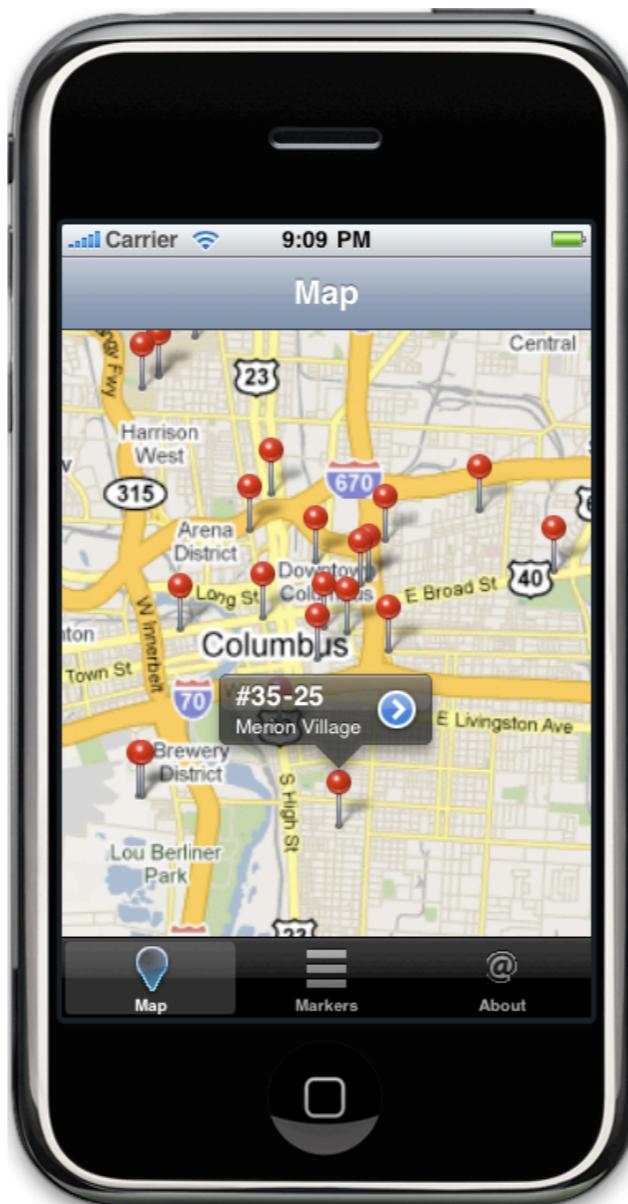


Introduction to MapKit



Christopher M. Judd
Judd Solutions

Christopher M. Judd

President/Consultant of **Judd Solutions**

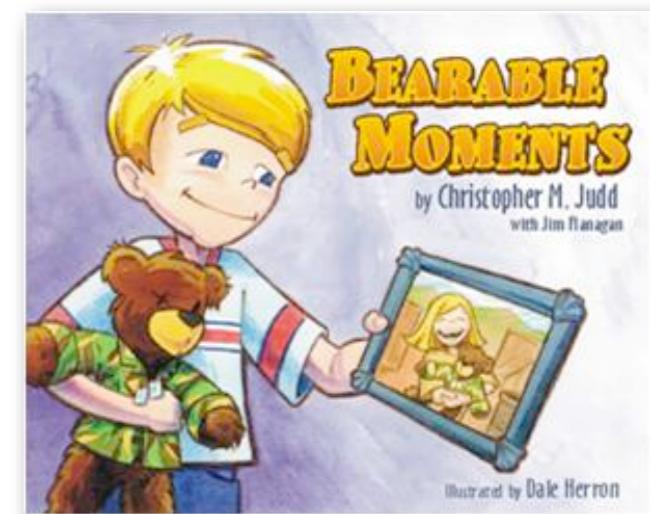
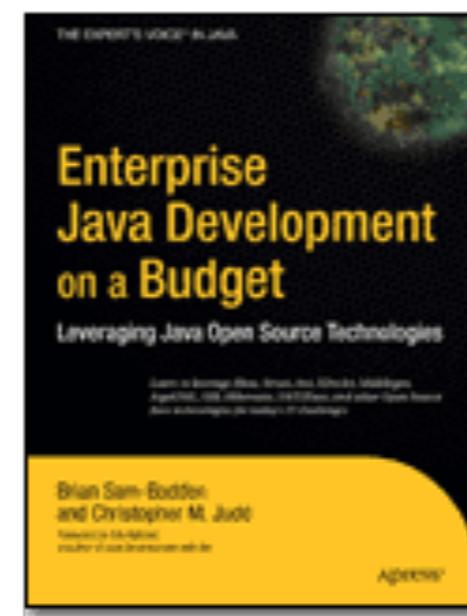
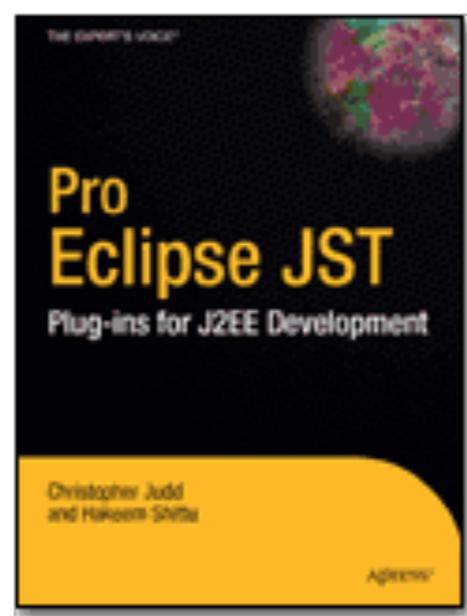
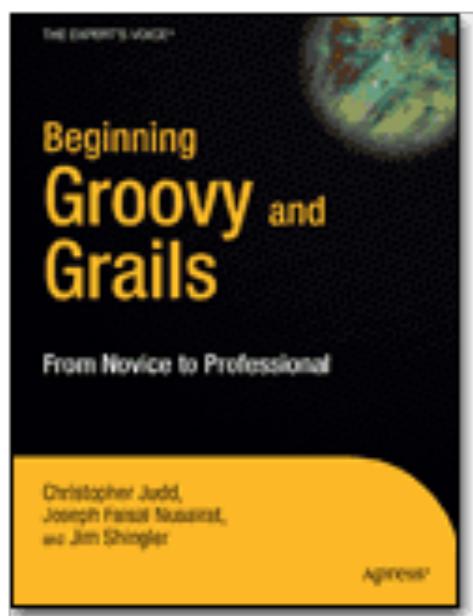


Central Ohio Java Users Group leader

Columbus



Developer User Group (CIDUG)



Remarkable Ohio

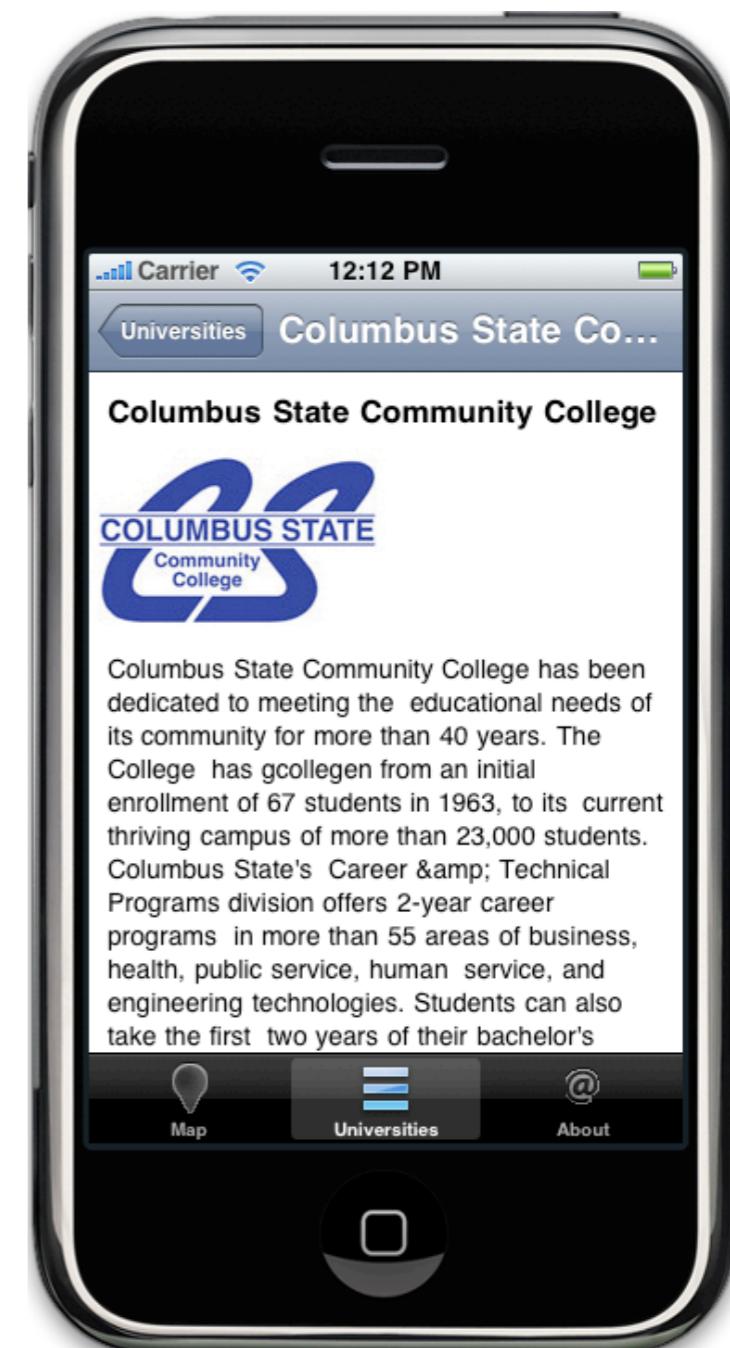
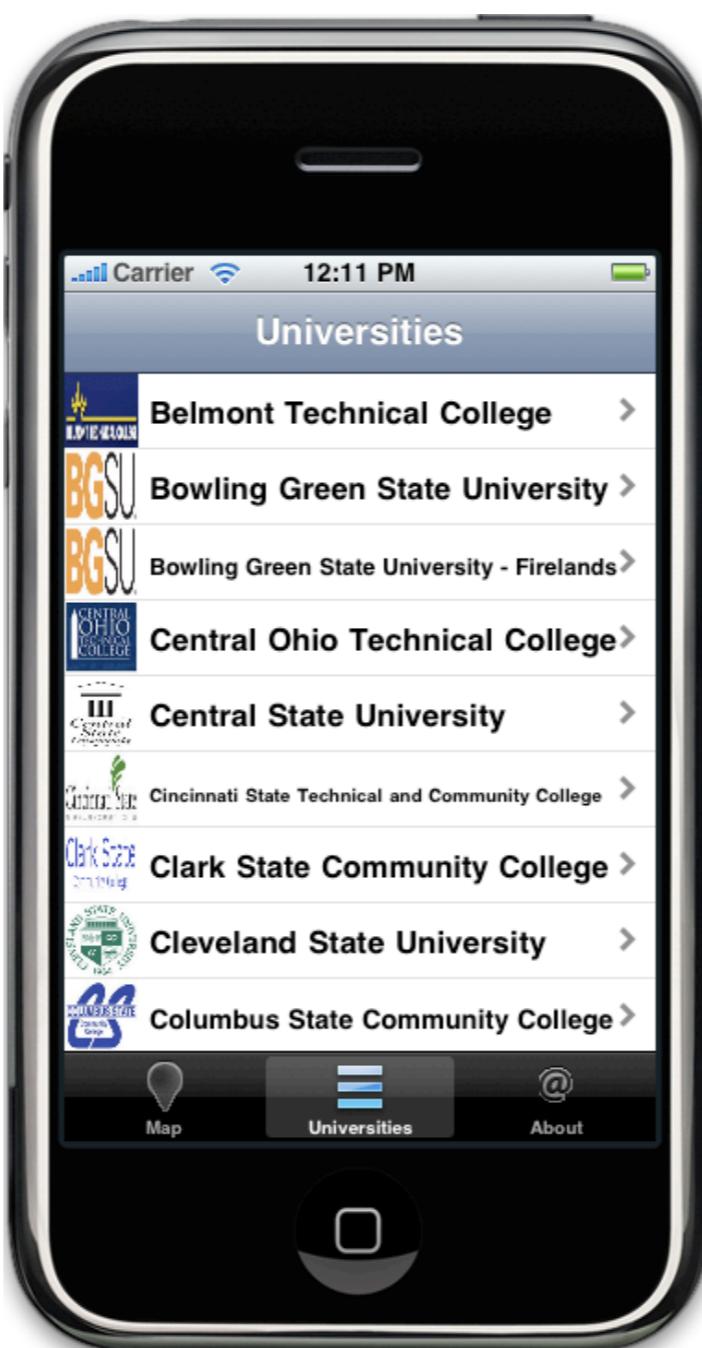
The image displays four screenshots of the Remarkable Ohio mobile application, showing its various features and a historical marker sign.

