Developing Apps with ArcGIS Runtime SDK for Android

Dan O’Neill and Gunther Heppner
Introductions

• What we do
  - Alaska – Dan O’Neill - @jdoneill - https://github.com/doneill

• What you do
• Session Surveys
Agenda

• Maps App
  - ArcGIS Online Subscription
  - Portal API Patterns
    - ArcGIS Basemaps
    - Licensing API
  - Analysis Functions
    - Routing and Geocoding
    - Measure Tool

• Offline Developer Patterns
  - Offline Basemaps
  - Custom Tile Layers
Maps App

Dan O’Neill
Maps App

• Modern Map Navigation Centric App
• Built with ArcGIS Runtime SDK for Android
• Access your ArcGIS Organizations custom Maps
• Features
  • Switching Basemaps
  • Searching for Places
  • Routing Navigation
  • Measure distances/areas
Map App Demo

Dan O’Neill & Gunther Heppner
ArcGIS Online Subscription

Gunther Heppner

Specify the URL to your organization or your ArcGIS portal

e.g., http://www.arcgis.com
Get access to an organization’s geospatial content and services, including:

- web maps
- basemaps
- groups
- utility services (geocoding, printing)
- ability to search an organization’s content
ArcGIS Online Subscription

Access to an organizational subscription requires:

- sign in via OAuth2 using the SDK’s OAuthView
  - requires a client id that identifies the application
- use the SDK’s Portal API to access and search content
OAuth2 Demo
Gunther Heppner

```java
// create an OAuthView and show it
OAuthView oAuthView = new OAuthView(this,
    new CallbackListener<UserCredentials>()

@override
public void onCallback(final UserCredentials credentials) {
    if (credentials != null) {
        Portal portal = new Portal(manager);
        PortalInfo portalInfo = portal.fetchPortalInfo();
        try {
            PortalUser portalUser = portal.fetchUser();
        } catch (Exception e) {
            onError(e);
        }
        // hold on to the initialized AccountManager
        AccountManager.getInstance();
        // enable standard license level
        if (portalInfo != null) {
            ArcGISRuntime.License.setLicenseLevel(portalInfo.getLicenseLevel());
        }
        // we are done signing in
        finish();
    }
    @Override
    public void onError(Exception e) {
    }
}
```
Portal API Patterns

Dan O’Neill
Portal Patterns – API Classes

• **Portal**
  - Provides the connection information to the portal (Uri, Token).

• **PortalInfo**
  - Includes information such as the name, logo, featured items and supported protocols (http vs https) for this portal.

• **PortalGroup**
  - Represents a group in a portal.

• **PortalItem**
  - Represents an item stored in a portal.

• **PortalQueryParams**
  - Creates query parameters suitable for finding content contained in a portal.

• **PortalQueryResult**
  - Contains the results of queries performed on a portal.
Portal Code Demo

Gunther Heppner

private void fetchBasemapItems() throws Exception {
    // Create a Portal object
    String url = getString(R.string.portal_url);
    Portal portal = new Portal(url, null);

    // Create a PortalQueryParams to query for items
    PortalQueryParams queryParams = new PortalQueryParams();
    queryParams.setCanSearchPublic(true);
    queryParams.setSortField("name").setSortOrder(PortalQuerySortOrder.ASCENDING);
    queryParams.setQuery(createQueryString());

    // Find items that match the query
    PortalQueryResultSet queryResultSet = portal.query(queryParams);
    if (isCancelled()) {
        return;
    }

    // Loop through query results
    for (PortalItem item : queryResultSet.getResults()) {
        // Fetch item thumbnail from server
        byte[] data = item.fetchThumbnail();
        if (isCancelled()) {
            return;
        }
        if (data != null) {
            // Decode thumbnail and add this item to list
            Bitmap bitmap = BitmapFactory.decodeByteArray(data, 0, data.length);
            BasemapItem portalItemData = new BasemapItem(bitmap);
            Log.i(TAG, "Item id = " + item.getTitle());
            mBasemapItemList.add(portalItemData);
        }
    }
}
Licensing
Gunther Heppner
Runtime Licensing

