百度地图移动版 API for ios
开发指南

2012.6.29

目录

1 简介

1.1 什么是百度地图 API

百度地图移动版 API（IOS）是一套基于 iOS3.0 及以上设备的应用程序接口，通过该接口，您可以轻松访问百度服务和数据，构建功能丰富、交互性强的地图应用程序。百度地图移动版 API 不仅包含构建地图的基本接口，还提供了诸如地图定位、本地搜索、路线规划等数据服务，你可以根据自己的需要进行选择，目前支持 Iphone3.0 以上的版本，对 iPad 暂不支持。

面向的读者

API 是提供给那些具有一定 iOS 编程经验和了解面向对象概念的读者使用。此外，读者还应该对地图产品有一定的了解。您在使用中遇到任何问题，都可以通过 API 贴吧或交流群反馈给我们。

1.2 获取 API Key

用户在使用 API 之前需要获取百度地图移动版 API Key，APIKey 可跨平台使用，如果您已经有 Android 平台的授权 Key，可直接在 iOS 平台使用。该 Key 与你的百度账户相关联，您必须先有百度帐户，才能获得 API KEY。并且，该 KEY 与您引用 API 的程序名称有关，具体流程请参照获取密钥。

1.3 兼容性

支持 iOS3.0 及以上系统，百度地图 API 接口与 iOS 内置的 MapKit 包兼容，开发者只需很小的改动即可完成从 MapKit 到百度地图 API 的迁移。并且迁移到百度地图 API 之后很多 MapKit 中只有 iOS4.0 以上版本才能使用的特性接口也可以正常使用了。

2 在您的程序中显示地图

完整的 Demo 例程可参考相关下载。

2.1 引入百度 MapApi 的头文件

首先将百度 MapAPI 提供的头文件和静态库(.a)文件拷贝到您的工程目录下，在 XCode 中添加新的文件 Group，引入百度 MapAPI 提供的头文件。

在您需要使用百度 MapAPId 的文件中添加以下代码

```
#import "BMapKit.h"
```

2.2 引入静态库文件

百度 MapAPI 提供了模拟器和真机两中环境所使用的静态库文件，分别存放在 libs/Release-iphonesimulator 和 libs/Release-iphoneos 文件夹下。有两种方式可以引入静态库文件：

第一种方式：直接将对应平台的.a 文件托拽至 XCode 工程左侧的 Groups&Files 中，缺点是每次在真机和模拟器编译时都需要重新添加.a 文件；

第二种方式：

1.将 API 的 libs 文件夹拷贝到您的 Application 工程跟目录下

2.在 XCode 的 Project -> Edit Active Target -> Build -> Linking -> Other Linker Flags 中添加 **-lbaidumapapi**

3.设置静态库的链接路径，在 XCode 的 Project -> Edit Active Target -> Build -> Search Path -> Library Search Paths 中添加您的静态库目录，比如 **"$(SRCROOT)/../libs/Release$(EFFECTIVE_PLATFORM_NAME)"**，$(SRCROOT)宏代表您的工程文件目录，$(EFFECTIVE_PLATFORM_NAME)宏代表当前配置是 OS 还是 simulator

注:静态库中采用 ObjectC++实现，因此需要您保证您工程中至少有一个.mm 后缀的源文件(您可以将任意一个.m 后缀的文件改名为.mm)，或者在工程属性中指定编译方式，即将 XCode 的 Project -> Edit Active Target -> Build -> GCC4.2 - Language -> Compile Sources As 设置为 "Objective-C++"

2.3 引入 CoreLocation.framework 和 QuartzCore.framework

百度 MapAPI 中提供了定位功能和动画效果，因此您需要在您的 XCode 工程中引入 CoreLocation.framework 和 QuartzCore.framework。 添加方式：右键点击 Xcode 工程左侧的 Frameworks 文件夹，add->Existing Frameworks,在弹出窗口中选中这两个 framework，点击 add 即可。