- Screenshot 1 (Left):** Shows a physical Ohio Historical Marker sign for "SITE OF OHIO'S FIRST STATEHOUSE". The sign is brown with gold lettering and is mounted on a post. The Ohio History logo and "eTech Ohio" are visible at the top.
- Screenshot 2 (Second from Left):** Shows the "Map" screen with a satellite view of Columbus, Ohio. A blue dot marks the user's location, and several red dots mark historical markers. One specific marker is highlighted with a callout: "#60-25 Ohio State School for the Blind".
- Screenshot 3 (Third from Left):** Shows the "Markers" screen, which lists historical markers categorized by county. The "Adams" category is selected, displaying markers like #05-1 Adams County Mineral Springs, #01-1 Bradford Tavern, and #19-1 Camp Hamer.
- Screenshot 4 (Right):** Shows a detailed view of the "#60-25 : Ohio State School for the Blind" marker. It includes a photograph of the marker sign, which reads "OHIO HISTORICAL MARKER" and "OHIO STATE SCHOOL FOR THE BLIND". The text on the sign describes the establishment of the school in 1832.

Free

Developed for eTech Ohio and Ohio Historical Center

University System Of Ohio

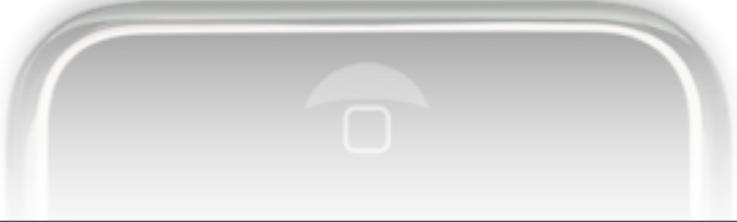


Free
Developed for eTech Ohio and University System Of Ohio

Chmod

Free

Judd Solutions



December 2008 issue

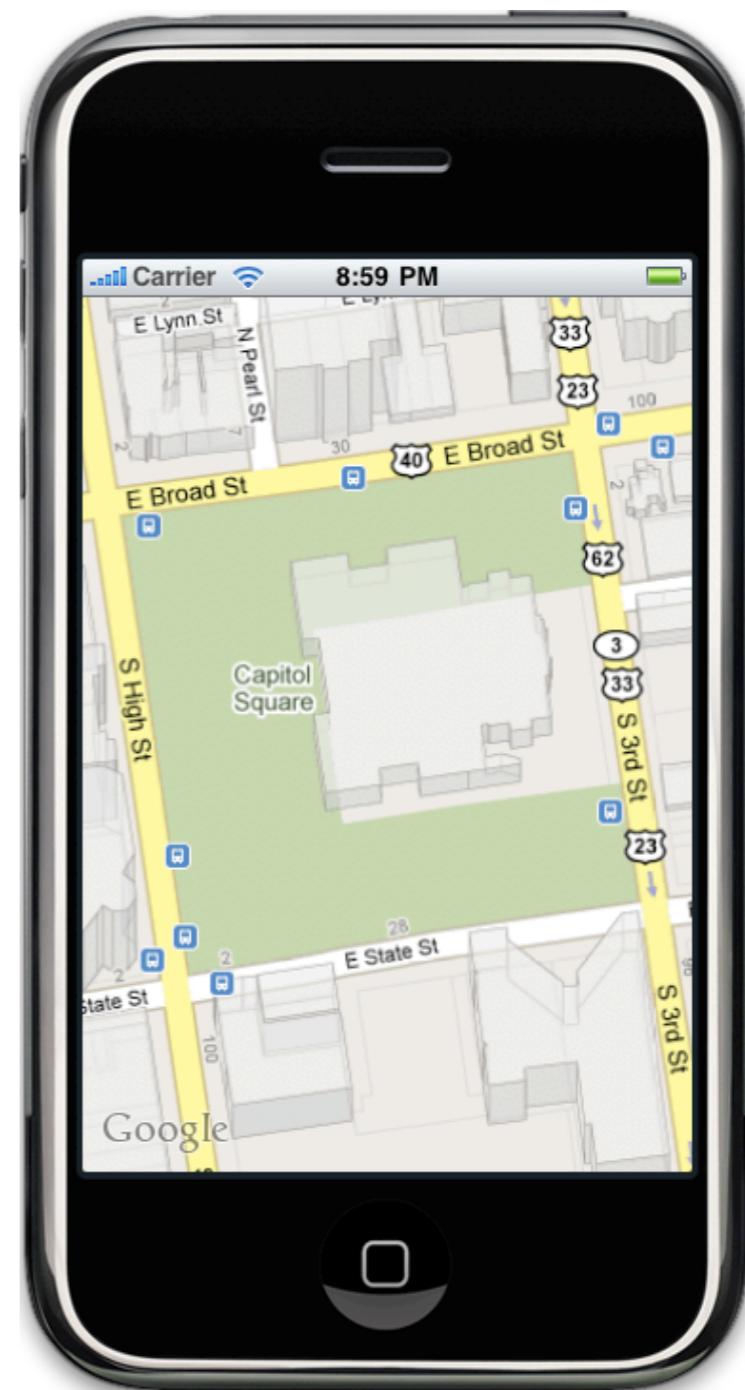


MapKit Basics

MapKit

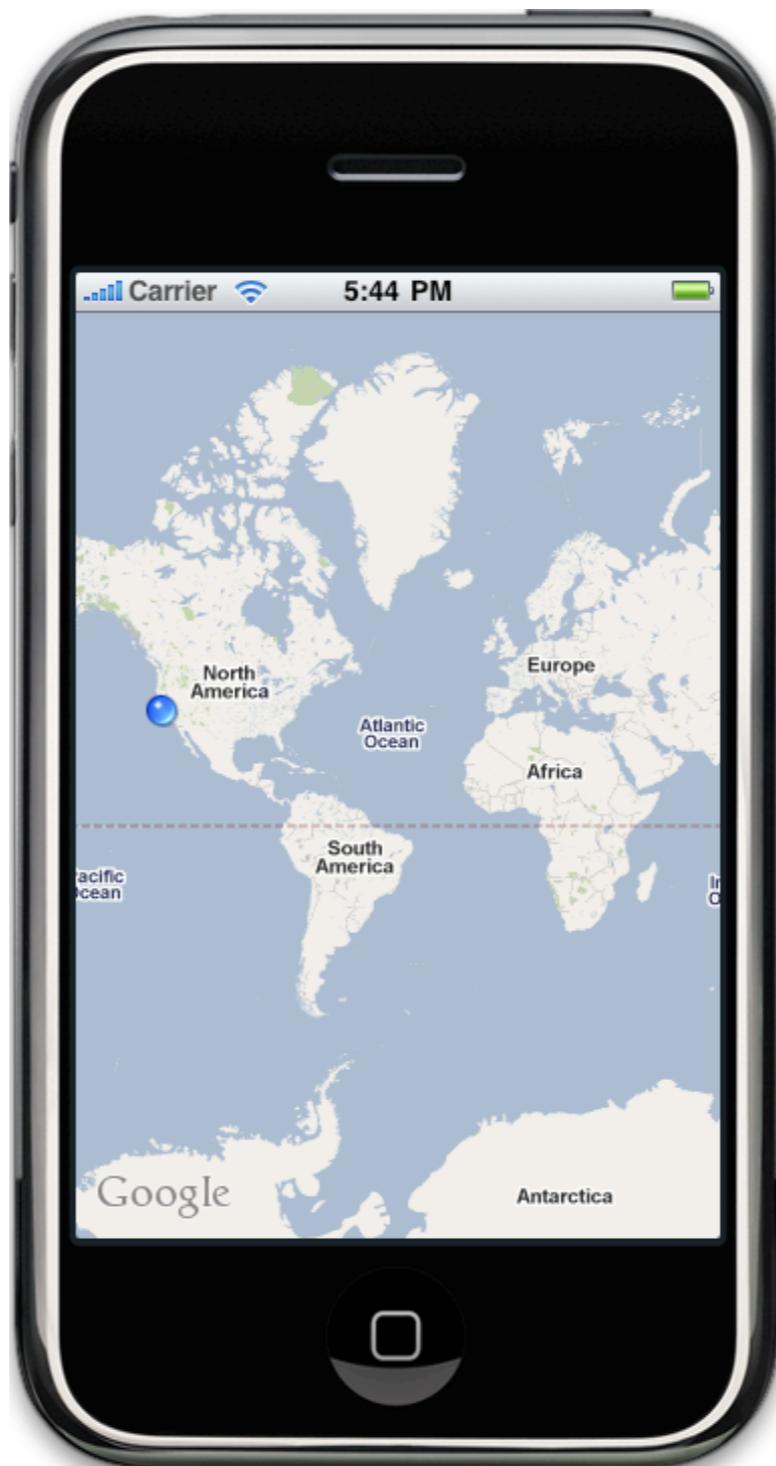


MKMapView



Google maps

Current Location

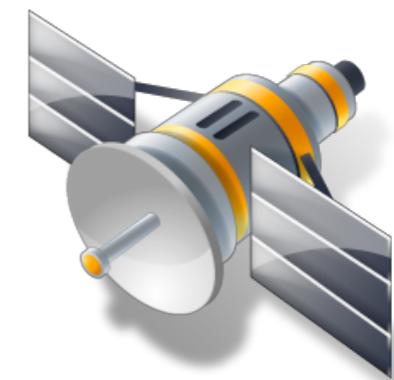


1 kilometer



100 meters

SKYHOOK
WIRELESS



10 meters

Core Location Framework

Map Types



Standard



Satellite

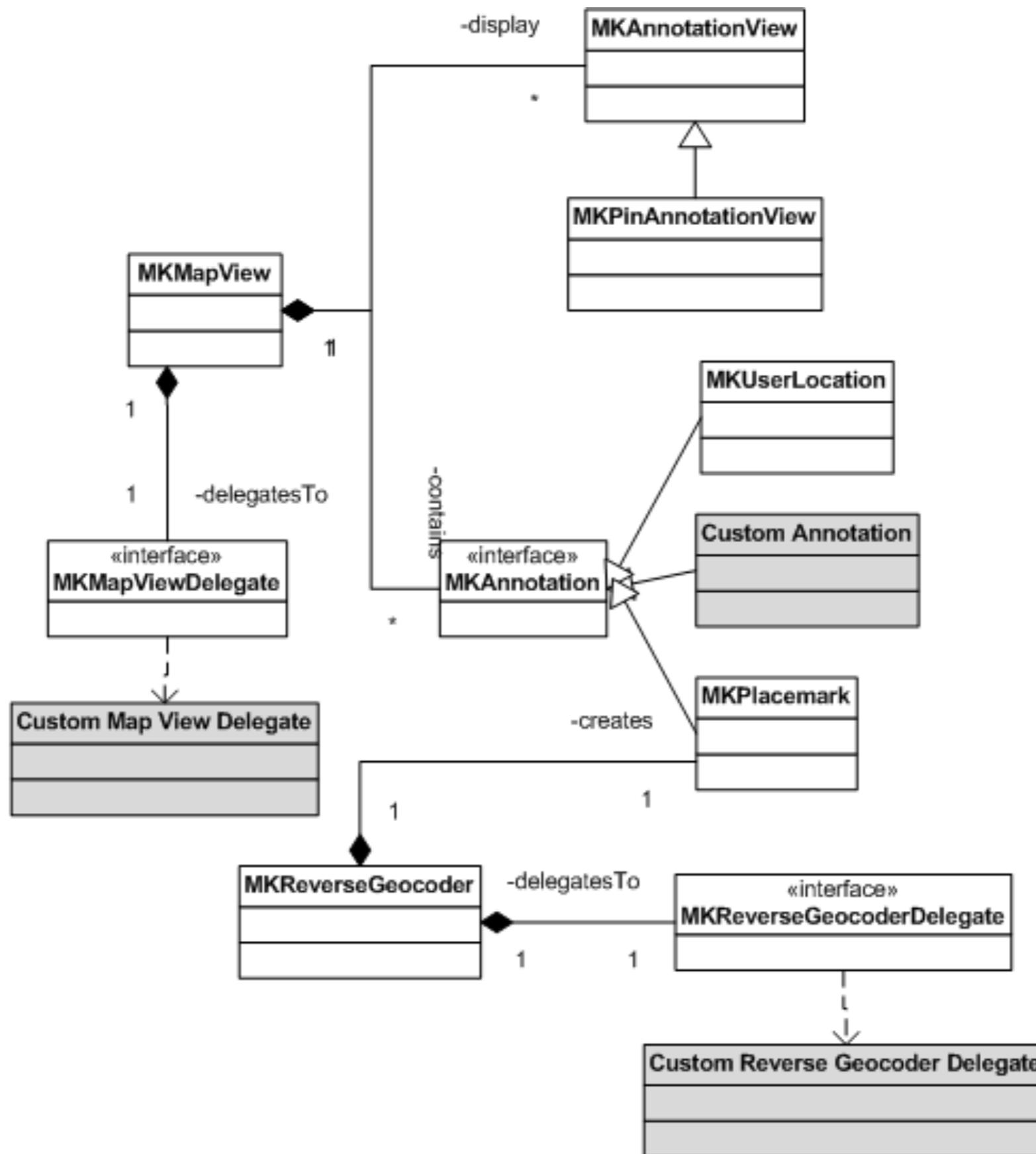


Hybrid

Annotations



Classes and Protocols



Adding Maps

Adding Maps Steps

1. Add MapView component to view 
2. Add MapKit Framework to target libraries 
3. Run Application 



Add Map View Component

The image shows the SketchUp interface. On the left is a large, empty view window with a light gray background, labeled "MKMapView". To the right is the "Library" panel, which has tabs for "Objects" and "Media". The "Objects" tab is selected. Under the "Cocoa Touch" category, there is a list of objects: "Controllers", "Data Views", "Inputs & Values", "Windows, Views & Bars", and "Custom Objects". Below this list, there are three entries:

- Web View** - Displays embedded web content and enables content navigation.
- Map View** - Displays maps and provides an embeddable interface to navigate map content.
- Text View** - Displays multiple lines of editable text and sends an action message to a target object when Return is tapped.

