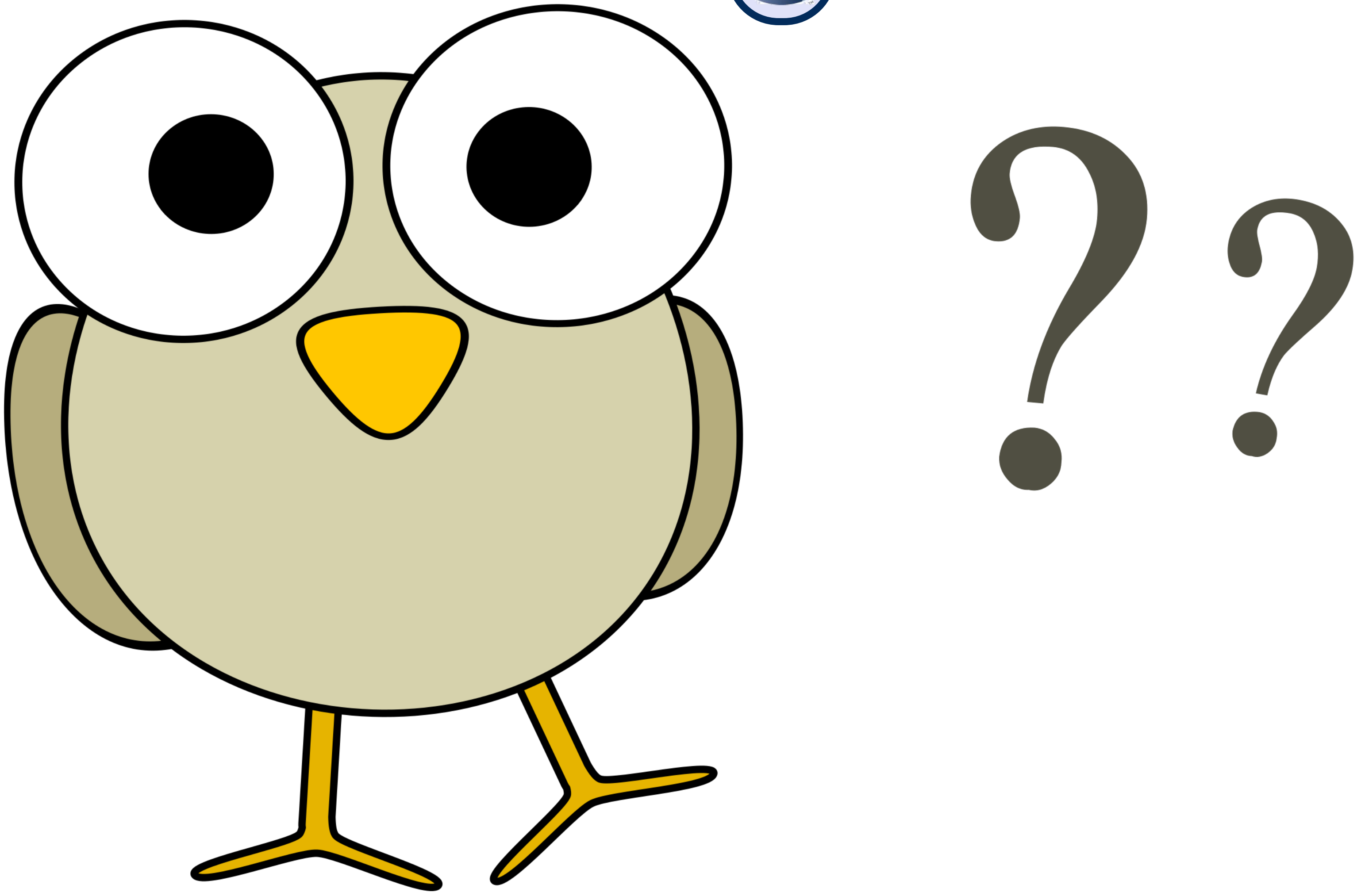
Mobile Applications CSCI 448 Lecture 27

Location Services

Previously in CSCI 448

- CameraX
  - Preview
  - Image Analysis
  - Image Capture
  - Video Capture

Questions?



