iPhone与iPad应用开发课程
精通iOS开发

第十一讲 地图和定位应用开发

主讲人：关东升

eorient@sina.com

主要知识点

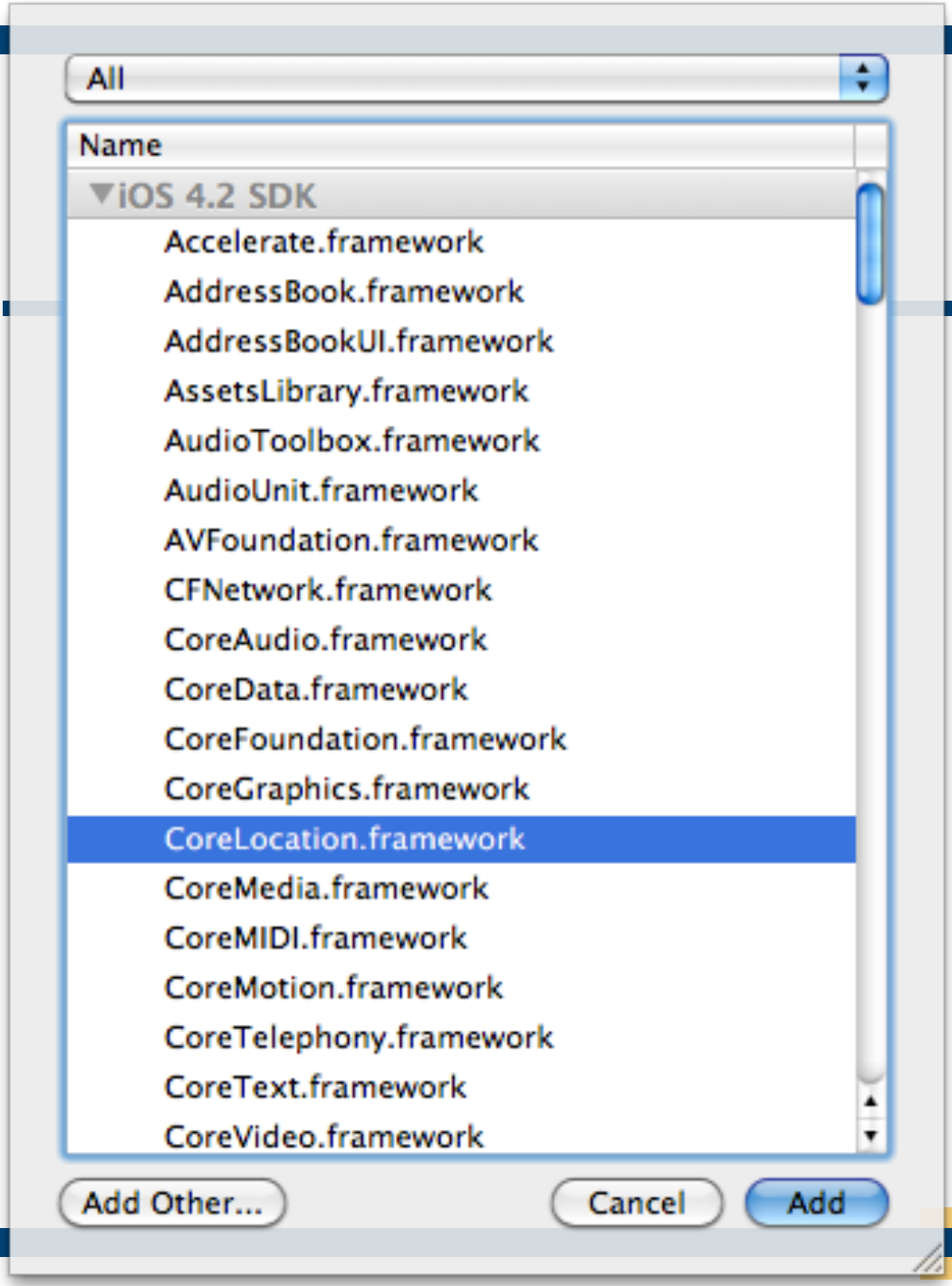
- iOS定位服务
- iOS地图

iOS定位服务

- iOS中有三个定位服务组件：
  - Wifi定位，通过查询一个Wifi路由器的地理位置的信息。比较省电，iPod touch和iPad也可以采用。
  - 蜂窝基站定位，通过移动运用商基站定位。也适合有3G版本的iPod touch和iPad。
  - GPS卫星定位，通过3-4颗GPS定位位置定位，最为准确，但是耗电量大，不能遮挡。

