Location-based Frameworks

Dr. Sarah Abraham
Location Data

- Provides information on device’s GPS location
- Provides information about surrounding area
- Requires user’s permission for privacy concerns
Core Location

- Framework that determines current location and heading of device
- Power-intensive operation
  - Standard location service
  - Significant change location service
Accessing Location

- User must grant permission for app to use location services

- Can ask for permission:
  - When app is running
  - When app is not running (iOS will start app upon reaching a destination)

- String requesting permission stored in Info.plist:
  - `NSLocationWhenInUseUsageDescription`
  - `NSLocationAlwaysUsageDescription`
Core Location Upon App Start

1. Check that location services are available

2. If so, request authorization

3. If the user has not authorized the app, a popup will request permission

4. User permission stored in Settings -> Privacy -> Location Services -> [app name]

5. If the user has authorized the app, `didChangeAuthorizationStatus` method runs allowing app to begin location monitoring
CoreLocationManagerDelegate

- Defines methods to receive heading and location updates
  - `didUpdateLocations (CLLocation)`
    - Tells delegate new location data is available
  - `didUpdateHeading (CLHeading)`
    - Tells delegate new heading data is available
  - `didVisit (CLVisit)`
    - Tells delegate new visit-related event was received
Additional Location Delegate Methods

Receives and handles:

- Location errors
- Pauses in location data updates
- Entering and exiting specified regions (CLRegion)
- Monitoring for specified regions (CLRegion)
- Checking for beacons (CLBeacon)
- Changes in app authorization status
CLLocation

- Object that represents location data generated by CLLocationManager
- Geographical coordinates
- Altitude
- Speed and heading
- Accuracy of measurements
- Time of sampling
MapKit

- API that handles:
  - Map display
  - Location plotting
  - Route drawing
  - Object rendering (overlays)
Using MapKit

- Import MapKit
- Include MKMapView in view controller
- Set view controller to implement MKMapViewDelegate protocol
- Add user location information, annotations and type of map view
Define locations of interest at specified coordinates

**MKPointAnnotation** provides simple annotation (location, title and subtitle)

Customize pins using **MKAnnotation** protocol within a custom class
Custom MKAnnotation

1. Create annotation class as NSObject subclass
2. Implement MKAnnotation protocol
3. Create annotation object and add to map
4. Implement viewForAnnotation method to handle annotation detailCalloutAccessoryView
Google Maps

- Possible to use Google Maps instead of Apple Maps

- Must have:
  - iOS Google Map SDK API and Google Places API Web Service enabled
  - API key for Google Maps SDK associated with bundle identifier
  - Google Places API key
  - Google Maps SDK installed in project
Making the Libraries Play Nicely

- Google Developer Console <https://console.developers.google.com/apis/library>
- Cocoapods Dependency Manager <https://cocoapods.org>
- Bridging Headers to allow Swift project to talk to Map’s Objective-C library code
- <https://www.raywenderlich.com/109888/google-maps-ios-sdk-tutorial>