iOS 5 Programming Cookbook

第六章

核心定位以及地图相关的知识

版本 1.0

翻译时间：2012-06-18

DevDiv 翻译： kyelup cloudhsu 耐心摩卡
wangli2003j3 xiebaochun dymx101

DevDiv 校对： laigb kyelup

DevDiv 编辑： BeyondVincent

写在前面

目前，移动开发被广大的开发者们看好，并大量的加入移动领域的开发。

鉴于以下原因：

- 国内的相关中文资料缺乏
- 许多开发者对 E 文很是感冒
- 电子版的文档利于技术传播和交流

DevDiv.com 移动开发论坛特此成立了翻译组，翻译组成员具有丰富的移动开发经验和英语翻译水平。组员们利用业余时间，把一些好的相关英文资料翻译成中文，为广大移动开发者尽一点绵薄之力，希望能对读者有些许作用，在此也感谢组员们的辛勤付出。

关于 DevDiv

DevDiv 已成长为国内最具人气的综合性移动开发社区

更多相关信息请访问 DevDiv 移动开发论坛。

技术支持

首先 DevDiv 翻译组对您能够阅读本文以及关注 DevDiv 表示由衷的感谢。

在您学习和开发过程中，或多或少会遇到一些问题。DevDiv 论坛集结了一流的移动专家，我们很乐意与您一起探讨移动开发。如果您有什么问题和技术需要支持的话，请访问 DevDiv 移动开发论坛 或者发送邮件到 BeyondVincent@DevDiv.com，我们将尽力所能及的帮助你。

关于本文的翻译

感谢 kyelup、cloudhsu、耐心摩卡、wangli2003j3、xiebaochun 和 dymx101 对本文的翻译，同时非常感谢 laigb 和 kyelup 在百忙中抽出时间对翻译初稿的认真校验，指出了文章中的错误。才使本文与读者尽快见面。由于书稿内容多，我们的知识有限，尽管我们进行了细心的检查，但是还是会存在错误，这里恳请广大读者批评指正，并发送邮件至 BeyondVincent@devdiv.com，在此我们表示衷心的感谢。

DevDiv翻译

读者查看下面的帖子可以持续关注章节翻译更新情况

[DevDiv 翻译]iOS 5 Programming Cookbook 翻译各章节汇总

DevDiv翻译

目录

DevDiv翻译

DevDiv翻译

前言

参考帖子
[DevDiv翻译]iOS 5 Programming Cookbook翻译_前言

DevDiv翻译

第 1 章 基础入门

参考帖子

[DevDiv 翻译]iOS 5 Programming Cookbook 翻译_第一章_基础入门

DevDiv翻译

第 2 章 使用控制器和视图

参考帖子

[DevDiv 翻译]iOS 5 Programming Cookbook 翻译 第二章 使用控制器和视图(上)

[DevDiv 翻译]iOS 5 Programming Cookbook 翻译 第二章 使用控制器和视图(下)

DevDiv翻译

第 3 章　构造和使用 Table View

参考帖子

[DevDiv 翻译]iOS 5 Programming Cookbook 翻译_第三章_构造和使用 Table View

DevDiv翻译

第 4 章 Storyboards

参考帖子

[DevDiv 翻译]iOS 5 Programming Cookbook 翻译_第四章 _Storyboards

DevDiv翻译

第 5 章　并发

参考帖子

[DEVDIV 翻译] iOS 5 Programming Cookbook 翻译_第五章_并发

DevDiv翻译

第 6 章 定位核心以及地图相关的知识

6.0 介绍

Core Location 以及 Map 框架包通常能给我们的应用程序添加定位和地图相关的服务。Core Location 框架包通常是使用硬件设备来进行定位服务的，Map 框架包通常能够使你的应用程序做一些地图展示与交互的相关功能。地图的定位服务一般需要依赖设备的硬件组成部分。如果有定位的硬件设备，那么肯定是可以利用地图框架包来进行地图的一些相关的操作.

为了能够在项目中使用到位置服务以及地图展示的相关功能，你必须要导入 Core Location 和 Map 这两个框架包。如果你不知道怎么做，那么请参照如下步骤。

1.点击你的项目工程图标文件。
2.然后选择 target 选项，如图 6-1 所示。
3.然后选择 Build Phase 模块栏。
4.然后点开 Link Binary With Libraries 栏目，在点击+号按钮。