Core Location

- Core Location是iPhone、iPad等开发定位服务应用程序的框架。我们要在Xcode中添加“CoreLocation.framework”存在的框架。
- 主要使用的类是：CLLocationManager，通过CLLocationManager实现定位服务。


All
Name
iOS 4.2 SDK
Accelerate.framework
AddressBook.framework
AddressBookUI.framework
AssetsLibrary.framework
AudioToolbox.framework
AudioUnit.framework
AVFoundation.framework
CFNetwork.framework
CoreAudio.framework
CoreData.framework
CoreFoundation.framework
CoreGraphics.framework
CoreLocation.framework
CoreMedia.framework
CoreMIDI.framework
CoreMotion.framework
CoreTelephony.framework
CoreText.framework
CoreVideo.framework
Add Other...
Cancel
Add

定位服务实例

实例代码
WhereAmI

WhereAmIViewController.h

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

@interface WhereAmIViewController : UIViewController <CLLocationManagerDelegate> {
    CLLocationManager *locationManager;
    UITextField *longitudeTextField;
    UITextField *latitudeTextField;
}

@property(nonatomic, retain) CLLocationManager *locationManager;
@property(nonatomic, retain) IBOutlet UITextField *longitudeTextField;
@property(nonatomic, retain) IBOutlet UITextField *latitudeTextField;

@end
```

说明

- CLLocationManagerDelegate是定位服务的委托，常用的位置变化回调方法是：

  locationManager:didUpdateToLocation:fromLocation:

  locationManager:didFailWithError:

- CLLocationManager 是定位服务管理类，通过它可以设置定位服务的参数、获取经纬度等。

m中View加载和卸载方法

```
- (void)viewDidLoad {
    self.locationManager = [[[CLLocationManager alloc] init] autorelease];
    self.locationManager.delegate = self;
    locationManager.desiredAccuracy = kCLLocationAccuracyBest;
    locationManager.distanceFilter = 1000.0f;
    [self.locationManager startUpdatingLocation];
}

- (void)viewDidUnload {
    [locationManager stopUpdatingLocation];
}
```

说明

- CLLocationManager 是的startUpdatingLocation方法启动所有定位硬件，对应的方法是stopUpdatingLocation，通过调用该方法关闭定位服务器更新，为了省电必须在不用的时候调用该方法关闭定位服务。
- 此外，我们还可以在这里设定定位服务的参数，包括：distanceFilter和desiredAccuracy。
- distanceFilter，这个属性用来控制定位服务更新频率。单位是“米”。
- desiredAccuracy，这个属性用来控制定位精度，精度越高耗电量越大。

定位精度

- desiredAccuracy精度参数可以iOS SDK通过常量实现：
  - kCLLocationAccuracyNearestTenMeters，10米
  - kCLLocationAccuracyHundredMeters ，100米
  - kCLLocationAccuracyKilometer ，1000米
  - kCLLocationAccuracyThreeKilometers，3000米
  - kCLLocationAccuracyBest ， 最好的精度
  - kCLLocationAccuracyBestForNavigation，导航情况下最好精度，iOS 4 SDK新增加。一般要有外接电源时候才能使用。

委托方法用于实现位置的更新

```
- (void)locationManager:(CLLocationManager *)manager
    didUpdateToLocation:(CLLocation *)newLocation
           fromLocation:(CLLocation *)oldLocation {

