



Esri International Developer Summit  
Palm Springs, CA

# Building Android Apps

## Runtime SDK for Android

Dan O'Neill & Alan Lucas

# Introductions

- What do you do
- What do we do
  - Android Development Team
    - Edinburgh – Alan Lucas - <https://github.com/alan-ed>
    - Alaska – Dan O’Neill - @jdoneill - <https://github.com/doneill>
- Session Surveys > <http://www.esri.com/events/devsummit/session-rater>

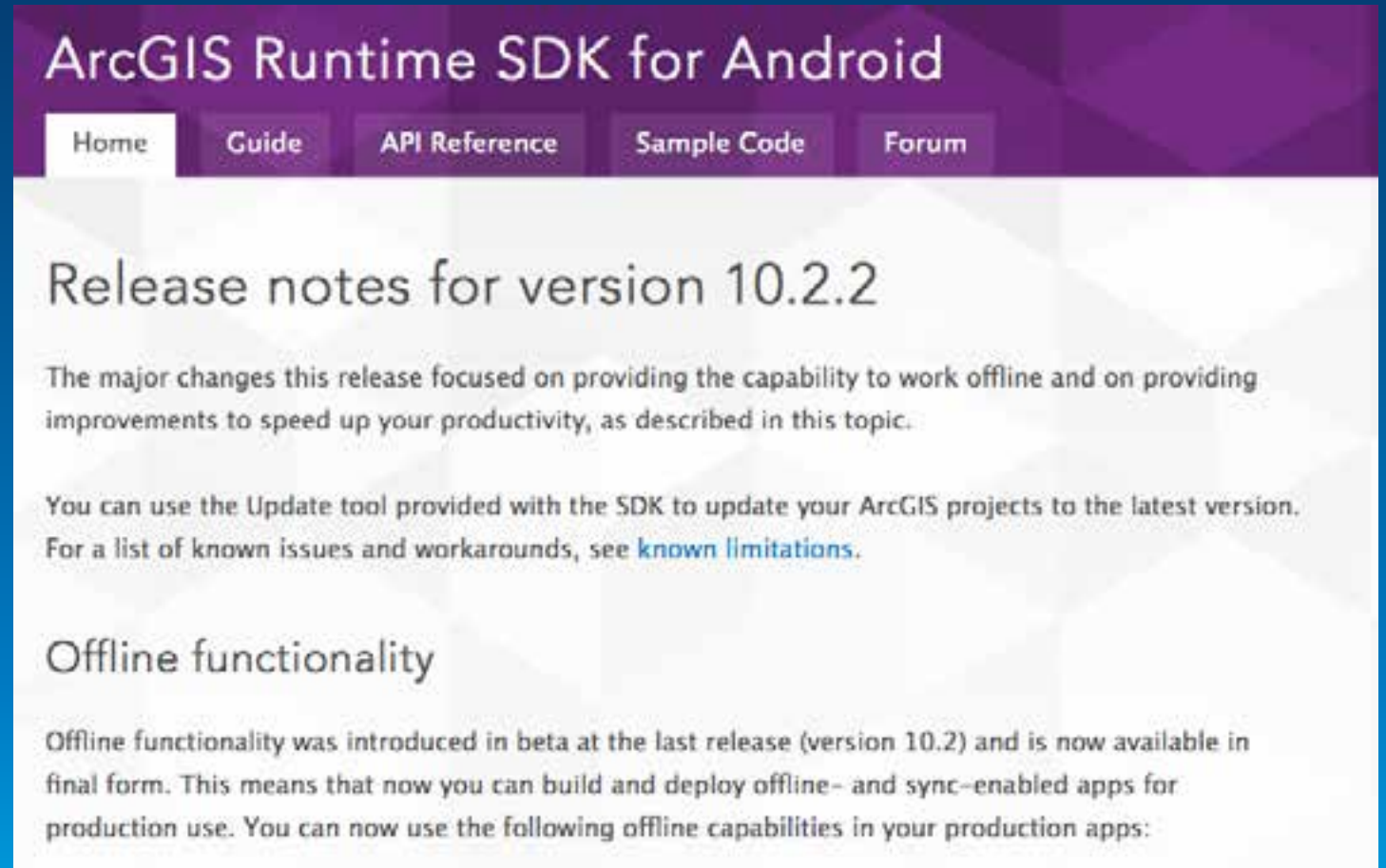


# Agenda

- **What's new at 10.2.2**
  - Licensing API
  - Offline Patterns
- **Map Apps Demo**
- **Portal API Patterns**
  - Basemaps
- **Analysis Functions**
  - Routing and Geocoding
  - Measure
- **Contribute to Github**
- **What's coming**

