MapKit

On the iPhone

MapKit ...on the iphone

Two Players:

Apple and Google

The MapKit.framework is an API to access Google-Maps on the iPhone.

Important: The MapKit framework uses Google services to provide map data. Use of this protocol and the associated interfaces binds you to the Google Maps/Google Earth API terms of service. You can find these terms of service at http://code.google.com/apis/maps/iphone/terms.html.

MapKit ...on the iphone

Relative small API

- MKMapView is central,
- MKAnnotationView and subclasses for customization,, with the MKAnnotation protocoll for own classes.
- MKReverseGeocoder and MKPlaceMark to interact with Google or the adressbook

MapKit ...on the iphone

API of the MapKit

- some special types like regions,
- heavily uses delegation,
- and protocolls,
- is not really intuitive.

MapKit ...on the iphone

MKMapView as center point provides the functionality as known from Google-Maps:

- shows maps and satelite-images
- zooming, panning and scrolling

MapKit ...on the iphone

Method to center a MKMapView

```
- (void) centerPointToFit:(CLLocation *)newPoint
{
    MKCoordinateRegion region = self.mapView.region;
    CLLocationCoordinate2D pos = newPoint.coordinate;
    CLLocationCoordinate2D top = region.center;
    CLLocationCoordinate2D bottom;
    bottom.latitude = top.latitude + region.span.latitudeDelta;
    bottom.longitude = top.longitude + region.span.longitudeDelta;

    if(!((top.latitude < pos.latitude && pos.latitude < bottom.latitude) &&
         (top.longitude < pos.longitude && pos.longitude < bottom.longitude)))
      {
        [self.mapView setCenterCoordinate:pos animated:YES];
      }
}
```

MapKit ...on the iphone

Subclassing of MKMapView is not recommended.

- Event handling is almost completely implemented.
- There is a heap of views and subviews inside a MKMapView.

MapKit ...customization

The implementation of some standard and special behavior is indirectly accomplished by using annotation.

- Standards for common tasks.
- Flexible views if necessary.
- Event handling

MapKit ...customization

Most important, conversion from view- to global-coordinates:

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {

UITouch* aTouch = [touches anyObject];
CGPoint location = [aTouch locationInView:self];

CLLocationCoordinate2D coordinate = [mapview convertPoint:location
toCoordinateFromView:self];
…
```

Don’t bother with the region.

MapKit ...customization

Annotations

- Standards for common tasks.
- Flexible views if necessary.
- Event handling is possible.

MapKit ...customization

Annotations and the MapKitView

- Annotations are subviews of the MapKitView.
- The MapKitView keeps a list of all its annotations, but not the views.
- Creation and handling of the annoation views are initiated through requests from the MapKitView to its delegate.

MapKit ...customization

MKAnnotationView is an abstract superclass to handle all kinds of annotations.

- Protocol <MKAnnotation> to handle the content of an annotation.
- methods to handle selection and the callout bubble
- reusing

MapKit ...customization

One important point: The user location is an annotation as well.

Always check:

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView
           viewForAnnotation:(id <MKAnnotation>)annotation
{
    if(annotation != mapView.userLocation) {
…
```

MapKit ...customization

MKPinAnnotationView are markers easy to implement and sufficient for simple tasks.

- marker available in 3 colors, red, green and purple
- Callout bubble shows title and subtitle of annotation
- drops can be animated.

MapKit ...customization

More sophisticated markers utilize the left and right view in the callout bubble.

