**iOS原生地图开发进阶——使用导航和附近兴趣点检索**

摘要 大部分需要导航功能的APP都采用的第三方的数据，百度地图的SDK就非常简单易用，但是如果APP要做到国际化，百度SDK的支持和mapKit相比就逊色许多。

[iOS](http://www.oschina.net/search?scope=blog&q=iOS) [地图](http://www.oschina.net/search?scope=blog&q=%E5%9C%B0%E5%9B%BE) [地图导航](http://www.oschina.net/search?scope=blog&q=%E5%9C%B0%E5%9B%BE%E5%AF%BC%E8%88%AA) [附近检索](http://www.oschina.net/search?scope=blog&q=%E9%99%84%E8%BF%91%E6%A3%80%E7%B4%A2)

目录[-]

 [iOS原生地图开发进阶——使用导航和附近兴趣点检索](http://my.oschina.net/u/2340880/blog/416158#OSC_h2_1)

 [一、线路导航](http://my.oschina.net/u/2340880/blog/416158#OSC_h3_2)

 [1、从几个类的关系说起](http://my.oschina.net/u/2340880/blog/416158#OSC_h4_3)

 [2、进行线路导航](http://my.oschina.net/u/2340880/blog/416158#OSC_h4_4)

 [二、附近兴趣点检索](http://my.oschina.net/u/2340880/blog/416158#OSC_h3_5)

## iOS原生地图开发进阶——使用导航和附近兴趣点检索

iOS中的mapKit框架对国际化的支持非常出色。在前些篇博客中，对这个地图框架的基础用法和标注与覆盖物的添加进行了详细的介绍，这篇博客将介绍两个更加实用的功能的开发：线路导航与兴趣点搜索。前几篇博客的链接如下：

地图基础用法详解：[http://my.oschina.net/u/2340880/blog/415360](http://my.oschina.net/u/2340880/blog/415360" \t "_self)。

添加大头针与自定义标注：[http://my.oschina.net/u/2340880/blog/415441](http://my.oschina.net/u/2340880/blog/415441" \t "_self)。

添加地图覆盖物：[http://my.oschina.net/u/2340880/blog/415611](http://my.oschina.net/u/2340880/blog/415611" \t "_self)。

### 一、线路导航

#### 1、从几个类的关系说起

(1)MKPlacemark

一个地点信息类，如下：

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | @interface MKPlacemark : CLPlacemark <MKAnnotation>  //初始化方法，通过给定一个经纬度和地点信息字典  - (instancetype)initWithCoordinate:(CLLocationCoordinate2D)coordinate                   addressDictionary:(NSDictionary \*)addressDictionary;  //国家编码  @property (nonatomic, readonly) NSString \*countryCode;  @end |

(2)MKMapItem

地点节点类，包含此节点的许多地点信息，如下：

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | @interface MKMapItem : NSObject  //当前节点的地点信息对象  @property (nonatomic, readonly) MKPlacemark \*placemark;  //是否是当前位置  @property (nonatomic, readonly) BOOL isCurrentLocation;  //节点名称  @property (nonatomic, copy) NSString \*name;  //电话号码  @property (nonatomic, copy) NSString \*phoneNumber;  //网址  @property (nonatomic, strong) NSURL \*url;  //将当前位置创建为节点  + (MKMapItem \*)mapItemForCurrentLocation;  //由一个位置信息创建节点  - (instancetype)initWithPlacemark:(MKPlacemark \*)placemark;    @end |

(3)MKDirectionsRequest

导航请求类

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | @interface MKDirectionsRequest : NSObject  //起点节点  - (MKMapItem \*)source NS\_AVAILABLE(10\_9, 6\_0);  - (void)setSource:(MKMapItem \*)source NS\_AVAILABLE(10\_9, 7\_0);  //目的地节点  - (MKMapItem \*)destination NS\_AVAILABLE(10\_9, 6\_0);  - (void)setDestination:(MKMapItem \*)destination NS\_AVAILABLE(10\_9, 7\_0);    @end |

这个类还有一些扩展的设置属性：

@property (nonatomic) MKDirectionsTransportType transportType;