# What's New at 10.2.2

- **Licensing API**
  - Basic
  - Standard
- **Offline functionality**
  - Create local geodatabase
  - Work with features
  - Sync edits



The screenshot shows the ArcGIS Runtime SDK for Android website. The header is purple with the title "ArcGIS Runtime SDK for Android" in white. Below the header is a navigation menu with five items: "Home", "Guide", "API Reference", "Sample Code", and "Forum". The main content area is white and features the heading "Release notes for version 10.2.2". The text below the heading states: "The major changes this release focused on providing the capability to work offline and on providing improvements to speed up your productivity, as described in this topic." It then says: "You can use the Update tool provided with the SDK to update your ArcGIS projects to the latest version. For a list of known issues and workarounds, see [known limitations](#)." Below this is a section titled "Offline functionality" with the text: "Offline functionality was introduced in beta at the last release (version 10.2) and is now available in final form. This means that now you can build and deploy offline- and sync-enabled apps for production use. You can now use the following offline capabilities in your production apps:"

# Licensing

Dan O'Neill

level has been  
successfully set to Basic.

OK

# Runtime Licensing

## Development and Deployment Workflow



1. Download and Install



2. Develop and Test



3. Deploy and Distribute

## License levels and functionality

License Level	Available functionality
Developer (development and testing only)	All functionality (watermarks and debug messages will be generated, nag screens with local server*)
Basic	Connected - all functionality Offline - map viewing only
Standard	Connected and offline - all functionality, includes: <ul style="list-style-type: none"><li>• Local locators (geocoding)</li><li>• Local routing</li><li>• Local geodatabase editing</li><li>• Local geodatabase sync operations</li></ul>

# Licensing – Basic API

```
38 public class MainActivity extends Activity {
39
40     // TODO: initialize CLIENT_ID with a valid client id string
41     //
42     private static final String CLIENT_ID = null;
43
44     @Override
45     protected void onCreate(Bundle savedInstanceState) {
46         super.onCreate(savedInstanceState);
47
48         // Set the client id string on the ArcGISRuntime class. This will set the license level to Basic.
49         // ArcGISRuntime.setClientId() needs to be called before any other calls to the ArcGIS SDK for Android are made.
50         //
51         LicenseResult licenseResult = ArcGISRuntime.setClientId(CLIENT_ID);
52
53         LicenseLevel licenseLevel = ArcGISRuntime.License.getLicenseLevel();
54
55         if (licenseResult == LicenseResult.VALID && licenseLevel == LicenseLevel.BASIC) {
56             AlertDialogFragment.showMessageDialog(getString(R.string.basic_license_succeeded), getFragmentManager());
57         } else {
58             AlertDialogFragment.showMessageDialog(getString(R.string.valid_client_id_required), getFragmentManager());
59         }
60
61         setContentView(R.layout.activity_main);
62     }
63 }
```



# 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
  1. **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

```
107 private void signInWithOAuth() {
108     mOAuthView = new OAuthView(this, PORTAL_URL, CLIENT_ID, OAUTH_EXPIRATION_NEVER,
109         new CallbackListener<UserCredentials>() {
110
111         @Override
112         public void onError(Throwable e) {
113             AlertDialogFragment.showMessageDialog(getString(R.string.oauth_login_failed), getFragmentManager());
114         }
115
116         @Override
117         public void onCallback(UserCredentials credentials) {
118             if (credentials != null) {
119                 setStandardLicenseWithLicenseInfo(credentials);
120             } else {
121                 AlertDialogFragment.showMessageDialog(getString(R.string.oauth_login_failed), getFragmentManager());
122             }
123         }
124     });
125
126     mViewContainer.addView(mOAuthView);
127 }
```

