# iOS原生地图开发指南

目录[-]

 [iOS原生地图开发详解](http://my.oschina.net/u/2340880/blog/415360#OSC_h2_1)

 [一、初始化地图视图与相关属性方法介绍](http://my.oschina.net/u/2340880/blog/415360#OSC_h3_2)

 [1、初始化地图视图](http://my.oschina.net/u/2340880/blog/415360#OSC_h4_3)

 [2、系统提供的三种地图样式](http://my.oschina.net/u/2340880/blog/415360#OSC_h4_4)

 [3、设置地图的中心和比例尺](http://my.oschina.net/u/2340880/blog/415360#OSC_h4_5)

 [4、坐标转换方法](http://my.oschina.net/u/2340880/blog/415360#OSC_h4_6)

 [5、MKMapView常用方法和属性](http://my.oschina.net/u/2340880/blog/415360#OSC_h4_7)

 [二、MKMapViewDelegate相关方法解读](http://my.oschina.net/u/2340880/blog/415360#OSC_h3_8)

**iOS原生地图开发详解**

在上一篇博客中:[http://my.oschina.net/u/2340880/blog/414760](http://my.oschina.net/u/2340880/blog/414760" \t "_self)。对iOS中的定位服务进行了详细的介绍与参数说明，在开发中，地位服务往往与地图框架结合使用，这篇博客主要对iOS官方的地图框架MapKit.framework进行介绍。

**一、初始化地图视图与相关属性方法介绍**

**1、初始化地图视图**

地图视图的展示依赖于MKMapView这个类，这个类继承于UIView，因此和其他View的使用方法类似。在我们需要展现地图的地方：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4  5 | - (void)viewDidLoad {      [super viewDidLoad];      MKMapView \* mapView =[[MKMapView alloc]initWithFrame:self.view.frame];      [self.view addSubview:mapView];  } |

运行发现，一张世界地图就在我们的设备上了，apple内置的地图数据是由高德提供的。

**2、系统提供的三种地图样式**

可以通过MKMapView的mapType这个属性设置地图的模式：

@property (nonatomic) MKMapType mapType;

枚举如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4  5 | typedef NS\_ENUM(NSUInteger, MKMapType) {      MKMapTypeStandard = 0,//标准式的行政地图(会显示城市，街道等)      MKMapTypeSatellite,//标准的卫星地图      MKMapTypeHybrid//混合地图(在卫星图上显示街道等名称)  }; |

**3、设置地图的中心和比例尺**

在百度地图等第三方地图服务的SDK中，都会提供一个类似zoomLevel比例尺的属性。通过官方的API设置这个属性有些麻烦，但是也更加灵活。首先，设置地图的中心位置和比例尺是通过region这个属性实现的。region结构体如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4 | typedef struct {      CLLocationCoordinate2D center;//地图中心的经纬度      MKCoordinateSpan span;//地图显示的经纬度范围  } MKCoordinateRegion; |

这个结构体中包含了两个结构体，其中CLLocationCoordinate2D很好理解，就是简单的经纬度，解释如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4 | typedef struct {      CLLocationDegrees latitude;//纬度，北纬为正，南纬为负      CLLocationDegrees longitude;//经度，东经为正，西经为负  } CLLocationCoordinate2D; |

MKCoordinateSpan这个结构体比较复杂，如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4 | typedef struct {      CLLocationDegrees latitudeDelta;//纬度范围      CLLocationDegrees longitudeDelta;//经度范围  } MKCoordinateSpan; |

这个结构体定义的应该是一个范围，因为北纬南纬加起来180°，所以纬度范围的取值应为0-180。同理，经度范围的取值范围为0-360。

通过上面的介绍，我们举个例子，将北京市设为地图的中心区域，并且比例设置为显示北京大小。通过百度，首先知道北京市界的地理坐标为：北纬39”26’至41”03’，东经115”25’至 117”30’。北京市区坐标为：北纬39.9”，东经116. 3”。代码如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1 | mapView.region=MKCoordinateRegionMake(CLLocationCoordinate2DMake(39.26, 116.3), MKCoordinateSpanMake(1.8, 2.05)); |

运行后可以看到，北京市基本上是在地图中心的，效果如下：

[](http://static.oschina.net/uploads/space/2015/0515/105726_Qmuq_2340880.png)

注意：MKCoordinateSpan的显示范围是取决于大的一边的，比如如果我们这样写：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1 | MKCoordinateSpanMake(1.8, 360); |