      latitudeTextField.text = [NSString stringWithFormat:@"%3.5f",
                                          newLocation.coordinate.latitude];
      longitudeTextField.text = [NSString stringWithFormat:@"%3.5f",
                                           newLocation.coordinate.longitude];
}
```

说明

- 该委托方法不仅可以获得当前位置（newLocation），还可以获得上次的位置（oldLocation ），CLLocation 对象coordinate.latitude属性获得经度，coordinate.longitude属性获得纬度。
- [NSString stringWithFormat:@"%3.5f”, newLocation.coordinate.latitude] 中的%3.5f是输出整数部分是3位，小数部分是5位的浮点数。

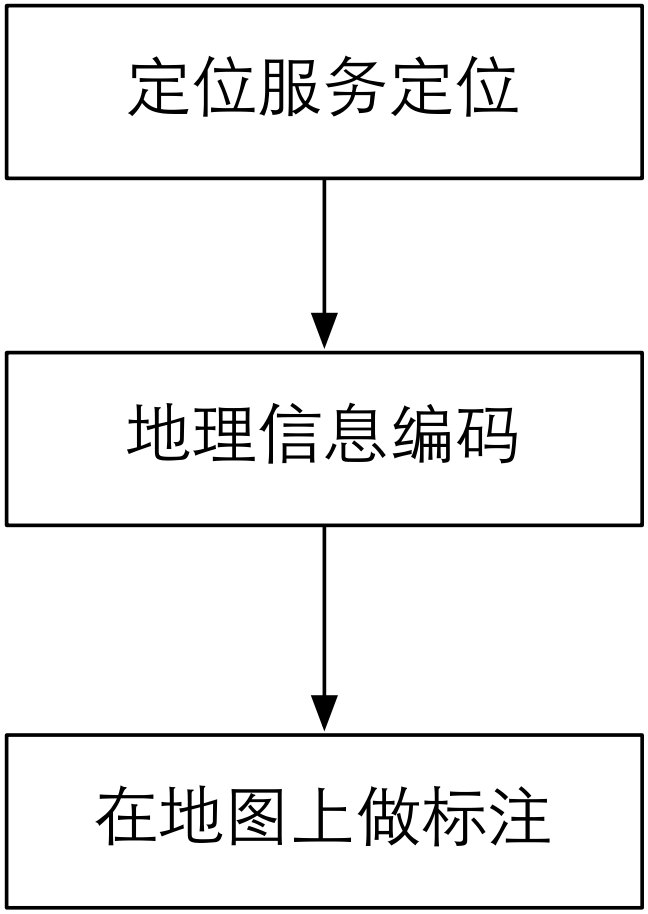
地图概述

- iOS应用程序中使用Map Kit API开发地图应用程序。
- 其核心是MKMapView类使用。
- 多数情况下地图会与定位服务结合使用。

地图实例

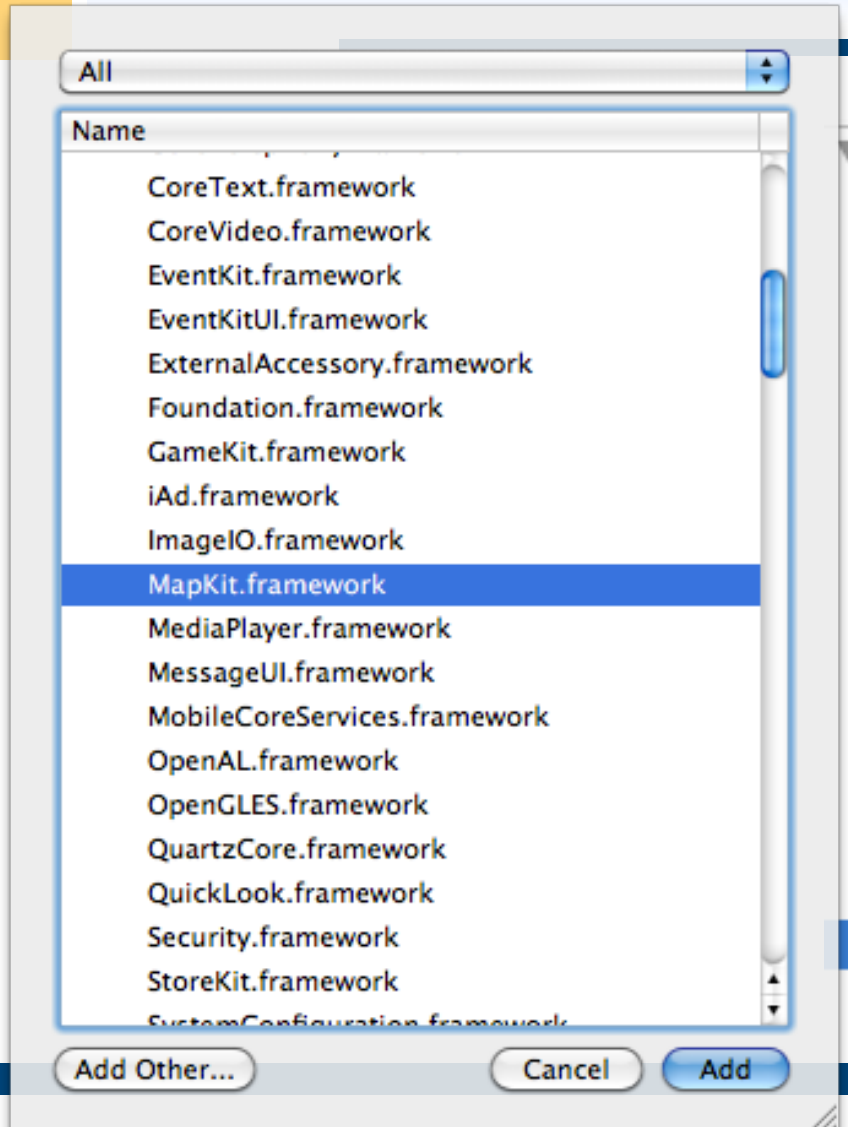
参考实例：MapMe

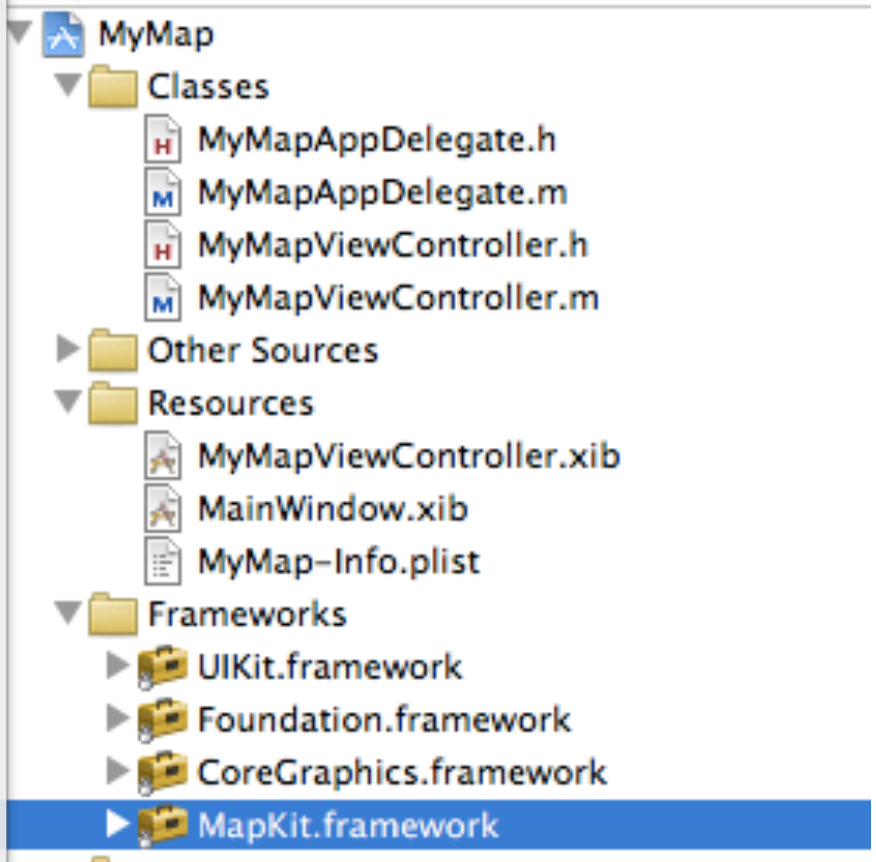
地图开发一般过程




View
MKMapView
创建工程添加
MKMapView控件。
Library
Objects
Classes
Media
Library
External Object – Provides a placeholder for an object that exists outside of the document.
Library – Cocoa Touch – Data Views
Table View – Displays data in a list of plain, sectioned, or grouped rows.
Table View Cell – Defines the attributes and behavior of cells (rows) in a table view.
Image View – Displays a single image, or an animation described by an array of images.
Web View – Displays embedded web content and enables content navigation.