## Licensing – Standard Portal Pattern

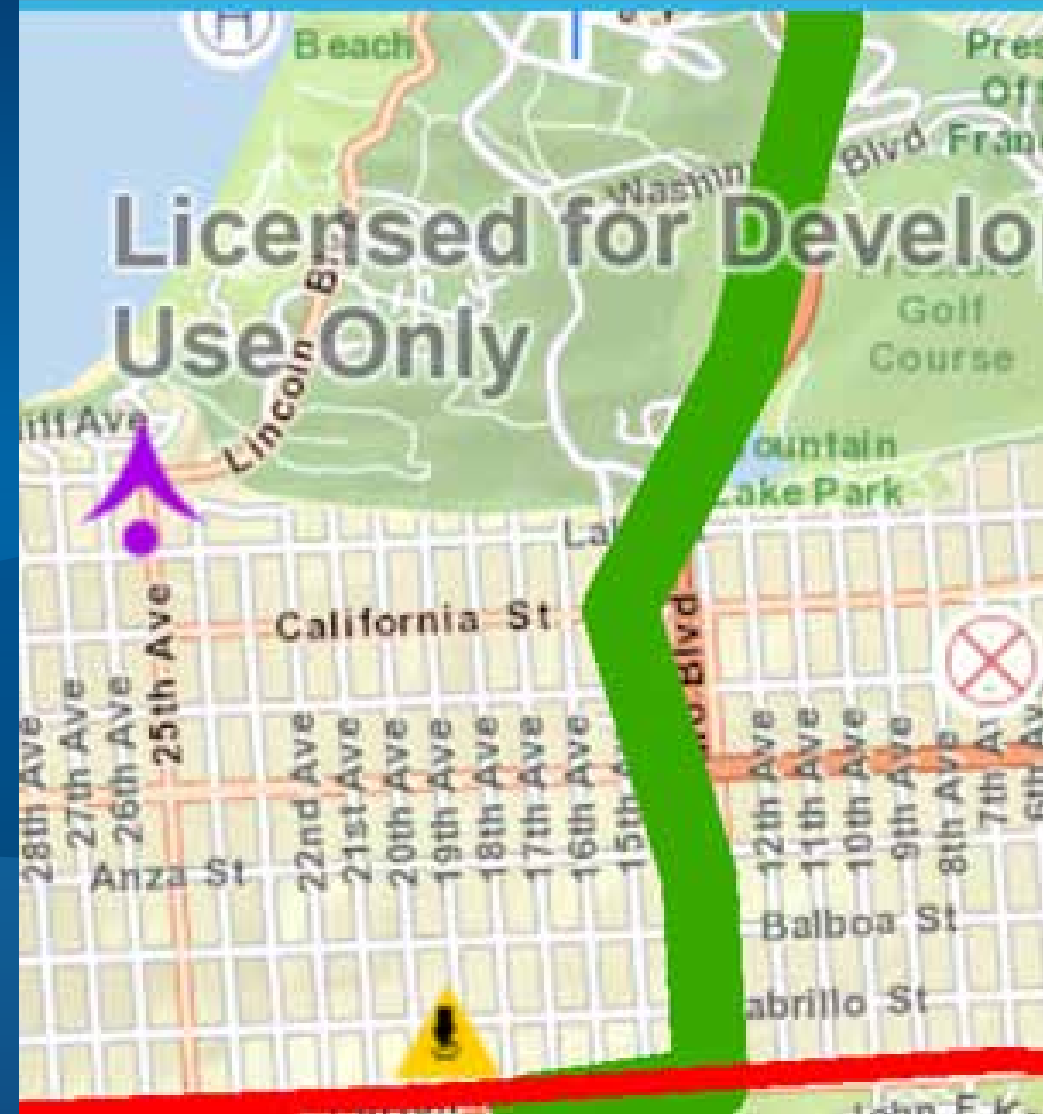
```
143 private void setStandardLicenseWithLicenseInfo(UserCredentials credentials) {
144     Portal portal = new Portal(PORTAL_URL, credentials);
145     PortalInfo portalInfo = null;
146
147     try {
148         portalInfo = portal.fetchPortalInfo();
149     } catch (Exception e) {
150         MessageDialogFragment.showMessage(getString(R.string.standard_license_failed), getFragmentManager());
151
152         return;
153     }
154
155     LicenseInfo licenseInfo = portalInfo.getLicenseInfo();
156
157     LicenseResult licenseResult = ArcGISRuntime.License.setLicense(licenseInfo);
158     LicenseLevel licenseLevel = ArcGISRuntime.License.getLicenseLevel();
159
160     if (licenseResult == LicenseResult.VALID && licenseLevel == LicenseLevel.STANDARD) {
161         MessageDialogFragment.showMessage(getString(R.string.standard_license_succeeded), getFragmentManager());
162     } else {
163         MessageDialogFragment.showMessage(getString(R.string.standard_license_failed), getFragmentManager());
164     }
165
166     showMap();
167 }
```

