

# Building Great Android App UIs and UXs



# WELCOME

## Presenters

Will Crick @willcrick

Shelly Gill @shellyegill

Dan O'Neill @jdoneill

# Agenda

- App patterns for location based experience
- Map-centric apps
- Map-as-navigation apps
- Map-as-context apps
- Mapless apps
- Material Design



# First, questions for you!

- Android devs already? / New to android dev?
- Already using our SDK?
- Released an android app in the store? Enterprise app?
- Using eclipse? / Using Android studio?

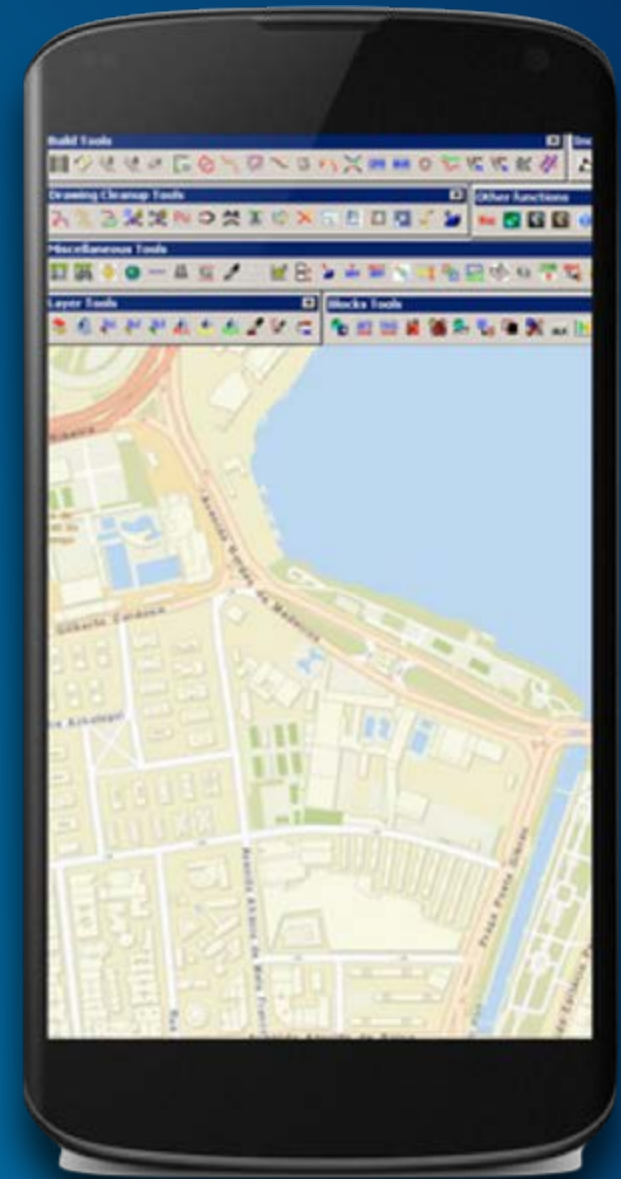
# Location based experience

## #LBX

Will Crick

# What is a good UI/UX for a location based app?

- **Who are you building your app for?**
  - **GIS analysts - tools, tools tools on a toolbar**
  - **Knowledge workers - minimum tool set to get the job done**
  - **Consumers - just what they need and make it look good**
- **Where are your users using it?**
  - **Office based - wifi, indoors**
  - **Outdoors - offline, glare, gloves?**
- **Whatever you do, dont make it slow or annoying!**
  - **Efficient tooling, fast screen loads, fast lists, fast maps!**
  - **Make alerts appropriate for usage**



# Before you start cranking code...

Ask yourself some questions...

- **What sort of location based app are you building?**
- **Location Based Experience (#LBX)**
  - **What value does location have to the user?**
  - **How will the user interact with location?**
  - **How does the app consume location?**
- **What are the patterns i can learn from?**
  - <https://developers.arcgis.com/android/guide/determine-your-app-map-pattern.htm>

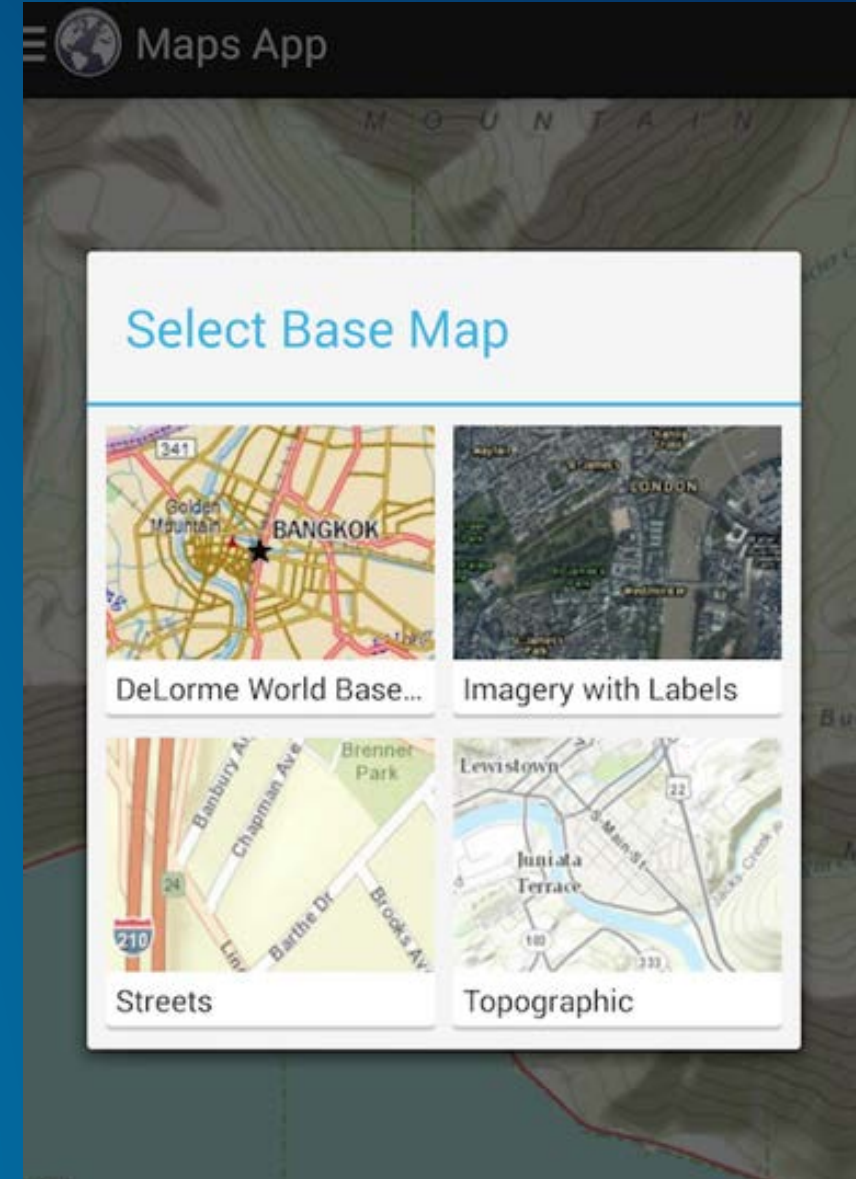
See [Building Great Apps for Mobile Devices: Tips and Tricks - Primrose B @ 5.30pm today!](#)



# Map-centric

Its all about the map

- **Get to the map quickly**
- **Care about cartography (as always)**
  - legend should be last resort
- **Save the state of the map**
  - last used, extent, active tools etc...
- **Full screen map on mobile, side panel on tablet?**
- **Provide tools in appropriate ways**
  - Use gestures, default tap action,
- **Offline maps?**
  - a) Take data offline from an online map
  - b) Pre-prepare offline data ready for app startup