Learning Outcomes For Today

- Describe how an app queries the user’s location
- Provide good UI/UX when asking the user for their location
- Create an app that queries the user’s location

On Tap For Today

- Getting a Location
- Practice

On Tap For Today

- Getting a Location
- Practice

Location

- Where you are

- `android.location` API
  - Fine Location: Based off of GPS position
  - Coarse Location: Based off of cellular tower

Google Play Location

- Apps make a request such as
  - Use as much battery for as accurate position as possible
  - Use as little battery as possible for an approximate position
- Fused Location Provider handles switching between Coarse/Fine Location as appropriate

```
val fusedLocationProviderClient =
          LocationServices.getFusedLocationProviderClient(context)
```

Google Play Services

- Google Play provides a Fused Location Provider
- Google Play must be installed on your device and updated

Using Play Services I

- Add dependency for
  `play-services-location`
  to project

- Do not add entire `play-services` dependency
  - Too many functions, apk has 64k method limit
  - Need to enable multidex

Using Play Services II

- Check if Play Services API is available on device
  - Pull up Play Store to download/update if necessary

```
val apiAvailability = GoogleApiAvailability.getInstance()
val errorCode = apiAvailability.isGooglePlayServicesAvailable(this)
if (errorCode != ConnectionResult.SUCCESS) {
  val pendingIntent = apiAvailability.getErrorResolutionPendingIntent(this, ConnectionResult(errorCode))
  if (pendingIntent != null)
    launcher.launch( IntentSenderRequest.Builder(pendingIntent).build() )
  }
}
```

Requesting a Location

- Create a (one time) location request

```
val locationRequest = LocationRequest
    .Builder(Priority.PRIORITY_HIGH_ACCURACY, 0L)
    .setMaxUpdates(1)
    .build()
```

Request parameters

- Priority
  - **`LOW_POWER`**: city level accuracy
  - **`BALANCED_POWER_ACCURACY`**: city-block level accuracy
  - **`HIGH_ACCURACY`**: most accurate available
  - **`NO_POWER`**: most accurate with no additional power consumption
- Interval – how frequently to update
- # of updates – how many times to update

Check if Location is Enabled

```kotlin
val builder = LocationSettingsRequest.Builder()
  .addLocationRequest(locationRequest)
val client = LocationServices.getSettingsClient(activity)
client.checkLocationSettings(builder.build()).apply {
  addOnSuccessListener { response ->
    val isLocationUsable = response.locationSettingsState?.isLocationUsable ?: false
  }
  addOnFailureListener { exc ->
    if (exc is ResolvableApiException) {
      launcher.launch( IntentSenderRequest.Builder(exc.resolution).build() )
    }
  }
}
```

Make the Request

- Add permissions to manifest

  ACCESS_FINE_LOCATION
  ACCESS_COARSE_LOCATION

```
if( hasLocationPermissions )
    fusedLocationProviderClient
        .requestLocationUpdates(locationRequest,
                                locationCallback,
                                Looper.getMainLooper())
```

Finally get the Location!

```kotlin
val locationCallback = object : LocationCallback() {
  override fun onLocationResult(locationResult: LocationResult) {
    super.onLocationResult(locationResult)
    val location = locationResult.lastLocation
  }
}
```

Android Design Patterns

- Behavioral Patterns
  1. Command – UI Event Handling, Retrofit Request Callback, Activity Result Callback, Permissions Callback, Location
  2. Observer – State, Flow, LiveData
  3. Template Method - IScreenSpec
- Creational Patterns
  4. Builder – Compose NavGraph, WorkRequest, Constraints, Retrofit, LocationRequest
  5. Factory – ViewModelFactory
  6. Singleton – ViewModelProvider, Repository, Room Database
- Structural Patterns
  7. Decorator – View Model
  8. Façade – DAO, Repository

Getting Location Info

- Physical Device:
  - Provided by your carrier
- Android Studio Virtual Device
  - Extended controls → location
  - (*But need real internet connection to load map*)
  - Demo!

On Tap For Today

- Getting a Location
- Practice

GeoLocatr

- Lab10A:
  - New app, larger piece of the two
  - Get and display user's location
- Lab10B:
  - Plot location on map (next time)