# Offline Patterns

Dan O'Neill



# OfflineEditor



# Offline Patterns – Create offline maps

- **Two 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 read only content
    - Network datasets
- **Add geodatabase layers to a map**
- **Add features**

## Offline Patterns – Create Local Geodatabase

```
145 private static void createGeodatabase() {
146     // set up the parameters to generate a geodatabase
147     GenerateGeodatabaseParameters params = new GenerateGeodatabaseParameters(layerIds, mapView.getExtent(),
148     mapView.getSpatialReference(), returnAttachments, syncModel, mapView.getSpatialReference());
149
150     // a callback which fires when the task has completed or failed.
151     CallbackListener<String> gdbResponseCallback = new CallbackListener<String>() {
152         @Override
153         public void onError(final Throwable e) {
154             Log.e(TAG, "Error creating geodatabase");
155             dialog.dismiss();
156         }
157
158         @Override
159         public void onCallback(String path) {
160             Log.i(TAG, "Geodatabase is: " + path);
161             dialog.dismiss();
162             // update map with local feature layer from geodatabase
163             updateFeatureLayer(path);
164         }
165     };
166
167     // a callback which updates when the status of the task changes
168     GeodatabaseStatusCallback statusCallback = new GeodatabaseStatusCallback() {
169         @Override
170         public void statusUpdated(GeodatabaseStatusInfo status) {
171             Log.i(TAG, status.getStatus().toString());
172         }
173     };
174
175     // create the fully qualified path for geodatabase file
176     localGdbFilePath = createGeodatabaseFilePath();
177
178     // get geodatabase based on params
179     submitTask(params, localGdbFilePath, statusCallback, gdbResponseCallback);
180 }
181
182 }
```

## Offline Patterns – Create Local Geodatabase

```
184  /*
185  * Request database, poll server to get status, and download the file
186  */
187  private static void submitTask(GenerateGeodatabaseParameters params, String file,
188      GeodatabaseStatusCallback statusCallback, CallbackListener<String> gdbResponseCallback) {
189      // submit task
190      gdbSyncTask.generateGeodatabase(params, file, false, statusCallback, gdbResponseCallback);
191  }
```

```
198  private static void updateFeatureLayer(String featureLayerPath) {
199      // create a new geodatabase
200      Geodatabase localGdb = null;
201      try {
202          localGdb = new Geodatabase(featureLayerPath);
203      } catch (FileNotFoundException e) {
204          e.printStackTrace();
205      }
206
207      // Geodatabase contains GdbFeatureTables representing attribute data
208      // and/or spatial data. If GdbFeatureTable has geometry add it to
209      // the MapView as a Feature Layer
210      if (localGdb != null) {
211          for (GeodatabaseFeatureTable gdbFeatureTable : localGdb.getGeodatabaseTables()) {
212              if (gdbFeatureTable.hasGeometry())
213                  mMapView.addLayer(new FeatureLayer(gdbFeatureTable));
214          }
215      }
216      // display the path to local geodatabase
217      pathView.setText(featureLayerPath);
218
219  }
```

# Offline Patterns – Work with Features

- Select features

```
37  mMapView.setOnSingleTapListener(new OnSingleTapListener() {
38      private static final long serialVersionUID = 1L;
39
40      public void onSingleTap(float x, float y) {
41          // gets the first 1000 features at the clicked point on the map, within 5 pixels
42          long[] selectedFeatures = featureLayer.getFeatureIDs(x, y, 5, 1000);
43          // select the features
44          featureLayer.selectFeatures(selectedFeatures, false);
45      }
46  }
```

- Update features

```
21  public void updateFeature(long featureID, Point newLocation) {
22      try {
23          //calls the API method updateFeature taking a feature ID and the new geometry
24          gdbFeatureTable.updateFeature(featureID, newLocation);
25      } catch (TableException e) {
26          // report/handle exception
27          Log.e(TAG, "", e);
28      }
29  }
```

- Delete features

```
33  geodatabaseFeatureTable.deleteFeatures(featureLayer.getSelectionIDs());
```