Map View – Displays maps and provides an embeddable interface to navigate map content.

添加MapKit类库


All
Name
CoreText.framework
CoreVideo.framework
EventKit.framework
EventKitUI.framework
ExternalAccessory.framework
Foundation.framework
GameKit.framework
iAd.framework
ImageIO.framework
MapKit.framework
MediaPlayer.framework
MessageUI.framework
MobileCoreServices.framework
OpenAL.framework
OpenGLES.framework
QuartzCore.framework
QuickLook.framework
Security.framework
StoreKit.framework
Add Other...
Cancel
Add


MyMap
Classes
MyMapAppDelegate.h
MyMapAppDelegate.m
MyMapViewController.h
MyMapViewController.m
Other Sources
Resources
MyMapViewController.xib
MainWindow.xib
MyMap-Info.plist
Frameworks
UIKit.framework
Foundation.framework
CoreGraphics.framework
MapKit.framework

MapMeViewController.h

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

@interface MapMeViewController : UIViewController
    <CLLocationManagerDelegate, MKReverseGeocoderDelegate, MKMapViewDelegate> {
    MKMapView           *mapView;
        UIActivityIndicatorView *activityIndicationView;
}
@property (nonatomic, retain) IBOutlet MKMapView *mapView;
@property (nonatomic, retain) IBOutlet UIActivityIndicatorView *activityIndicationView;
- (IBAction)findMe;
@end
```

