Core Location

• Features
  • Get the device's current location
  • Receive notifications for location changes
  • Region monitoring/geofencing
Location
Getting the Device's Current Location

- Standard location service
- Configurable and general-purpose
- "Significant-change" location service
- Lower power; monitors cell tower changes
Getting the Device's Current Location

• Acquiring location data is power-intensive!

• Leaving the standard location service on for extended periods will drain the battery!

• Use the significant change service whenever possible, if you need regular updates
Availability of Location Services

```c
[CLLocationManager locationServicesEnabled];
```
Availability of Location Services

- When are location services unavailable?
  - User may have disabled them entirely in Settings
  - User can deny access for a specific app
  - The device might be in Airplane mode
- If you try to start location services anyway, iOS will prompt the user to confirm reenabling
Starting the Standard Location Service

- (void) startStandardUpdates
{
    // Create the location manager if this object does not
    // already have one.

    if (locationManager == nil)
        locationManager = [[[CLLocationManager alloc] init] autorelease];

    locationManager.delegate = self;
    locationManager.desiredAccuracy = kCLLocationAccuracyKilometer;

    // Set a movement threshold for new events.
    locationManager.distanceFilter = 500;

    [locationManager startUpdatingLocation];
}
Starting the Significant-Change Service

- (void) startSignificantChangeUpdates
{
    // Create the location manager if this object does not
    // already have one.
    if (locationManager == nil)
        locationManager = [[CLLocationManager alloc] init];

    locationManager.delegate = self;
    [locationManager startMonitoringSignificantLocationChanges];
}
Receiving Location Data

- (void) locationManager: (CLLocationManager *) manager
didUpdateToLocation: (CLLocation *) newLocation
fromLocation: (CLLocation *) oldLocation;
Region Monitoring

- Core Location (since iOS 4) includes support for monitoring a programmer-defined (circular) region for location updates
- Delegate receives messages when the device location enters or exits the region
Battery Conservation Tips

• Turn off location services when you're not using them

• Favor the significant-change service whenever possible

• Use the least accurate resolution values you need

• Turn off location services if accuracy does not improve over time
Heading
Getting the Heading

- Devices with a magnetometer can get the device's heading
- Gives you the direction of magnetic north
- If you want true north values, you'll also need the device's location
Availability of Heading

[CLLocationManager headingAvailable];
- (void) startHeadingEvents
{
    if (locationManager == nil)
        locationManager = [[[CLLocationManager alloc] init] autorelease];

    // Start location services to get the true heading.
    locationManager.distanceFilter = 1000;
    locationManager.desiredAccuracy = kCLLocationAccuracyKilometer;
    [locationManager startUpdatingLocation];

    // Start heading updates.
    if ([CLLocationManager headingAvailable])
    {
        locationManager.headingFilter = 5;
        [locationManager startUpdatingHeading];
    }
}
Receiving Heading Data

- (void) locationManager: (CLLocationManager *)
  manager
    didUpdateHeading: (CLHeading *) newHeading;
Geocoding
Geocoding

- Location data usually consists of two numeric values (latitude and longitude)
  - Useful and easy for computers
  - Unintuitive for humans

- Geocoding is the translation of coordinates to "placemarks" (street, city, state, etc.) and vice-versa
Reverse Geocoding

- Reverse geocoding is the translation of coordinates to a human-friendly format
- Since iOS 4.1, this is done using `MKReverseGeocoder`
- Since iOS 5, `MKReverseGeocoder` is deprecated, in favor of `CLGeocoder`
Reverse Geocoding

• The reverse geocoders are very easy to use:
  • Give it a coordinate
  • Let it (asynchronously) perform the lookup
  • Query the results for address information
Forward Geocoding

- Since iOS 5, `CLGeocoder` can perform *forward* geocoding - translation of user-readable locations (such as a street address) into coordinates.
Maps
MKMapView

- Fully functional map view
- Supports zooming and panning
  - These can be disabled/limited by setting properties of the map view
Setting the Visible Region

- `MKMapView` has a property, `region`, that can be set to assign what region of the map to show

```c
typedef struct
{
    CLLocationCoordinate2D center;
    MKCoordinateSpan span;
} MKCoordinateRegion;
```
Map Views

• Built-in support for showing the user's current location

• Set using the property `showsUserLocation`
Demo!
Annotations

• As views, map views can contain subviews
• Subviews *don't* "stick" to the map by coordinates
• *Annotations* are like subviews, but are instead attached to coordinates
• *Overlays* are geometric shapes used to show larger geographic regions (such as parks, cities, states, etc.)
Annotations

• "Annotations offer a way to highlight specific coordinates on the map and provide additional information about them."

• Consist of an image drawn on the map and an optional callout bubble.
Annotations

- Two distinct objects needed to display an annotation on a map:
  - An *annotation object* - an object that conforms to the protocol `MKAnnotation`
  - An *annotation view* - the view used to draw the annotation on the map
Annotations

- Annotation views are not added to the map directly
- Annotation objects are added to the map view and the map view asks its delegate for the corresponding annotation view
Adding an Annotation

1. Define an annotation object
   - Use/subclass MKPointAnnotation
   - Define a custom object that conforms to MKAnnotation
Adding an Annotation

2. Define an annotation view

- Create an instance of `MKAnnotationView` and set its image property
- Create a standard pin annotation (`MKPinAnnotationView`)
- Subclass `MKAnnotationView` and provide custom drawing code
Adding an Annotation

3. Implement `mapView:viewForAnnotation:` in your map view's delegate

   • Given an annotation, provide an annotation view

4. Add the annotation object to the map (using `addAnnotation:` or `addAnnotations:`)
Demo!
Overlays
Overlays

- Map overlays are similar, but more complicated
- See the documentation on how to create map overlays
Resources

• Location Awareness Programming Guide

• Core Location Framework Reference

• MapKit Framework Reference
  • http://developer.apple.com/library/ios/#documentation/MapKit/Reference/MapKit_Framework_Reference/_index.html