# Offline Patterns – Sync Edits

## Get Sync Parameters

```
55 final SyncGeodatabaseParameters syncParams = geodatabase.getSyncParameters();
```

## Setup Callbacks

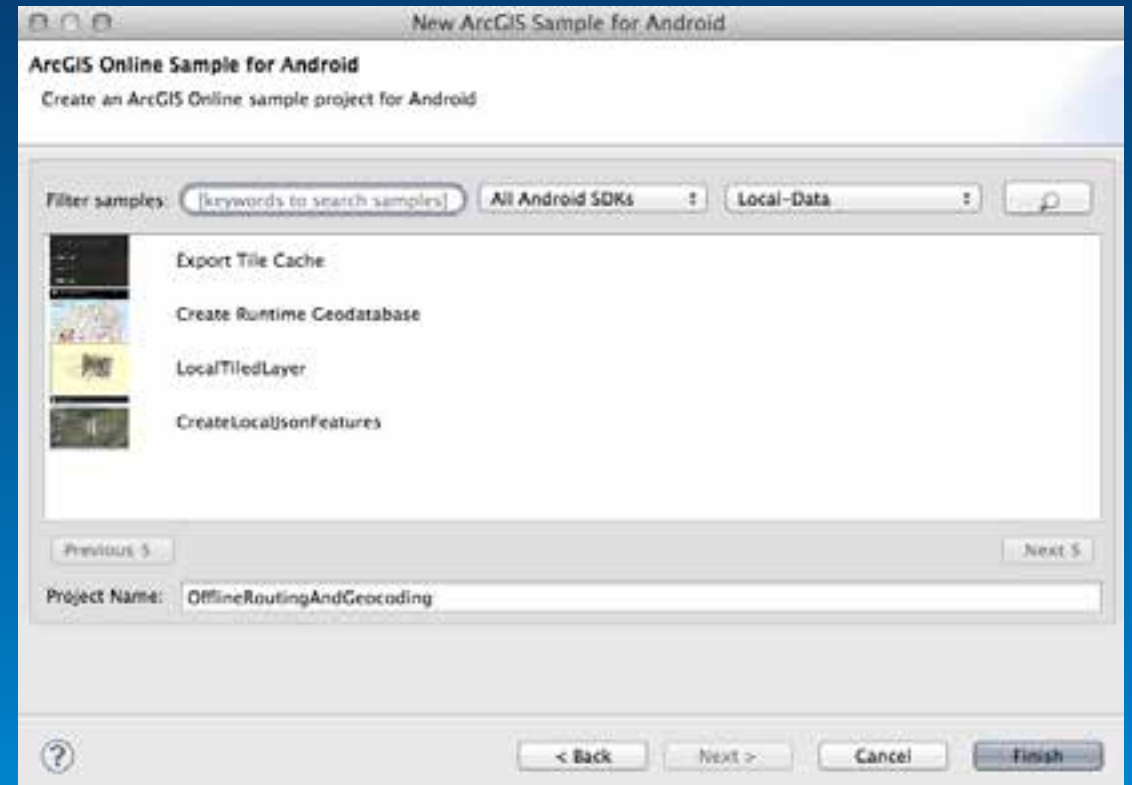
```
61 GeodatabaseStatusCallback statusCallback = new GeodatabaseStatusCallback() {
62
63     @Override
64     public void statusUpdated(GeodatabaseStatusInfo status) {
65         updateProgressBarUI("Latest status: " + status.getStatus(), true);
66     }
67 };
68
69 // callback that fires when the synchronization process completes or fails
70 CallbackListener<Geodatabase> syncResponseCallback = new CallbackListener<Geodatabase>() {
71
72     @Override
73     public void onError(Throwable e) {
74         // report/handle the error as desired
75         Log.e(TAG, "An error occurred: ", e);
76     }
77
78     @Override
79     public void onCallback(Geodatabase geodatabase) {
80         // notify the user, optionally do something with the geodatabase
81     }
82 };
```

## Start Sync

```
87 // you may already have a GeodatabaseSyncTask object if you've generated a geodatabase
88 if (geodatabaseSyncTask == null) {
89     geodatabaseSyncTask = new GeodatabaseSyncTask(FEATURE_SERVICE_URL, null);
90 }
91 // -----
92 // Start sync
93 // -----
94 geodatabaseSyncTask.syncGeodatabase(syncParams, geodatabase, statusCallback, syncResponseCallback);
```

# Offline demos in the SDK

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



# Map App Demo

Alan Lucas

Licensed for Developer  
Use Only

## Select Base Map



DeLorme World Base...