说明

- CLLocationManagerDelegate是定位服务委托。
- MKMapViewDelegate是地图视图委托，主要方法：
  - -mapView:viewForAnnotation:
  - -mapViewDidFailLoadingMap:withError:
- MKReverseGeocoderDelegate是给地理坐标获得标志点信息的委托，用于地理信息编码（即：从坐标获得地点获得信息），主要委托方法：
  - – reverseGeocoder:didFindPlacemark:
  - – reverseGeocoder:didFailWithError:

m文件中的视图加载和卸载

```
- (void)viewDidLoad {
    mapView.mapType = MKMapTypeStandard;
   // mapView.mapType = MKMapTypeSatellite;
        //    mapView.mapType = MKMapTypeHybrid;
        mapView.delegate = self;
}
- (void)viewDidUnload {
    self.mapView = nil;
    self.activityIndicationView = nil;
}
- (void)dealloc {
    [mapView release];
    [activityIndicationView release];
    [super dealloc];
}
```

说明

- mapView.mapType = MKMapTypeStandard;是指定地图的类型，iOS提供了三种风格的地图:
  - MKMapTypeStandard标准地图模式
  - MKMapTypeSatellite卫星地图模式
  - MKMapTypeHybrid具有街道等信息的卫星地图模式
- mapView.delegate = self;是将委托对象指定为自身。

按钮事件

```
- (IBAction)findMe {
    CLLocationManager *lm = [[CLLocationManager alloc] init];
    lm.delegate = self;
    lm.desiredAccuracy = kCLLocationAccuracyBest;
    [lm startUpdatingLocation];

      activityIndicationView.hidden = NO;
      [activityIndicationView startAnimating];
}
```

说明

- 点击按钮时候通过定位服务获取当前位置信息。通过lm.delegate = self;是将委托对象指定为自身。
- 因此，点击事件发生时候将会回调CLLocationManagerDelegate委托的-locationManager:didUpdateToLocation:fromLocation:方法。

回调位置更新方法

```
- (void)locationManager:(CLLocationManager *)manager
    didUpdateToLocation:(CLLocation *)newLocation
           fromLocation:(CLLocation *)oldLocation {

    MKCoordinateRegion viewRegion = MKCoordinateRegionMakeWithDistance
(newLocation.coordinate, 2000, 2000);
    MKCoordinateRegion adjustedRegion = [mapView regionThatFits:viewRegion];
    [mapView setRegion:adjustedRegion animated:YES];

    manager.delegate = nil;
    [manager stopUpdatingLocation];

    MKReverseGeocoder *geocoder = [[MKReverseGeocoder alloc]
initWithCoordinate:newLocation.coordinate];
    geocoder.delegate = self;
    [geocoder start];
}
```

说明

- MKCoordinateRegionMakeWithDistance (newLocation.coordinate, 2000, 2000); 该函数能够创建一个MKCoordinateRegion结构体，第一个参数是一个CLLocationCoordinate2D结构指定了目标区域的中心点，第二个是目标区域南北的跨度单位是米，第三个是目标区域东西的跨度单位是米。后两个参数的调整会影响地图缩放。