- Controls and other views are available
- Handling only through identifier and tags in the delegate methods of the MKMapView
- 2 actions per annotation are no problem

MapKit

delegate method to create an annotation


```
- (MKAnnotationView *)mapView:(MKMapView *)mapView
          viewForAnnotation:(id <MKAnnotation>)annotation
{
MKPinAnnotationView *newPin = (MKPinAnnotationView *)[mapView
              dequeueReusableAnnotationViewWithIdentifier:@"annotation1"];
 if(nil == newPin) {
   newPin = [[[MKPinAnnotationView alloc] initWithAnnotation:poi
                                reuseIdentifier:@"annotation1"]autorelease];
   newPin.pinColor = MKPinAnnotationColorPurple;
   newPin.rightCalloutAccessoryView =
              [UIButton buttonWithType:UIButtonTypeInfoLight];
   newPin.rightCalloutAccessoryView.tag = kMKAnnotationViewInfo;

   newPin.leftCalloutAccessoryView =
              [self mapAnnotationButtonForImage:[poi thumbnailImage]];
   newPin.leftCalloutAccessoryView.tag = kMKAnnotationImage;
 }
 newPin.animatesDrop = YES;
 [newPin setCanShowCallout:YES];
return newPin;
}
```

with left and right call out views.

MapKit ...customization

Subclasses of MKAnnotationView can handle almost any task any view can handle.

- Drawings
- Animations
- Events inside the view
- Standard events via the callout bubble

MapKit ...customization

Example class, header:

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

@interface ExampleAnnotationView : MKAnnotationView {
    UIColor *_fillColor;
    UIColor *_strokeColor;
    CGFloat _duration;
}
@property(nonatomic, retain) UIColor *fillColor;
@property(nonatomic, retain) UIColor *strokeColor;
@property(nonatomic, assign) CGFloat duration;;

- (id)initWithAnnotation:(id <MKAnnotation>)annotation
               fillColor:(UIColor *)newFillColor
             strokeColor:(UIColor *)newStrokeColor
                    size: (CGSize)size
                duration:(CGFloat)newDuration
          reuseIdentifier:(NSString *)reuseIdentifier;
@end
```

MapKit ...customization

Attention to the initializer, because it is a subclass of MKAnnotationView:

```
- (id)initWithAnnotation:(id <MKAnnotation>)annotation
                fillColor:(UIColor *)newFillColor
              strokeColor:(UIColor *)newStrokeColor
                     size: (CGSize)size
                 duration:(CGFloat)newDuration
          reuseIdentifier:(NSString *)reuseIdentifier
{
    if(self = [super initWithAnnotation:annotation
                        reuseIdentifier:(NSString *)reuseIdentifier])
      {
        self.fillColor = newFillColor;
        self.strokeColor = newStrokeColor;
        self.duration = newDuration;
      }
    return self;
}
```

MapKit

Placemark

MKPlacemark are objects to store certain informations about a location assigned in degrees.

- conforms to the AdressBook
- Additional functionality like administrative area.
- result value of the MKReverseGeocoder

MapKit

ReverseGeocoder

MKReverseGeocoder are one-shot objects to retrieve informations about a location directly from Google.

- directly only nation and state/province are provided, as part of the MKAnnotation-protocoll
- Information is send only through a delegate (one shot).

MapKit

ReverseGeocoder

Though directly as annotation the ReverseGeocoder are not really useful, the delegate in addition receives a placemark.

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

There we go...

MapKit

ReverseGeocoder

Suggestion:
Convenience allocator method:

```
+ (MKReverseGeocoder *)reverseGeocoderWithCoords:
                                (CLLocationCoordinate2D)coords{
    MKReverseGeocoder *revGeocoder = [[MKReverseGeocoder alloc]
                                      initWithCoordinate:coords];
    return [revGeocoder autorelease];
}

+ (MKReverseGeocoder *)reverseGeocoderWithLocation: (CLLocation *)loc
{
    MKReverseGeocoder * revGeocoder = [[MKReverseGeocoder alloc]
                                  initWithCoordinate:loc.coordinate];
    return [revGeocoder autorelease];
}
```

...because they are one shot objects.

MapKit

Geocoder

A Geocoder would obtain geographical positions from an adress or similar information.

But the MapKit, where to look?

There is no such thing!

MapKit

No wait…

www.cloudmade.com

Third party framework with Geocoder, more mapstyles, additional maps and some licenses.

See for yourself

MapKit

Think different:

Phones on maps as pointing device.
locative publishing, gaming etc.

Not new in this sense, but different.