Imagery with Labels



Streets



Topographic

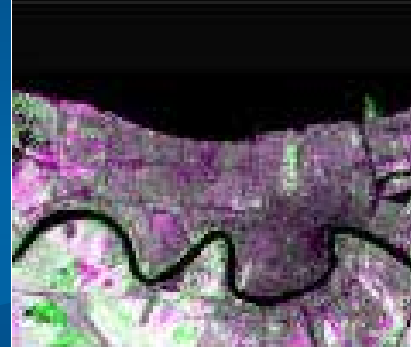
# Portal API Patterns

Dan O'Neill

FeaturedUserGroup Sar



Landsat Tim



Landsat ND



Glacier Retr

# 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

Alan Lucas

```
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 in
    PortalQueryParams queryParams = new PortalQueryParams();
    queryParams.setCanSearchPublic(true);
    queryParams.setSortField("name").setSortOrder(PortalQueryParams.SORT_ORDER_ASCENDING);
    queryParams.setQuery(createQueryString());

    // Find items that match the query
    PortalQueryResultSet<PortalItem> 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(item.getId(), item.getTitle(), bitmap);
            Log.i(TAG, "Item id = " + item.getTitle());
            mBasemapItemList.add(portalItemData);
        }
    }
}
}
```

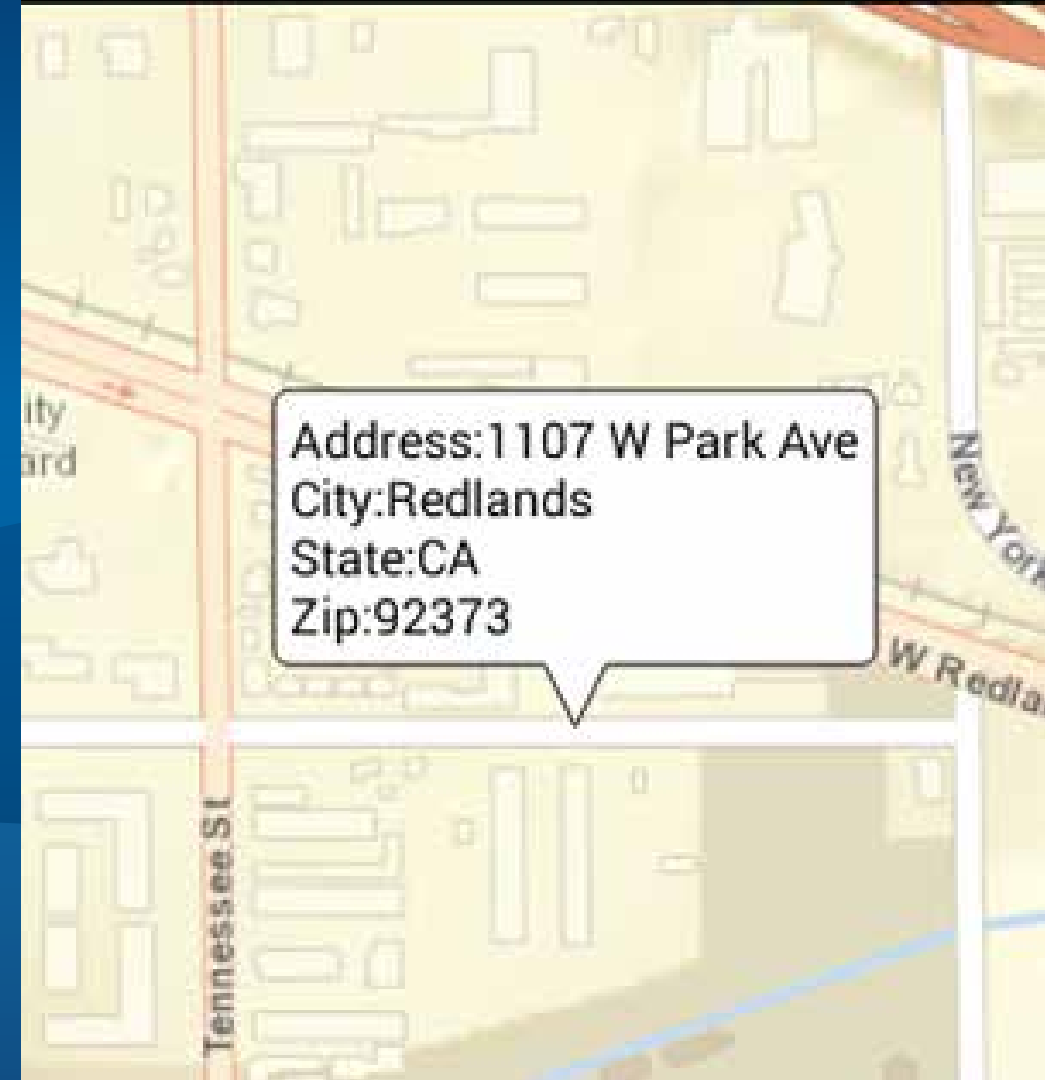