说明

- [[MKReverseGeocoder alloc] initWithCoordinate:newLocation.coordinate]; 创建地理编码对象geocoder，通过该对象可以把坐标转换成为地理信息的描述。
- geocoder.delegate = self;指定编码的处理是自身对象。
- [geocoder start];开始编码处理。

MKReverseGeocoderDelegate

- 是地理编码委托对象，该委托的方法：
  - 成功时候调用-reverseGeocoder:didFindPlacemark:
  - 失败时候调用-reverseGeocoder:didFailWithError:

成功编码回调方法

```
- (void)reverseGeocoder:(MKReverseGeocoder *)geocoder didFindPlacemark:
(MKPlacemark *)placemark {

    MapLocation *annotation = [[MapLocation alloc] init];
    annotation.streetAddress = placemark.thoroughfare;
    annotation.city = placemark.locality;
    annotation.state = placemark.administrativeArea;
    annotation.zip = placemark.postalCode;
    annotation.coordinate = geocoder.coordinate;
    [mapView addAnnotation:annotation];

   [annotation release];
    geocoder.delegate = nil;
    [geocoder autorelease];

        [activityIndicationView stopAnimating];
}
```

说明

- 成功编码后需要在该方法中创建标注对象（MapLocation）。MapLocation 是我们自定义的实现MKAnnotation协议标注对象。
- 该方法的placemark是MKPlacemark获得很多地理信息，详细见下表。
- [mapView addAnnotation:annotation]; 为地图添加标注，该方法将会触发mapView:viewForAnnotation:方法回调。

MKPlacemark类属性

addressDictionary	地址信息的dictionary
thoroughfare	指定街道级别信息
subThoroughfare	指定街道级别的附加信息
locality	指定城市信息
subLocality	指定城市信息附加信息
administrativeArea	行政区域
subAdministrativeArea	行政区域附加信息
country	国家信息
countryCode	国家代号
postalCode	邮政编码

失败编码回调方法

```
- (void)reverseGeocoder:(MKReverseGeocoder *)geocoder didFailWithError:
(NSError *)error {
    UIAlertView *alert = [[UIAlertView alloc]
                          initWithTitle:@"地理解码错误息"
                                                message:@"地理代码不能识别"
                                                delegate:nil
                          cancelButtonTitle:@"Ok"
                          otherButtonTitles:nil];
    [alert show];
    [alert release];

    geocoder.delegate = nil;
    [geocoder autorelease];

        [activityIndicationView stopAnimating];
}
```

MKMapViewDelegate

- 是地图视图委托对象，本例子我们使用的方法：
  - - mapView:viewForAnnotation:为地图设置标注时候回调方法。
  - -mapViewDidFailLoadingMap:withError:地图加载错误时候回调方法。

地图标注回调方法

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

    MKPinAnnotationView *annotationView
                = (MKPinAnnotationView *)[mapView
                      dequeueReusableAnnotationViewWithIdentifier:@"PIN_ANNOTATION"];
    if(annotationView == nil) {
                annotationView = [[[MKPinAnnotationView alloc] initWithAnnotation:annotation
                      reuseIdentifier:@"PIN_ANNOTATION"] autorelease];
    }
    annotationView.canShowCallout = YES;
    annotationView.pinColor = MKPinAnnotationColorPurple;
    annotationView.animatesDrop = YES;
    return annotationView;
}
```

说明

- 与表格视图单元格处理类似，地图标注对象由于会很多，因此需要重复利用，通过dequeueReusableAnnotationViewWithIdentifier方法可以查找可重复利用的标注对象，以达到节省内存的目的。
- annotationView.canShowCallout = YES;指定标注上的插图，点击图钉有气泡显示。
- annotationView.pinColor 设置图钉的颜色。
- annotationView.animatesDrop动画效果。

地图加载失败回调方法