Below the "Map View" entry, there is a detailed description of the MKMapView class:

An **MKMapView** object provides an embeddable map interface, similar to the one provided by the Maps application. You use this class as-is to display map information and to manipulate the map contents from your application. You can center the map on a given coordinate, specify the size of the area you want to display, and annotate the map with custom information.

At the bottom of the Library panel are two buttons: a gear icon for settings and a magnifying glass icon for filtering.



Add MapKit Framework

The screenshot shows the Xcode interface with the 'ColumbusHistoryMap' target selected in the left sidebar. The main window displays the 'General' tab of the target settings.

General Tab Fields:

- Name: ColumbusHistoryMap
- Type: Application

Direct Dependencies: This section is currently empty.

Linked Libraries:

	Type
Foundation.framework	Required
UIKit.framework	Required
CoreGraphics.framework	Required
MapKit.framework	Required

A large black arrow points from the top-left towards the 'Targets' section in the left sidebar.



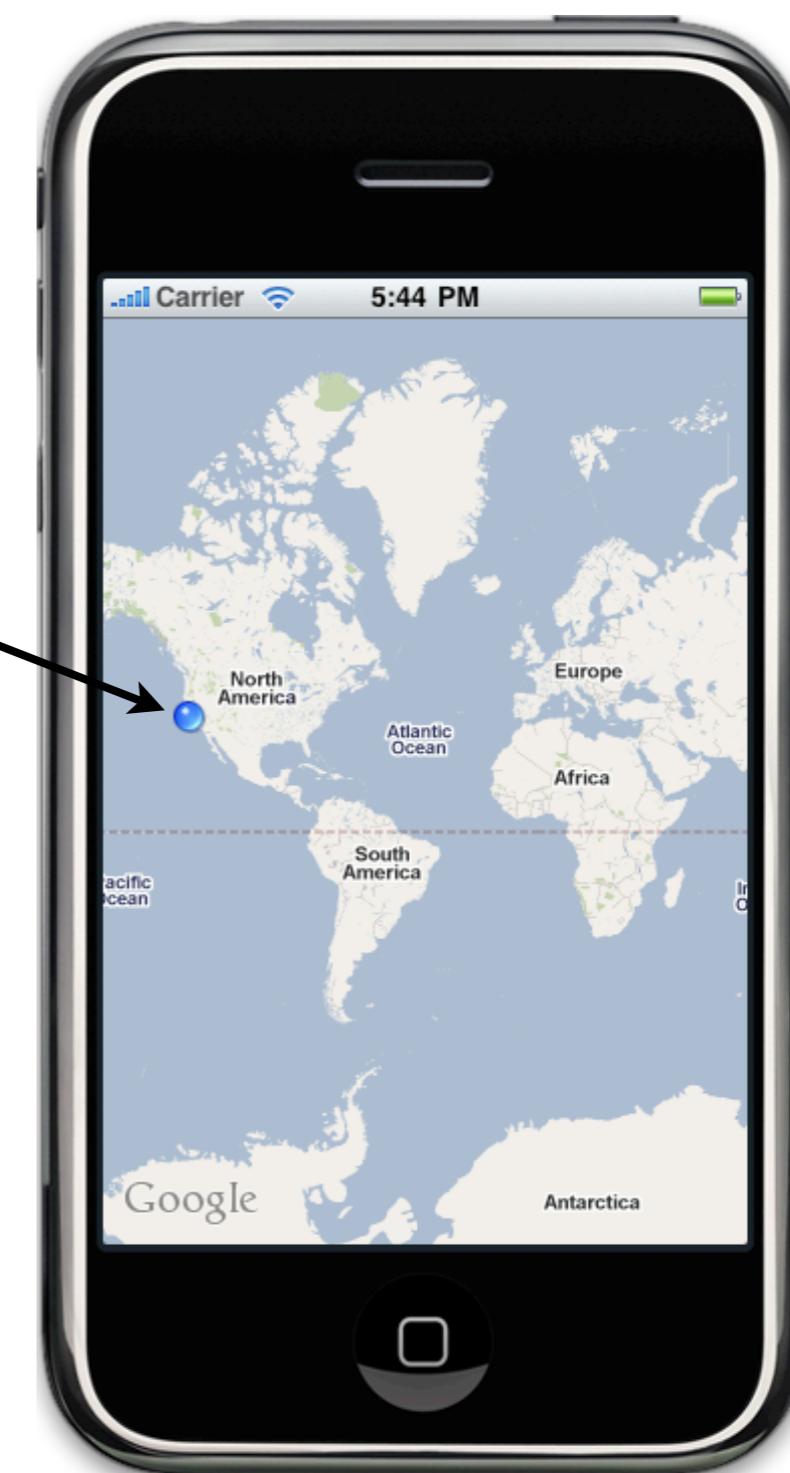
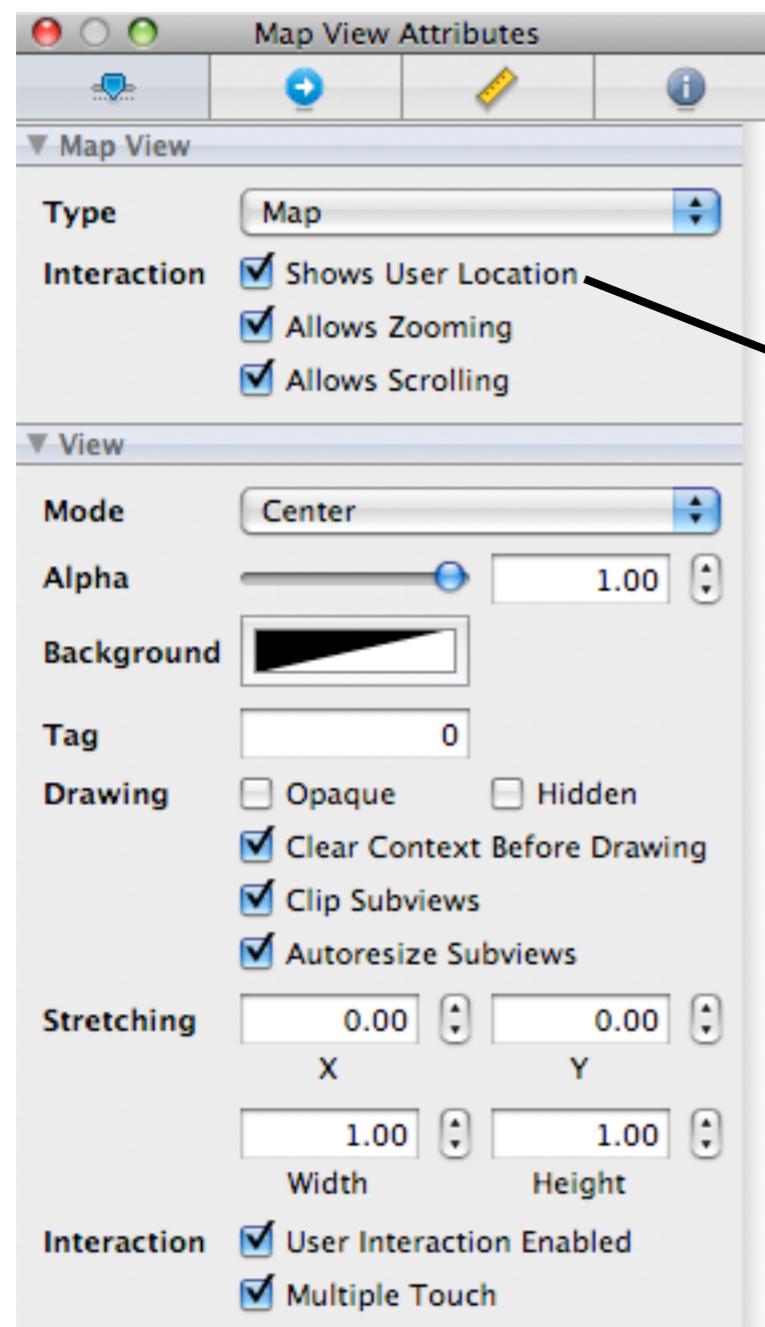
Run Application