2.4 引入 mapapi.bundle 资源文件

该步骤为可选，mapapi.bundle 中存储了定位、默认大头针标注 View 及路线关键点的资源图片。如果您不需要使用内置的图片显示功能，则可以不添加此 bundle 文件。您也可以根据具体需求任意替

换 或 删 除 该 bundle 中 的 图 片 文 件 。
添加方式：将 mapapi.bundle 拷贝到您的工程目录，直接将该 bundle 文件托拽至 XCode 工程左侧的 Groups&Files 中即可。

2.5 初始化 BMKMapManager

在您的 AppDelegate.h 文件中添加 BMKMapManager 的定义

```
@interface BaiduMapApiDemoAppDelegate : NSObject
<UIApplicationDelegate> {
    UIWindow *window;
    UINavigationController *navigationController;
    BMKMapManager* _mapManager;
}
```

在您的 AppDelegate.m 文件中添加对 BMKMapManager 的初始化，并填入您申请的授权 Key，示例如下

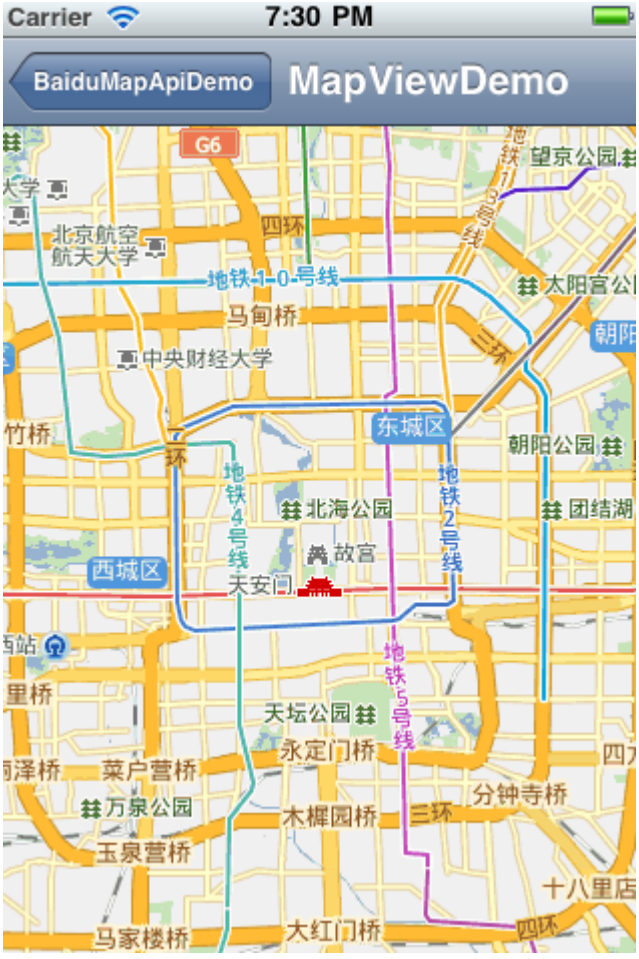
```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // 要使用百度地图，请先启动 BaiduMapManager
    _mapManager = [[BMKMapManager alloc]init];
    // 如果要关注网络及授权验证事件，请设定     generalDelegate 参数
    BOOL ret = [_mapManager start:@"在此处输入您的授权 Key"
    generalDelegate:nil];
    if (!ret) {
            NSLog(@"manager start failed!");
    }
    // Add the navigation controller's view to the window
    and display.
        [self.window addSubview:navigationController.view];
            [self.window makeKeyAndVisible];
            return YES;
        }
```

2.6 创建 BMKMapView

在您的 ViewController.m 文件中添加 BMKMapView 的创建代码，示例如下

```
0. - (void)viewDidLoad {
1.     [super viewDidLoad];
2.     BMKMapView* mapView = [[BMKMapView
   alloc]initWithFrame:CGRectMake(0, 0, 320, 480)];
3.     self.view = mapView;
4. }
```