# Map-as-app-navigation

The map is an app navigation tool

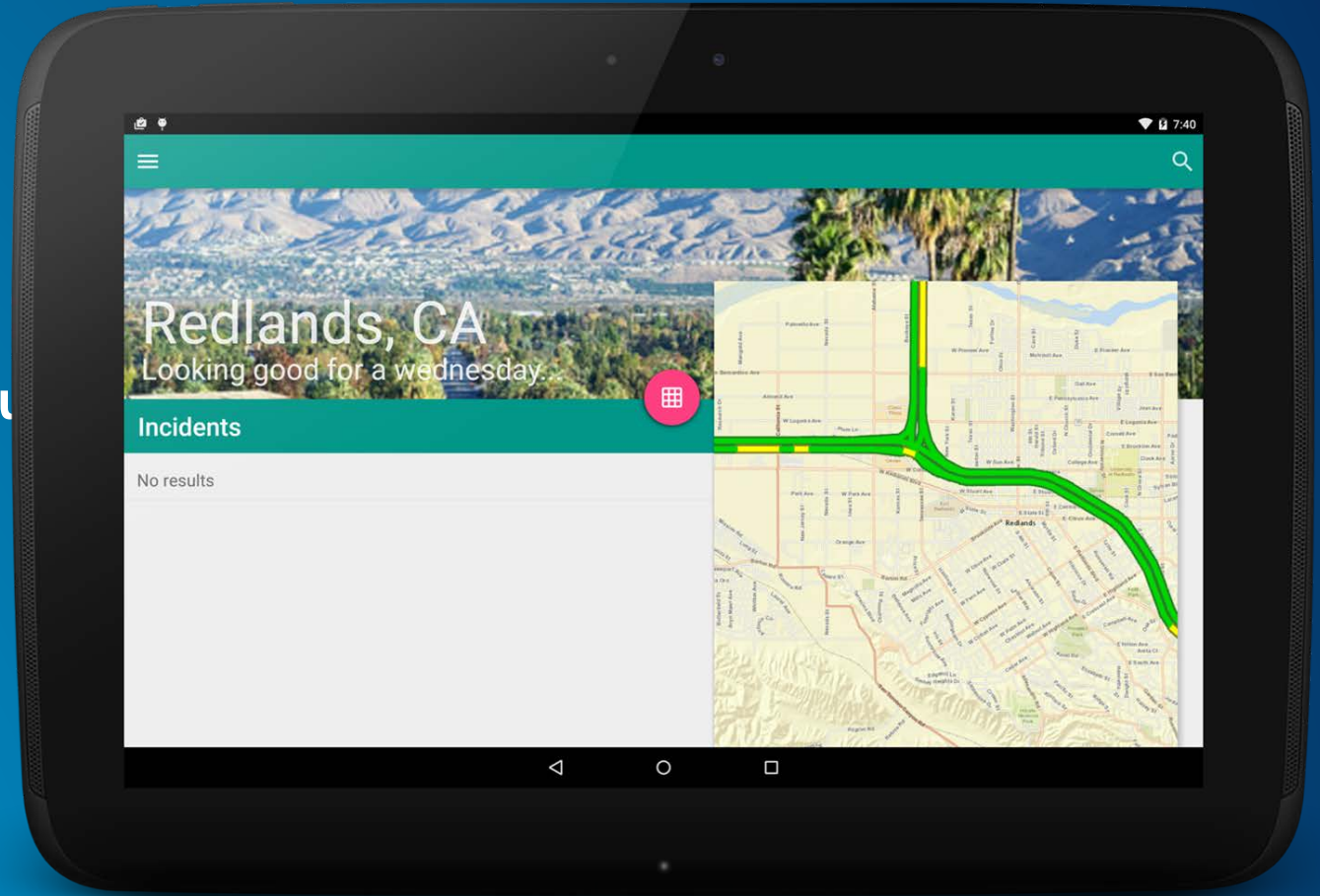
- **Map is just part of the apps screen flow**
  - has little value on its own
  - e.g. sketch graphic for use in spatial query on a layer
- **Focused toolset**
  - Simple, no distractions
  - Default actions only?
- **Simple cartography**
  - Intuitive, no distractions
- **Consider the use of popups to show information**



# Maps-as-context

Map is an auxiliary view that MAY add information

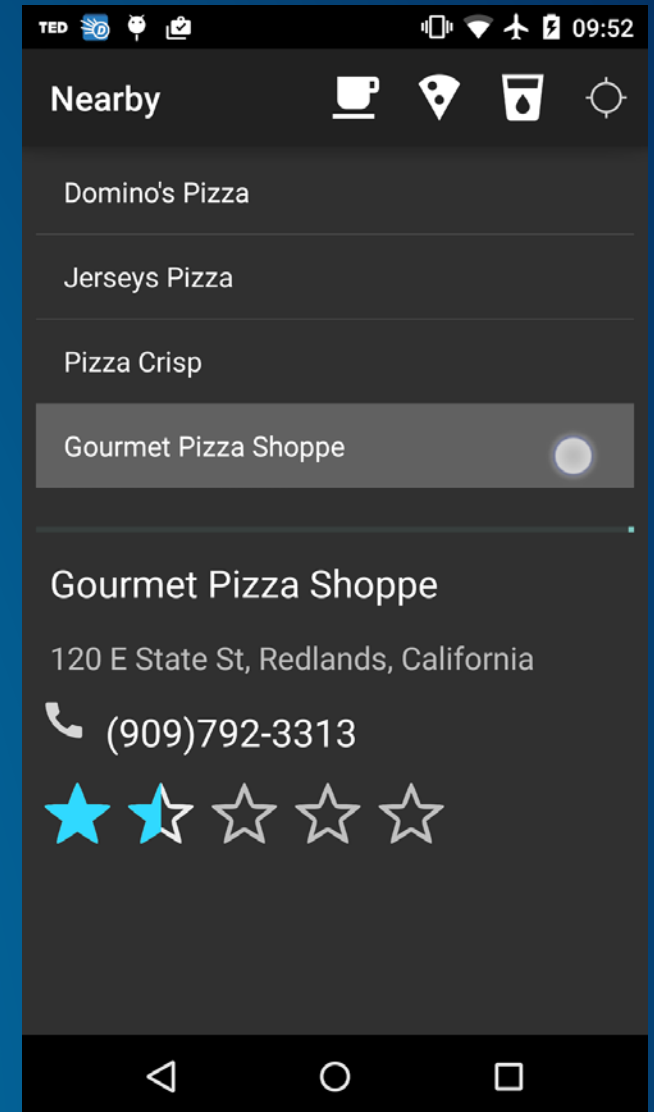
- Embedded in other information
- Map is driven by other content
  - Location, content and filters are populated
- Few or no tools