Display Current
Location

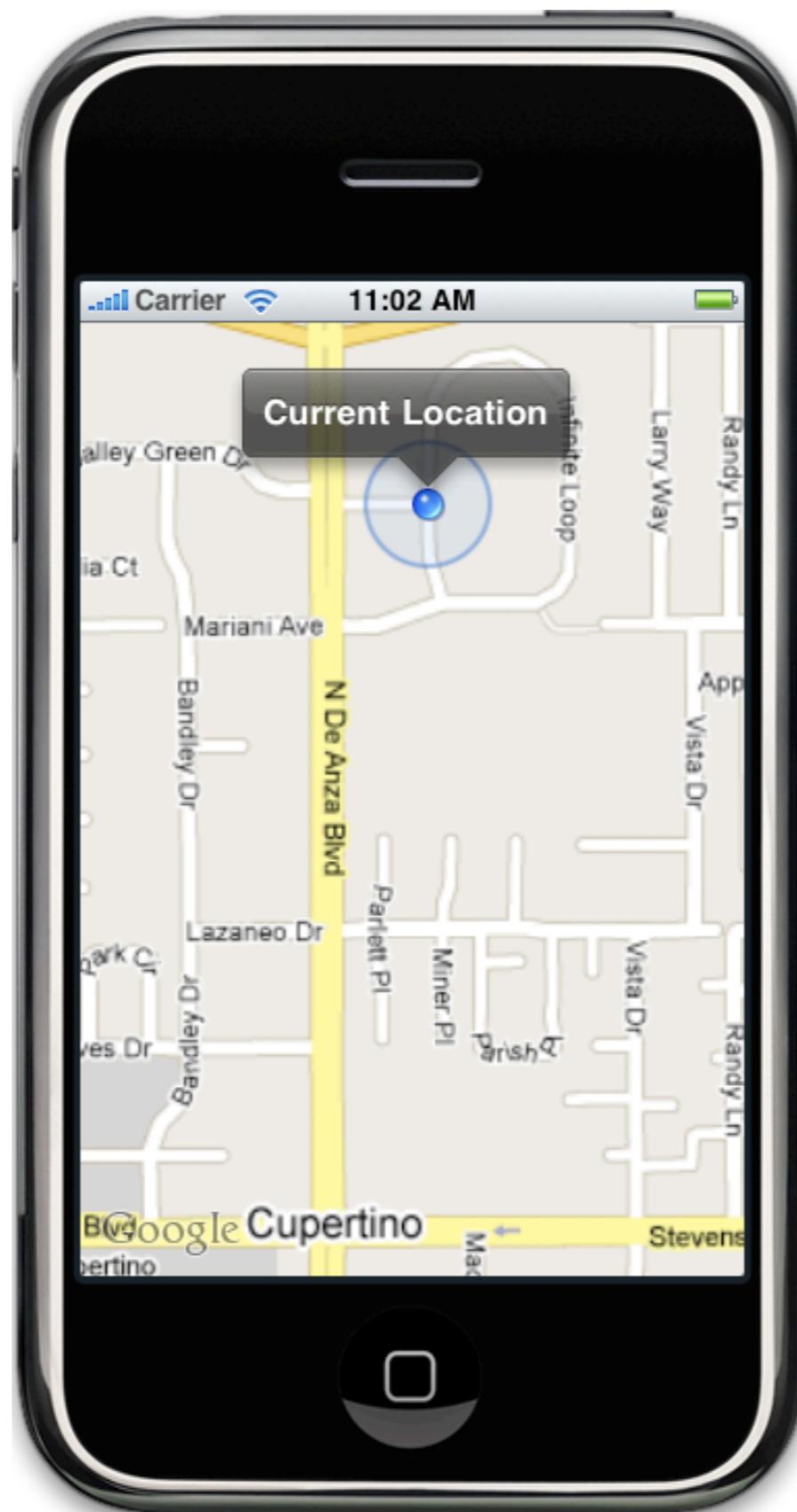


Show User Location

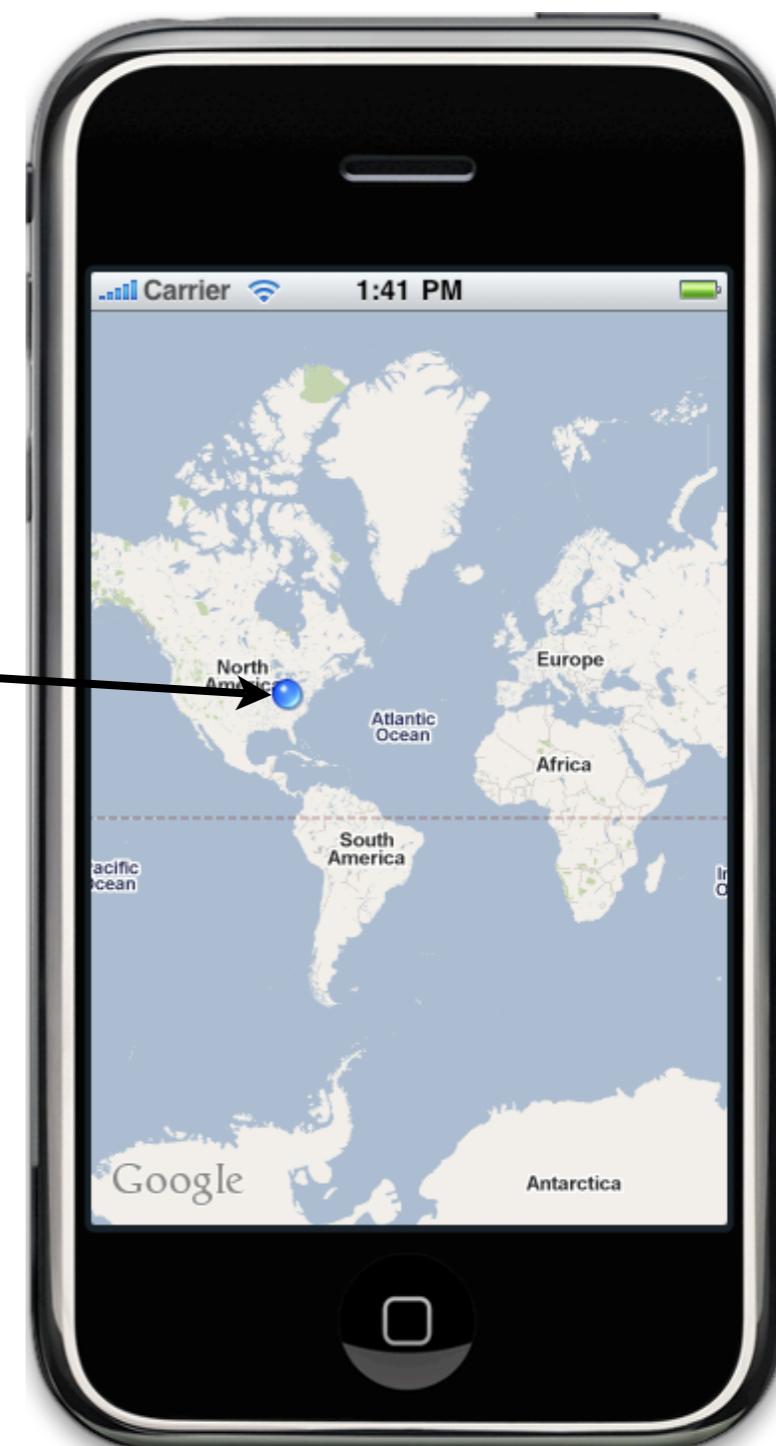
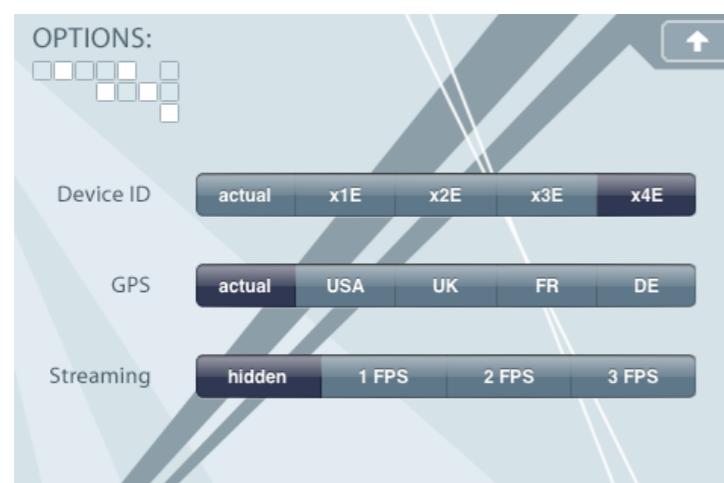
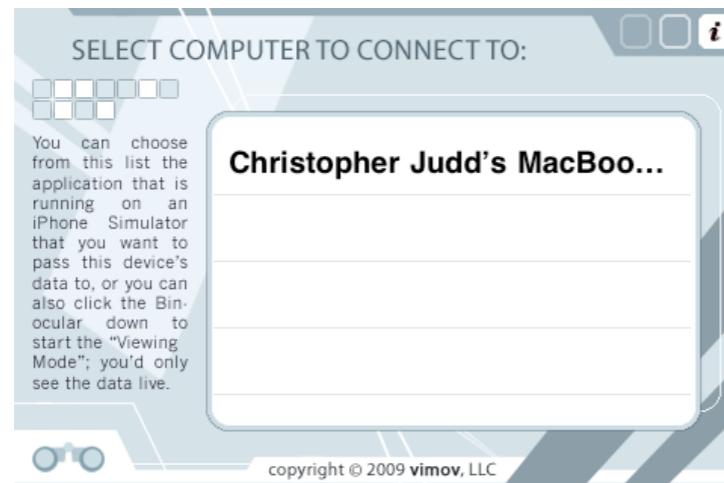




Simulator Location



iSimulate



\$9.99 in App Store

Change Map Types



Map View Attributes

Type: Map

Interaction:

- Shows User Location
- Allows Zooming
- Allows Scrolling

View

Mode: Center

Alpha: 1.00

Background:

Tag: 0

Drawing:

- Opaque
- Hidden
- Clear Context Before Drawing
- Clip Subviews
- Autoresizing Subviews

Stretching:

0.00	X	0.00	Y
1.00	Width	1.00	Height

Interaction:

- User Interaction Enabled
- Multiple Touch



Map

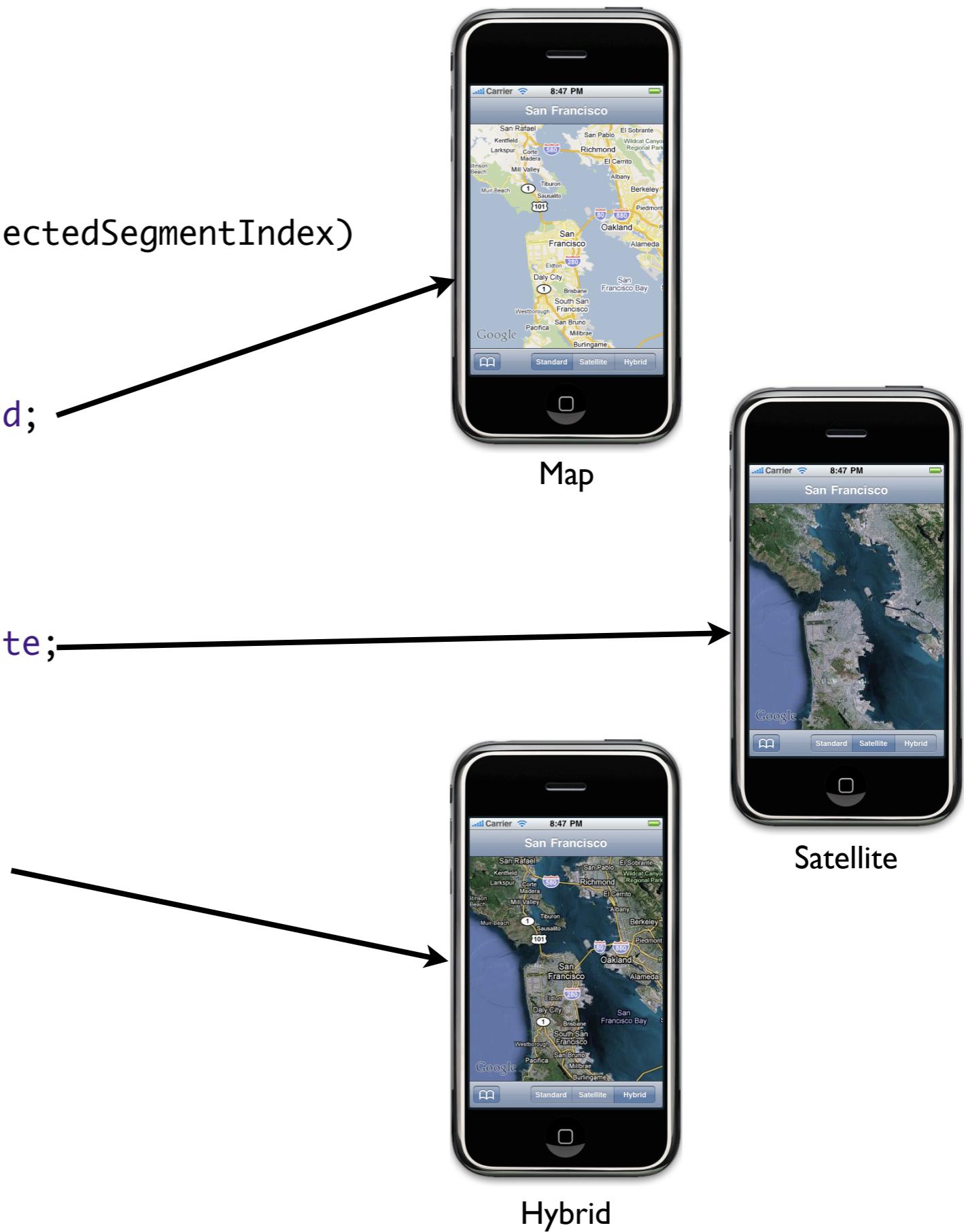


Satellite



Hybrid

```
switch (((UISegmentedControl *)sender).selectedSegmentIndex)
{
    case 0:
    {
        mapView.mapType = MKMapTypeStandard;
        break;
    }
    case 1:
    {
        mapView.mapType = MKMapTypeSatellite;
        break;
    }
    default:
    {
        mapView.mapType = MKMapTypeHybrid;
        break;
    }
}
```

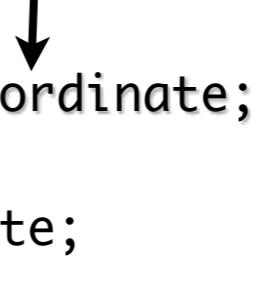
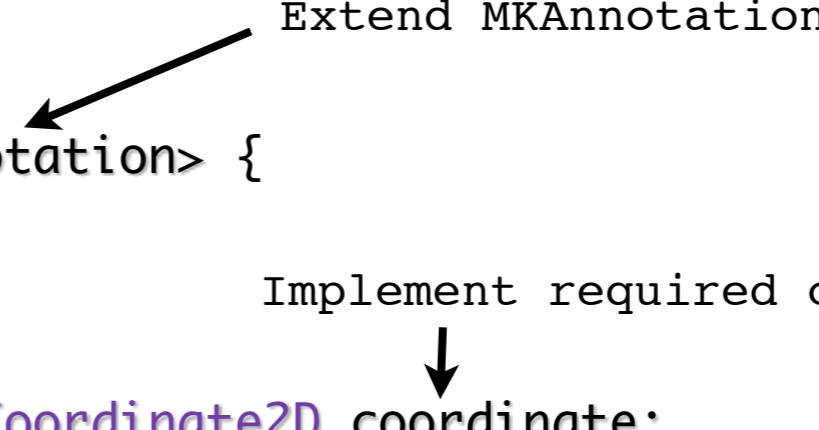


Adding Annotations

Basic Custom Annotation

HistoryMarker.h

```
@interface HistoryMarker : NSObject <MKAnnotation> {  
    CLLocationCoordinate2D _coordinate;  
}  
  
@property (nonatomic, readonly) CLLocationCoordinate2D coordinate;  
  
- (id)initWithCoordinate:(CLLocationCoordinate2D)coordinate;  
  
@end
```