Development and Deployment Workflow

1. Download and Install
2. Develop and Test
3. Deploy and Distribute
## License levels and functionality

<table>
<thead>
<tr>
<th>License Level</th>
<th>Available functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer (development and testing only)</td>
<td>All functionality (watermarks and debug messages will be generated, nag screens with local server*)</td>
</tr>
<tr>
<td>Basic</td>
<td>Connected - all functionality</td>
</tr>
<tr>
<td></td>
<td>Offline - map viewing only</td>
</tr>
<tr>
<td>Standard</td>
<td>Connected and offline - all functionality, includes:</td>
</tr>
<tr>
<td></td>
<td>• Local locators (geocoding)</td>
</tr>
<tr>
<td></td>
<td>• Local routing</td>
</tr>
<tr>
<td></td>
<td>• Local geodatabase editing</td>
</tr>
<tr>
<td></td>
<td>• Local geodatabase sync operations</td>
</tr>
</tbody>
</table>
How to license your app at the basic level

- [http://developers.arcgis.com](http://developers.arcgis.com)

- Under Application section, create a New Application (or select existing)

- Click on Runtime SDK Licensing

- Copy the Client ID and use it to set your clientID
  - `ArcGISRuntime.setClientID("0x7W");`
Licensing – Basic API

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    LicenseResult licenseResult = ArcGISRuntime.setClientId(CLIENT_ID);
    LicenseLevel licenseLevel = ArcGISRuntime.License.getLicenseLevel();
```
How to license your app at the standard level

• You have 2 options:

1. Use an organization account (ArcGIS Online or Portal for ArcGIS)
   - Requires users of your app to log in with their account

2. Use a license string obtained from Customer Service or your international distributor
   - License burnt into the app
   - Extensions can also be added with this option

For more info speak to sales or product management
Licensing – Standard Portal Pattern

```java
Portal portal = new Portal(PORTAL_URL, credentials);
PortalInfo portalInfo = null;
portalInfo = portal.fetchPortalInfo();
LicenseInfo licenseInfo = portalInfo.getLicenseInfo();
LicenseResult licenseResult = ArcGISRuntime.License.setLicense(licenseInfo);
LicenseLevel licenseLevel = ArcGISRuntime.License.getLicenseLevel();
```
Analysis

Dan O’Neill
Why analyze?

- Convert data into information
- Adds value
  - Answers questions
  - More than just points on a map
- Services available in ArcGIS Online
  - Build into your apps
Geocoding Patterns – Fine Grain

• ArcGIS Geocoding service
  - Global coverage
• Locator uses this in default constructor
• Find method allows for address and POI search
• Point and radius parameters
  - Results in radius are promoted
  - Sorted by distance to point
  - Results outside radius still returned
Geocoding Pattern - Simplification

- ArcGIS Android Toolkit API
- Provides GeocodeHelper class
- Find address for given location
- Simplifies the workflow
  - No need to set parameters and get result from a Locator
Routing API - Workflow

• Create a Route Task
• Set up Route Task Parameters
• Set stops
• Calculate route
• Get results
• Display route on map
• Get directions and display to user
Routing – Geocoding

Code Demo

Dan O’Neill

```java
protected RouteResult doInBackground(List<LocatorFindParameters> params)
{
    // Perform routing request on background thread
    mException = null;

    // Define route objects
    List<LocatorGeocodeResult> geocodeStartResult = null;
    List<LocatorGeocodeResult> geocodeEndResult = null;
    Point startPoint = null;
    Point endPoint = null;
    RouteParameters routeParams = null;

    // Create a new locator to geocode start/end points
    Locator locator = Locator.createOnlineLocator();

    try {
        // Geocode start position, or use My Location (from GPS)
        LocatorFindParameters startParam = params[0].get(0);
        if (startParam.getText().equals(getString(R.string.entrystart))
            startPoint = (Point) GeometryEngine.project(mlocator, startParam.getLatitude(), startParam.getLongitude());
        } else {
            geocodeStartResult = locator.find(startParam);
            startPoint = geocodeStartResult.get(0).getLocation();
            if (isCancelled()) { return null; }
        }