# Location-as-search

Use location to drive a list of items

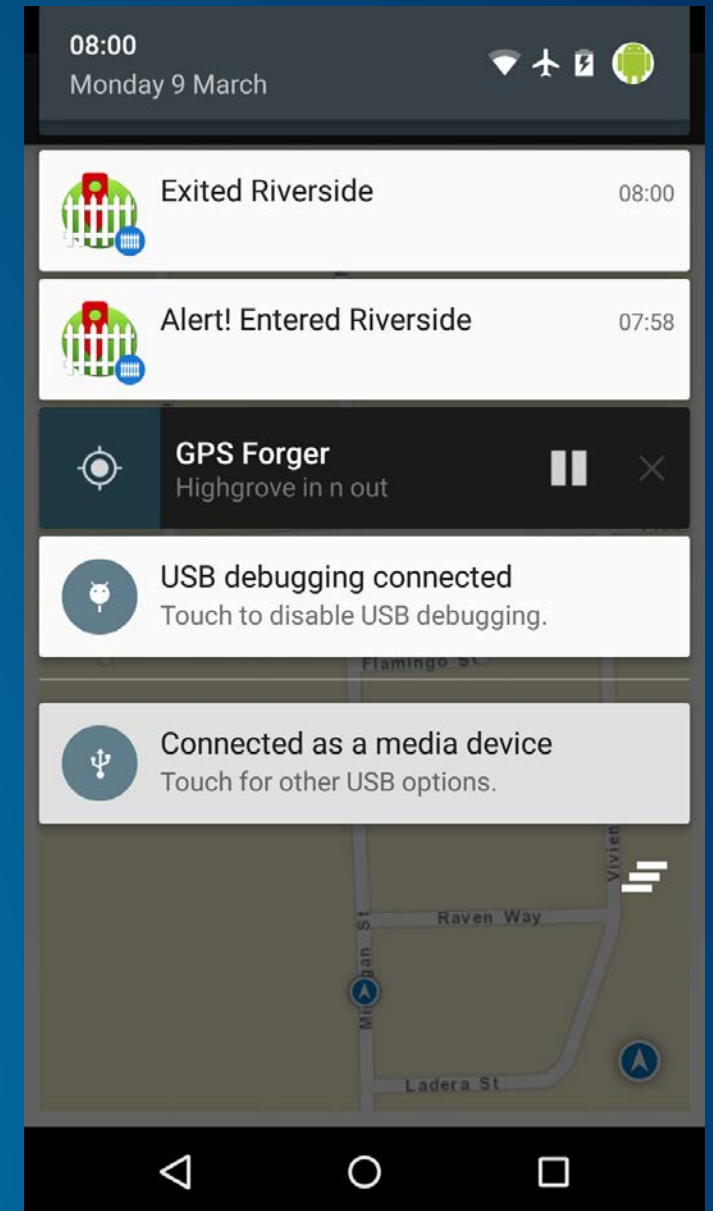
- **Current location provides basis for a search from data/service**
- **Retrieve location from the device location manager**
  - One time, at time of search
- **Sorting of the list?**
  - Nearest, largest etc...
- **Add spatial information to the list**
  - Distance, route time, size, pictures etc...



# Location-as-alerts

Use location to alert the user about proximity

- Can build on location-as-search apps
- Retrieve location from the device location manager
  - What is your strategy?
  - Be kind on the battery...
  - What is your tolerance?
- Provide alerts using background service & system notifications
- Work offline
  - typically yes - needs to work wherever!
- How often do the fences change?
  - How often do you need to update your fences?





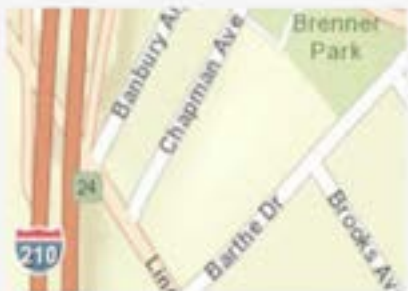
## Select Base Map



DeLorme World Base...



Imagery with Labels



Streets



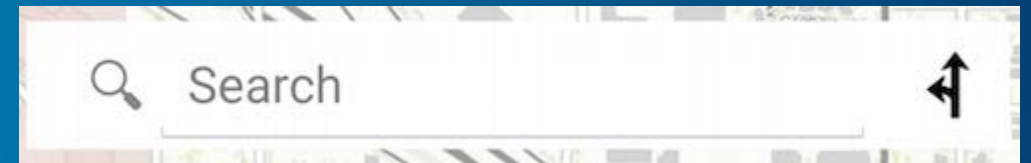
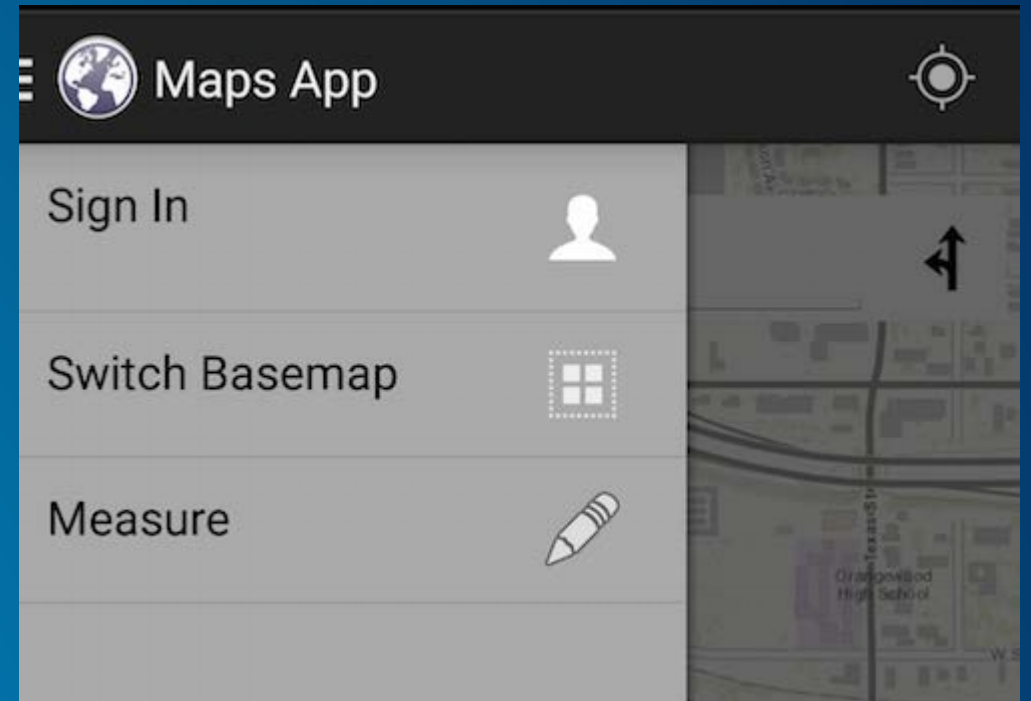
Topographic

# Map-centric Maps App demo & code

Dan O'Neill

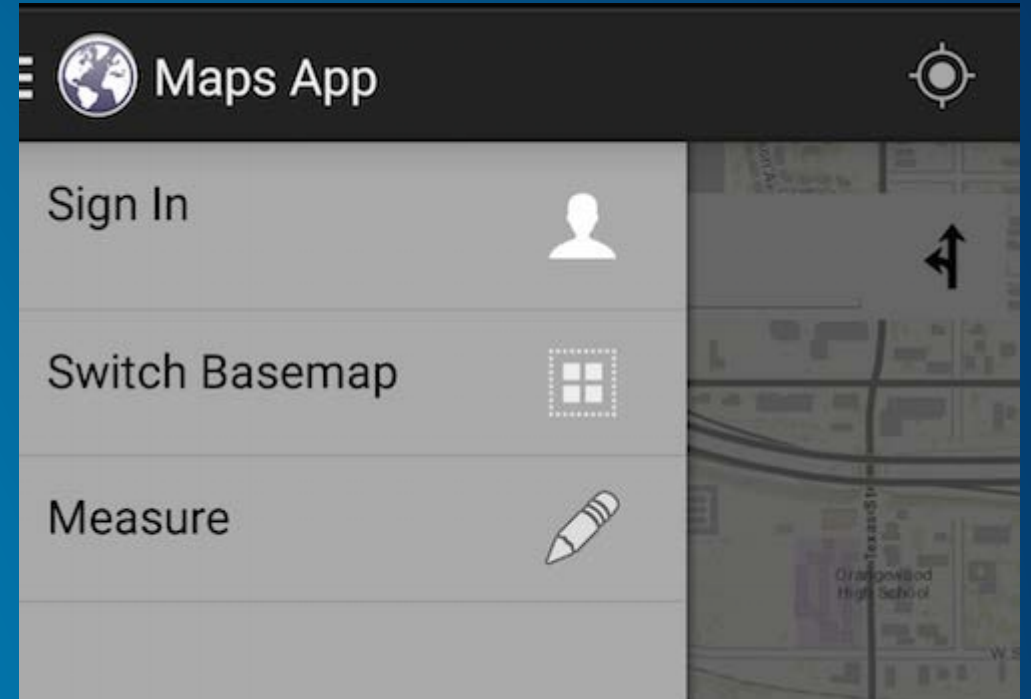
# Maps App

- **Material**
  - Navigation Drawer
- **Search Widget**
- **Floating Compass Button**
- **Fragments**
  - Routing Dialog
  - Directions Dialog
  - Basemaps Dialog



