TODAY

• Core Location

• MapKit
CORE LOCATION

• Authorize location services
  • .authorizedAlways
  • .authorizedWhenInUse – stops when app closed (not background)
• Must add corresponding description string to plist
  • e.g. NSLocationAlwaysUsageDescription

CLLlocationManagerDelegate
// needed for any location service call
func locationManager(_ manager: CLLocationManager, didFailWithError error: Error) {
  print("error: \(error)")
}
// needed for deferred
func locationManager(_ manager: CLLocationManager, didFinishDeferredUpdatesWithError error: Error?) {
  print("Finished deferred: \(error)")
}
// authorizations changed asynchronously after request
func locationManager(_ manager: CLLocationManager, didChangeAuthorization status: CLAuthorizationStatus) {
  if status == .authorizedWhenInUse || status == .authorizedAlways {
    startTracking()
  }
}
// called one or more new location updates
func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {

MAPKIT

• Easy to bring up a map
  • show specific region MKMapView.setRegion(..)
  • current position (option when adding mapview in storyboard)
  • options for type
    • .standard
    • .satellite
• Add to the map via
  • annotations
  • overlays
MAPKIT ANNOTATIONS

- Annotations
  - create object that conforms to MKAnnotation protocol
  - Add annotation object to mapview w/ addAnnotation(_:)
    ```swift
    mapView?.addAnnotations(places) // from a plist
    ```
  - Set view controller to conform to MKMapViewDelegate
  - And implementation for mapView(_: viewFor:) to provide view for the annotation
    ```swift
    func mapView(_ mapView: MKMapView, viewFor annotation: MKAnnotation) -> MKAnnotationView? {
        if annotation is MKUserLocation { return nil }
        else {
            let aView = mapView.dequeueReusableAnnotationView(withIdentifier: "aView") ?? MKAnnotationView()
            aView.image = UIImage(named: "place icon")
            return aView
        }
    }
    ```

MAPKIT OVERLAYS

- Create overlays
  - MKCircle(), MKPolygon(), MKPolyLine(), MKTileOverlay()
  - MKMapView.addOverlays(…)
- Create renderers for each used type
  ```swift
  func mapView(_ mapView: MKMapView, rendererFor overlay: MKOverlay) -> MKOverlayRenderer {
    let renderer = MKCircleRenderer(overlay: overlay)
    renderer.fillColor = UIColor.black.withAlphaComponent(0.5)
    renderer.strokeColor = UIColor.blue
    renderer.lineWidth = 2
    return renderer
  }
  ```
DEANONYMIZATION

- “Even This Data Guru Is Creeped Out….”
  - two anonymous folk allowed Google Maps to collect data all the time

Results
- in a few hours had:
  - names
  - work / home addresses
  - knew whether they had kids / pets
  - personal photos from social networks
  - mortgage info….