编译，运行，效果如下图所示：



默认地图已经可以支持多点触摸，双击放大，多点单击缩小等操作，并都附带动画效果。

3 注意事项

1 .静态库中采用 ObjectC++实现，因此需要您保证您工程中至少有一个.mm 后缀的源文件(您可以将任意一个.m 后缀的文件改名为.mm)，或者在工程属性中指定编译方式，即将 XCode 的 Project -> Edit Active Target -> Build -> GCC4.2 - Language -> Compile Sources As 设置为 "Objective-C++"

2.如果您只在 Xib 文件中使用了 BMKMapView，没有在代码中使用 BMKMapView，编译器在链接时不会链接对应符合，需要在工程属性中显式设定：在 XCode 的 Project -> Edit Active Target -> Build -> Linking -> Other Linker Flags 中添加-all_load

3.授权 Key 的申请：授权 Key 可跨平台使用，如果您已经申请过 Android 的 key，可直接在 iOS 中使用；如果还没有授权 Key，请到 http://dev.baidu.com/wiki/static/imap/key/ 页面申请

4 卫星图图层

目前 IOS API 支持卫星图和地图相互切换。
切换为卫星图：

```
[mapView setMapType:BMKMapTypeSatellite];
```

切换为地图：

```
[mapView setMapType:BMKMapTypeStandard];
```

5 实时路况图层

目前支持以下 11 个城市的实时路况信息：北京，上海，广州，深圳，南京，南昌，成都，重庆，武汉，大连，常州。在地图中通过以下代码设置显示实时路况图层：

打开实时路况图层：

```
[mapView setMapType:BMKMapTypeTrafficOn];
```

关闭实时路况图层：

```
[mapView setMapType:BMKMapTypeTrafficOff];
```

6 地图覆盖物

覆盖物概述

地图上自定义的标注点和覆盖物我们统称为地图覆盖物。您可以通过定制 BMKAnnotation 和 BMKOverlay 来添加对应的标注点和覆盖物。地图覆盖物的设计遵循数据与 View 分离的原则，

BMKAnnotation 和 BMKOverlay 系列的类主要用来存放覆盖物相关的数据，BMKAnnotaionView 和 BMKOverlayView 系列类为覆盖物对应的 View。

6.1 添加标注

BMKAnnotation 为标注对应的 protocal，您可以自定义标注类实现该 protocal。百度地图 API 也预置了基本的标注点：BMKPointAnnotation，和一个大头针标注 View：BMKPinAnnotationView，您可以直接使用来显示标注。示例如下：
修改您的 ViewController.h 文件，添加以下代码，使您的 ViewController 实现 BMKMapViewDelegate 协议：

```
#import <UIKit/UIKit.h>
#import "BMapKit.h"
@interface AnnotationDemoViewController : UIViewController
    <BMKMapViewDelegate>{
 IBOutlet BMKMapView* mapView;
}
```

