

# Esri Developer Summit

March 26–29, 2012 | Palm Springs, California

[esri.com/events/devsummit](http://esri.com/events/devsummit)



## Building Applications with ArcGIS Runtime SDK for Java Part I

Mary Harvey, Ralf Gottschalk & Elise Acheson



# Agenda

- **Part 1 – 10:30am**
  - **Overview**
  - **SDK**
  - **Building the Map**
  - **Interacting with the Map**
  - **Deployment**
- **Part 2 – 1pm**
  - **Query**
  - **Geoprocessing**
  - **Editing**
  - **Local Server and Services Architecture**
  - **Product Summary & Business Model**
  - **Release plan and roadmap**

# ArcGIS Runtime

- Set of lightweight components
- Exploit performance of the operating system and hardware
- Integrate with ArcGIS system



# Product Functionality

- Mapping
- GPS
- Query
- Geocoding
- Editing
- Geoprocessing

**...Determines deployment size and licensing**

# What can you build?

Ralf Gottschalk



- Mapping
- GPS
- Query
- Geocoding
- Editing
- Geoprocessing

# What is the ArcGIS Runtime SDK for Java?

- **Software developer kit for building focused workflow-orientated GIS applications**
- **Technology**
  - **New Java SE Desktop SDK for Windows and Linux**
  - **Swing**
  - **Eclipse**
- **New GIS architecture based on ArcGIS**
  - **Small footprint**
  - **Native 64-bit and 32-bit**
  - **Asynchronous programming patterns**
  - **Side by side SDKs**



# Getting Started

- **Part of ESRI Developer Network (EDN)**
  - **Download from EDN website / DVD**
- **Prerequisites**
  - **Windows and Linux**
  - **Java SE Development Kit Version 6 or 7**
  - **Eclipse IDE for Java Developers (Indigo)**
  - **Windows - DirectX End-User Runtime 9.0c**
  - **Linux/Windows – OpenGL**
- **Install the SDK**
- **License for Machine for Development (timeout)**



# ArcGIS Runtime SDK for Java

Elise Acheson



- **Start Menu**
- **Samples**
- **Documentation**
- **IDE Integration**
- **Resource Center**



# It's All About Services!

- **Supported Services at 1.0**

<b>Services</b>	<b>ArcGIS Server/Online</b>	<b>Local Services</b>
Map Service	ü	ü
Feature Service	ü	ü
Geocode Service	ü	ü
Geoprocessing Service	ü	ü

- **Local Server with Local Services**
- **Asynchronous programming patterns**
- **Manage the services to maximize your workflow**

# Java API - Packages

Package(com.esri.....)	Details
.runtime	ArcGIS Runtime initialization & license
.map	Visible Map Control, Map Overlay and ArcGIS for Server Layer Types
.local	Local layer types Local service management Runtime Local Server management
.core	Core functionality: Graphics, geometry, symbology, renderer and tasks such as Geocode, Identify, Query, Geoprocessing
toolkit	UI controls to support map navigation, layer interaction, editing

# Building the Map

```
esri.symbol.SimpleLineSymbol,
new dojo.Color([0,0,0,0.5]),
polySymbol,
feature.setSymbol(polySymbol);
} else if(f == 1) {
var polySymbolGreen =
polySymbolGreen.setOutline(
symbol.SimpleLineSymbol(esri.symbol.
Color([0,0,0,0.5]), 1));
polySymbolGreen.setSymbol(polySymbolGreen);
feature.setSymbol(polySymbolGreen);
} else if(f == 2) {
var polySymbolBlue = new esri.symbol.SimpleLineSymbol(
new dojo.Color([0,0,0,0.5]), 1);
polySymbolBlue.setSymbol(polySymbolBlue);
feature.setSymbol(polySymbolBlue);
}
```

# Map Control

- **Swing Control – JMap**
- **2D map display - supports ArcGIS Cartography**
- **DirectX or OpenGL – Hardware utilization**
- **Default Constructor**
  - **Spatial reference – first layer**
  - **Full extent – Union of all layers in the map**
- **Place into top level Swing components - JFrame**
  - **`JFrame.getContentPane().add(JMap)`**
- **Properties**
  - **Spatial Reference, Rotation, Zoom, Dispose**