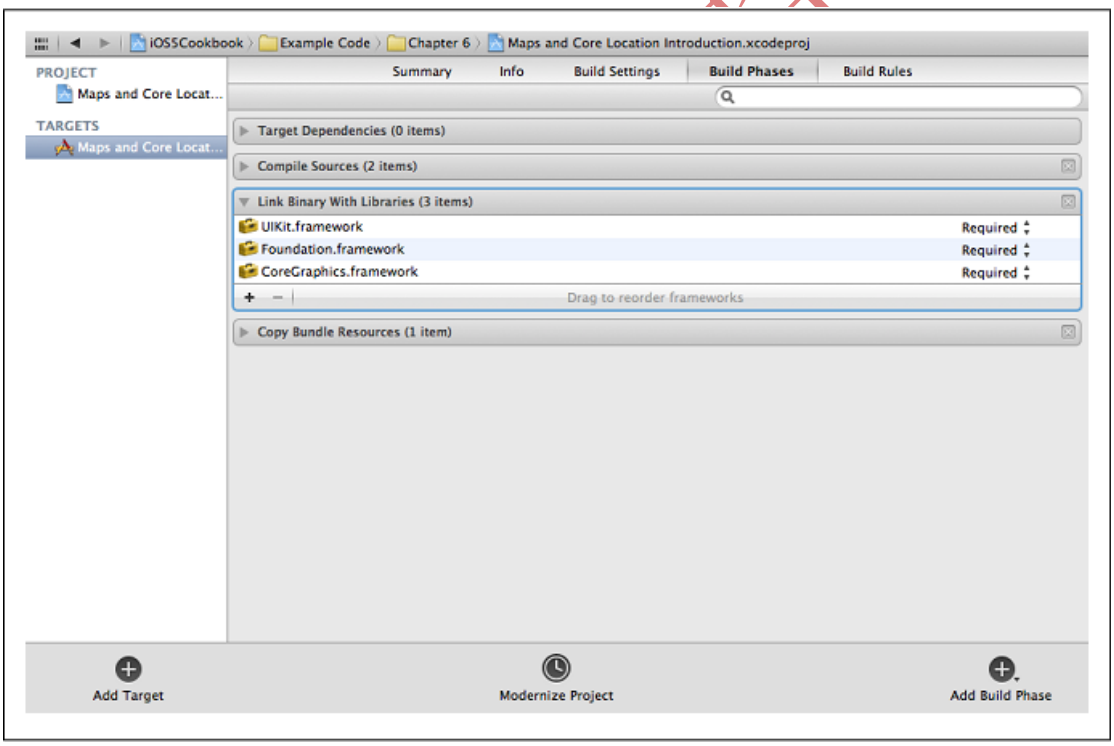

图 6-1 添加相关的框架包

5.在对话框中你将看到所有支持的框架和静态库，找到并选择 CoreLocation 和 Mapket 框架然后按下添加，如下图所示。

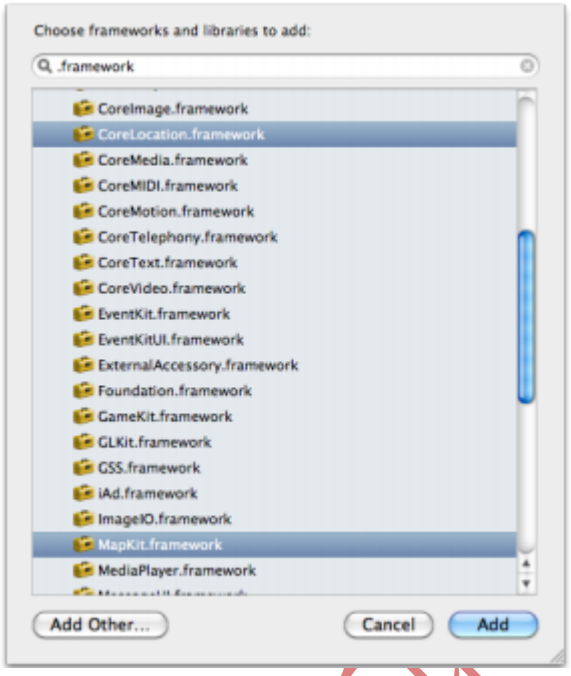

图 6-2 通过弹出的框添加框架包

在添加完这两个包之后，你需要在你的.h 文件或.m 文件中添加头文件的引用(在你的头文件中如果有涉及任何有关于这两个框架的引用)。

```
#import <CoreLocation/CoreLocation.h>
#import <MapKit/MapKit.h>
```

6.1. 创建一个地图的视图

6.1.1. 问题

你需要往你的程序中添加一个地图展示的功能。

6.1.2. 方案

创建一个 MKMapView 的类，然后把他以一个子视图的形式添加到你的控制器文件中。如下是一个视图控制器创建了一个静态 MKMapView 并全屏显示的.h 文件例子。

```
#import <CoreLocation/CoreLocation.h>
#import <MapKit/MapKit.h>
@interface Creating_a_Map_ViewViewController : UIViewController
@property (nonatomic, strong) MKMapView *myMapView;
@end
```

这个是一个有 MKMapkit 类型变量的根视图控制器的例子，然后在视图控制器的实现文件(.m 文件)，我们将初始化地图并设置它的类型为卫星模式，如下。

```
    #import "Creating_a_Map_ViewViewController.h"
@implementation Creating_a_Map_ViewViewController
@synthesize myMapView;
- (void)didReceiveMemoryWarning{
    [super didReceiveMemoryWarning];
}
- (void)viewDidLoad{
    [super viewDidLoad];

    self.view.backgroundColor = [UIColor whiteColor];

    self.myMapView = [[MKMapView alloc]
                      initWithFrame:self.view.bounds];
    /* Set the map type to Satellite */
    self.myMapView.mapType = MKMapTypeSatellite;

    self.myMapView.autoresizingMask =
        UIViewAutoresizingFlexibleWidth |
        UIViewAutoresizingFlexibleHeight;

    /* Add it to our view */
    [self.view addSubview:self.myMapView];

}
- (void)viewDidUnload{
    [super viewDidUnload];
    self.myMapView = nil;
}
- (BOOL)shouldAutorotateToInterfaceOrientation
          :(UIInterfaceOrientation)interfaceOrientation{
    return YES;
}
@end
```