HistoryMarker.m

```
@implementation HistoryMarker  
  
@synthesize coordinate = _coordinate;  
  
- (id)initWithCoordinate:(CLLocationCoordinate2D)coordinate {  
    if((self = [super init])) {  
        _coordinate = coordinate;  
    }  
    return self;  
}  
  
@end
```

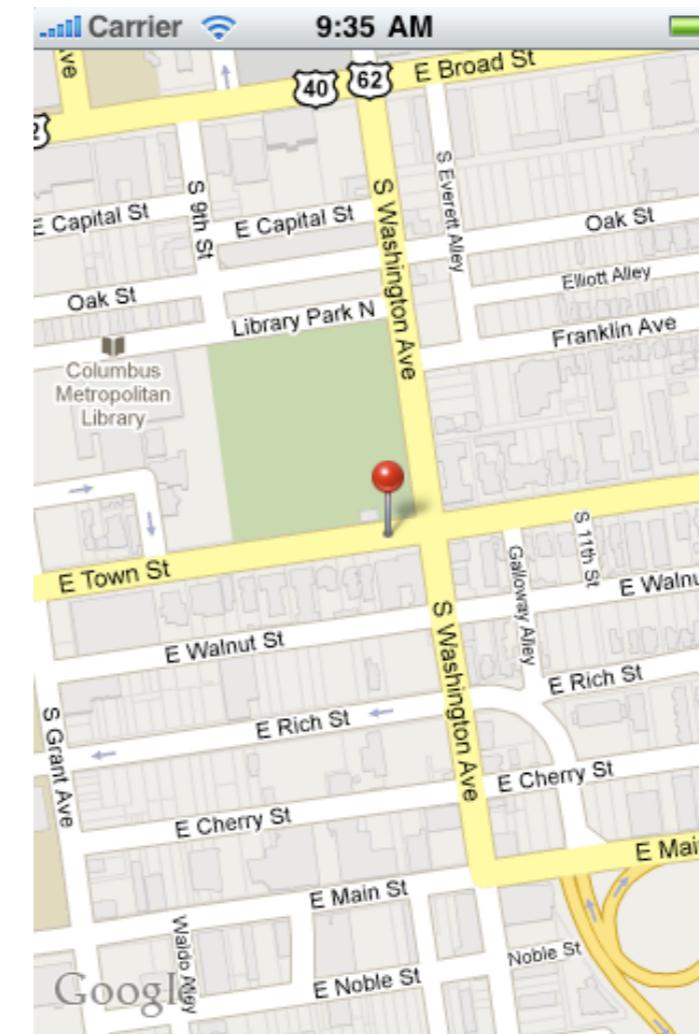
Add Annotation

*Controller.m

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

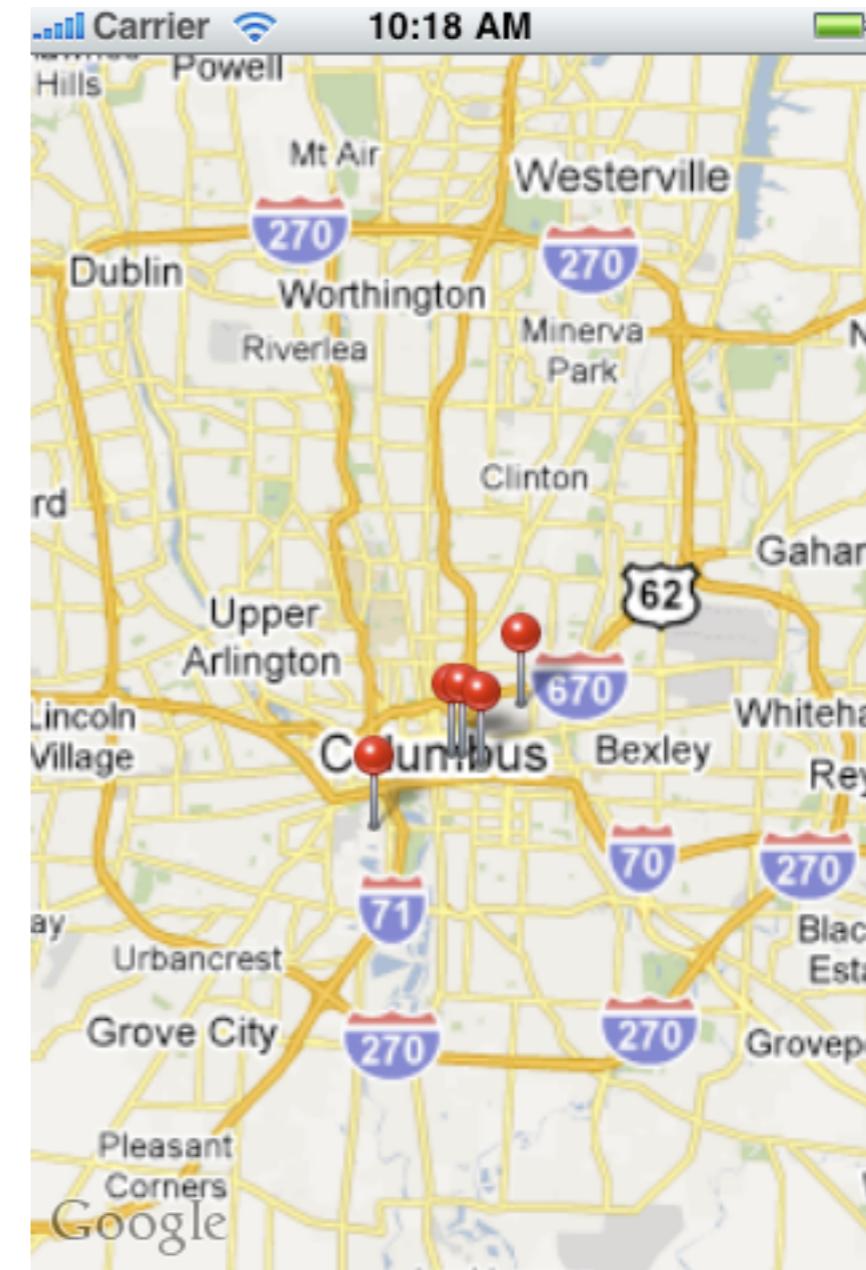
    CLLocationCoordinate2D coordinate = {39.960307, -82.98685};
    HistoryMarker* marker = [[HistoryMarker alloc] initWithCoordinate:coordinate];

    [_mapView addAnnotation:marker];
}
```



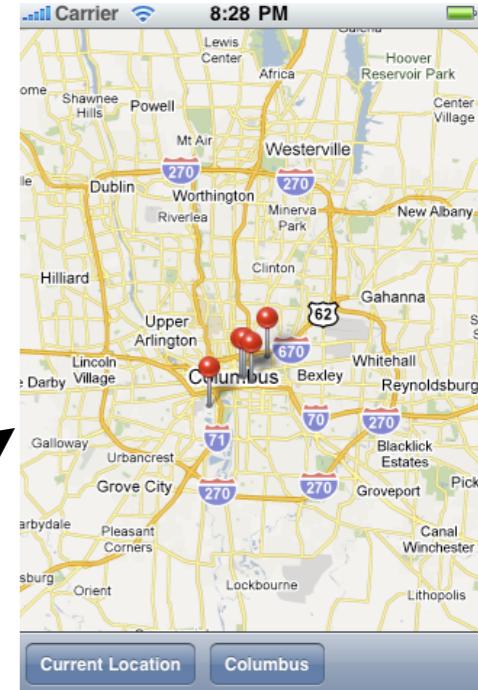
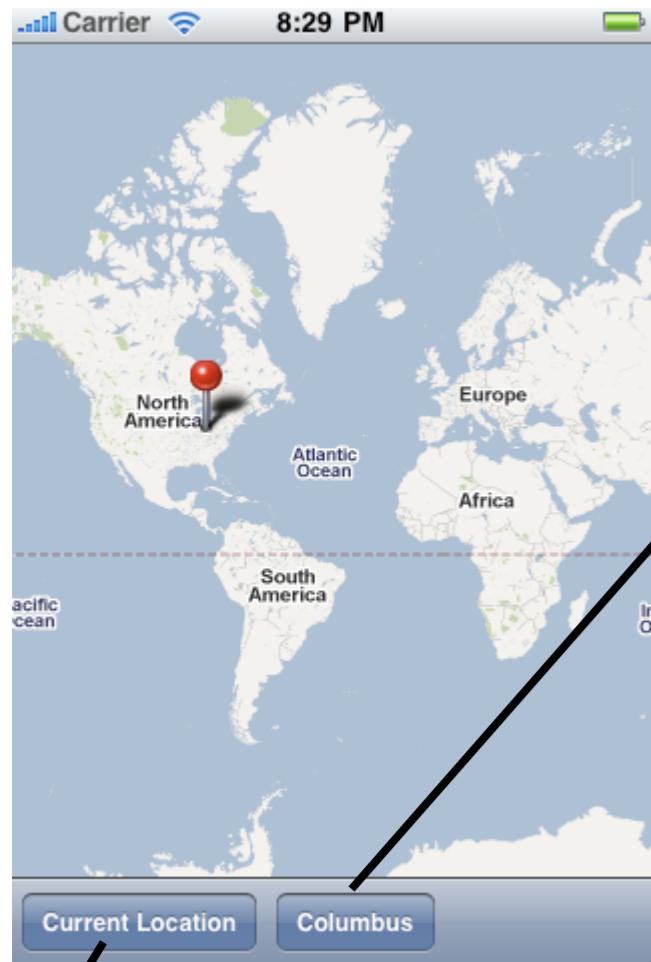
Positioning Map

Multiple Pins Can be hard to see from far away



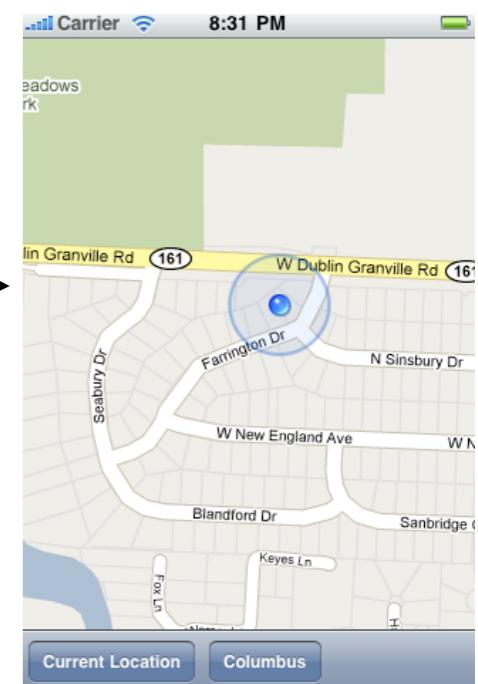
Applications with a geographical focus should frame the region

Zooming



```
- (IBAction)zoomColumbus:(id)sender {  
    CLLocationCoordinate2D columbusCenterCoordinate = {39.971793, -82.983396};  
    MKCoordinateSpan span = MKCoordinateSpanMake( 0.390456, 0.390456);  
    MKCoordinateRegion region = MKCoordinateRegionMake(columbusCenterCoordinate, span);  
    [_mapView setRegion:region animated:TRUE];  
}
```

```
- (IBAction)zoomCurrentLocation:(id)sender {  
  
    MKCoordinateRegion region = MKCoordinateRegionMakeWithDistance(  
        _mapView.userLocation.location.coordinate, 500, 500);  
    [_mapView setRegion:region animated:TRUE];  
  
}
```



Decorating Annotations

Changing Annotation Appearance

*Controller.h

```
@interface ColumbusHistoryMapViewController : UIViewController <MKMapViewDelegate> {  
    IBOutlet MKMapView* _mapView;  
}  
  
@property (nonatomic, retain) MKMapView* mapView;  
  
@end
```

Implement MKMapViewDelegate protocol



*Controller.m

```
@implementation ColumbusHistoryMapViewController  
  
- (void)viewDidLoad {  
    [super viewDidLoad];  
    _mapView.delegate = self; // Assign delegate  
    // Details removed for brevity  
}  
  
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id <MKAnnotation>)annotation {  
    if(annotation == mapView.userLocation) { return nil; } // Implement viewForAnnotation  
    // Details removed for brevity  
    return annotationView;  
}  
  