# Maps App - Navigation Drawer

- **Create a Drawer Layout**
  - main content view
- **Initialize the Drawer List**
  - **FrameLayout**
  - **ListView**
- **Handle Navigation click events**
  - **set OnItemClickListener on Drawer**
- **Listen for Open Close events**
  - **onDrawerOpen**
  - **onDrawerClosed**



# Maps App - Navigation Drawer

## Create Drawer Layout

- Set up in the Activity
  - maps\_app\_activity.xml

```
<android.support.v4.widget.DrawerLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/maps_app_activity_drawer_layout"
  android:layout_width="match_parent"
  android:layout_height="match_parent" >

  <!-- The main content view -->

  <FrameLayout
    android:id="@+id/maps_app_activity_content_frame"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

  <!-- The navigation drawer -->

  <ListView
    android:id="@+id/maps_app_activity_left_drawer"
    style="@style/drawer_listView_style"
    android:layout_width="240dp"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:background="@android:color/darker_gray"
    android:choiceMode="singleChoice"
    android:divider="@color/esri_gray"
    android:dividerHeight="1px" />

</android.support.v4.widget.DrawerLayout>
```



# Maps App - Navigation Drawer

## Initialize the Drawer List

- Inflate view from XML

Set the Lists click listener with **setOnItemClickListener()**

```
mDrawerLayout = (DrawerLayout) findViewById(R.id.maps_app_activity_drawer_layout);  
  
// Set the list's click listener  
mDrawerList.setOnItemClickListener(new DrawerItemClickListener());
```

# Maps App - Navigation Drawer

## Listen for Open Close Events

```
@Override
public void onDrawerClosed(View drawerView) {
    super.onDrawerClosed(drawerView);

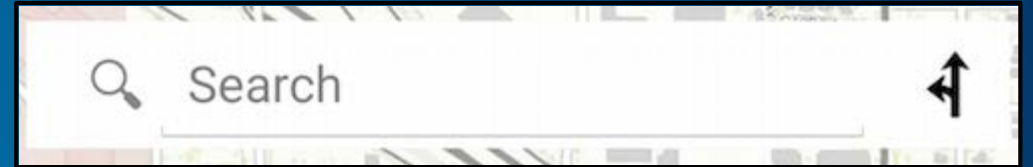
    invalidateOptionsMenu(); // calls onPrepareOptionsMenu()
}

@Override
public void onDrawerOpened(View drawerView) {
    super.onDrawerOpened(drawerView);

    invalidateOptionsMenu(); // calls onPrepareOptionsMenu()
}
};
```

# Maps App - Search Widget

- **Configure Search Widget**
  - **Inflate Layout SearchView**
  - **Set Search Parameters**
- **Set up Listeners**
  - **QueryTextListener**
    - **TextSubmit**
  - **Route Button**
    - **OnClickListener**



# Maps App - Search Widget

- MapFragment
  - Search widget is an instance of **SearchView**

```
// Setup the listener when the search button is pressed on the keyboard
mSearchview.setOnQueryTextListener(new OnQueryTextListener() {

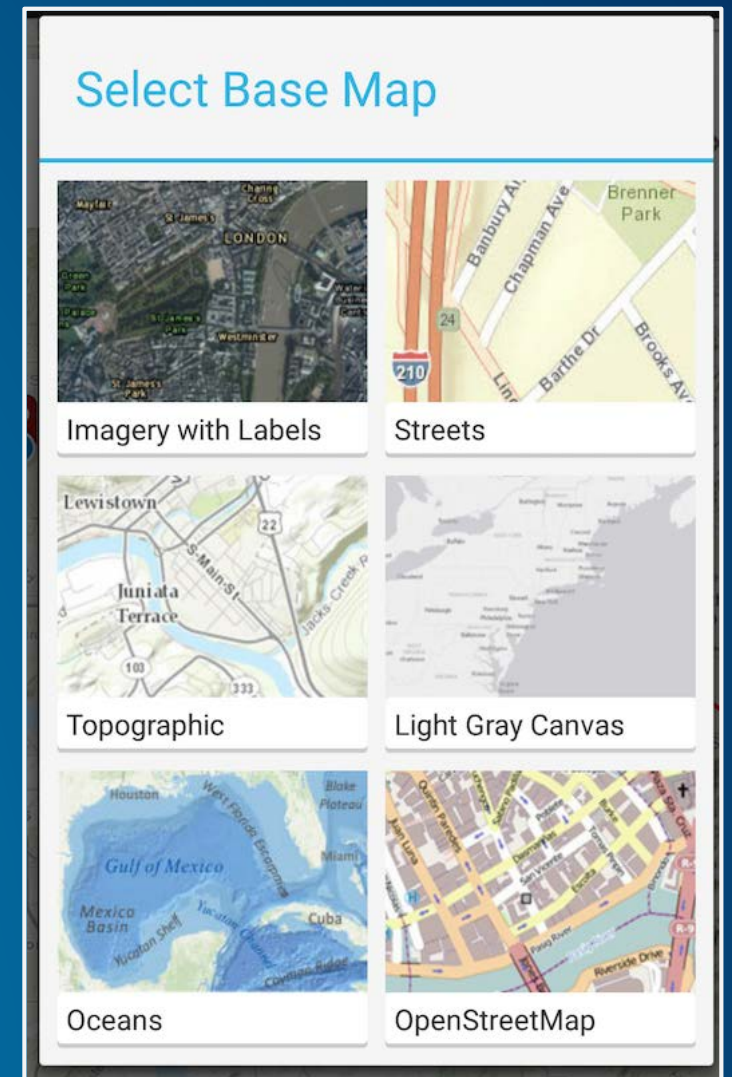
    @Override
    public boolean onQueryTextSubmit(String query) {
        onSearchButtonClicked(query);
        mSearchview.clearFocus();
        return true;
    }

    @Override
    public boolean onQueryTextChange(String newText) {
        return false;
    }
});
```



# Maps App - Fragments Basemap

- **Basemap Dialog is a fragment**
  - Floats on top of Activity Window
- **Backed by a Basemaps Adapter**
  - Registers the listener for clicks on thumbnails
- **Listens for Click Events**
  - Opens the Basemap