6.1.3. 讨论

创建一个 MKMapView 实例对象是一个最简单有效的方法，我们同时也可以创建一个窗体，然后利用他的构造器，把这个实例对象以一个子窗体的形式添加到上面。这样我们也能看到这个地图的展示。

MKMapView 是 UIView 的一个子类，因此你能够操作任何一个地图视图就像操作一个普通的视图一样的。

可能你还没有注意到，MKMapView 这个类有一个 mapType 的属性，这个属性一边是用来设置你的地图到底是以什么样的形式展示。有卫星云图，有普通的地图等等。如下图，就是一个卫星云图。

图 6-3 卫星云图

我们可以通过 MKMapView 的 mapType 属性来切换地图的展现形式。有如下的属性值可用：

MKMapTypeStandard
用这个类型来显示普通地图（这个是默认的）。
MKMapTypeSatellite
用这个来类型来显示卫星云图（如图 6-3）。
MKMapTypeHybrid
用这个类型来显示普通地图覆盖于卫星云图之上，这个地图的展现形式属于复合形式。

6.1.4. 参考

无

6.2. 捕获 Map 视图上的一些动作事件

6.2.1. 问题

你想捕获一些地图视图上的一些动作事件，并想根据这些事件做一些操作。

6.2.2. 方案

定义一个 delegate 的对象，然后实现 MKMapViewDelegate 协议里面的相关放然。

```
   /* Create a map as big as our view */
self.myMapView = [[MKMapView alloc]
                        initWithFrame:self.view.bounds];
/* Set the map type to Satellite */
self.myMapView.mapType = MKMapTypeSatellite;
self.myMapView.delegate = self;
self.myMapView.autoresizingMask =
   UIViewAutoresizingFlexibleWidth |
   UIViewAutoresizingFlexibleHeight;
```

DevDiv翻译

```
/* Add it to our view */
[self.view addSubview:self.myMapView];
```

把上面这段代码添加到一个视图控制器的 viewDidLoad 方法里面，很容易就能运行。不过前提是你需要在头文件中添加类似如下的内容。

```
    /* Create a map as big as our view */
self.myMapView = [[MKMapView alloc]
                  initWithFrame:self.view.bounds];
/* Set the map type to Satellite */
self.myMapView.mapType = MKMapTypeSatellite;
self.myMapView.delegate = self;
self.myMapView.autoresizingMask =
  UIViewAutoresizingFlexibleWidth |
  UIViewAutoresizingFlexibleHeight;
/* Add it to our view */
[self.view addSubview:self.myMapView];
```

6.2.3. 讨论

任何实现了 MKMapViewDelegate 这个类的必须要实现它定义的几个方法。以便当地图视图上有一些动作操作的时候能够接收并监听这些方法。然后我们可以根据捕获到的时间，然后添加相关的视图展示效果和操作。例如 mapViewWillStartLoadingMap 这个方法。这个方法一般是当地图加载完成的时候会调用。如下是 MKMapViewDelegate 这个协议中的一些方法。

mapViewWillStartLoadingMap:

这个方法是当地图界面将要加载的时候会调用。

mapView:viewForAnnotation:

这个方法你可以在 6.4 小节中找到更详细的知识，这个方法是当地图上有一些动画效果展示或者加载的时候会调用这个方法。

mapViewWillStartLocatingUser:

这个方法是准备进行一个位置定位的时候会调用的方法。

mapView:regionDidChangeAnimated:

这个方法调用，一般是当用户的地理位置发生变化的时候会调用。

6.2.4. 参考

无

6.3. 用硬件设备获取用户当前的地理位置

6.3.1. 问题

你需要通过你的设备获取到当前位置的经纬度。

6.3.2. 方案

使用 CLLocationManager 这个类，参考代码如下。

```
    if ([CLLocationManager locationServicesEnabled]){
    self.myLocationManager = [[CLLocationManager alloc] init];
    self.myLocationManager.delegate = self;

    self.myLocationManager.purpose =
    @"To provide functionality based on user's current location.";

    [self.myLocationManager startUpdatingLocation];
} else {
    /* Location services are not enabled.
      Take appropriate action: for instance, prompt the
      user to enable the location services */
    NSLog(@"Location services are not enabled");
}
```

在上面这段代码中，myLocationManager 是 CLLocationManager 的一个属性，当前这个类也是 LocationManager 的一些协议实现类。

6.3.3. 讨论

Core Location 框架提供了让开发者能够利用 iOS 设备来进行位置服务。因为在 iOS 中，用户通常是禁止了设备的定位功能，因此，当你在使用 CLLocationManager 这个类的时候一般都是有一个提示功能，让用户选择打开定位的功能。