// Details removed for brevity  
  
@end
```

Assign delegate

Implement viewForAnnotation

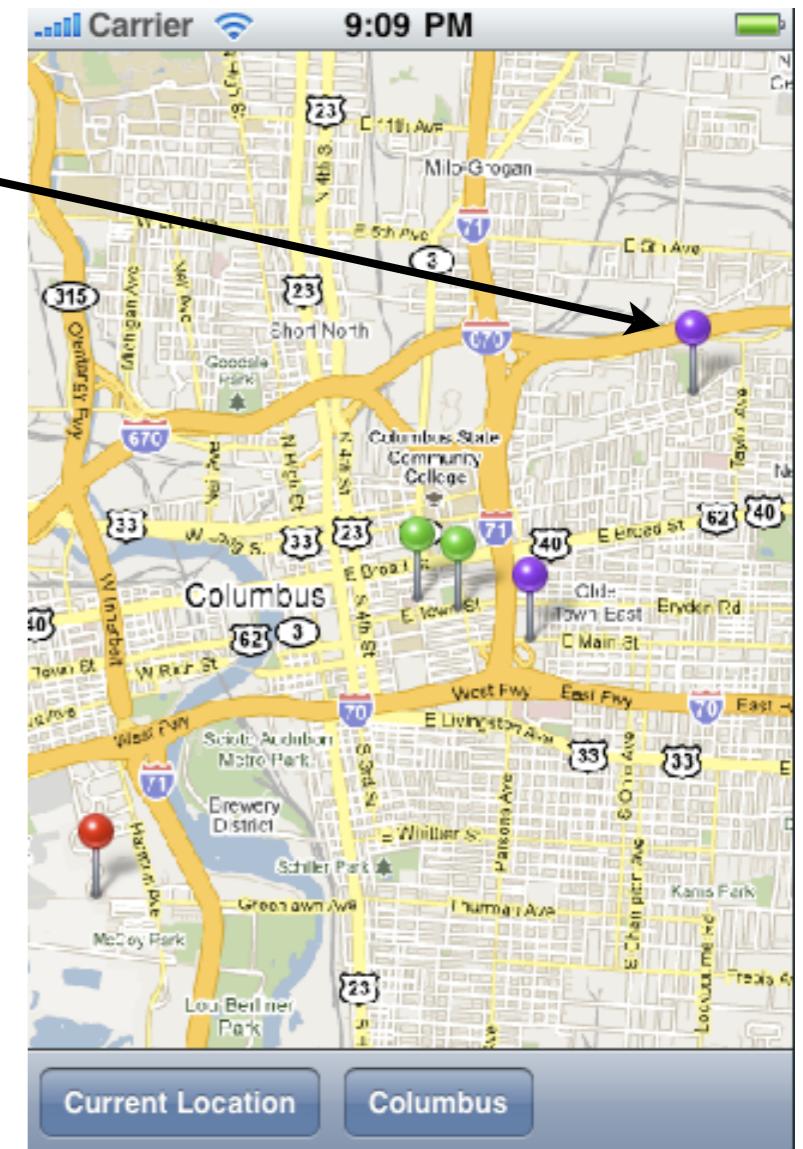
Returning nil will perform default behavior

Coloring Pins

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id <MKAnnotation>)annotation {  
    if(annotation == mapView.userLocation) { return nil; }  
  
    MKPinAnnotationView *annotationView = nil;  
    annotationView = (MKPinAnnotationView*)[mapView dequeueReusableCellWithIdentifier:@"historyMarker"];  
    if(nil == annotationView) {  
        annotationView = [[MKPinAnnotationView alloc] initWithAnnotation:annotation reuseIdentifier:@"historyMarker"];  
    }  
  
    annotationView.pinColor = MKPinAnnotationColorPurple;  
  
    return annotationView;  
}
```

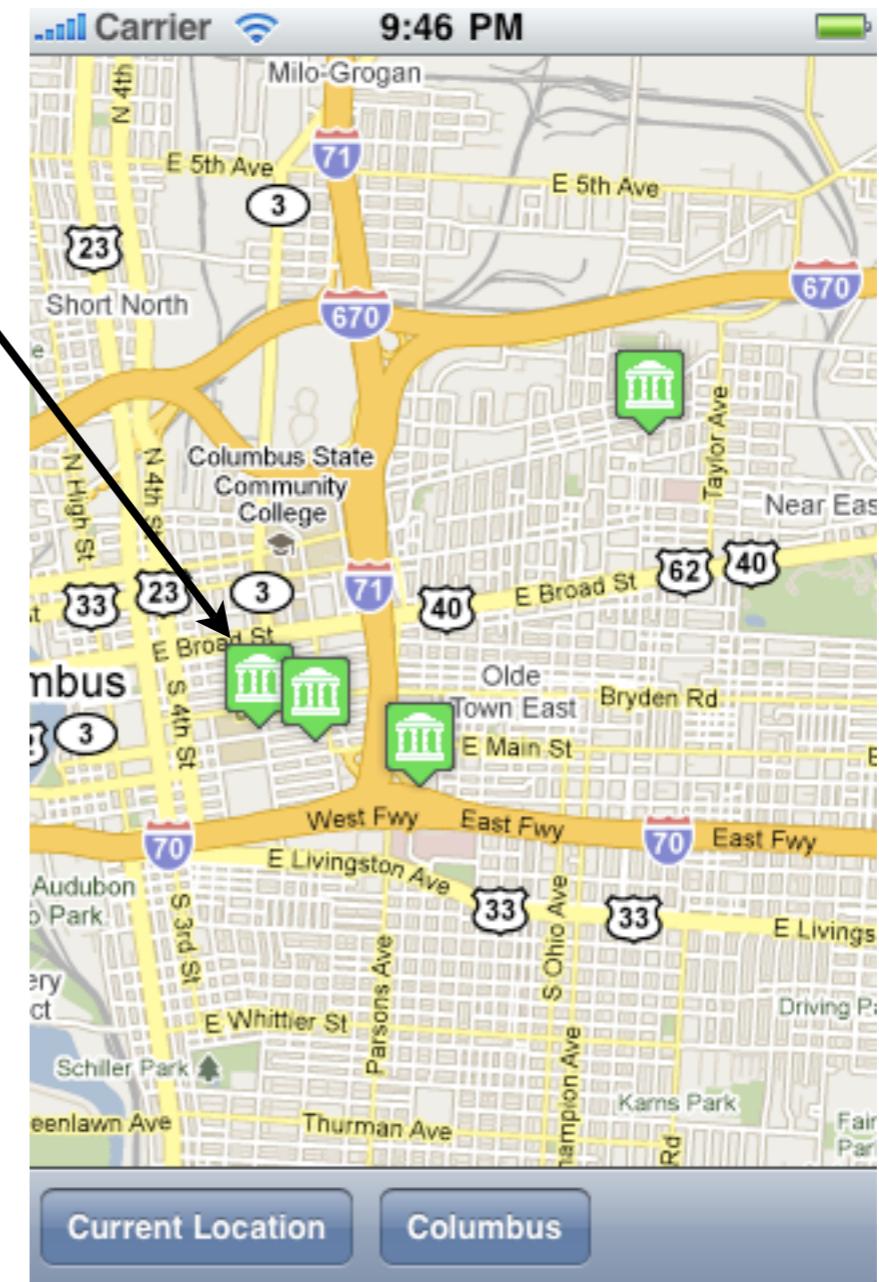
Limited to 3 colors

MKPinAnnotationColorPurple
MKPinAnnotationColorRed
MKPinAnnotationColorGreen



Annotation Images

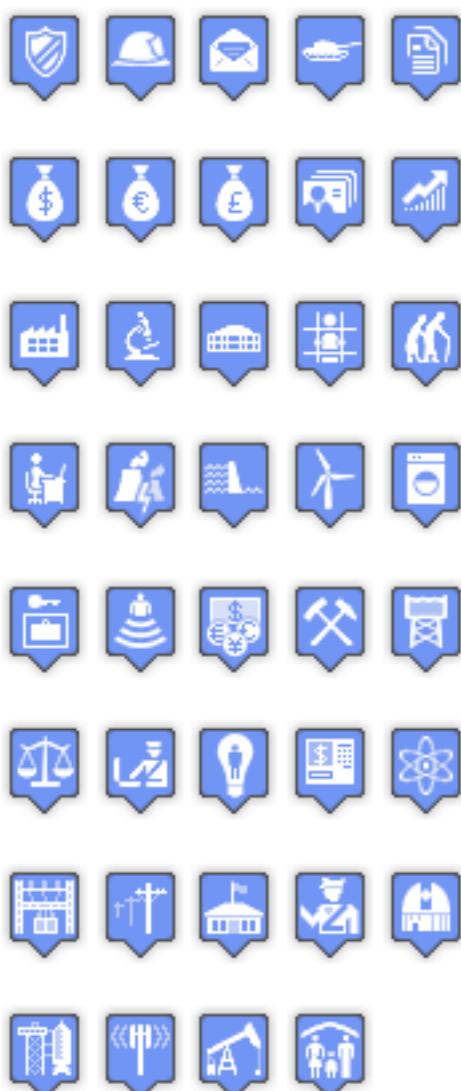
```
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id <MKAnnotation>)annotation {  
    if(annotation == mapView.userLocation) { return nil; }  
  
    MKAnnotationView *annotationView = nil;  
    annotationView = (MKAnnotationView*)[mapView dequeueReusableCellWithIdentifier:@"historyMarker"];  
    if(nil == annotationView) {  
        annotationView = [[MKAnnotationView alloc] initWithAnnotation:annotation reuseIdentifier:@"historyMarker"];  
    }  
  
    annotationView.image = [UIImage imageNamed:@"history.png"];  
  
    return annotationView;  
}
```



Free Map Icons

<http://code.google.com/p/google-maps-icons/>

Administration, Office & Industry



Culture & Entertainment



All the icons [described and tagged](#).

Education & Kids



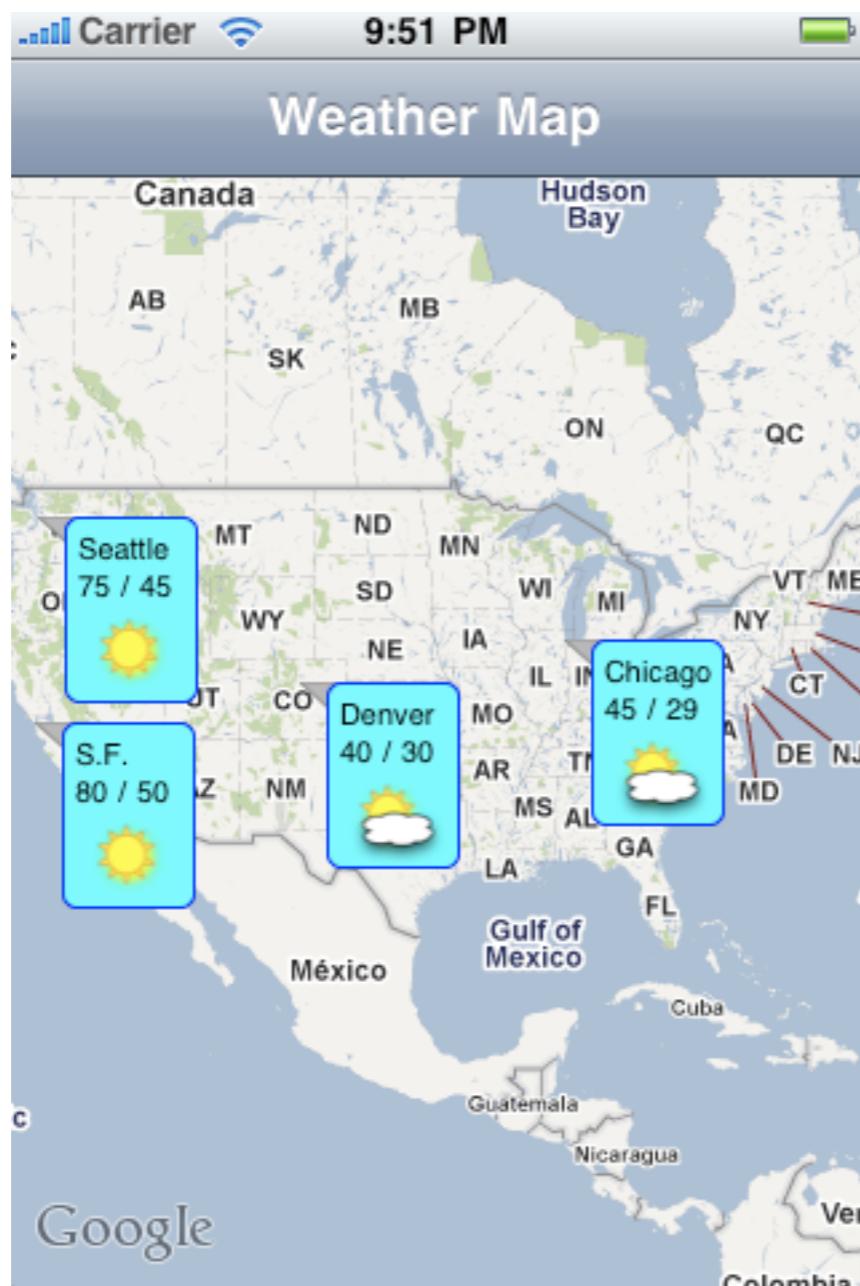
All the icons [described and tagged](#).

Events & Weather



All the icons [described and tagged](#).

All the icons in this Icon Category [described and tagged](#).