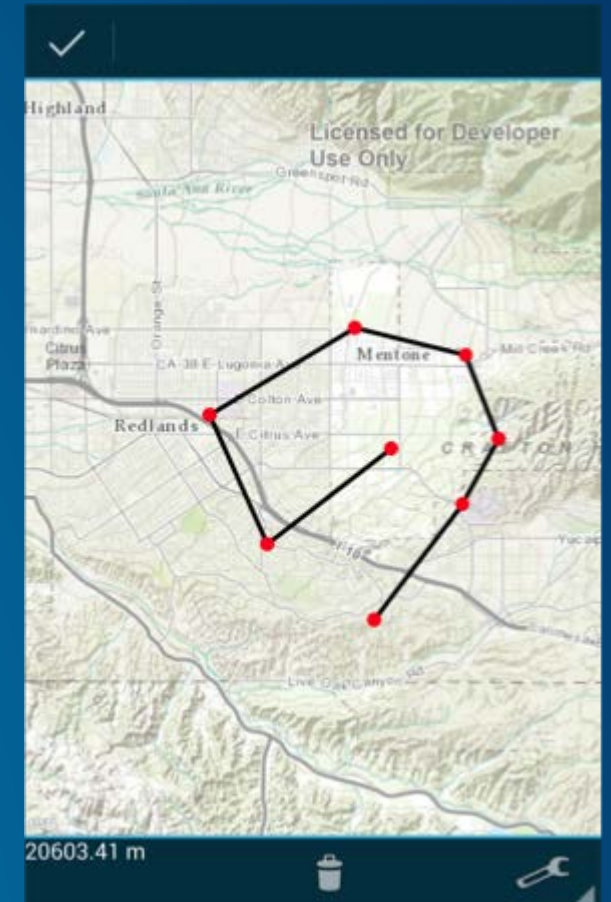
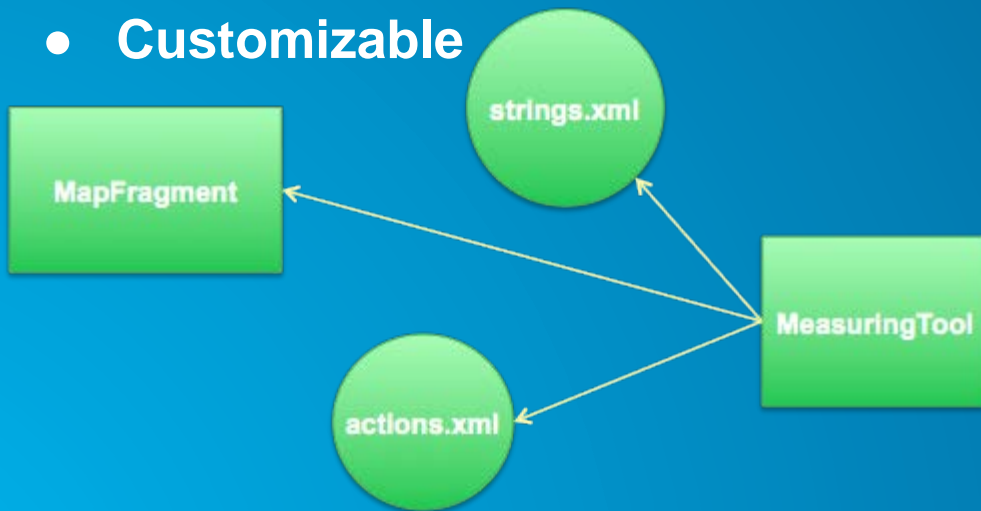
# Maps App - Basemap Fragment

- Inflate layout
- Create View
- Register listener
- Set title

```
public View getView(final int position, View convertView, ViewGroup parent) {  
  
    // Inflate view unless we have an old one to reuse  
    View newView = convertView;  
    if (convertView == null) {  
        LayoutInflater inflater = (LayoutInflater)  
            mContext.getSystemService(Context.LAYOUT_INFLATER_SERVICE);  
        newView = inflater.inflate(R.layout.basemap_image, null);  
    }  
  
    // Create view for the thumbnail  
    ImageView image = (ImageView)  
        newView.findViewById(R.id.basemap_grid_item_thumbnail_imageview);  
    image.setImageBitmap(items.get((position)).itemThumbnail);  
  
    // Register listener for clicks on the thumbnail  
    image.setOnClickListener(new OnClickListener() {  
        @Override  
        public void onClick(final View view) {  
            mListener.onBasemapItemClicked(position);  
        }  
    });  
  
    // Set the title and return the view we've created  
    TextView text = (TextView) newView.findViewById(R.id.basemap_grid_item_title_textview);  
    text.setText(items.get((position)).item.getTitle());  
    return newView;  
}
```

# Maps App - Configurable Measure Tool

- Measure Tool provided by ArcGIS Android Toolkit
- Contextual action bar
- Works in map centric apps
- Easy to integrate into your own app
- Customizable





## Maps App - Configurable Measure Tool

```
// initialize some resources for the measure tool, optional.
Unit[] linearUnits = new Unit[] {
    Unit.create(LinearUnit.Code.CENTIMETER),
    Unit.create(LinearUnit.Code.METER),
    Unit.create(LinearUnit.Code.KILOMETER),
    Unit.create(LinearUnit.Code.INCH),
    Unit.create(LinearUnit.Code.FOOT),
    Unit.create(LinearUnit.Code.YARD),
    Unit.create(LinearUnit.Code.MILE_STATUTE) };
SimpleMarkerSymbol markerSymbol = new SimpleMarkerSymbol(
    Color.BLUE, 10,
    com.esri.core.symbol.SimpleMarkerSymbol.STYLE.DIAMOND);
SimpleLineSymbol lineSymbol = new SimpleLineSymbol(Color.YELLOW, 3);
SimpleFillSymbol fillSymbol = new SimpleFillSymbol(Color.argb(100,
    0, 225, 255));
fillSymbol.setOutline(new SimpleLineSymbol(Color.TRANSPARENT, 0));
```



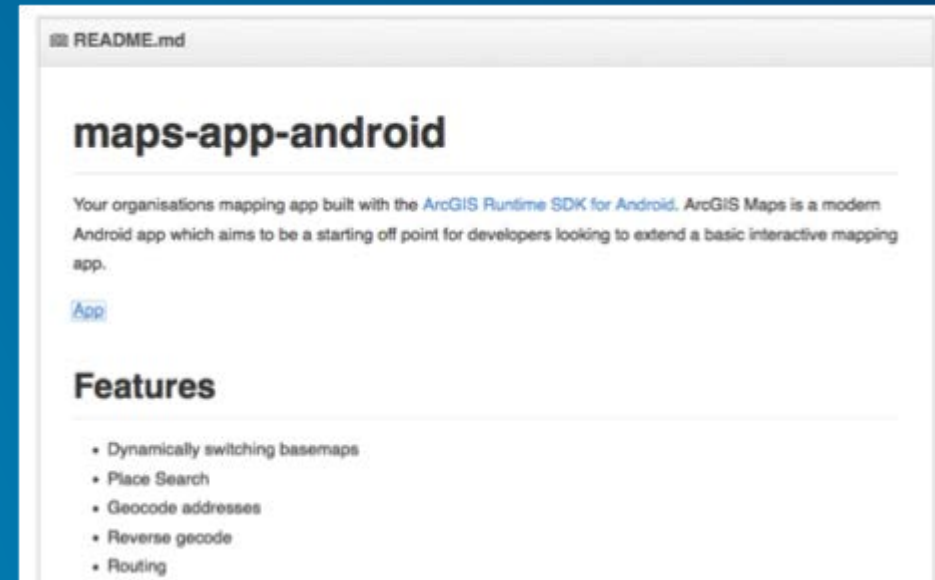
## Maps App - Configurable Measure Tool

```
// create the tool, required.  
MeasuringTool measuringTool = new MeasuringTool(MapFragment.mMapView);  
// customize the tool, optional.  
measuringTool.setLinearUnits(linearUnits);  
measuringTool.setMarkerSymbol(markerSymbol);  
measuringTool.setLineSymbol(lineSymbol);  
measuringTool.setFillSymbol(fillSymbol);  
  
// fire up the tool, required.  
startActionMode(measuringTool);
```

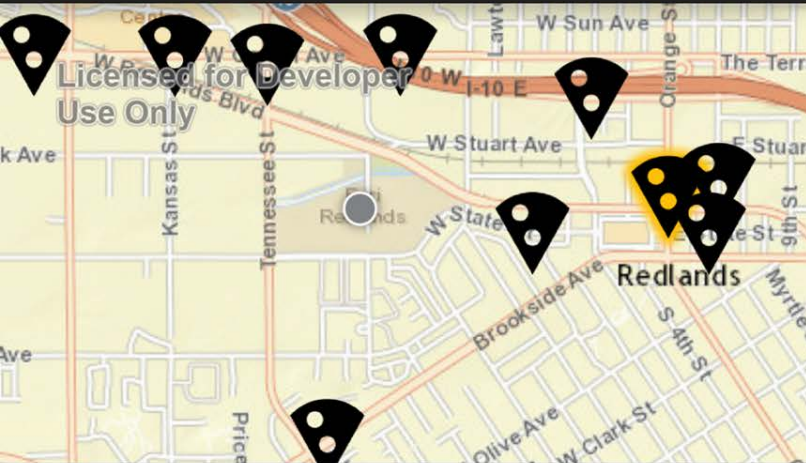
# Maps App on GitHub



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



Nearby



Jerseys Pizza

214 Orange St, Redlands, California

0.773241.2 miles away

(909)335-7076



# Map-as-app-navigation

## Nearby demo & code

Shelly Gill

# Map-as-app-navigation

Food and drink information nearby

- Updated Nearby sample
  - <https://github.com/Esri/arcgis-runtime-samples-android/tree/master/Nearby>
- Map present but less than half screen
  - Map allows user to navigate information
  - Select beer, coffee, or pizza to show locations on the map
  - Information about nearest item is shown below map
  - Tap on different item to change displayed information
  - Tap to call the establishment
- Change map extent to search different area
- Uses ArcGIS Online global locator service