最后依然会显示整个世界地图。

- (void)setRegion:(MKCoordinateRegion)region animated:(BOOL)animated;

这个方法可以在设置后给地图加上动画效果

@property (nonatomic) CLLocationCoordinate2D centerCoordinate;

设置地图的中心点位置

- (void)setCenterCoordinate:(CLLocationCoordinate2D)coordinate animated:(BOOL)animated;

设置地图的中心点位置，并附带动画效果

**4、坐标转换方法**

- (CGPoint)convertCoordinate:(CLLocationCoordinate2D)coordinate toPointToView:(UIView \*)view;

将经纬度转换为视图上的坐标

- (CLLocationCoordinate2D)convertPoint:(CGPoint)point toCoordinateFromView:(UIView \*)view;

将视图上的坐标转换为经纬度

- (CGRect)convertRegion:(MKCoordinateRegion)region toRectToView:(UIView \*)view;

将地理显示的区域转换为视图上的坐标区域

- (MKCoordinateRegion)convertRect:(CGRect)rect toRegionFromView:(UIView \*)view;  
将视图上的坐标区域转换为地理区域

**5、MKMapView常用方法和属性**

@property (nonatomic, getter=isZoomEnabled) BOOL zoomEnabled;

设置是否允许捏合手势进行地图缩放

@property (nonatomic, getter=isScrollEnabled) BOOL scrollEnabled;

设置是否允许滑动

@property (nonatomic, getter=isRotateEnabled) BOOL rotateEnabled;

设置是否允许旋转地图

@property (nonatomic, getter=isPitchEnabled) BOOL pitchEnabled;

设置是否支持3D效果

@property (nonatomic) BOOL showsPointsOfInterest;

设置是否显示兴趣点，例如学校，医院等

@property (nonatomic) BOOL showsBuildings;

设置是否显示建筑物轮廓，只在标准的地图中有效

@property (nonatomic) BOOL showsUserLocation;

是否显示用户位置

@property (nonatomic) MKUserTrackingMode userTrackingMode;

- (void)setUserTrackingMode:(MKUserTrackingMode)mode animated:(BOOL)animated;

设置更新用户位置的模式,当显示用户位置设置为YES，这个方法也设置了后，地图框架为我们直接集成了定位，地图上就会显示我们的位置，模式的枚举如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4  5 | typedef NS\_ENUM(NSInteger, MKUserTrackingMode) {      MKUserTrackingModeNone = 0, // 不跟踪用户位置      MKUserTrackingModeFollow, // 跟踪用户位置      MKUserTrackingModeFollowWithHeading, // 当方向改变时跟踪用户位置  } |

@property (nonatomic, readonly) MKUserLocation \*userLocation;

获取用户位置的标注

@property (nonatomic, readonly, getter=isUserLocationVisible) BOOL userLocationVisible;

获取用户位置是否可见

- (void)addAnnotation:(id <MKAnnotation>)annotation;

在地图上添加一个标注

- (void)addAnnotations:(NSArray \*)annotations;  
在地图上添加一组标注  
- (void)removeAnnotation:(id <MKAnnotation>)annotation;

移除一个标注

- (void)removeAnnotations:(NSArray \*)annotations;

移除一组标注

@property (nonatomic, readonly) NSArray \*annotations;

获取所有标注数组

- (MKAnnotationView \*)viewForAnnotation:(id <MKAnnotation>)annotation;

获取标注的视图

- (MKAnnotationView \*)dequeueReusableAnnotationViewWithIdentifier:(NSString \*)identifier;

获取复用的标注

- (void)selectAnnotation:(id <MKAnnotation>)annotation animated:(BOOL)animated;

选中一个标注

- (void)deselectAnnotation:(id <MKAnnotation>)annotation animated:(BOOL)animated;

取消选中一个标注

@property (nonatomic, copy) NSArray \*selectedAnnotations;

选中标注的数组

- (void)addOverlay:(id <MKOverlay>)overlay level:(MKOverlayLevel)level;

添加一个地图覆盖物，level是设置一个层级，枚举如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4 | typedef NS\_ENUM(NSInteger, MKOverlayLevel) {      MKOverlayLevelAboveRoads = 0, // 覆盖物位于道路之上      MKOverlayLevelAboveLabels//覆盖物位于标签之上  } |

- (void)addOverlays:(NSArray \*)overlays level:(MKOverlayLevel)level;

