地图及定位解释：

a.
地图和定位是两种功能，他们分属**MapKit**和**CoreLocation**框架。
要显示地图就要给ViewController添加一个**MapView**。

```
self.myMapView=[[MKMapView alloc] initWithFrame:self.view.bounds];
self.myMapView.mapType=MKMapTypeStandard;
self.myMapView.autoresizingMask=UIViewAutoresizingFlexibleWidth|UIViewAutoresizingFlexible
self.myMapView.delegate=self;
self.myMapView.userInteractionEnabled=YES;
self.myMapView.showsUserLocation=YES;
[self.myMapView setUserTrackingMode:MKUserTrackingModeFollowWithHeading animated:YES ];
[self.view addSubview:myMapView];
```

MapView是UIView的子类，它负责google(apple)地图的显示。

b.
定位的使用需要配合定位的协议和一个**CLLocationManager**的类一起进行。你需要实例化一个CLLocationManager类。

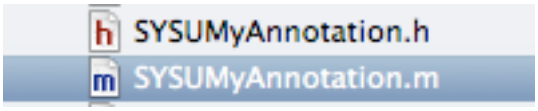
```
if ([CLLocationManager locationServicesEnabled]) {
    myManager=[[CLLocationManager alloc] init];
    self.myManager.delegate=self;
    self.myManager.purpose=@"To provide functionality base on user's location

//Location功能正式开启。其功能实现主要依靠于 location的8个协议方法，后面我只用了一个。
    [self.myManager startUpdatingLocation];
```

记得在ViewDidLoad里面使用***stopUpdatingLocation***方法）打开定位功能，这样iOS就自动开始定位服务了。相关的信息可以在协议方法里获得。

c.
地图上的标签(pin)是MapKit的一部分，它的属性一般都在MapView的协议方法下定义的。Every annotation that is added to an instance of MKMapView has a corresponding view that gets displayed on the map view.意味着你可以赋予MapView一个Annotation,然后Map会添加一个view给它，使之visual化。An annotation view is an object of type MKAnnotationView, which is a subclass of UIView（就像UIImage和UIImageView的区别）

d.
Annotation 和 **AnnotationView**的区别：Annotations are not the same as annotation views. An annotation is the location that you want to show on a map and an annotation view is the view that represents that annotation on the map.我们既可以自定义Annotation,又可以自定义MKAnnotationView，不同之处在于Annotation可以使用一个头文件＋实现文件的类来自定义，MKAnnotationView则是在协议方法里面使用系统提供的类实例化，在给它的属性赋值。

下面2个文件定义了Annotation。



实例化Annotation，然后add给mapView，如下

```
//创建SYSUAnnotation, 这是我们自定义的Annotation
SYSUMyAnnotation *annotation=[[SYSUMyAnnotation alloc] initWithCoordinate:location title:@"MyAnnotation
    subtitle:@"SubTitle"];

// 定义annotation的pin颜色属性为紫色（如果使用MKPinAnnotation的话，实际上下面的代码使用了MKAnnotation，用于我们自定义pi
//    annotation.pinColor=MKPinAnnotationColorPurple;

[self.myMapView addAnnotation:annotation];
```

这样mapView的协议方法就会被调用

```
#pragma mark - MKAnnotation Delegate
-(MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation>)annotation
```

在里面完成AnnotationView的定义，并return。这样，地图上就看到大头针了！

1.纯代码添加MapView。（代码上面有）

实例化一个MKMapView-》属性设置（推荐查document）--》addSubview（UIView及其子类都有的一个方法，MKMapView是UIView的一个子类）

地图属性之一：mapType(定义地图类型：卫星图 MKMapTypeStandard，平面图 MKMapViewTypeSatellite，卫星平面叠加图 MKMapTypeHybird)

2.MapView的协议方法有很多。

3. Get User Location(Device Location)

It belongs to CoreLocation framework, witch provides a functionality to deal with location of device base on the **CLLocationManager** class and **CLLocationManagerDelegate**.