任何实现 CLLocationManager 类的协议实现者都必须要遵循 CLLocationManagerDelegate 这个协议中的方法。

请分别参考如下代码，我们需要在.h 头文件和.m 文件中添加的代码。

```
    #import <UIKit/UIKit.h>
#import <CoreLocation/CoreLocation.h>
@interface Pinpointing_the_Location_of_a_DeviceViewController
                : UIViewController <CLLocationManagerDelegate>
@property (nonatomic, strong) CLLocationManager *myLocationManager;
@end
```

```
    #import "Pinpointing_the_Location_of_a_DeviceViewController.h"
@implementation Pinpointing_the_Location_of_a_DeviceViewController
@synthesize myLocationManager;
- (void)didReceiveMemoryWarning{
    [super didReceiveMemoryWarning];
}
- (void)locationManager:(CLLocationManager *)manager
      didUpdateToLocation:(CLLocation *)newLocation
                fromLocation:(CLLocation *)oldLocation{

    /* We received the new location */

    NSLog(@"Latitude = %f", newLocation.coordinate.latitude);
    NSLog(@"Longitude = %f", newLocation.coordinate.longitude);

}
- (void)locationManager:(CLLocationManager *)manager
          didFailWithError:(NSError *)error{
```

```
    /* Failed to receive user's location */

}
- (void)viewDidLoad {
    [super viewDidLoad];

    if ([CLLocationManager locationServicesEnabled]){
        self.myLocationManager = [[CLLocationManager alloc] init]
        self.myLocationManager.delegate = self;

        self.myLocationManager.purpose =
        @"To provide functionality based on user's current locati

        [self.myLocationManager startUpdatingLocation];
    } else {
        /* Location services are not enabled.
          Take appropriate action: for instance, prompt the
          user to enable the location services */
        NSLog(@"Location services are not enabled");
    }
}
- (void) viewDidUnload{
    [super viewDidUnload];
    [self.myLocationManager stopUpdatingLocation];
    self.myLocationManager = nil;
}
- (BOOL)shouldAutorotateToInterfaceOrientation
            :(UIInterfaceOrientation)interfaceOrientation{
    return YES;
}
@end
```

如上代码，CLLocationManager 的 startUpdateLocation 方法通过它的代理 didUpdateToLocation:fromLocation: 和 locationManager:didFailWithError:方法来报告用户定位成功或失败。

CLLocationManager 必须要在 SDK4.0 以后的版本中才能使用 CLLocationManager 这个类有一个参数为 purpose,这个参数能够配置一些相关的告示信息，这个一般用来提醒用户，让用户给这个程序地图位置服务功能授权。我们也可以配置一些区域不用的提示语言来提高用户体验度。

6.3.4. 参考

无

6.4. 在地图视图上添加锚点

6.4.1. 问题

你需要在地图上标注一些点来特别说明这些地理位置

6.4.2. 方案

在地图中内置一些注释信息

请参考如下步骤

1.创建一个新的类，命名为 MyAnnotation。

2.确保这个类要实现 MKAnnotation 协议。

3.给这个类定义一个类型为 CLLocationCoordinate2D 的属性，命名为 coordinate。特别要注意这个参数需要标示为只读类型的。因为 MKAnnotation 这个协议中定义的 Coordinate 也是只读类型的。

4.同理，定义两个 NSString 类型的属性，分别命名为 title 和 subtitle。这两个参数用来保存锚点的标题和内容信息。

5.给这个类添加一个初始话方法，这个方法需要传递一个 CLLocationCoordinate2D 类型的参数。在这个方法中把我们在步骤 3 定义的那个属性传递进来..由于这个属性是只读的，他并不能在这个类以外进行赋值。因此，这个初始化方法也就是一个桥梁，使我们能够正常的往这个了类中进行传递值。同理 title 和 subtitle 也要进行类似的操作。

6.初始化 MyAnnotation 这个类，然后把他添加到你的地图中，通过 Annotation 这个方法。

6.4.3. 讨论

相关的代码如下，我们分别需要在.h 和.m 文件中进行类似如下的修改。

```
 #import <Foundation/Foundation.h>
#import <MapKit/MapKit.h>
@interface MyAnnotation : NSObject <MKAnnotation>
@property (nonatomic, readonly) CLLocationCoordinate2D coordinate;
@property (nonatomic, copy, readonly) NSString *title;
@property (nonatomic, copy, readonly) NSString *subtitle;
- (id) initWithCoordinates:(CLLocationCoordinate2D)paramCoordinates
                     title:(NSString *)paramTitle
                  subTitle:(NSString *)paramSubTitle;
@end
```

.m 文件修改类似如下。

```
 #import "MyAnnotation.h"
@implementation MyAnnotation
CLLocationCoordinate2D coordinate;
@synthesize coordinate, title, subtitle;
- (id) initWithCoordinates:(CLLocationCoordinate2D)paramCoordinates
                     title:(NSString *)paramTitle
                  subTitle:(NSString *)paramSubTitle{

  self = [super init];

  if (self != nil){
    coordinate = paramCoordinates;
```

```
        title = paramTitle;
        subtitle = paramSubTitle;
    }

    return(self);

}
@end
```