# Routing and Geocoding

Dan O'Neill

An input address: house #, street, city, state, zip

York St Redlands CA 92373

Go



# 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

Alan Lucas

```
protected RouteResult doInBackground(List<LocatorFindPa
// 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;

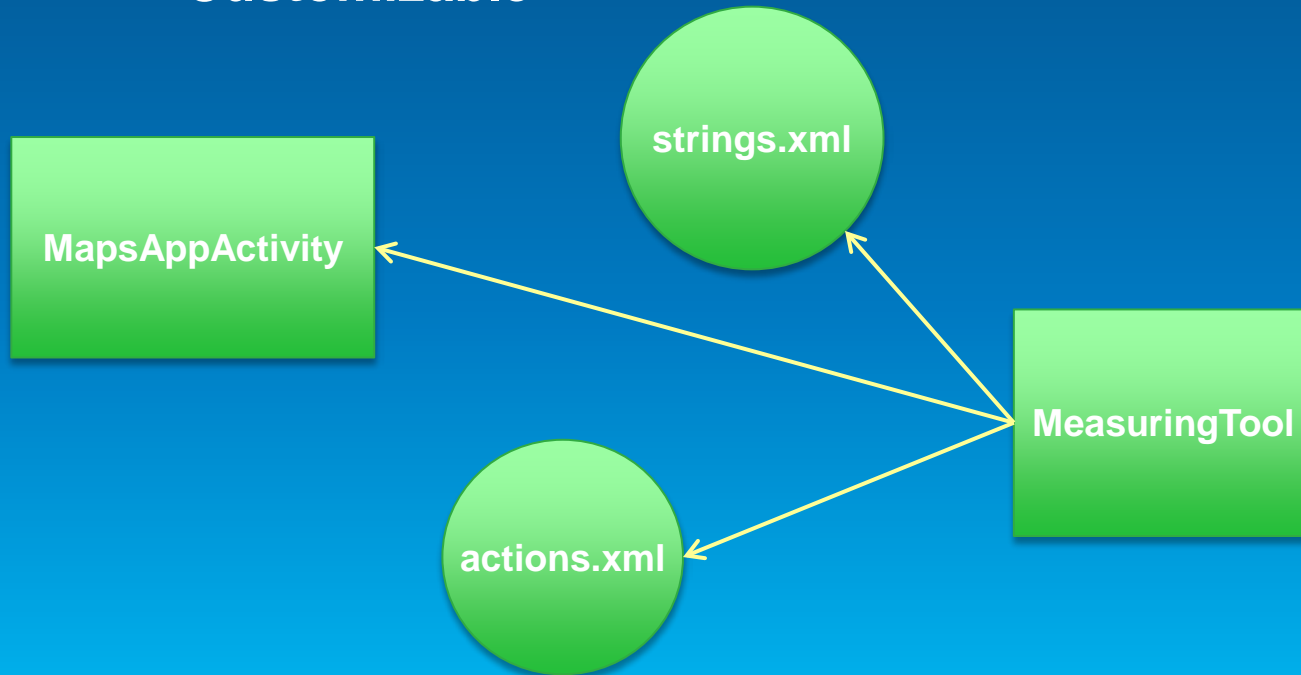
// Create a new locator to geocode start/end points
Locator locator = Locator.createOnlineLocator();

try {
// Geocode start position, or use My Location (from
LocatorFindParameters startParam = params[0].get(0);
if (startParam.getText().equals(getString(R.string.
startPoint = (Point) GeometryEngine.project(mLoca
} 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) {
mException = 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



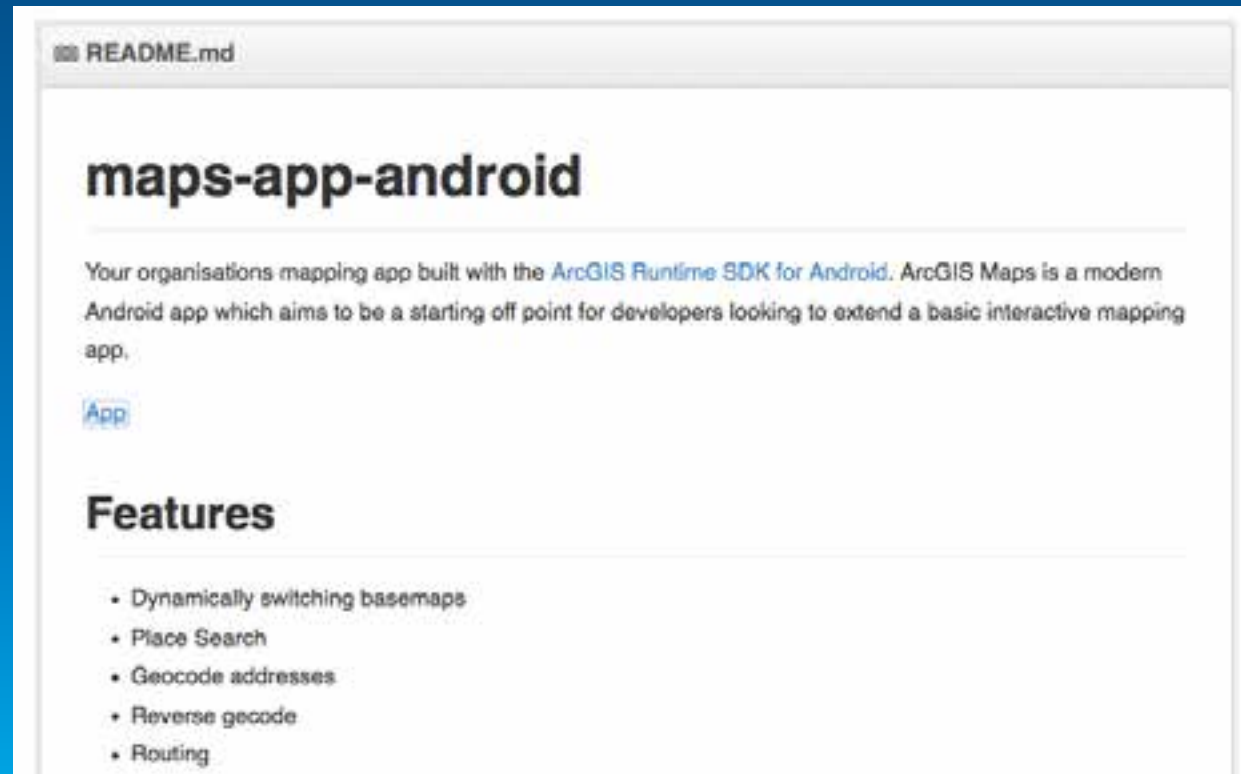
# Measure Integration Code Demo

Dan O'Neill