Be Creative

Callouts

Enable Callouts

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id <MKAnnotation>)annotation {  
    if(annotation == mapView.userLocation) { return nil; }  
  
    MKAnnotationView *annotationView = nil;  
    annotationView = (MKAnnotationView*)[mapView dequeueReusableCellWithIdentifier:@"historyMarker"];  
    if(nil == annotationView) {  
        annotationView = [[MKAnnotationView alloc] initWithAnnotation:annotation reuseIdentifier:@"historyMarker"];  
    }  
  
    annotationView.image = [UIImage imageNamed:@"history.png"];  
    annotationView.canShowCallout = YES;  
  
    return annotationView;  
}  
  


Implement title and  
optionally subtitle on  
MKAnnotation  
Implementation



↓



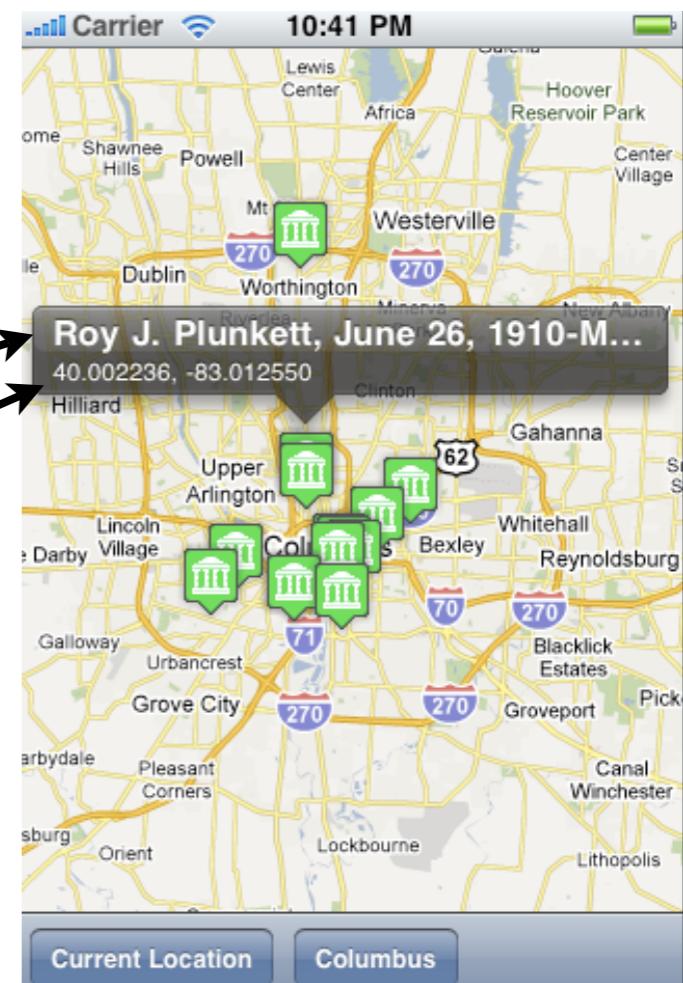
```
@implementation HistoryMarker
- (NSString *)title {
 return _name;
}

- (NSString *)subtitle {
 return [NSString stringWithFormat:@"%f, %f", _coordinate.latitude, _coordinate.longitude];
}

@end
```



Set canShowCallout


```

Add Behavior to Callouts

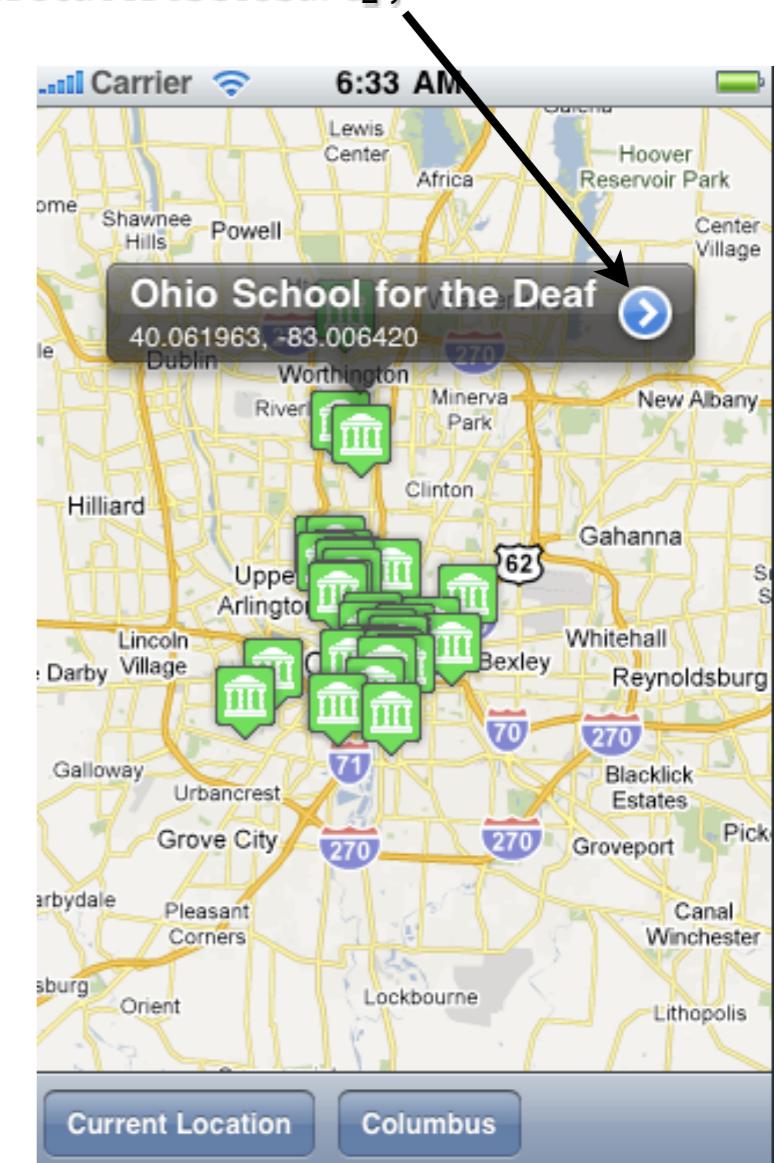
```
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id <MKAnnotation>)annotation {  
    if(annotation == mapView.userLocation) { return nil; }  
  
    MKAnnotationView *annotationView = nil;  
    annotationView = (MKAnnotationView*)[mapView dequeueReusableCellWithIdentifier:@"historyMarker"];  
    if(nil == annotationView) {  
        annotationView = [[MKAnnotationView alloc] initWithAnnotation:annotation reuseIdentifier:@"historyMarker"];  
    }  
  
    annotationView.rightCalloutAccessoryView = [UIButton buttonWithType:UIButtonTypeDetailDisclosure];  
    annotationView.image = [UIImage imageNamed:@"history.png"];  
    annotationView.canShowCallout = YES;  
  
    return annotationView;  
}
```

Implement calloutAccessoryControlTapped

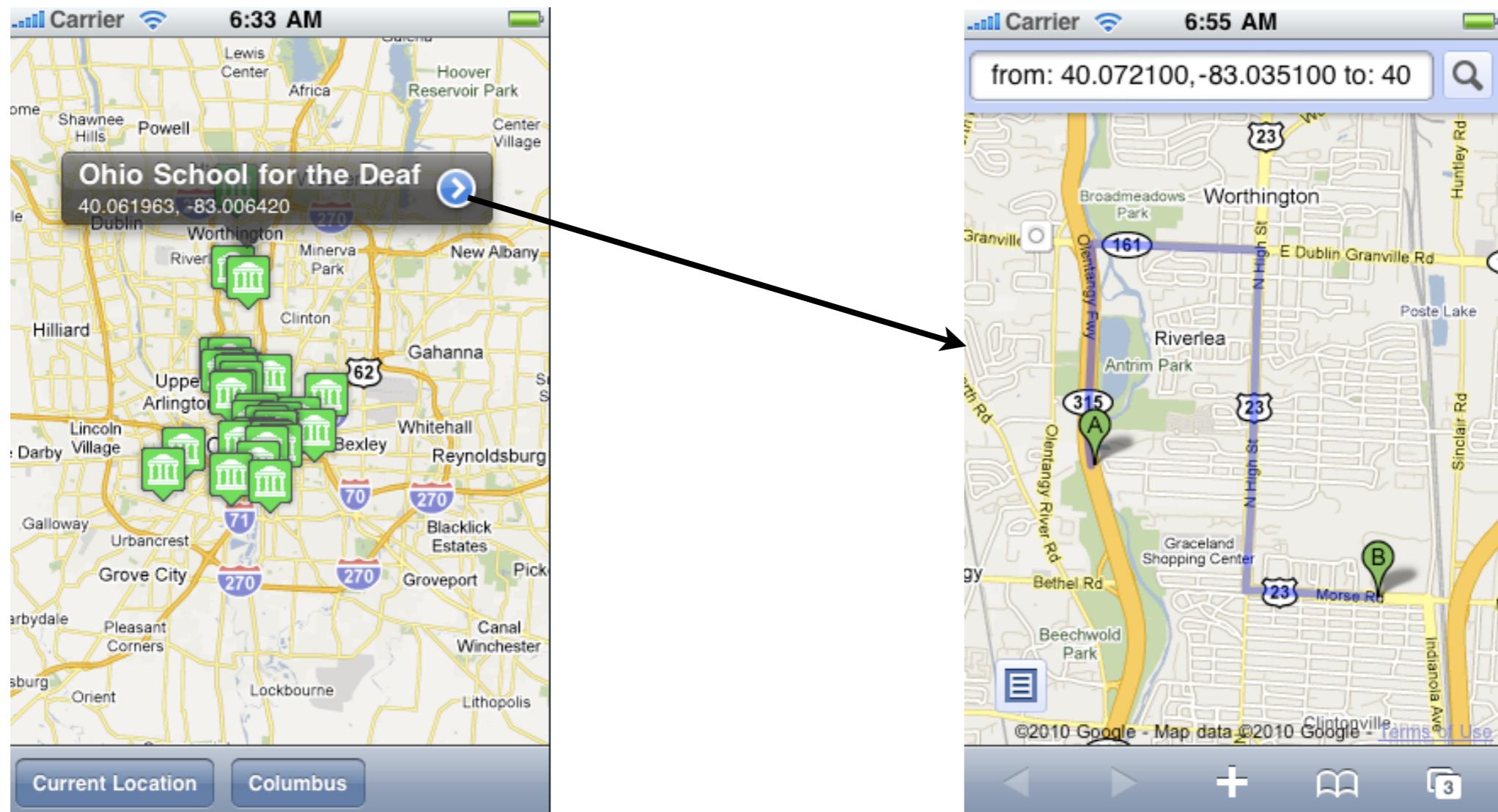


```
- (void)mapView:(MKMapView *)mapView annotationView:(MKAnnotationView *)view  
calloutAccessoryControlTapped:(UIControl *)control {
```

```
HistoryMarker* historyMarker = (HistoryMarker*)view.annotation;  
NSLog(@"Historical Marker %@ was selected.", historyMarker.name);  
//Could open a detail view, navigate to a website, call, etc.
```



Directions