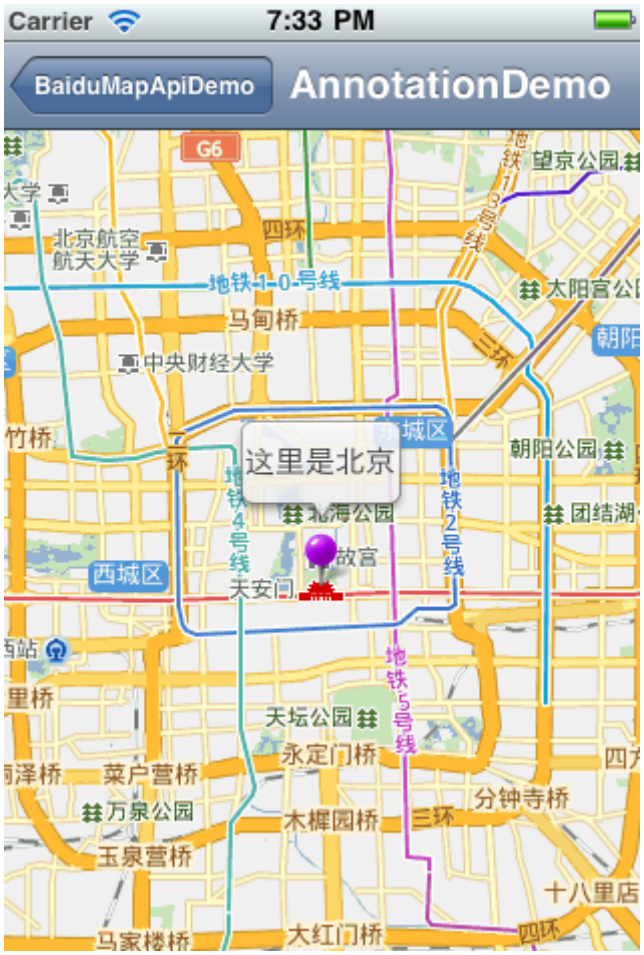
```
@end
```

修改您的 ViewController.m 文件，实现 BMKMapViewDelegate 的 mapView:viewForAnnotation:函数，并在 viewDidLoad 添加标注数据对象

```
// Implement viewDidLoad to do additional setup after loading      th
e view, typically from a nib.
- (void)viewDidLoad {
 [super viewDidLoad];
 // 设置 mapView 的 Delegate
 mapView.delegate = self;
 // 添加一个 PointAnnotation
 BMKPointAnnotation* annotation = [[BMKPointAnnotation alloc]init];
 CLLocationCoordinate2D coor;
 coor.latitude = 39.915;
 coor.longitude = 116.404;
                annotation.coordinate = coor;
                annotation.title = @"这里是北京";
```

```
12.                 [mapView addAnnotation:annotation];
13.         }
14.         // Override
15.         - (BMKAnnotationView *)mapView:(BMKMapView *)mapView viewFo
    rAnnotation:(id <BMKAnnotation>)annotation
16.         {
17.                 if ([annotation isKindOfClass:[BMKPointAnnotation
    class]]) {
18.                         BMKPinAnnotationView *newAnnotationView =
    [[BMKPinAnnotationView alloc] initWithAnnotation:annotation reuseId
    entifier:@"myAnnotation"];
19.                         newAnnotationView.pinColor = BMKPinAnnotati
    onColorPurple;
20.                         newAnnotationView.animatesDrop = YES;// 设置
    该标注点动画显示
21.                         return newAnnotationView;
22.                 }
23.                 return nil;
24.         }
```

运行后，会在地图显示对应的标注点，点击会弹出气泡，效果如图：



6.2 删除标注

通过 removeAnnotation:函数实现对已添加标注的删除功能，示例如下：

```
0.          if (annotation != nil) {
1.          [mapView removeAnnotation:annotation];
```

6.3 添加折线

修改您的 ViewController.h 文件，添加以下代码，使您的 ViewController 实现 BMKMapViewDelegate 协议：

```
#import <UIKit/UIKit.h>
#import "BMapKit.h"
```

```
@interface OverlayDemoViewController : UIViewController <BMKMapViewDelegate>{
 IBOutlet BMKMapView* mapView;
}
@end
```