然后我们需要在我们的地图视图控制器类中添加这个类，并且添加相关的方法。代码如下。

```
    #import "Displaying_Pins_on_a_Map_ViewViewController.h"
#import "MyAnnotation.h"
@implementation Displaying_Pins_on_a_Map_ViewViewController
@synthesize myMapView;
- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Release any cached data, images, etc that aren't in use.
}
- (void)viewDidLoad {
    [super viewDidLoad];
  /* Create a map as big as our view */
    self.myMapView = [[MKMapView alloc]
                                initWithFrame:self.view.bounds];

    self.myMapView.delegate = self;

    /* Set the map type to Standard */
    self.myMapView.mapType = MKMapTypeStandard;

    self.myMapView.autoresizingMask =
        UIViewAutoresizingFlexibleWidth |
        UIViewAutoresizingFlexibleHeight;

    /* Add it to our view */
    [self.view addSubview:self.myMapView];

    /* This is just a sample location */
    CLLocationCoordinate2D location =
        CLLocationCoordinate2DMake(50.82191692907181, -0.13811767101287842);

    /* Create the annotation using the location */
    MyAnnotation *annotation =
    [[MyAnnotation alloc] initWithCoordinates:location
                                            title:@"My Title"
                                         subTitle:@"My Sub Title"];

    /* And eventually add it to the map */
    [self.myMapView addAnnotation:annotation];

}
- (void) viewDidUnload{
    [super viewDidUnload];
    self.myMapView = nil;
}
```

```
- (BOOL)shouldAutorotateToInterfaceOrientation
            :(UIInterfaceOrientation)interfaceOrientation{
   return YES;
}
@end
```

添加完成如上代码之后，编译我们的项目并在模拟器中运行,你将会看到如下效果。

图 6.4 在地图上添加一个锚点

6.4.4. 参考

无

6.5. 在地图上添加一些不同颜色的锚点

6.5.1. 问题

地图上的锚点默认的都是红色的，你想根据你自己的意愿来添加一些不同颜色的锚点。

6.5.2. 方案

通过 mapView:viewForAnnotation:这个方法来返回一个 MKPinAnnotationView 实例对象，来进行一些相关的操作。每一个动画都是添加到一个 MKMapView 实例化对象的视图上，然后加以显示的。这些视图一般称为动画视图。一个动画视图是一个 MKAnnotationView 类型的对象。也是 UIView 的子类。如果一个实现类的对象是一个地图的视图，并且实现了 mapVIew:viewForAnnotation:这个方法，那么这个实现类将会返回一个 MKAnnotationView 的实例，然后你可以自己添加一个动画的视图效果，从而使他们在地图视图上得以显示。

6.5.3. 讨论

为了能够在地图视图中为我们的锚点自定义显示的颜色，我们必须要获得一个

MKPinAnnotationView 这个实例对象而不是以往的 MKAnnotationView 对象。我们需要通过 mapView:viewForAnnotation 这个方法来获得这个实例对象。具体实现方式请参考如下代码。

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView
            viewForAnnotation:(id <MKAnnotation>)annotation{

  MKAnnotationView *result = nil;

  if ([annotation isKindOfClass:[MyAnnotation class]] == NO){
     return result;
  }

  if ([mapView isEqual:self.myMapView] == NO){
     /* We want to process this event only for the Map View
       that we have created previously */
     return result;
  }

  /* First typecast the annotation for which the Map View has
    fired this delegate message */
  MyAnnotation *senderAnnotation = (MyAnnotation *)annotation;

  /* Using the class method we have defined in our custom
    annotation class, we will attempt to get a reusable
 identifier for the pin we are about
    to create */
  NSString *pinReusableIdentifier =
  [MyAnnotation
    reusableIdentifierforPinColor:senderAnnotation.pinColor];

  /* Using the identifier we retrieved above, we will
    attempt to reuse a pin in the sender Map View */
  MKPinAnnotationView *annotationView = (MKPinAnnotationView *)
  [mapView
    dequeueReusableAnnotationViewWithIdentifier:pinReusableIdentifier];

  if (annotationView == nil){
     /* If we fail to reuse a pin, then we will create one */
     annotationView =    [[MKPinAnnotationView alloc]
                          initWithAnnotation:senderAnnotation
                          reuseIdentifier:pinReusableIdentifier];

     /* Make sure we can see the callouts on top of
       each pin in case we have assigned title and/or
       subtitle to each pin */
     [annotationView setCanShowCallout:YES];
  }

  /* Now make sure, whether we have reused a pin or not, that
    the color of the pin matches the color of the annotation */
  annotationView.pinColor = senderAnnotation.pinColor;
  result = annotationView;

  return result;
}
```

这个动画视图应该通过给它一个 NSString 类型的标示符来重复利用。根据你的实际需求来选择你所需要的锚点类型从而能够在地图视图上正常的显示，不同类型的锚点从而达到不同颜色的一个效果。你应该通过 dequeueReusableAnnotationViewWithIdentifier 这个方法来获得一个实例对象来重复的使用锚点。

我们应该在自定义的 MyAnnotation 类里针对每个锚点的唯一标识来设定回收机制，如下是 MyAnnotation 类的头文件:

```
   #import <Foundation/Foundation.h>
#import <MapKit/MapKit.h>
/* These are the standard SDK pin colors. We are setting
  unique identifiers per color for each pin so that later we
  can reuse the pins that have already been created with the same
  color */
#define REUSABLE_PIN_RED       @"Red"
#define REUSABLE_PIN_GREEN    @"Green"
#define REUSABLE_PIN_PURPLE @"Purple"
@interface MyAnnotation : NSObject <MKAnnotation>
@property (nonatomic, unsafe_unretained, readonly)
   CLLocationCoordinate2D coordinate;
@property (nonatomic, copy) NSString    *title;
@property (nonatomic, copy) NSString    *subtitle;
@property (nonatomic, unsafe_unretained) MKPinAnnotationColor    pinColor;
- (id) initWithCoordinates:(CLLocationCoordinate2D)paramCoordinates
                                title:(NSString*)paramTitle
                             subTitle:(NSString*)paramSubTitle;
+ (NSString *)      reusableIdentifierforPinColor
                           :(MKPinAnnotationColor)paramColor;
@end
```

MyAnnotation 这个类其实并不是一个注释的视图。它是在你想在地图上显示的定位并且就是你要显示。当我们创建一个动画效果，我们可以来通过我们创建的 pinColor 这个属性来指定颜色等特征。forAnnotation 这个属性用来传递动画展示的具体效果。通过获得一个动画的引用，我们可以考本一个类似 MyAnnotation 的实例对象。通过 pinColor 这个属性来恢复一些颜色效果,.基于这个特征，我们额可以创建一个 MKPinAnnotationView 的一个实例化对象，并且绑定一个颜色效果，返回到地图的视图上。

相关代码如下。

```
@implementation MyAnnotation
@synthesize coordinate;
@synthesize title;
@synthesize subtitle;
@synthesize pinColor;
+ (NSString *)      reusableIdentifierforPinColor
                           :(MKPinAnnotationColor)paramColor{

   NSString *result = nil;

   switch (paramColor){
      case MKPinAnnotationColorRed:{
         result = REUSABLE_PIN_RED;
         break;
      }
      case MKPinAnnotationColorGreen:{
```

```
            result = REUSABLE_PIN_GREEN;
            break;
        }
        case MKPinAnnotationColorPurple:{
            result = REUSABLE_PIN_PURPLE;
            break;
        }
    }

    return result;
}
- (id) initWithCoordinates:(CLLocationCoordinate2D)paramCoordinates
                     title:(NSString*)paramTitle
                  subTitle:(NSString*)paramSubTitle{

    self = [super init];

    if (self != nil){
        coordinate = paramCoordinates;
        title = paramTitle;
        subtitle = paramSubTitle;
        pinColor = MKPinAnnotationColorGreen;
    }
- (id) initWithCoordinates:(CLLocationCoordinate2D)paramCoordinates
                     title:(NSString*)paramTitle
                  subTitle:(NSString*)paramSubTitle{

    self = [super init];

    if (self != nil){
        coordinate = paramCoordinates;
        title = paramTitle;
        subtitle = paramSubTitle;
        pinColor = MKPinAnnotationColorGreen;
    }

    return self;

}
@end
```

通过实现 MyAnnotation 这个类，然后在添加类似下面实例的代码，我们基本就可以完成效果了。

```
#import <UIKit/UIKit.h>
#import <MapKit/MapKit.h>
@interface Displaying_Pins_with_Different_Colors_on_a_Map_ViewViewController
                : UIViewController <MKMapViewDelegate>
@property (nonatomic, strong) MKMapView *myMapView;
@end

#import "Displaying_Pins_with_Different_Colors_on_a_Map_ViewViewController.h"
#import "MyAnnotation.h"
@implementation
    Displaying_Pins_with_Different_Colors_on_a_Map_ViewViewController
@synthesize myMapView;
- (void)didReceiveMemoryWarning{
```

```
    [super didReceiveMemoryWarning];
}
- (MKAnnotationView *)mapView:(MKMapView *)mapView
                viewForAnnotation:(id <MKAnnotation>)annotation{

    MKAnnotationView *result = nil;

    if ([annotation isKindOfClass:[MyAnnotation class]] == NO){
        return result;
    }

    if ([mapView isEqual:self.myMapView] == NO){
        /* We want to process this event only for the Map View
          that we have created previously */
        return result;
    }

    /* First typecast the annotation for which the Map View has
      fired this delegate message */
    MyAnnotation *senderAnnotation = (MyAnnotation *)annotation;

    /* Using the class method we have defined in our custom
      annotation class, we will attempt to get a reusable
      identifier for the pin we are about
      to create */
   NSString *pinReusableIdentifier =
    [MyAnnotation
      reusableIdentifierforPinColor:senderAnnotation.pinColor];

    /* Using the identifier we retrieved above, we will
      attempt to reuse a pin in the sender Map View */
    MKPinAnnotationView *annotationView = (MKPinAnnotationView *)
    [mapView
      dequeueReusableAnnotationViewWithIdentifier:pinReusableIdentifier];

    if (annotationView == nil){
        /* If we fail to reuse a pin, then we will create one */
        annotationView =    [[MKPinAnnotationView alloc]
                                    initWithAnnotation:senderAnnotation
                                    reuseIdentifier:pinReusableIdentifier];

        /* Make sure we can see the callouts on top of
          each pin in case we have assigned title and/or
          subtitle to each pin */
        [annotationView setCanShowCallout:YES];
    }

    /* Now make sure, whether we have reused a pin or not, that
      the color of the pin matches the color of the annotation */
    annotationView.pinColor = senderAnnotation.pinColor;

    result = annotationView;
    return result;
}
- (void)viewDidLoad {
    [super viewDidLoad];