设置路线检索类型，枚举如下：

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5 | typedef NS\_OPTIONS(NSUInteger, MKDirectionsTransportType) {      MKDirectionsTransportTypeAutomobile     = 1 << 0,//适合驾车时导航      MKDirectionsTransportTypeWalking        = 1 << 1,//适合步行时导航      MKDirectionsTransportTypeAny            = 0x0FFFFFFF//任何情况  }; |

@property (nonatomic) BOOL requestsAlternateRoutes;

设置是否搜索多条线路

@property (nonatomic, copy) NSDate \*departureDate;

设置出发日期

@property (nonatomic, copy) NSDate \*arrivalDate;

设置到达日期

（4）MKDirections

从apple服务器获取数据的连接类

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | @interface MKDirections : NSObject  //初始化方法  - (instancetype)initWithRequest:(MKDirectionsRequest \*)request NS\_DESIGNATED\_INITIALIZER;  //开始计算线路信息  - (void)calculateDirectionsWithCompletionHandler:(MKDirectionsHandler)completionHandler;  //开始计算时间信息  - (void)calculateETAWithCompletionHandler:(MKETAHandler)completionHandler;  //取消  - (void)cancel;  //是否正在计算  @property (nonatomic, readonly, getter=isCalculating) BOOL calculating;  @end |

(5)MKDirectionsResponse

线路信息结果类

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5 | @interface MKDirectionsResponse : NSObject  @property (nonatomic, readonly) MKMapItem \*source;//起点  @property (nonatomic, readonly) MKMapItem \*destination;//终点  @property (nonatomic, readonly) NSArray \*routes; //线路规划数组  @end |

(6)MKETResponse

时间信息结果类

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | @interface MKETAResponse : NSObject  @property (nonatomic, readonly) MKMapItem \*source;//起点  @property (nonatomic, readonly) MKMapItem \*destination;//终点  @property (nonatomic, readonly) NSTimeInterval expectedTravelTime;//耗时    @end |

(7)MKRoute

线路信息类，导航的线路结果是这个类型的对象

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | @interface MKRoute : NSObject    @property (nonatomic, readonly) NSString \*name; //线路名称  @property (nonatomic, readonly) NSArray \*advisoryNotices; //注意事项  @property (nonatomic, readonly) CLLocationDistance distance; //距离  @property (nonatomic, readonly) NSTimeInterval expectedTravelTime;//耗时  @property (nonatomic, readonly) MKDirectionsTransportType transportType; //检索的类型    @property (nonatomic, readonly) MKPolyline \*polyline; // 线路覆盖物    @property (nonatomic, readonly) NSArray \*steps; // 线路详情数组    @end |

（8）MKRouteStep

线路详情信息类，线路中每一步的信息都是这个类的对象

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | @interface MKRouteStep : NSObject    @property (nonatomic, readonly) NSString \*instructions; // 节点信息  @property (nonatomic, readonly) NSString \*notice; // 注意事项    @property (nonatomic, readonly) MKPolyline \*polyline; //线路覆盖物    @property (nonatomic, readonly) CLLocationDistance distance; // 距离    @property (nonatomic, readonly) MKDirectionsTransportType transportType; // 导航类型    @end |

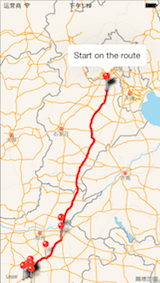
看到上面如此多的类，你可能会觉得一头雾水，那么不用着急，类虽然繁杂，但他们之间的逻辑非常清晰，下面就通过一个例子来进行线路导航。