修改您的 ViewController.m 文件，实现 BMKMapViewDelegate 的 mapView:viewForOverlay:函数，并在 viewDidLoad 添加折线数据对象：

```
- (void)viewDidLoad {
 [super viewDidLoad];
 // 设置delegate
 mapView.delegate = self;
 // 添加折线覆盖物
 CLLocationCoordinate2D coors[2] = {0};
 coors[0].latitude = 39.315;
 coors[0].longitude = 116.304;
 coors[1].latitude = 39.515;
 coors[1].longitude = 116.504;
                BMKPolyline* polyline = [BMKPolyline polylineWithCoordinates:coors count:2];
                [mapView addOverlay:polyline];
        }
        // Override
        - (BMKOverlayView *)mapView:(BMKMapView *)mapView viewForOverlay:(id <BMKOverlay>)overlay{
                if ([overlay isKindOfClass:[BMKPolyline class]]){
                        BMKPolylineView* polylineView = [[[BMKPolylineView alloc] initWithOverlay:overlay] autorelease];
                        polylineView.strokeColor = [[UIColor purpleColor] colorWithAlphaComponent:1];
                        polylineView.lineWidth = 5.0;
                        return polylineView;
                }
                return nil;
        }
```

运行后，效果如图：

6.4 添加多边形

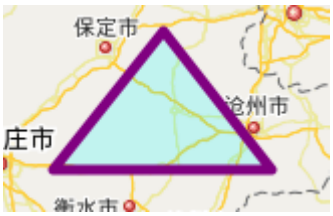
修改您的 ViewController.h 文件，添加以下代码，使您的 ViewController 实现 BMKMapViewDelegate 协议：

修改您的 ViewController.m 文件，实现 BMKMapViewDelegate 的 mapView:viewForOverlay:函数，并在 viewDidLoad 添加多边形数据对象：

```
0. - (void)viewDidLoad {
1.  [super viewDidLoad];
2.  // 设置 delegate
3.  mapView.delegate = self;
4.  // 添加多边形覆盖物
5.  CLLocationCoordinate2D coords[3] = {0};
6.  coords[0].latitude = 39;
7.  coords[0].longitude = 116;
8.  coords[1].latitude = 38;
9.  coords[1].longitude = 115;
10.                 coords[2].latitude = 38;
11.                 coords[2].longitude = 117;
12.                 BMKPolygon* polygon = [BMKPolygon polygonWithCoordi
    nates:coords count:3];
13.                 [mapView addOverlay:polygon];
14.         }
15.         // Override
16.         - (BMKOverlayView *)mapView:(BMKMapView *)mapView viewForOv
    erlay:(id <BMKOverlay>)overlay{
17.                 if ([overlay isKindOfClass:[BMKPolygon class]]){
18.                         BMKPolygonView* polygonView = [[[BMKPolygo
    nView alloc] initWithOverlay:overlay] autorelease];
```

```
                        polygonView.strokeColor = [[UIColor purpleC
    olor] colorWithAlphaComponent:1];
                        polygonView.fillColor = [[UIColor cyanColo
    r] colorWithAlphaComponent:0.2];
                        polygonView.lineWidth = 5.0;
                        return polygonView;
                }
                return nil;
        }
```

运行后，效果如图：



6.5 添加圆

修改您的 ViewController.h 文件，添加以下代码，使您的 ViewController 实现 BMKMapViewDelegate 协议：

修改您的 ViewController.m 文件，实现 BMKMapViewDelegate 的 mapView:viewForOverlay:函数，并在 viewDidLoad 添加园数据对象：

```
- (void)viewDidLoad {
 [super viewDidLoad];
 // 设置delegate
 mapView.delegate = self;
 // 添加圆形覆盖物
 CLLocationCoordinate2D coor;
 coor.latitude = 39.915;
 coor.longitude = 116.404;
 BMKCircle* circle = [BMKCircle circleWithCenterCoordinate:coor radi
    us:5000];
 [mapView addOverlay:circle];
        }
        // Override
        - (BMKOverlayView *)mapView:(BMKMapView *)mapView viewForOv
    erlay:(id <BMKOverlay>)overlay{
```

```
13.                 if ([overlay isKindOfClass:[BMKCircle class]]){
14.                         BMKCircleView* circleView = [[[BMKCircleVi
    ew alloc] initWithOverlay:overlay] autorelease];
15.                         circleView.fillColor = [[UIColor cyanColor]
     colorWithAlphaComponent:0.5];
16.                         circleView.strokeColor = [[UIColor blueColo
    r] colorWithAlphaComponent:0.5];
17.                         circleView.lineWidth = 10.0;
18.                         return circleView;
19.             }
20.             return nil;
21.         }
```