```
- (void)mapView:(MKMapView *)mapView annotationView:(MKAnnotationView *)view
calloutAccessoryControlTapped:(UIControl *)control {
    HistoryMarker* historyMarker = (HistoryMarker*)view.annotation;
    NSString* url = [NSString stringWithFormat:@"http://maps.google.com/maps?daddr=%f,%f&saddr=%f,%f",
                     historyMarker.coordinate.latitude, historyMarker.coordinate.longitude,
                     _mapView.userLocation.location.coordinate.latitude,
                     _mapView.userLocation.location.coordinate.longitude];
    [[UIApplication sharedApplication] openURL:[NSURL URLWithString:url]];
}
```

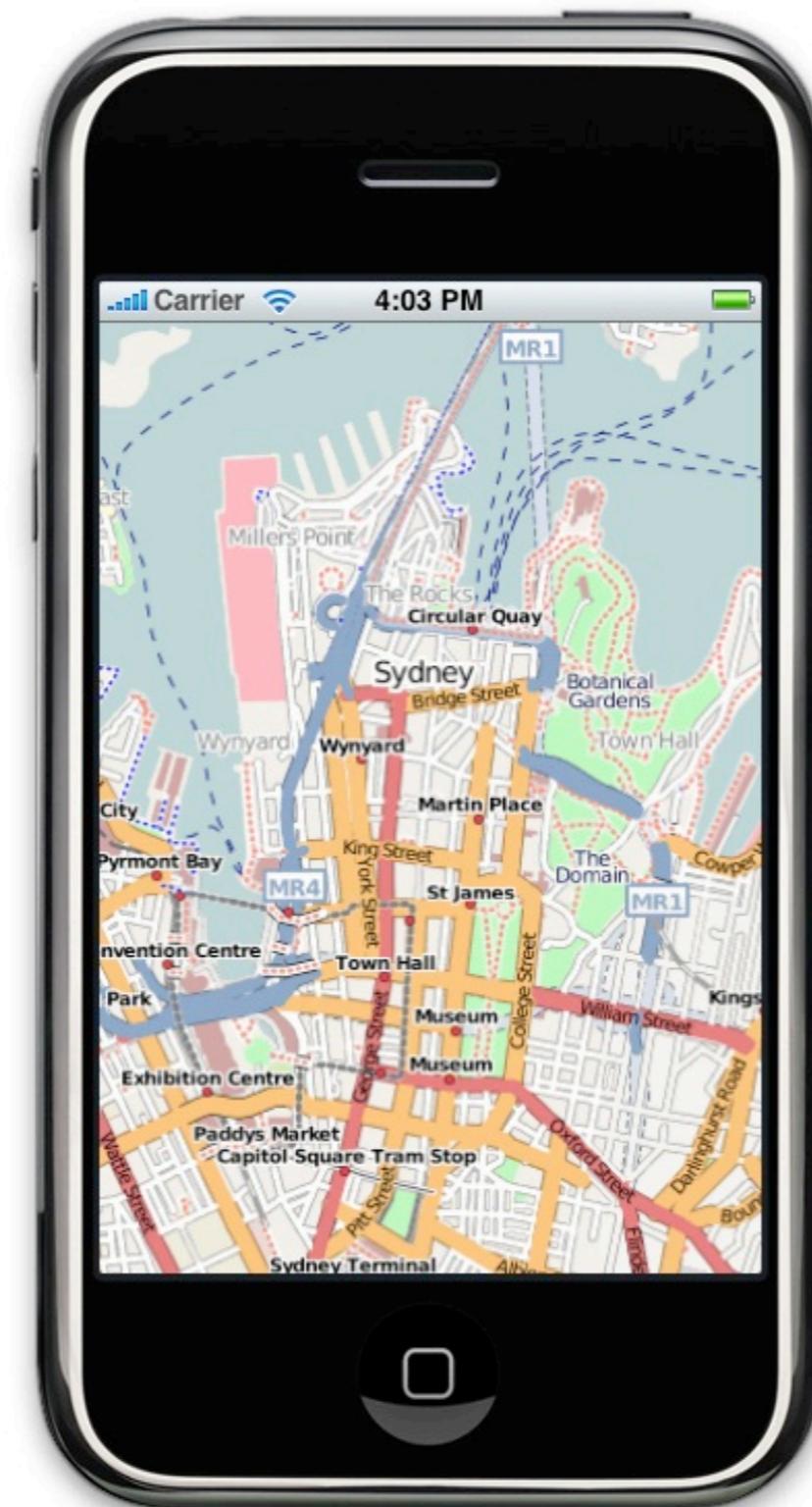
Alternatives



route-me

Open source iPhone-native slippy map.

New Free BSD



OpenStreetMap



The Free Wiki World Map

bing™



<http://code.google.com/p/route-me/>



Bring your own maps? Bring **CloudMade** Maps.

The iPhone Maps Library lets you build interactive mapping applications for the iPhone. With this API you can:

- Build applications that give users a rich mapping experience on the iPhone
- Benefit from our tile servers which deliver mobile optimized maps to your users
- Easily integrate with the iPhone's location API to show your user's position in real time
- Get more with CloudMade maps than with Google

[See some examples](#)

New Release Coming Soon...

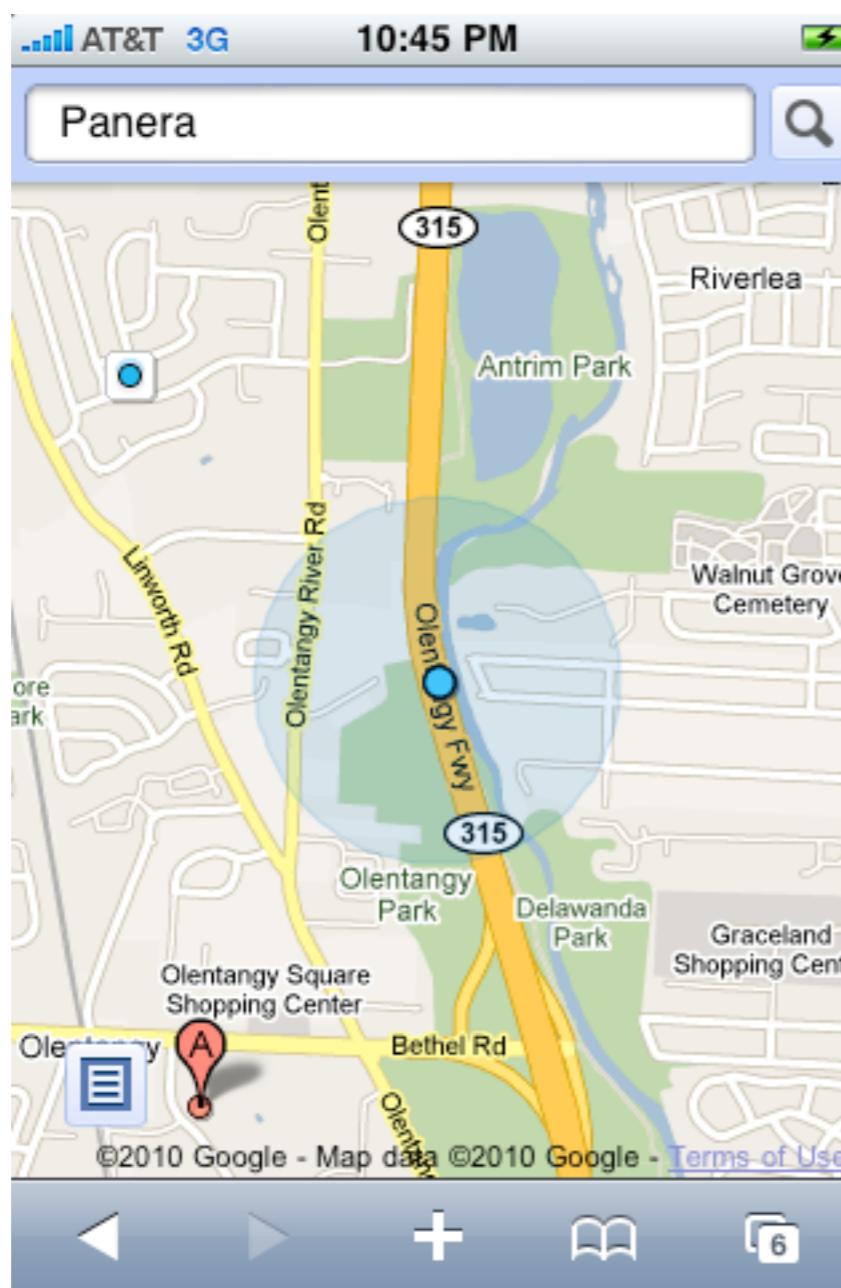
Includes:

- Native support
- Sponsored Points of Interest and more...



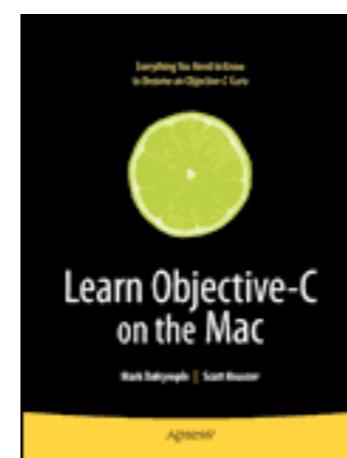
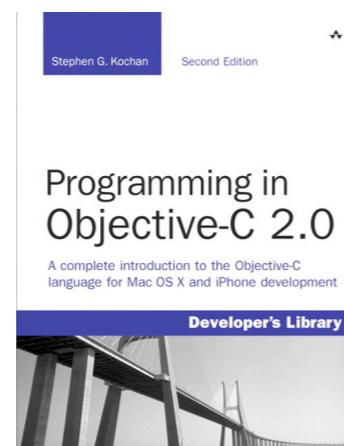
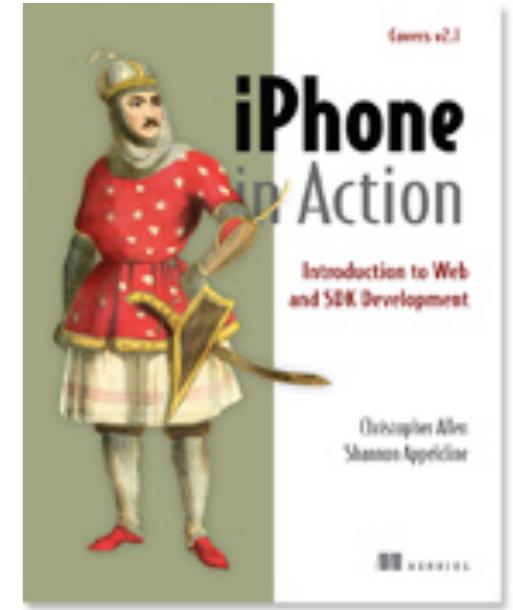
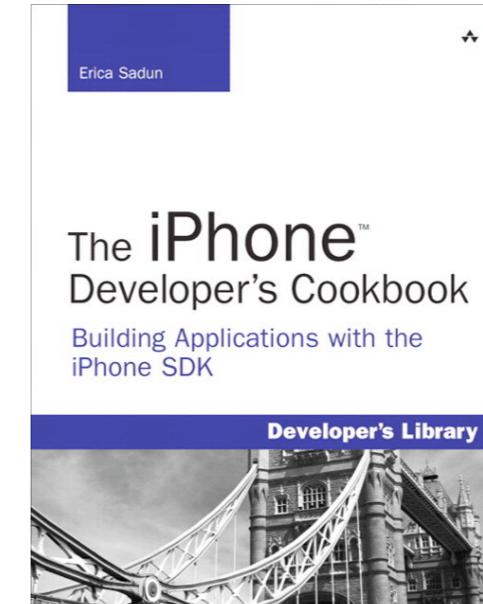
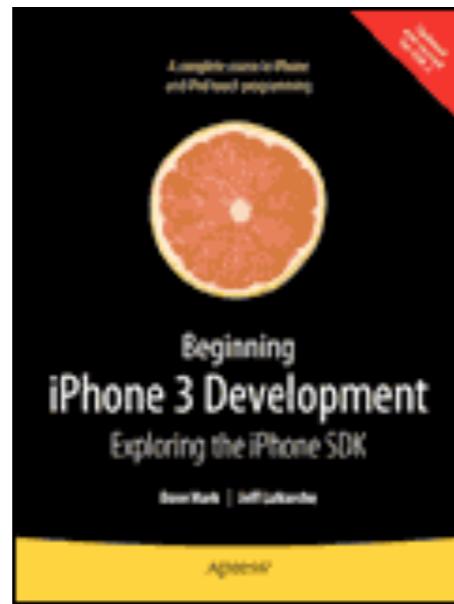
Feature	Google	CloudMade
Multiple map styles	✗	✓
Customizable map styles	✗	✓
Vehicle, Pedestrian, Cycle Routing	✗	✓
Forward Geocoding	✗	✓
Reverse Geocoding	✓	✓
Open source, user extendable library	✗	✓

Google maps

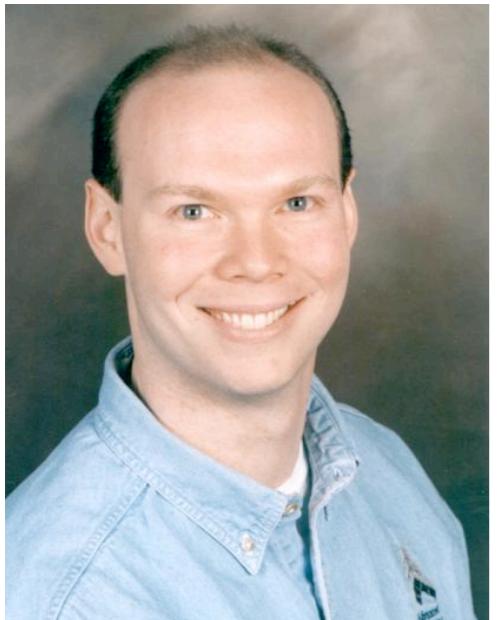


A screenshot of a Google Maps detailed view for "Panera Bread: Northwest" on an AT&T 3G mobile device at 10:46 PM. The page shows the address 875 Bethel Road, Columbus, OH 43214-1903, phone number (614) 457-6800, and website panerabread.com. It features a small image of a sandwich. Below the details are links for "Directions", "Search nearby", "Save to...", and "more▼". The map below shows the location's proximity to Olentangy Square Shopping Center, Bethel Rd, and Olentangy River. The bottom of the screen shows standard mobile navigation icons.

Resources



The Objective-C Programming Language



Christopher M. Judd
Judd Solutions
President/Consultant/Author
email: cjudd@juddsolutions.com
web: www.juddsolutions.com
blog: juddsolutions.blogspot.com
twitter: [javajudd](https://twitter.com/javajudd)