#### 2、进行线路导航

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67 | - (void)viewDidLoad {      [super viewDidLoad];      //地图初始化设置      mapView =[[MKMapView alloc]initWithFrame:self.view.frame];      mapView.region=MKCoordinateRegionMake(CLLocationCoordinate2DMake(39.26, 116.3), MKCoordinateSpanMake(5, 5));      mapView.mapType=MKMapTypeStandard;      mapView.delegate=self;      [self.view addSubview:mapView];        //导航设置      CLLocationCoordinate2D fromcoor=CLLocationCoordinate2DMake(39.26, 116.3);      CLLocationCoordinate2D tocoor = CLLocationCoordinate2DMake(33.33, 113.33);      //创建出发点和目的点信息      MKPlacemark \*fromPlace = [[MKPlacemark alloc] initWithCoordinate:fromcoor                                                         addressDictionary:nil];      MKPlacemark \*toPlace = [[MKPlacemark alloc]initWithCoordinate:tocoor addressDictionary:nil];      //创建出发节点和目的地节点      MKMapItem \* fromItem = [[MKMapItem alloc]initWithPlacemark:fromPlace];      MKMapItem \* toItem = [[MKMapItem alloc]initWithPlacemark:toPlace];      //初始化导航搜索请求      MKDirectionsRequest \*request = [[MKDirectionsRequest alloc]init];      request.source=fromItem;      request.destination=toItem;      request.requestsAlternateRoutes=YES;      //初始化请求检索      MKDirections \*directions = [[MKDirections alloc]initWithRequest:request];      //开始检索，结果会返回在block中      [directions calculateDirectionsWithCompletionHandler:^(MKDirectionsResponse \*response, NSError \*error) {          if (error) {              NSLog(@"error:%@",error);          }else{              //提取导航线路结果中的一条线路              MKRoute \*route =response.routes[0];              //将线路中的每一步详情提取出来              NSArray \* stepArray = [NSArray arrayWithArray:route.steps];              //进行遍历              for (int i=0; i<stepArray.count; i++) {                  //线路的详情节点                  MKRouteStep \* step = stepArray[i];                  //在此节点处添加一个大头针                  MKPointAnnotation \* point = [[MKPointAnnotation alloc]init];                  point.coordinate=step.polyline.coordinate;                  point.title=step.instructions;                  point.subtitle=step.notice;                  [mapView addAnnotation:point];                  //将此段线路添加到地图上                  [mapView addOverlay:step.polyline];              }          }      }];  }  //地图覆盖物的代理方法  -(MKOverlayRenderer \*)mapView:(MKMapView \*)mapView rendererForOverlay:(id<MKOverlay>)overlay{      MKPolylineRenderer \*renderer = [[MKPolylineRenderer alloc] initWithPolyline:overlay];        renderer.strokeColor = [UIColor redColor];        renderer.lineWidth = 4.0;        return  renderer;  }  //标注的代理方法  -(MKAnnotationView \*)mapView:(MKMapView \*)mapView viewForAnnotation:(id<MKAnnotation>)annotation{      MKPinAnnotationView \* view= [[MKPinAnnotationView alloc]initWithAnnotation:annotation reuseIdentifier:@"anno"];      view.canShowCallout=YES;      return view;  } |

效果如下：

[](http://static.oschina.net/uploads/space/2015/0518/132045_dJBv_2340880.png)

### 二、附近兴趣点检索

兴趣点检索的逻辑和导航线路检索的逻辑相似，直接通过代码来演示：

[?](http://my.oschina.net/u/2340880/blog/416158)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | //创建一个位置信息对象，第一个参数为经纬度，第二个为纬度检索范围，单位为米，第三个为经度检索范围，单位为米      MKCoordinateRegion region = MKCoordinateRegionMakeWithDistance(tocoor, 5000, 5000);      //初始化一个检索请求对象      MKLocalSearchRequest \* req = [[MKLocalSearchRequest alloc]init];      //设置检索参数      req.region=region;      //兴趣点关键字      req.naturalLanguageQuery=@"hotal";      //初始化检索      MKLocalSearch \* ser = [[MKLocalSearch alloc]initWithRequest:req];      //开始检索，结果返回在block中      [ser startWithCompletionHandler:^(MKLocalSearchResponse \*response, NSError \*error) {          //兴趣点节点数组          NSArray \* array = [NSArray arrayWithArray:response.mapItems];          for (int i=0; i<array.count; i++) {              MKMapItem \* item=array[i];              MKPointAnnotation \* point = [[MKPointAnnotation alloc]init];              point.title=item.name;              point.subtitle=item.phoneNumber;              point.coordinate=item.placemark.coordinate;              [mapView addAnnotation:point];          }      }]; |

效果如下：

[](http://static.oschina.net/uploads/space/2015/0518/132653_jZOv_2340880.png)