下面的代码开启了定位功能

```
//Location功能正式开启。其功能实现主要依靠于 location的d
    [self.myManager startUpdatingLocation];
```

这个协议方法是负责定位的方法之一

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

启动后，设备会开始跟踪坐标变化，这个方法就是其中会被调用的一个。有好几个，会在不同的情况下调用，具体可以查协议方法的文档

4. Point Out A Specific Location On A Map View Using A Build-in Annotations
上面解释过了。

5. Custom Our Own Pin With Image

```
UIImage *pinImage=[UIImage imageNamed:@"Icon.png"];
if (pinImage!=nil) {
    annota.image=pinImage;
}
```

就是把我们的图片赋给AnnotationView的image属性

6.获取具体的地址

```
self.myGeocoder=[[CLGeocoder alloc] init];
[self.myGeocoder reverseGeocodeLocation:GLocation completionHandler:^(NSArray *placemarks, NSError *error) {
    if (error==nil&&[placemarks count]>0) {
        CLPlacemark *placemark=[placemarks objectAtIndex:0];
       NSLog(@"%@",placemark);
       NSLog(@"%@ : %@ : %@", placemark.subLocality,placemark.name, placemark.locality);
        annotation.title=placemark.name;
        annotation.subtitle=placemark.subLocality;
    } else if(error==nil&&[placemarks count]==0){
        NSLog(@"No Results returns");
    }else if (error!=nil){
        NSLog(@"Error=%@",error);
    }
}];
```

返回地址信息被放在CLPlacemark。其实地址信息是存在一个dictionary里面来的。然后还有许多像name,locality,country这样的属性存放小块一点的信息。

7.通过3次点击在指定处添加PIN
手势和图片动作绑定放在下面解释。

8.长按图标实现一键插萝卜。
一步就是搞定Gesture了，首先定义一个LongPress型的GestureRecognizer

```
@property (nonatomic, strong) UILongPressGestureRecognizer *insertCarrot;
```

然后是初始化并设置各个属性

```
  self.insertCarrot=[[UILongPressGestureRecognizer  alloc] initWithTarget:self action:@selector(PushCarrot:)];
  self.insertCarrot.minimumPressDuration=1.0f;
//  self.insertCarrot.numberOfTapsRequired=1;
  self.insertCarrot.delegate=self;

  [self.rightCornerView addGestureRecognizer:self.insertCarrot];
```

我们的mapView成为了target，而它的action是PushCarrot:
所以当用户长按我们的图像的时候，mapView就会使用PushCarrot:

萝卜函数如下。原理是：通过mapView拿到用户坐标，然后转换成CLLocationCoordinate2D类型。这样就能初始化我们的Annotation了，接下来就是上面提到过的代码的事情了

```
//插萝卜函数
-(void)PushCarrot:(UILongPressGestureRecognizer  *)parasender{

    //不做下面的判断，函数会被激活两次。即：长按开始一次，结束（手指离开）一次
    if(parasender.state==UIGestureRecognizerStateBegan){
        MKUserLocation *usrLocation=self.myMapView.userLocation;
        CLLocation *location=usrLocation.location;
        CLLocationCoordinate2D location2D=CLLocationCoordinate2DMake(location.coordinate.latitude, location.coordin
            longitude);
        NSLog(@"%f : %f",location2D.latitude,location2D.longitude);
        SYSUMyAnnotation *tapAnnotation=[[SYSUMyAnnotation alloc] initWithCoordinate:location2D title:@"兔子军团"
            subtitle:@"萝卜世界"];
        [self.myMapView addAnnotation:tapAnnotation];
}
}
```

9.修改Mode。

初始化时，mapView有一个userTrackingMode的属性，我们可以设置为followWithHeading，这个是实时定位方向和位置。（还有两个，一个是none，不定位，一个是follow，只定位位置） 在地图放缩移动的时候，会自动取消实时定位，mode变为none。所以我们要在用户想回到实时定位的时候变回去。只需要在合适的时候，让用户点击自我定位的按键（思路和上面一样，是个图像，绑定了GestureRecognizer），我们只要简单的把mapView的mode属性改掉就好。