```
-    (void)mapViewDidFailLoadingMap:(MKMapView *)theMapView
            withError:(NSError *)error {
    UIAlertView *alert = [[UIAlertView alloc]
                          initWithTitle:@"地图加载错误"
                                                message:[error localizedDescription]
                          delegate:nil
                          cancelButtonTitle:@"Ok"
                          otherButtonTitles:nil];
    [alert show];
    [alert release];
}
```

自定义地图标注对象

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

@interface MapLocation : NSObject <MKAnnotation, NSCoding> {
    NSString *streetAddress;
    NSString *city;
    NSString *state;
    NSString *zip;

    CLLocationCoordinate2D coordinate;
}
@property (nonatomic, copy) NSString *streetAddress;
@property (nonatomic, copy) NSString *city;
@property (nonatomic, copy) NSString *state;
@property (nonatomic, copy) NSString *zip;
@property (nonatomic, readwrite) CLLocationCoordinate2D coordinate;
@end
```

说明

- 作为地图标注对象实现MKAnnotation协议是必须的，只有实现该协议才能使该类成为标注类。实现NSCoding协议是可选的，实现该协议可以使标注对象能够复制。
- 里面的属性有哪些要看你自己的需要。

MapLocation.m

```
- (NSString *)title {
    return @"您的位置!";
}
-    (NSString *)subtitle {
    NSMutableString *ret = [NSMutableString string];
    if (streetAddress)
        [ret appendString:streetAddress];
    if (streetAddress && (city || state || zip))
        [ret appendString:@" • "];
    if (city)
        [ret appendString:city];
    if (city && state)
        [ret appendString:@", "];
    if (state)
        [ret appendString:state];
    if (zip)
        [ret appendFormat:@", %@", zip];
    return ret;
}
```

说明

- title 和subtitle 是MKAnnotation协议要求实现的方法。

MapLocation.m

```
- (void) encodeWithCoder: (NSCoder *)encoder {
    [encoder encodeObject: [self streetAddress] forKey: @"streetAddress"];
    [encoder encodeObject: [self city] forKey: @"city"];
    [encoder encodeObject: [self state] forKey: @"state"];
    [encoder encodeObject: [self zip] forKey: @"zip"];
}
- (id) initWithCoder: (NSCoder *)decoder  {
    if (self = [super init]) {
        [self setStreetAddress: [decoder decodeObjectForKey: @"streetAddress"]];
        [self setCity: [decoder decodeObjectForKey: @"city"]];
        [self setState: [decoder decodeObjectForKey: @"state"]];
        [self setZip: [decoder decodeObjectForKey: @"zip"]];
    }
    return self;
}
```

说明

- encodeWithCoder：和initWithCoder：是NSCoding协议要求实现方法。

Web地图

- 在iOS中我们还可以使用Web地图。

WhereAmI-WebMap

WhereAmIViewController.h

```
#import <UIKit/UIKit.h>
#import <CoreLocation/CoreLocation.h>
@interface WhereAmIViewController : UIViewController
<CLLocationManagerDelegate> {
        CLLocationManager *locationManager;
        UITextField *longitudeTextField;
        UITextField *latitudeTextField;
}
@property(nonatomic, retain) CLLocationManager *locationManager;
@property(nonatomic, retain) IBOutlet UITextField *longitudeTextField;
@property(nonatomic, retain) IBOutlet UITextField *latitudeTextField;

-(IBAction)go;

@end
```

WhereAmIViewController.m

```
-(IBAction)go {
    CLLocation *lastLocation = [locationManager location];
    if(!lastLocation)
    {
        UIAlertView *alert;
        alert = [[UIAlertView alloc]
                 initWithTitle:@"系统错误"
                       message:@"还没有接收到数据！"
                       delegate:nil cancelButtonTitle:nil
                 otherButtonTitles:@"OK", nil];

        [alert show];
        [alert release];
        return;
    }
```

WhereAmIViewController.m

```
    NSString *urlString = [NSString stringWithFormat:
                           @"http://maps.google.com/maps?q=Here+I+Am!@%f,%f",
                           lastLocation.coordinate.latitude,
                           lastLocation.coordinate.longitude];
    NSURL *url = [NSURL URLWithString:urlString];

    [[UIApplication sharedApplication] openURL:url];

}
```

说明

- http://maps.google.com/maps?q=Here+I+Am!@%f,%f是请求Web地图的网站，q后面上参数。
- [[UIApplication sharedApplication] openURL:url];打开iOS内置的浏览器，即在内置浏览器中打开地图。