```
item
    android:id="@+id/menu_search"
    android:actionLayout="@layout/search_layout"
    android:icon="@drawable/ic_menu_search"
    android:showAsAction="ifRoom|collapseActionView"
    android:title="@string/menu_search"/>
item
    android:id="@+id/route"
    android:icon="@drawable/action_route"
    android:showAsAction="ifRoom"
    android:title="@string/menu_route"/>
item
    android:id="@+id/basemaps"
    android:icon="@drawable/action_basemaps"
    android:showAsAction="ifRoom"
    android:title="@string/menu_basemaps"/>
item
    android:id="@+id/location"
    android:icon="@android:drawable/ic_menu_mylocation"
    android:showAsAction="ifRoom"
    android:title="@string/action_location"/>
item
    android:id="@+id/directions"
    android:icon="@drawable/ic_routing_take_center_fork_d"
    android:showAsAction="ifRoom"
    android:title="@string/action_directions"
    android:visible="false"/>
item
    android:id="@+id/action_settings"
    android:icon="@android:drawable/ic_menu_edit"
    android:orderInCategory="100"
    android:showAsAction="ifRoom"
    android:title="@string/action_settings"/>
```

# Maps App on Github

- <https://github.com/Esri/maps-app-android>
- Get involved
- Report Issues
- Contribute Code
  - Fork it
  - Clone it
  - Configure remotes
  - Send pull requests



## Next release...

- **Direct read of raster datasets\***
- **Direct read of vector data (kml, shapefiles)**
- **3D\***
- **Simplification**
- **Toolkits**
- **Common conceptual model**
- **More offline capabilities**
  - **Feature service table**
  - **Versioned data support**

\*Some apis will release items before others...

# Agenda

- **What's new at 10.2.2**
  - Licensing API
  - Offline Patterns
- **Portal API Patterns**
  - Basemaps
- **Analysis Functions**
  - Routing and Geocoding
  - Measure
- **Contribute to Github**
- **What's coming**



# Thank You

**Questions**

**Session Surveys**

**<http://www.esri.com/events/devsummit/session-rater>**



Understanding our world.