    /* Create a map as big as our view */
    self.myMapView = [[MKMapView alloc]
```

```
                                initWithFrame:self.view.bounds];

    self.myMapView.delegate = self;

    /* Set the map type to Standard */
    self.myMapView.mapType = MKMapTypeStandard;

    self.myMapView.autoresizingMask =
        UIViewAutoresizingFlexibleWidth |
        UIViewAutoresizingFlexibleHeight;

    /* Add it to our view */
    [self.view addSubview:self.myMapView];

    /* This is just a sample location */
    CLLocationCoordinate2D location;
    location.latitude = 50.82191692907181;
    location.longitude = -0.13811767101287842;

    /* Create the annotation using the location */
    MyAnnotation *annotation =
    [[MyAnnotation alloc] initWithCoordinates:location
                                        title:@"My Title"
                                     subTitle:@"My Sub Title"];

    annotation.pinColor = MKPinAnnotationColorPurple;

    /* And eventually add it to the map */
    [self.myMapView addAnnotation:annotation];

}
- (void)viewDidUnload{
    [super viewDidUnload];
    self.myMapView = nil;
}
- (BOOL)shouldAutorotateToInterfaceOrientation
            :(UIInterfaceOrientation)interfaceOrientation{
    return YES;
}
@end
```

运行，将会得到如下效果。

图 6.5 添加了自定义颜色和文字的锚点

6.5.4. 参考

无

6.6. 在地图视图上添加自定义的锚点

6.6.1. 问题

你想在地图上添加一些依你自己的定义的图片的形式来显示锚点，而不是 iOS SDK 自带的那种形式。

6.6.2. 方案

通过 UIImage 引入一个你自定义的图片，然后把它绑定到 MKAnnotationView 的 image 这个属性上去，以一个锚点的形式返回到你的地图视图上.代码如下:

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView
            viewForAnnotation:(id <MKAnnotation>)annotation{

MKAnnotationView *result = nil;

if ([annotation isKindOfClass:[MyAnnotation class]] == NO){
   return result;
}

if ([mapView isEqual:self.myMapView] == NO){
   /* We want to process this event only for the Map View
     that we have created previously */
   return result;
```

DevDiv翻译平台

```
    }

    /* First typecast the annotation for which the Map View has
     fired this delegate message */
    MyAnnotation *senderAnnotation = (MyAnnotation *)annotation;

    /* Using the class method we have defined in our custom
     annotation class, we will attempt to get a reusable
     identifier for the pin we are about to create */
    NSString *pinReusableIdentifier =
    [MyAnnotation
     reusableIdentifierforPinColor:senderAnnotation.pinColor];

    /* Using the identifier we retrieved above, we will
   attempt to reuse a pin in the sender Map View */
    MKPinAnnotationView *annotationView = (MKPinAnnotationView *)
    [mapView
     dequeueReusableAnnotationViewWithIdentifier:
     pinReusableIdentifier];

    if (annotationView == nil){
        /* If we fail to reuse a pin, then we will create one */
        annotationView =
        [[MKPinAnnotationView alloc]    initWithAnnotation:senderAnnotation
                                              reuseIdentifier:pinReusableIdentifier];

        /* Make sure we can see the callouts on top of
         each pin in case we have assigned title and/or
         subtitle to each pin */
        annotationView.canShowCallout = YES;

    }

    /* Now make sure, whether we have reused a pin or not, that
     the color of the pin matches the color of the annotation */
    annotationView.pinColor = senderAnnotation.pinColor;
    UIImage *pinImage = [UIImage imageNamed:@"BluePin.png"];
    if (pinImage != nil){
        annotationView.image = pinImage;
    }

    result = annotationView;

    return result;
}
```

如上代码中我们引入了一个图片，BluePin.png,如果对于 MyAnnotation 这个类有不了解的，请参考 6.5 小节。

6. 6. 3. 讨论

MKMapView 实例对象的协议类必须要实现 MKMapViewDelegate 这个协议，并且要实现 mapView:viewForAnnotation 这个方法。这个方法将会返回一个 MKAnnotationView 的实例对象，任何这个类的实现类都有一个叫做 image 的属性，我们可以通过这个属性来绑定我们自定义的图片从而能够在地图上展示.如下图所示。

图 6.6 自定义以图片展示形式的锚点

6.6.4. 参考

无

6.7. 通过一组经纬度数据得到一个地点名称

6.7.1. 问题

你有一组经纬度数据，你想把这组数据解析出来得到一个实在的地理位置名称。

6.7.2. 方案

通过一组经纬度数据得到一个实在的地理位置数据，我们通常称之为逆向地理编码。

创建一个 CLGeocoder 的实例对象，然后创建一个完整的块对象，这个对象必须要没有返回值，而且要接收两个参数。

一个 NSArray 类型的地点标记参数，这个参数将会用来保存你讲需要查询的地理位置。

一个 NSerror 类型的参数，这个参数将会返回一些地理位置编码信息是否正确的校验信息。

当介绍了 GLGeocoder 用法之后，我们需要使用 erverseGeocodeLocation:completionHandler 这个方法来进行反向地理位置编码。

