Best Development Practices and Patterns Using ArcGIS Runtime SDK for Java

Vijay Gandhi, Elise Acheson, Eric Bader
Overview

- Good to know
- Responsive UI & spatial reference
- Working with graphics
- Geoprocessing & asynchronous programming
- Working with feature layers
- Miscellaneous

Tips shown by improving a demo application
Good to know

- REST
- JSON
- Proxy
- Learn by example
Responsive UI

Elise Acheson
### Responsive UI

- **Tiled vs Dynamic layers**

<table>
<thead>
<tr>
<th>Tiled layers</th>
<th>Dynamic layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>pre-processed tiles (bitmaps)</td>
<td>vector data, images generated server-side on request</td>
</tr>
<tr>
<td>basemap data (static)</td>
<td>operational data (dynamic)</td>
</tr>
<tr>
<td>fast – performance independent of data complexity</td>
<td>performance depends on data complexity</td>
</tr>
<tr>
<td>visual only, no underlying data access</td>
<td>can access the underlying data: Find, Query, Identify</td>
</tr>
</tbody>
</table>
Spatial Reference

- Map’s spatial reference set by first layer added to map (can be tiled or dynamic layer)

- Check in/out-spatial reference for tasks
  - Query, Find, Identify, Geocode

- Use `jMap.getSpatialReference()` to be sure
Working with graphics
Vijay Gandhi
## Working with Graphics

- Use symbology that matches graphic’s geometry:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Geometry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimpleMarkerSymbol</td>
<td>Point</td>
<td>Symbolizes points with simple shapes</td>
</tr>
<tr>
<td>PictureMarkerSymbol</td>
<td>Point</td>
<td>Symbolizes points with images</td>
</tr>
<tr>
<td>SimpleLineSymbol</td>
<td>Polyline</td>
<td>Symbolizes lines with pre-defined styles</td>
</tr>
<tr>
<td>SimpleFillSymbol</td>
<td>Polygon</td>
<td>Fills polygons with a color and pre-defined styles</td>
</tr>
<tr>
<td>TextSymbol</td>
<td>Point</td>
<td>Displays text</td>
</tr>
</tbody>
</table>
Working with Graphics

• By default, Graphics appear in order of layers and graphics added

• Layer order
  - Currently in the order added
  - In 10.2, API to change order

• Graphics order
  - API on GraphicsLayer to change order
Know your GP service

- To know the parameter names, see the help URL or execute task or REST interface from a browser

Online:
- URL (REST endpoint) in a browser

Local:
- start the service, check when started
- get URL from console output, or programmatically
  - http://127.0.0.1:50000/PYVrpM/arcgis/rest/services/SimpleBuffer.gpk/GPServer/SimpleBuffer
Asynchronous programming patterns

- Don’t block UI thread on non-UI operations
- Use executeAsync() for tasks when possible
- Tasks include:
  - Search: Geocoding, Query, Identify, Find
  - Network Analysis: Routing, Closest Facility, Service Areas
  - Spatial Analysis: Geoprocessing
- Geoprocessing:
  - execute() – very quick tasks only, will freeze UI
  - executeAsync() – longer
  - submitJob() - long jobs / large result set
Working with feature layers

Vijay Gandhi
Feature Layers

Fetch only what you need from a feature service:

- Use the appropriate feature layer mode
- Use scale thresholds
- Do not return feature geometry if not required (query)
Miscellaneous

• Execute UI operations in UI Swing thread
• Can’t access online services
  - Set proxy using ProxySetup class or through JVM arguments
• Using many instances of JMap increases memory significantly
  - Call dispose() on JMap for instances not used anymore
• Consider using SDK’s open-sourced toolkit components instead of writing custom code
• Use relative paths for resources (local data, images, etc.)
• Set license using ArcGISRuntime class
• Use ArcGISRuntime class to change temp folder
Further information

• Samples
• (API reference, concept doc, Blog posts, forum)

• Oracle Tutorial: *Concurrency in Swing*
  [http://docs.oracle.com/javase/tutorial/uiswing/concurrency/index.html](http://docs.oracle.com/javase/tutorial/uiswing/concurrency/index.html)

• *Effective Java (2nd Edition)*
  - Java Best Practices book by Joshua Bloch

• Other references...

Offering ID: 297
Understanding our world.