运行后，效果如图：

6.6 删除 Overlay

通过 removeOverlay:函数实现对已添加标注的删除功能，示例如下：

```
0.          if (overlay != nil) {
1.          [mapView removeOverlay:overlay];
2.  }
```

7 服务类

百度地图 API 提供的搜索服务包括：POI 检索，多关键字检索，公交方案检索，驾车路线检索，步行路线检索，地理编码，反地理编码。

所有检索请求接口均为异步接口，您必须实现 BMKSearchDelegate 协议，在检索到结果后，API 会回

调 BMKSearchDelegate 对应的接口，通知调用者检索结果数据。

BMKSearchDelegate 对应的接口如下：

```
/**
 *返回 POI 搜索结果
 *@param poiResultList 搜索结果列表，成员类型为 BMKPoiResult
 *@param type 返回结果类型: BMKTypePoiList,BMKTypeAreaPoiList,BMKAreaMultiPoiList
 *@param error 错误号，@see BMKErrorCode
 */
- (void)onGetPoiResult:(NSArray*)poiResultList searchType:(int)type errorCode:(int)error{
}
/**
 *返回公交搜索结果
          *@param result 搜索结果
          *@param error 错误号，@see BMKErrorCode
          */
        - (void)onGetTransitRouteResult:(BMKPlanResult*)result errorCode:(int)error{
        }

        /**
          *返回驾乘搜索结果
          *@param result 搜索结果
          *@param error 错误号，@see BMKErrorCode
          */
        - (void)onGetDrivingRouteResult:(BMKPlanResult*)result errorCode:(int)error{
        }
        /**
          *返回步行搜索结果
          *@param result 搜索结果
          *@param error 错误号，@see BMKErrorCode
          */
        - (void)onGetWalkingRouteResult:(BMKPlanResult*)result errorCode:(int)error{
        }
        /**
          *返回地址信息搜索结果
          *@param result 搜索结果
```

```
33.          *@param error 错误号，@see BMKErrorCode
34.          */
35.         - (void)onGetAddrResult:(BMKAddrInfo*)result
    errorCode:(int)error{
36.         }
37.         /**
38.          *返回公交详情搜索结果
39.          *@param result 搜索结果
40.          *@param error 错误号，@see BMKErrorCode
41.          */
42.         -       (void)onGetBusDetailResult:(BMKBusLineResult*)result
    errorCode:(int)error{
43.         }
44.
```

7.1 POI 检索

百度地图 API 提供以下几类 POI 检索类型：城市内检索，周边检索，范围检索，多关键字检索。
此处以城市内检索为例说明：
在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码如下：

```
0. @interface PoiSearchDemoViewController : UIViewController<BMKMapV
iewDelegate, BMKSearchDelegate> {
1.  IBOutlet BMKMapView* _mapView;
2.  BMKSearch* _search;
3. }
```

在 ViewController.m 的 viewDidLoad 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取 POI 结果的方法，代码如下：

```
0. - (void)viewDidLoad {
1.  [super viewDidLoad];
2.  _mapView.delegate = self;
3.  _search = [[BMKSearch alloc]init];
4.  _search.delegate = self;
5.  //发起 POI 检索
```

```
6.  [_search poiSearchInCity:@"北京" withKey:@"西单" pageIndex:0];
7. }
8. - (void)onGetPoiResult:(NSArray*)poiResultList searchType:(int)typ
   e errorCode:(int)error
9. {
10.                 if (error == BMKErrorOk) {
11.                         BMKPoiResult* result = [poiResultList objec
   tAtIndex:0];
12.                         for (int i = 0; i < result.poiInfoList.cou
   nt; i++) {
13.                                 BMKPoiInfo* poi = [result.poiInfoL
   ist objectAtIndex:i];
14.                                 BMKPointAnnotation* item = [[BMKPo
   intAnnotation alloc]init];
15.                                 item.coordinate = poi.pt;
16.                                 item.title = poi.name;
17.                                 [_mapView addAnnotation:item];
18.                                 [item release];
19.                         }
20.                 }
21.         }
```