添加一组地图覆盖物

- (void)removeOverlay:(id <MKOverlay>)overlay;

移除一个地图覆盖物

- (void)removeOverlays:(NSArray \*)overlays;

移除一组地图覆盖物

- (void)insertOverlay:(id <MKOverlay>)overlay atIndex:(NSUInteger)index level:(MKOverlayLevel)level;

在索引处插入一个地图覆盖物

- (void)insertOverlay:(id <MKOverlay>)overlay aboveOverlay:(id <MKOverlay>)sibling;

将一个地图覆盖物插在到某个覆盖物之上

- (void)insertOverlay:(id <MKOverlay>)overlay belowOverlay:(id <MKOverlay>)sibling;

将一个地图覆盖物插入到某个覆盖物之下

- (void)exchangeOverlay:(id <MKOverlay>)overlay1 withOverlay:(id <MKOverlay>)overlay2;

替换一个地图覆盖物

@property (nonatomic, readonly) NSArray \*overlays;

地图覆盖物数组

- (NSArray \*)overlaysInLevel:(MKOverlayLevel)level;

层级属性下的东土覆盖物数组

**二、MKMapViewDelegate相关方法解读**

- (void)mapView:(MKMapView \*)mapView regionWillChangeAnimated:(BOOL)animated;

地图显示位置将要改变时调用的方法

- (void)mapView:(MKMapView \*)mapView regionDidChangeAnimated:(BOOL)animated;

地图显示位置已经改变时调用的方法

- (void)mapViewWillStartLoadingMap:(MKMapView \*)mapView;

地图将要加载时调用的方法

- (void)mapViewDidFinishLoadingMap:(MKMapView \*)mapView;

地图加载完成时执行的方法

- (void)mapViewDidFailLoadingMap:(MKMapView \*)mapView withError:(NSError \*)error;

地图加载失败时执行的方法

- (MKAnnotationView \*)mapView:(MKMapView \*)mapView viewForAnnotation:(id <MKAnnotation>)annotation;

渲染标注视图时调用的方法，可以通过这个方法自定义标注视图

- (void)mapView:(MKMapView \*)mapView didAddAnnotationViews:(NSArray \*)views;

标注添加完成后调用的方法

- (void)mapView:(MKMapView \*)mapView didSelectAnnotationView:(MKAnnotationView \*)view;

选中标注时调用的方法

- (void)mapView:(MKMapView \*)mapView didDeselectAnnotationView:(MKAnnotationView \*)view;

取消选中标注时调用的方法

- (void)mapViewWillStartLocatingUser:(MKMapView \*)mapView;

将要开始定位用户位置时调用的方法

- (void)mapViewDidStopLocatingUser:(MKMapView \*)mapView;

停止定位用户位置时调用的方法

- (void)mapView:(MKMapView \*)mapView didUpdateUserLocation:(MKUserLocation \*)userLocation;

更新用户位置时调用的方法

- (void)mapView:(MKMapView \*)mapView didFailToLocateUserWithError:(NSError \*)error;

更新用户位置失败时调用的方法

- (void)mapView:(MKMapView \*)mapView annotationView:(MKAnnotationView \*)view didChangeDragState:(MKAnnotationViewDragState)newState   
   fromOldState:(MKAnnotationViewDragState)oldState;

标注拖动状态改变调用的方法，MKAnnotationViewDragState的枚举如下：

[?](http://my.oschina.net/u/2340880/blog/415360)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | typedef NS\_ENUM(NSUInteger, MKAnnotationViewDragState) {      MKAnnotationViewDragStateNone = 0,      // 初始状态      MKAnnotationViewDragStateStarting,      // 开始拖动时      MKAnnotationViewDragStateDragging,      // 正在拖动      MKAnnotationViewDragStateCanceling,     // 取消拖动      MKAnnotationViewDragStateEnding         // 结束拖动  }; |

- (void)mapView:(MKMapView \*)mapView didChangeUserTrackingMode:(MKUserTrackingMode)mode animated:(BOOL)animated;

定位用户位置模式改变时调用的方法

- (MKOverlayView \*)mapView:(MKMapView \*)mapView viewForOverlay:(id <MKOverlay>)overlay;

渲染覆盖物视图时调用的方法，可以自定义覆盖物视图

- (void)mapView:(MKMapView \*)mapView didAddOverlayViews:(NSArray \*)overlayViews;

添加完成覆盖物数组执行的方法