我们在自己实现的 view controller 中添加如下代码。

```
#import <UIKit/UIKit.h>
#import <CoreLocation/CoreLocation.h>
@interface
   Converting_Longitude_and_Latitude_to_a_Meaningful_AddressViewController
    : UIViewController
```

DevDiv翻译平台

```
@property (nonatomic, strong) CLGeocoder *myGeocoder;
@end
```

然后我们需要在.m 文件中进行一个绑定。

```
@implementation
   Converting_Longitude_and_Latitude_to_a_Meaningful_AddressViewController
@synthesize myGeocoder;
...
```

然后需要在 viewDidLoad 加载方法中进行配置。

```
   CLLocation *location = [[CLLocation alloc]
                                    initWithLatitude:+38.4112810
                                    longitude:-122.8409780f];
   self.myGeocoder = [[CLGeocoder alloc] init];

   [self.myGeocoder
    reverseGeocodeLocation:location
    completionHandler:^(NSArray *placemarks, NSError *error) {

        if (error == nil &&
              [placemarks count] > 0){
                    CLPlacemark *placemark = [placemarks objectAtIndex:0];
           /* We received the results */
           NSLog(@"Country = %@", placemark.country);
           NSLog(@"Postal Code = %@", placemark.postalCode);
           NSLog(@"Locality = %@", placemark.locality);
        }
        else if (error == nil &&
                    [placemarks count] == 0){
           NSLog(@"No results were returned.");
        }
        else if (error != nil){
           NSLog(@"An error occurred = %@", error);
        }

     }];
}
- (void)viewDidUnload{
   [super viewDidUnload];
   self.myGeocoder = nil;
}
```

你可以自己看看一个叫做 placemarks 这个数组，这个数组，如果反向地理位置编码正常，那么这个数组将会包含一组 CLPlacemark 的数据，这个组数据通过我们给 reverseGeocodeLocation：completionHandler 这个方法传递的经纬度数据来进行地理位置匹配，因此如果我们要确保没有错误信息的发生，并且我们应该确定 placemarks 数组中至少包含一个 placemark.。

以上的代码将在窗口上输出反向地址解析。

```
Country = United States
Postal Code = 95472
```

```
Locality = Sebastopol
```

6.7.3. 讨论

每个程序每天都有限的反向地址解析请求次数，执行一个反向地址解析请求，你都要创建一个 CLGeocoder 静态实例。这个类请求了一个有效的网络连接来响应请求成功。反向地址解析的值将通过 reverseGeocodeLocation:completionHandler:block 方法来报告。

6.7.4. 参考

无

6.8. 通过一个有意义的地址得到一组经纬度数据

6.8.1. 问题

你想通过一个地理名称得到一组经纬度数据。

6.8.2. 方案

通过 GLGeocoder 这个类的 geocodeAddressString:completionHandler 这个方法来实现。

6.8.3. 讨论

反向地理编码是通过一组经纬度数据的到一个实在的地理位置名称。同样我们可以使用地理编码通过一个地理名称得到一组经纬度数据。我们需要使用 CoreLocation 这个资源包中的 CLGeocoder 这个类。

我们通过给 geocodeAddressString:completionHandler 这个方法传递一个 String 类型的地理位置名称来进行地理编码。让我们开始声明一个 CLGeocoder 类型的一个变量，代码如下。

```
@interface
   Converting_Meaningful_Addresses_to_Longitude_and_LatitudeViewController
   : UIViewController
@property (nonatomic, strong) CLGeocoder *myGeocoder;
@end
```

然后我们需要绑定这个变量。

```
@implementation
   Converting_Meaningful_Addresses_to_Longitude_and_LatitudeViewController
@synthesize myGeocoder;
...
```

然后我们需要在相关的方法中添加如下代码。

```
- (void)didReceiveMemoryWarning{
   [super didReceiveMemoryWarning];
   // Release any cached data, images, etc that aren't in use.
}
- (void)viewDidLoad{
   [super viewDidLoad];
```

VDJV翻译

```
    /* We have our address */
    NSString *oreillyAddress =
        @"1005 Gravenstein Highway North, Sebastopol, CA 95472, USA";

    self.myGeocoder = [[CLGeocoder alloc] init];

    [self.myGeocoder
      geocodeAddressString:oreillyAddress
      completionHandler:^(NSArray *placemarks, NSError *error) {

          if ([placemarks count] > 0 &&
                error == nil){
              NSLog(@"Found %lu placemark(s).", (unsigned long)[placemarks count]);
              CLPlacemark *firstPlacemark = [placemarks objectAtIndex:0];
              NSLog(@"Longitude = %f", firstPlacemark.location.coordinate.longitude);
              NSLog(@"Latitude = %f", firstPlacemark.location.coordinate.latitude);
          }
          else if ([placemarks count] == 0 &&
                       error == nil){
              NSLog(@"Found no placemarks.");
          }
          else if (error != nil){
              NSLog(@"An error occurred = %@", error);
          }

      }];

}
- (void)viewDidUnload{
    [super viewDidUnload];
    self.myGeocoder = nil;
}
```

当这些代码都添加完成之后，我们编译运行我们的程序，然后我们将会看到如下的效果。

```
Found 1 placemark(s).
Longitude = -122.841135
Latitude = 38.410373
```

6.8.4. 参考

无

点击这里访问：DevDiv.com 移动开发论坛