## Code to know: locators

- Use ArcGIS Online World Locator
- Can find:
  - Street addresses
  - Administrative place names
  - Postal / zip codes
  - Points of interest
  - Businesses
- `Locator.find`
- `Locator.suggest`

```
mLocator = Locator.createOnlineLocator();
```

```
LocatorFindParameters fParams =  
    new LocatorFindParameters(searchFor);  
fParams.setSearchExtent(searchExtent, mMapSr);  
fParams.setOutSR(mMapSr);  
fParams.setOutFields(mOutFields);  
if(mLDM != null) {  
    Point currentPoint = getAsPoint(mLDM.getLocation());  
    fParams.setLocation(currentPoint, mMapSr);  
}  
mLocator.find(fParams, findCallback);
```

## Code to know: graphics and locator results

- In find callback, iterate LocatorGeocodeResults
- Create Graphics from result:
  - location
  - symbol
  - attributes(note conversion of HashMap types)
- Use Multipoint as easy way to get results extent

```
final CallbackListener<List<LocatorGeocodeResult>> callback =
    new CallbackListener<List<LocatorGeocodeResult>>() {

    @Override
    public void onCallback(List<LocatorGeocodeResult> results) {

        mResultsLayer.removeAll();
        if (results.size() > 0) {
            MultiPoint fullExtent = new MultiPoint();
            Symbol symbol = null;
            if (mCurrentSearchType == SearchType.BAR) {
                symbol = mBarMapIcon;
            } else if (mCurrentSearchType == SearchType.PIZZA) {
                symbol = mPizzaMapIcon;
            } else if (mCurrentSearchType == SearchType.COFFEE) {
                symbol = mCoffeeMapIcon;
            }

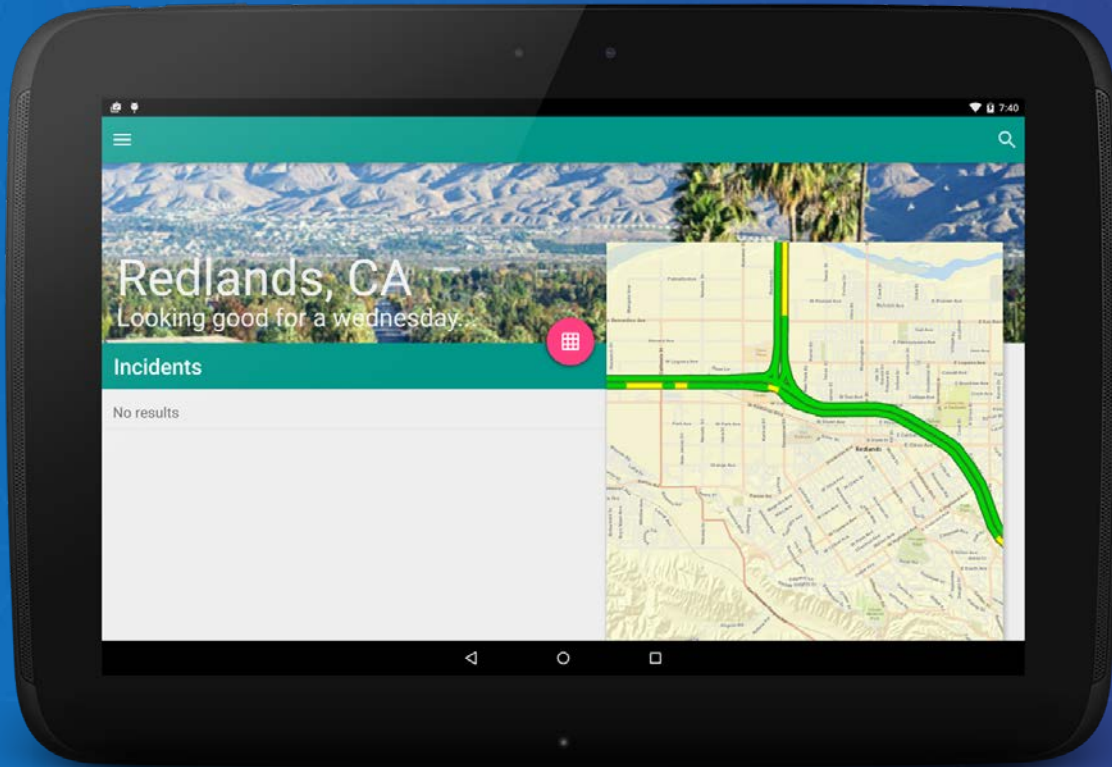
            for (LocatorGeocodeResult result : results) {
                Point resultPoint = result.getLocation();
                HashMap<String, Object> attrMap = new
                    HashMap<>(result.getAttributes());
                mResultsLayer.addGraphic(new Graphic(resultPoint,
                    symbol, attrMap));
                fullExtent.add(resultPoint);
            }
            mMapView.setExtent(fullExtent, 100);
        }
    }
}
```

## Code to know: graphic selection

- Set `MapView.setOnSingleTapListener`
- Use parameters to find graphics at screen coordinates and select them
- Use `Graphic.setAttributes` to store information, populate lower panel

```
final OnSingleTapListener mapTapListener = (x, y) -> {  
  
    int[] graphicIDs = mResultsLayer.getGraphicIDs(x, y, 25);  
    if (graphicIDs != null && graphicIDs.length > 0) {  
        if (graphicIDs.length > 1){  
            int id = graphicIDs[0]; // Only select first  
            graphicIDs = new int[] { id };  
        }  
  
        mResultsLayer.clearSelection();  
        mResultsLayer.setSelectedGraphics(graphicIDs, true);  
        Graphic gr = mResultsLayer.getGraphic(graphicIDs[0]);  
        updateContent(gr.getAttributes());  
    }  
}
```





# Map-as-context demo & code

Will Crick

<https://github.com/Esri/arcgis-runtime-demos-android/tree/master/2015-DS/TrafficApp>



## Create the map

- Using a webmap with secured layers
- Use a framelayout
- Add MapView to layout, pass in webmap and credentials

```
<FrameLayout
    android:id="@+id/map_frame"
    android:layout_width="550dp"
    android:layout_height="560dp"
    android:layout_marginLeft="700dp"
    android:layout_marginTop="110dp"
    android:background="@color/primary_material_light"
    android:elevation="6dp"/>
```

```
//setup map
FrameLayout mapFrame = (FrameLayout) findViewById(R.id.map_frame);
String mapUrl = getResources().getString(R.string.webmap_url);
mMapView = new MapView(getApplicationContext(), mapUrl, mUserCredentials, null, null);
mapFrame.addView(mMapView);
```

## Set the map extent on startup

- Do from result of a geocode (not actually implemented, but thats the pattern!)

```
//wait for mapview to load then set zoom and query for incidents
mMapView.setOnStatusChangeListener((source, status) -> {

    //Once map is loaded zoom to location
    if (source == mMapView && status == STATUS.INITIALIZED) {

        //Zoom to location
        Point point = new Point(-13046162, 4036352);
        mMapView.zoomToScale(point, 100000);
    }
}
```

## Query the incidents layer

- On app startup
- Get maps extent (centre point set by previous code)
- Use QueryTask within an AsyncTask
- Show results in a list
  