        // Geocode the destination
        LocatorFindParameters endParam = params[0].get(1);
        geocodeEndResult = locator.find(endParam);
        endPoint = geocodeEndResult.get(0).getLocation();
    } catch (Exception e) {
        return null;
    }
}
```
Analysis – Measure Tool

- Measure Tool is self contained
- Easy to integrate into your own app
- Candidate for first tool in our App Toolkit
- Customizable
Measure Integration
Code Demo

Dan O’Neill
Maps App on Github

- [https://github.com/Esri/maps-app-android](https://github.com/Esri/maps-app-android)
- Get involved
- Report Issues
- Contribute Code
  - Fork it
  - Clone it
  - Configure remotes
  - Send pull requests
Offline Patterns

Dan O’Neill
Offline Patterns

Service
- Programmatic pattern to support large number of users
- Can use the map extent to select parts of database to download to device

Desktop
- Consume Runtime content created in ArcGIS for Desktop
- Best for single user
- Network datasets for offline routing
Offline Workflows

- **Take Map Offline**
  - View / Basemap
  - Edit
  - Sync

- **Perform Offline Tasks**
  - Routing
  - Geocoding
  - Query

- **Create Your Own Layer**
  - Define layer and symbology
  - Persist it to device
Offline Basemaps

- From a Tiled Service
  - TPK or Compact Cache
  - Service must support `exportTiles` operation

- Requesting a Tile Cache parameters
  - Level Of Details (LOD)
  - Level ID’s supported
  - Extent of the Tile Cache
  - SpatialReference

```java
final ExportTileCacheTask exportTileCacheTask = new ExportTileCacheTask(tileURL, null);

ExportTileCacheParameters params = new ExportTileCacheParameters(createAsTilePackage, levels, ExportBy.ID, extentForTPK, mapView.getSpatialReference());
```
Offline Basemaps

- Submit a tile cache job
- Poll for status
- Download the completed cache to device

```java
exportTileCacheTask.generateTileCache(
    ExportTileCacheParameters params,
    CallbackListener<ExportTileCacheStatus> statusCallback,
    CallbackListener<String> downloadCallback,
    String tileCachePath)
```
Offline Basemap Demo
Gunther Heppner
Offline Custom Layers

• Base Abstract Class – **TiledServiceLayer**
  - Used by
    - ArcGISTiledMapServiceLayer
    - BingMapsLayer
    - OpenStreetMapLayer

• Fetch Tiles
  - Implement abstract getTile() method

```java
MBTilesLayer mbLayer = new MBTilesLayer(
    Environment.getExternalStorageDirectory().getPath() + "/ArcGIS/samples/mbtiles/world_countries.mbtiles";
```
Custom Layer Demo

Dan O’Neill
Offline demos in the SDK

- Local Tile Layer
- Create Local JSON Features
- Create Local Geodatabase
- Offline Editor
- Export Tile Cache
- Offline Routing and Geocoding
Next release…

• Direct read of raster datasets
  - Mosaic datasets
  - Raster Files – GeoTIFF, NITF, IMG, RPF, DTED, PNG, HFA, JPEG, JP2K
  - GeoPackage

• Direct read of vector data
  - Shapefile
  - KML
  - GeoPackage

• Simplification
  - Open Source Toolkit

• Analysis
  - Renderscript
  - Spatial Analysis
Related Session

• **Debugging and Troubleshooting ArcGIS Runtime SDK for iOS and Android**
  - **Wednesday July 16, 11:30AM**
  - Demo Theater – Technical Support Exhibit Hall B

• **Offline Routing and Geocoding in ArcGIS Runtime SDK’s**
  - **Wednesday July 16, 3:00PM & Thursday July 17, 10:00AM**
  - General Theater 2 Exhibit Hall A

• **Create Your Own Android App Tools Using ArcGIS Runtime SDKs**
  - **Thursday July 17, 9:30AM**
  - Demo Theater – Developer Island Exhibit Hall B

• **ArcGIS Runtime SDK’s: The Road Ahead**
  - **Thursday July 17, 1:30PM**
  - Room 07 A/B
Agenda

• **Maps App**
  - ArcGIS Online Subscription
  - Portal API Patterns
    - ArcGIS Basemaps
    - Licensing API
  - Analysis Functions
    - Routing and Geocoding
    - Measure Tool

• **Offline Developer Patterns**
  - Offline Basemaps
  - Custom Tile Layers
Thank You

• Please fill out the session survey:

Offering ID: 1172

Online – www.esri.com/ucsessionsurveys
Paper – pick up and put in drop box