运行效果如图：

示例代码请参考[相关下载](#) demo 工程中的 PoiSearchDemoViewController.mm 文件。

7.2 公交方案检索

在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码参考 POI 检索中的示例代码.

在 ViewController.m 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取公交路线结果的方法，代码如下：

```
0. - (void)viewDidLoad {
1. [super viewDidLoad];
2. _mapView.delegate = self;
3. _search = [[BMKSearch alloc]init];
4. _search.delegate = self;
5. //发起公交检索
6. BMKPlanNode* start = [[BMKPlanNode alloc]init];
7. start.name = @"天安门";
```

```
8.  BMKPlanNode* end = [[BMKPlanNode alloc]init];
9.  end.name = @"百度大厦";
10.                 [_search transitSearch:@"北京" startNode:start endNo
    de:end];
11.                 [start release];
12.                 [end release];
13.         }
14.         - (void)onGetTransitRouteResult:(BMKPlanResult*)result err
    orCode:(int)error
15.         {
16.                 // 在此处添加您对公交方案结果的处理
17.         }
```

将公交方案对应的路线和关键点绘制在地图上，效果如下图：

示例代码请参考相关下载 demo 工程中的 RouteSearchDemoViewController.mm 文件

7.3 驾车路线检索

在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码参考 POI 检索中的示例代码.

在 ViewController.m 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取驾车路线结果的方法，代码如下：

```
0. - (void)viewDidLoad {
1. [super viewDidLoad];
2. _mapView.delegate = self;
3. _search = [[BMKSearch alloc]init];
4. _search.delegate = self;
5. //发起公交检索
6. BMKPlanNode* start = [[BMKPlanNode alloc]init];
7. start.name = @"天安门";
8. BMKPlanNode* end = [[BMKPlanNode alloc]init];
9. end.name = @"百度大厦";
10.                 [_search drivingSearch:@"北京" startNode:start endC
   ity:@"北京" endNode:end];
11.                 [start release];
12.                 [end release];
13.         }
14.         - (void)onGetDrivingRouteResult:(BMKPlanResult*)result erro
   rCode:(int)error
15.         {
16.                 // 在此处添加您对驾车方案结果的处理
17.         }
```

将驾车方案对应的路线和关键点绘制在地图上，效果如下图：

示例代码请参考相关下载 demo 工程中的 RouteSearchDemoViewController.mm 文件

7.4 步行路线检索

在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码参考 POI 检索中的示例代码.

在 ViewController.m 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取步行路线结果的方法，代码如下：

```
0. - (void)viewDidLoad {
1. [super viewDidLoad];
2. _mapView.delegate = self;
3. _search = [[BMKSearch alloc]init];
4. _search.delegate = self;
5. //发起步行检索
6. BMKPlanNode* start = [[BMKPlanNode alloc]init];
```

```
7.  start.name = @"天安门";
8.  BMKPlanNode* end = [[BMKPlanNode alloc]init];
9.  end.name = @"百度大厦";
10.                 [_search walkingSearch:@"北京" startNode:start endC
    ity:@"北京" endNode:end];
11.                 [start release];
12.                 [end release];
13.         }
14.         - (void)onGetWalkingRouteResult:(BMKPlanResult*)result erro
    rCode:(int)error
15.         {
16.                 // 在此处添加您对步行方案结果的处理
17.         }
```