- This is the information the app is all about!

```
QueryParameters qParameters = new QueryParameters();
qParameters.setGeometry(mMapView.getExtent());
qParameters.setInSpatialReference(mMapView.getSpatialReference());
qParameters.setOutSpatialReference(mMapView.getSpatialReference());
qParameters.setOutFields(new String[]{"*"});
qParameters.setReturnGeometry(true);
qParameters.setSpatialRelationship(SpatialRelationship.INTERSECTS);

try {

    QueryTask qTask = new QueryTask("http://traffic.arcgis.com/arcgis
```

```
if (results != null && results.featureCount() > 0) {
    int size = (int) results.featureCount();

    Log.d("T_APP", "we have " + size + " results!");

    //add the results
    for (Object element : results) {
        if (element instanceof Feature) {
            Feature feature = (Feature) element;
            Graphic graphic = new Graphic(feature.getGeometry());
            //check for valid incidents which have a full description
            if (graphic.getAttributeValue("description") != null) {
                list.add(graphic);
            }
        }
    }
}
```

## Incidents click listener

- Get item from `ArrayAdapter<Graphic>`
- Zoom the map to the point geometry
- Add graphic so you know which one it is
  - in case there is many of them

```
//set the on click listener for the list view
mListView.setOnItemClickListener((parent, view, position, id) -> {

    //clear graphics layer
    mGraphicLayer.removeAll();

    //get the graphic
    Graphic graphic = adapter.getItem(position);

    //zoom the map to the graphic (they are already in the right SR)
    mMapView.zoomToScale((Point) graphic.getGeometry(), 50000);
    Graphic markerGraphic = new Graphic(graphic.getGeometry(), new SimpleMarkerSymbol(Color.CYAN, 15, SimpleMarkerSymbol.STYLE.CROSS));
    mGraphicLayer.addGraphic(markerGraphic);
}
```



## Set time extent on the map service layer

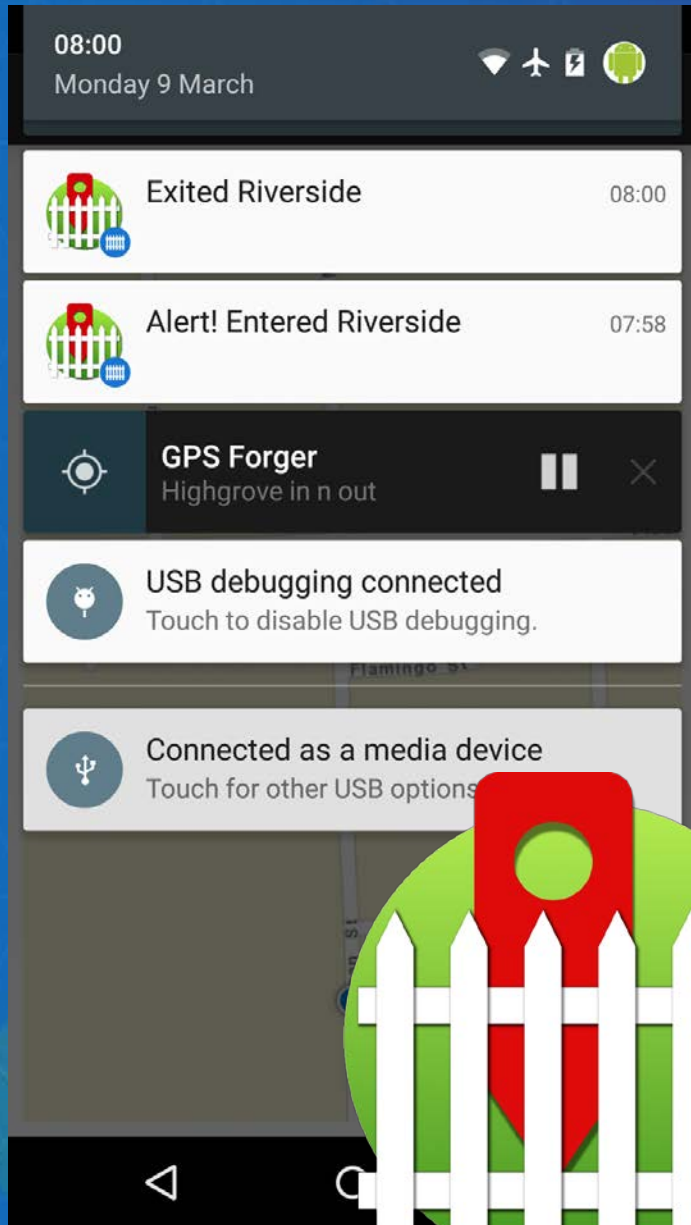
- Use date picker to get date
- Set time interval on map service layer
- Map becomes important here - as its the only way to visualize the historical information (no incidents saved in history - shame!)

```
Calendar newDate = Calendar.getInstance();
newDate.set(Calendar.YEAR, year);
newDate.set(Calendar.MONTH, monthOfYear);
newDate.set(Calendar.DAY_OF_MONTH, dayOfMonth);
TimeExtent timeInterval = new TimeExtent(newDate);
mTrafficLayer.setTimeInterval(timeInterval);
```

# Map-as-context

Driving conditions app

- **Use of a webmap**
- **Map driven by the date picker/click listener**
- **Used views, not fragments, could have used both**
- **When rotate, restart so that different resources can be loaded (not implemented yet!)**
- **Might also want to lock the map to fixed location (coming...)**
- **No map tools required!**



# Location-as-alerts

## Local geofence app demo & code

Shelly Gill

<https://github.com/Esri/arcgis-runtime-demos-android/tree/master/2015-DS/LocalGeofence>

# Location-as-alerts

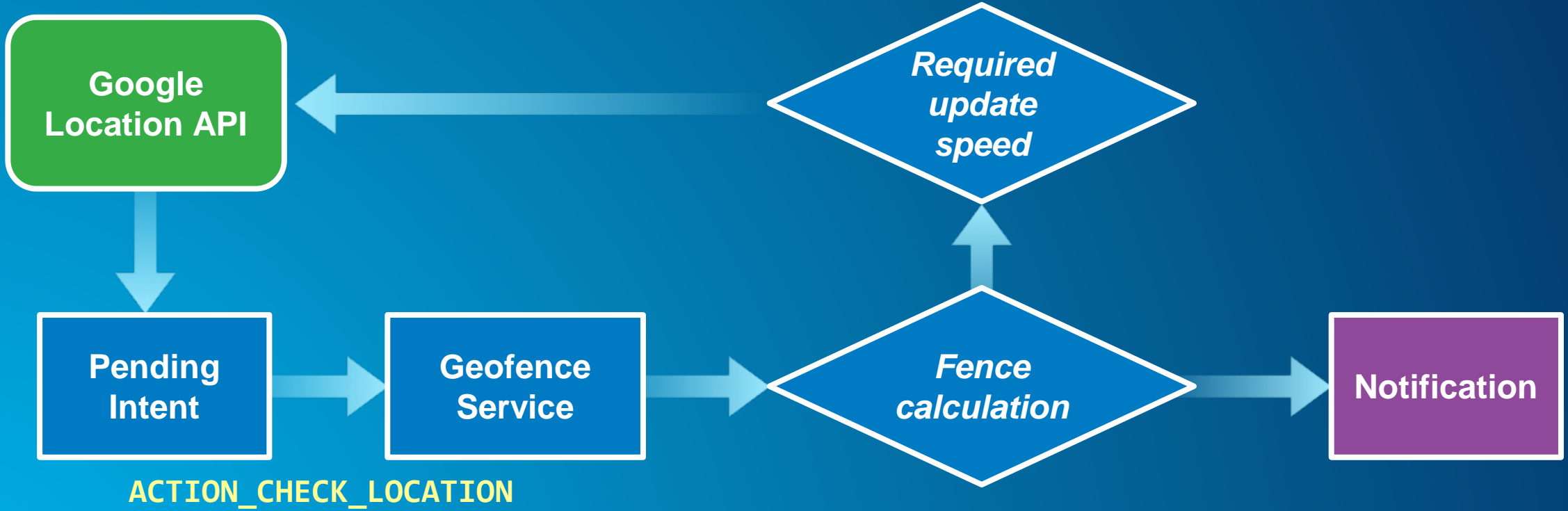
Local geofence app

- **Local Geofence demo**
  - <https://github.com/Esri/arcgis-runtime-demos-android/tree/master/2015-DS/LocalGeofence>
- **Uses offline geodatabase to store geofence features**
- **Notifications for alerting user when entering or exiting a polygon geofence**
  - Always available, outside app
  - User can control priority, sensitivity (lock screen)
- **Service for monitoring location updates**
  - Combines Google Fusion API and GeometryEngine
  - Increase update frequency when close by
- **Activity based UX allows user to choose fence**



