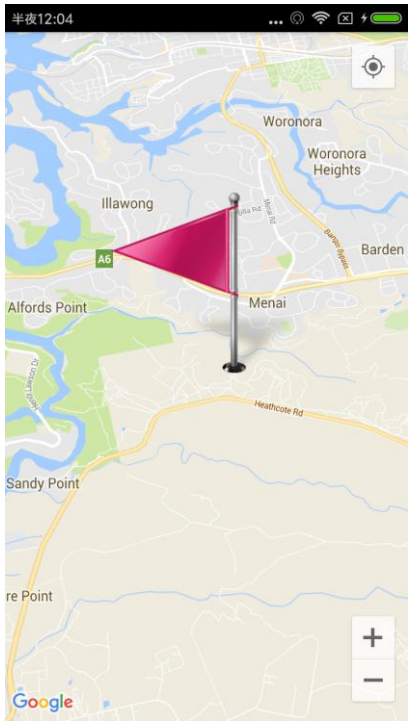



- float HUE_AZURE
- float HUE_BLUE
- float HUE_CYAN
- float HUE_GREEN
- float HUE_MAGENTA
- float HUE_ORANGE
- float HUE_RED
- float HUE_ROSE
- float HUE_VIOLET
- float HUE_YELLOW

# 地图标记 Marker

更改标记的图片需要将目标图片导入到项目中，在icon中设置显示图标

```
LatLng sydney = new LatLng(-34, 151);
mMap.addMarker(new MarkerOptions().position(sydney)
        .title("Marker in Sydney"))
        .setIcon(BitmapDescriptorFactory.fromResource(R.mipmap.marker));
```



# 地图标记 Marker

- 标记设计为具有交互能力。 它们默认接收 click 事件，通常与事件侦听器联用以调出**信息窗口**。

- 默认情况下，当用户点按标记时，地图工具栏出现在地图右下角，让用户可以快速访问 Google 地图移动应用。

```
mMap.addMarker(new MarkerOptions().position(sydney)
        .title("Marker in Sydney"))
        //设置标记颜色
          .setIcon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_CYAN));
        //自定义图片设置标记
        .setIcon(BitmapDescriptorFactory.fromResource(R.mipmap.marker));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
```

# 地图标记 Marker

此外，还可以使用setDraggable()函数设置标记可拖动。

```
mMap.addMarker(new MarkerOptions().position(sydney)
        .title("Marker in Sydney"))
        //设置可拖动
        .setDraggable(true);
```

# 地图标记 Marker

- **信息窗口**在标记上方的弹出窗口中显示文本或图像。 信息窗口始终固定在标记上。 其默认行为是在用户点按标记时显示。
- 添加信息窗口最简单的方法是设置相应标记的 title() 方法和setSnippet() 方法。

```
mMap.addMarker(new MarkerOptions().position(sydney)
        .title("Marker in Sydney"))
        .setSnippet("Population: *** ");
```

# 地图标记 Marker

- 在用户点按标记时，通过信息窗口向其显示信息。 一次只能显示一个信息窗口。 如果用户点击另一个标记，将隐藏当前信息窗口，并显示新的信息窗口。
- 可以通过对目标标记调用 **showInfoWindow()** 以编程方式显示信息窗口,调用 **hideInfoWindow()** 隐藏信息窗口。

```
Marker marker = mMap.addMarker(new MarkerOptions().position(sydney)
        .title("Marker in Sydney"));

marker.showInfoWindow();
marker.hideInfoWindow();
```

# 地图标记 Marker

在HelloMap中给当前位置增加标记。

```
//标记的当前位置显示函数
private void showLocation(LatLng latlng){
    if(latlng != null){
        mMap.addMarker(new MarkerOptions()
                .position(latlng)
                .title("当前位置"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latlng));
    }else{
    }
}
```

# 地图形状

Google Maps API for Android 提供了一些简单的方法,向地图添加形状。包括：

- **Polyline** 是一系列相连的线段，可组成您想要的任何形状，并可用于在地图上标记路径和路线
- **Polygon** 是一种封闭形状，可用于在地图上标记区域
- **Circle** 是投影在绘制于地图上的地球表面上地理位置准确的圆圈

# 地图形状

以Polyline为例，说明向地图添加形状的方法：

- Polyline 对象包含一组LatLng 位置，它创建的一系列线段以有序方式将这些位置连接起来。

- 如需创建多段线，首先需要创建一个 **PolylineOptions** 对象并为其添加位置点，位置点以LatLng 对象形式表示。 然后 Google Maps API 按照向PolylineOptions对象添加点的顺序在各点之间绘制线段。

# 地图形状

下面这段代码说明了如何向地图添加矩形：

```
//向地图添加矩形
PolylineOptions sudamericaRect = new PolylineOptions()
        .add(new LatLng(12.897489,-82.441406))  //P1
        .add(new LatLng(12.897489,-32.167969))  //P2
        .add(new LatLng(-55.37911,-32.167969))  //P3
        .add(new LatLng(-55.37911,-82.441406))  //P4
        .add(new LatLng(12.897489,-82.441406))  //P1
        .color(Color.parseColor("#f44336"));  //Red 500
Polyline polyline = mMap.addPolyline(sudamericaRect);
mMap.moveCamera(CameraUpdateFactory.newLatLng(new LatLng(-20.96144,-61.347656)));
```


参考资料:

官网：https://developers.google.com/maps/documentation/android-api/

Tutorial：http://www.hermosaprogramacion.com/2016/05/google-maps-android-api-v2/

# Google地图应用 V1 to V2

Google 在2012年发布了Google Maps API v2，其与v1有很大的不同。

- 设置地图的相关属性

  v1中设置地图的相关属性需要通过**MapController**设置，MapController是MapView的控制器，可以控制MapView的显示中心和缩放级别等功能

- 增加标记

  在v1中添加标记需要使用**Overlay**。Overlay是一个可显示于地图之上的可绘制的对象。

- 定义地图上的位置

  在v1中需要将经纬度封装在**GeoPoint**类对象中。

- 等等。

# 百度地图

- 百度地图 Android SDK是一套基于Android 2.3及以上版本设备的应用程序接口。 您可以使用该套 SDK开发适用于Android系统移动设备的地图应用，通过调用地图SDK接口，您可以轻松访问百度地图服务和数据，构建功能丰富、交互性强的地图类应用程序。
- 自v4.0起，适配Android Wear，支持Android穿戴设备，新增室内图相关功能。
- 2017年11月10日V4.5.2发布

# 百度地图

- 百度地图SDK提供实现丰富的LBS功能：

**地图**：提供地图的展示和缩放、平移、旋转、改变视角等地图操作；
**室内图**：提供展示公众建筑物室内地图的展示功能；
**Android Wear**：支持Android穿戴设备；
**POI检索**：可根据关键字，对POI数据进行周边、区域和城市内三种检索；
**室内POI检索**：支持设置城市和当前建筑物的室内POI检索；
**地理编码**：提供地理坐标和地址之间相互转换的能力；
**线路规划**：支持公交信息查询、公交换乘查询、驾车线路规划和步行路径检索；
**覆盖物**：提供多种地图覆盖物（自定义标注、几何图形、文字绘制、地形图图层、热力图图层等），满足开发者的各种需求；
**定位**：采用多种定位模式，使用定位SDK获取位置信息，使用地图SDK我的位置图层进行位置展示；

百度地图API地址：http://lbsyun.baidu.com/


Questions?