将步行方案对应的路线和关键点绘制在地图上，效果如下图：

示例代码请参考相关下载 demo 工程中的 RouteSearchDemoViewController.mm 文件

7.5 地理编码

在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码参考 POI 检索中的示例代码.
在 ViewController.m 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取地理编码结果的方法，代码如下：

```
0. - (void)viewDidLoad {
1.  [super viewDidLoad];
2.  _mapView.delegate = self;
3.  _search = [[BMKSearch alloc]init];
4.  _search.delegate = self;
5.  //发起地理编码
6.  [_search geocode:@"东长安街 33 号" withCity:@"北京"];
7. }
8. - (void)onGetAddrResult:(BMKAddrInfo*)result errorCode:(int)error
9. {
10.                 // 在此处添加您对地理编码结果的处理
11.         }
```

完整的示例代码请参考相关下载 demo 工程中的 GeocodeDemoViewController.mm 文件

7.6 反地理编码

在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码参考 POI 检索中的示例代码.
在 ViewController.m 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取反地理编码结果的方法，代码如下：

```
0. - (void)viewDidLoad {
1.  [super viewDidLoad];
2.  _mapView.delegate = self;
3.  _search = [[BMKSearch alloc]init];
4.  _search.delegate = self;
5.  //发起反地理编码
6.  CLLocationCoordinate2D pt = (CLLocationCoordinate2D){39.915101, 116.
    403981};
7.  [_search reverseGeocode:pt];
8. }
```

```
- (void)onGetAddrResult:(BMKAddrInfo*)result errorCode:(int)error
        {
                // 在此处添加您对反地理编码结果的处理
        }
```

完整的示例代码请参考相关下载 demo 工程中的 GeocodeDemoViewController.mm 文件

7.7 公交详情检索

在 ViewController.h 中声明 BMKSearch 对象，并将 ViewController 实现 BMKSearchDelegate 协议，代码参考 POI 检索中的示例代码.

在 ViewController.m 中创建 BMKSearch 对象，设置对应的 delegate，并实现 BMKSearchDelegate 协议中获取公交详情结果的方法。发起公交详情搜索前，先进行 POI 检索，检索结果中 poi.epoitype == 2 时表示该类型为公交线路，才可发起公交搜索，代码如下：

```
        - (void)viewDidLoad {
                [super viewDidLoad];
                _mapView.delegate = self;
                _search = [[BMKSearch alloc]init];
                _search.delegate = self;
                //发起poi检索
                [_search poiSearchInCity:@"北京" withKey:@"717" pageIndex:0];
        }

        - (void)onGetPoiResult:(NSArray*)poiResultList searchType:(int)type errorCode:(int)error
        {
                if (error == BMKErrorOk) {
                        BMKPoiResult* result = [poiResultList objectAtIndex:0];
                        for (int i = 0; i < result.poiInfoList.count; i++) {
                                BMKPoiInfo* poi = [result.poiInfoList objectAtIndex:i];
                                if(poi.epoitype == 2)
                                   {
                                        break;
                                    }
                        }
```

```
                    // 发起公交详情搜索
                    if(poi != nil && poi.epoitype == 2 )
                    {
                        NSLog(poi.uid);
                        BOOL flag = [_search busLineSearch:@"北京"
withKey:poi.uid];
                        if (!flag) {
                            NSLog(@"search failed!");
                            }
                        }
                }
        }
        - (void)onGetBusDetailResult:(BMKBusLineResult *)busLineRes
ult errorCode:(int)error

        {
                // 在此处添加您对公交详情结果的处理
        }
```

}将公交详情对应的路线和关键点绘制在地图上，效果如下图：

示例代码请参考相关下载 demo 工程中的 BusLineSearchViewController.mm 文件

8 定位

您可以通过以下代码来开启定位功能：

```
0. [mapView setShowsUserLocation:YES];
```

定位成功后，可以通过 mapView.userLocation 来获取位置数据。

完整的示例代码请参考相关下载 demo 工程中的 LocationDemoViewController.mm 文件