# Geofence app process

`FusedLocationApi.requestLocationUpdates`



## Code to know: GeometryEngine

- Proximity2DResult gets nearest location on fence boundary
- geodesicDistance
- within
- project of fence feature to WGS84

```
private static boolean isCloseToFence(Point location) {
    Proximity2DResult proximity =
        GeometryEngine.getNearestCoordinate(
            mFenceWgs84, location, true);
    double distanceGeodesic =
        GeometryEngine.geodesicDistance(location,
            proximity.getCoordinate(), mWgs84Sr,
            mProximityUnits);

    return (distanceGeodesic < IS_CLOSE_DISTANCE);
}
```

## Code to know: Notifications

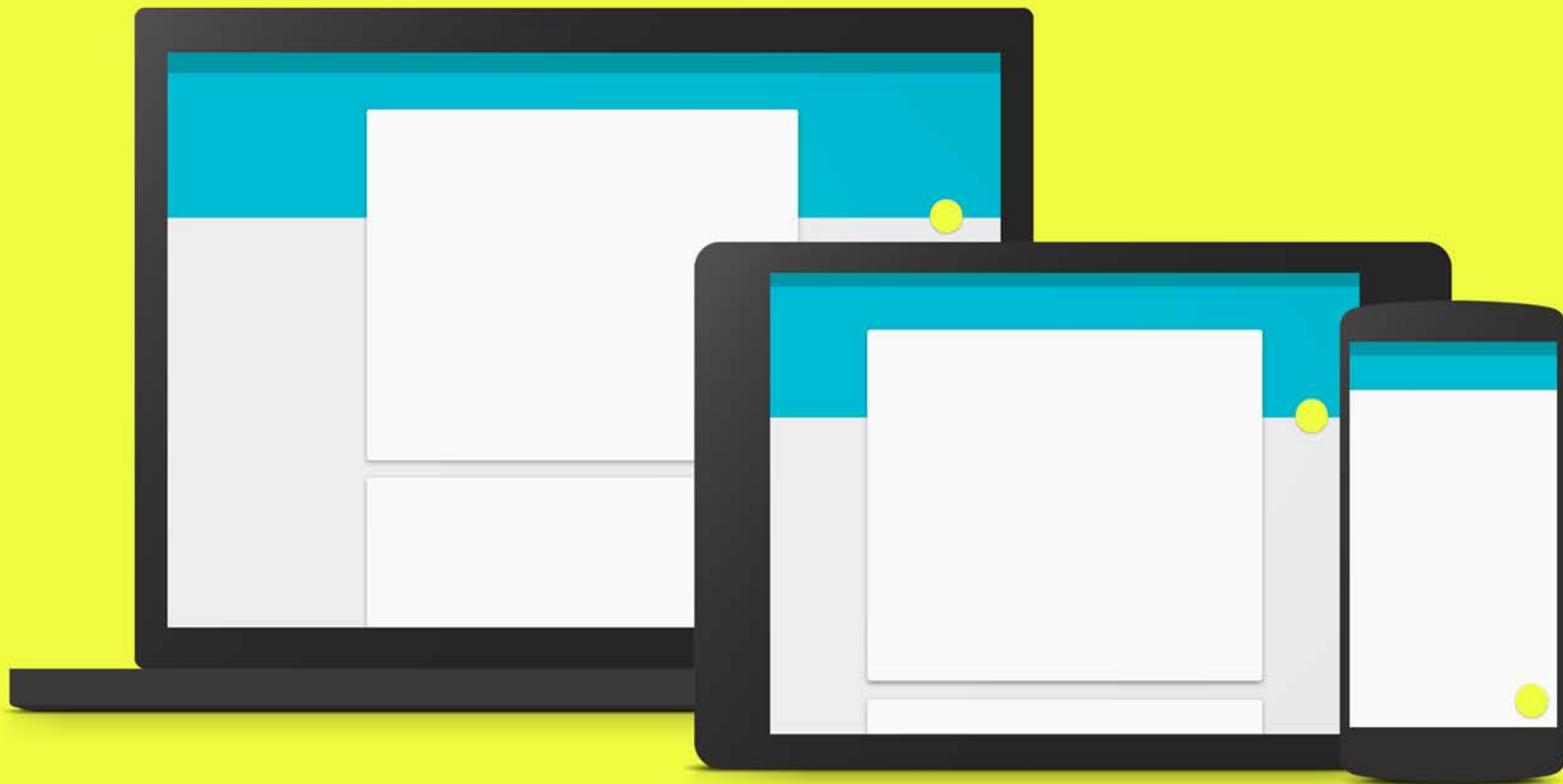
- Build and send notification using Feature attribute
- Notification API changes within SDK supported SDK versions

```
Notification.Builder nBuilder = new Notification.Builder(this);
nBuilder.setContentTitle("Alert! Entered " + LocalGeofence.getFeatureName());
nBuilder.setContentText(LocalGeofence.getSubtitle());
nBuilder.setSmallIcon(R.drawable.ic_fence_simple);

Bitmap largeIcon = BitmapFactory.decodeResource(getResources(),
    R.drawable.ic_geofence_bright);
nBuilder.setLargeIcon(largeIcon);

// Support Android SDK version 14 and up: API changes at 16
if (Build.VERSION.SDK_INT < 16) {
    notificationManager.notify(NOTIFICATION_ID, nBuilder.getNotification());
}
else {
    notificationManager.notify(NOTIFICATION_ID, nBuilder.build());
}
```

# What is material design?



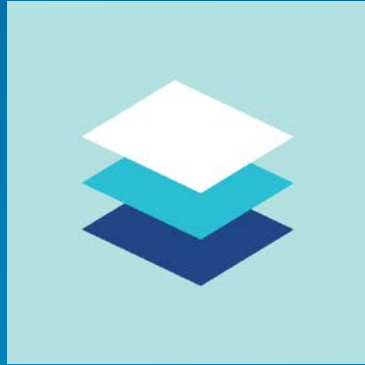


## Google's answer to Apples design dominance...

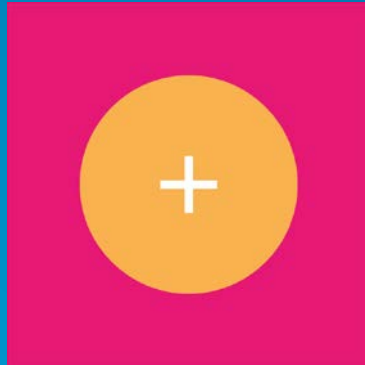
**“...a visual language for our users that synthesizes the classic principles of good design with the innovation and possibility of technology and science.”**

- **Unified experience across platforms, device sizes and human computer interaction (touch, voice, mouse & keyboard)**
- **Creating a Google brand AND an Android brand**

# Principles



Material is the metaphor – modern tech inspired by paper and ink  
(not skeuomorphic  
– does not replicate)



Bold, graphic, intentional – print based design guides visual treatment



Motion provides meaning – user actions initiate movement



# Material Design

Dan/Will

- **ArcGIS Basemaps app**
  - **Material design UI components used**
    - **RecyclerView**
    - **CardView**
    - **Floating Action Button**
    - **Animations**
    - **Ripples**
- **Map as context app**
  - **Used material theme and appcompat for backwards compatibility**
  - **Material theme is picky, only some views support params**
- **Need to use lots of custom drawables/shapes, hopefully templates are coming soon**





**Thank You**

Questions?



**Thanks!**

Please review session!