# Building the map



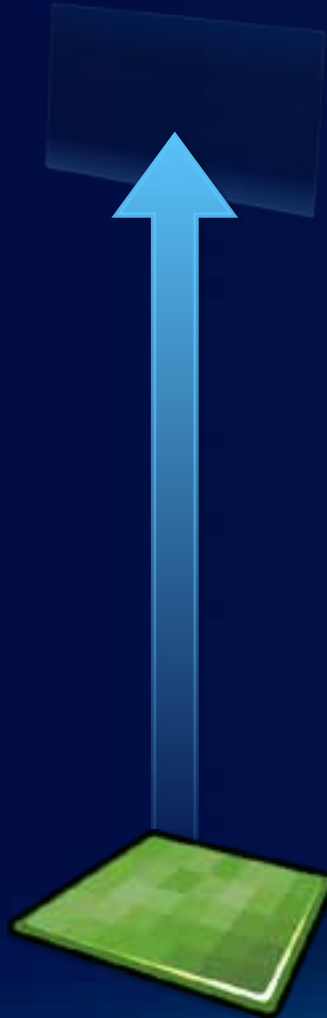
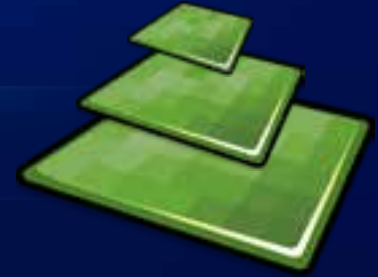
- **Map Control**
- **Live / temporary data**
  - Vehicles, people, events...
- **Operational data**
  - Facilities, zones, networks...
- **Basemap**
  - Imagery, topography...

**Graphics**

**Dynamic**

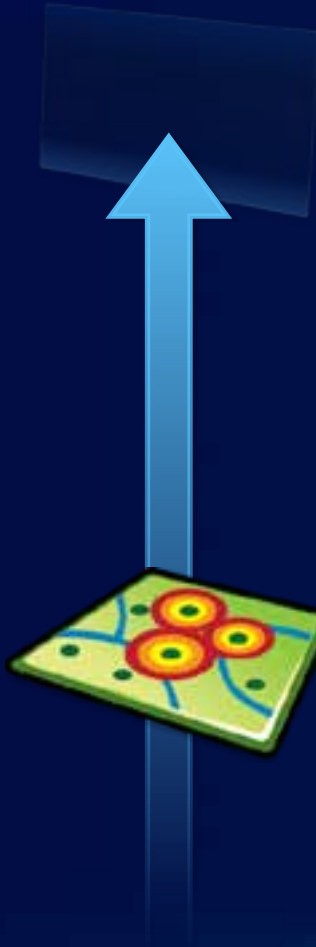
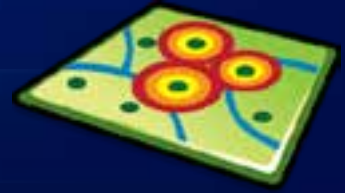
**Tiled**

# Tiled Basemaps



- **A visual context for operational data**
- **Seamless / continuous data**
  - E.g. Topography, Imagery, Streets
- **Entire map pre-rendered as tiles at defined scales (online or .tpk)**
- **Coordinate system baked into tiled map**
  - Cannot be re-projected on the fly
  - Should match intended coordinate system of application & other layers

# Dynamic operational layers



- **Map image dynamically rendered from data**
  - Delivered on per request basis
- **Developer can control**
  - Extent, visible layers, symbology, data source
- **Can be reprojected on the fly based on the requested spatial reference**
- **Online MapServer or MPK**
- **Supported data (analyzers)**

# Building the Map

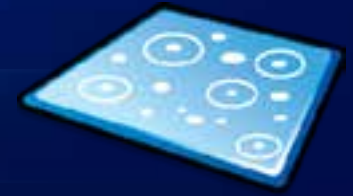
Ralf Gottschalk



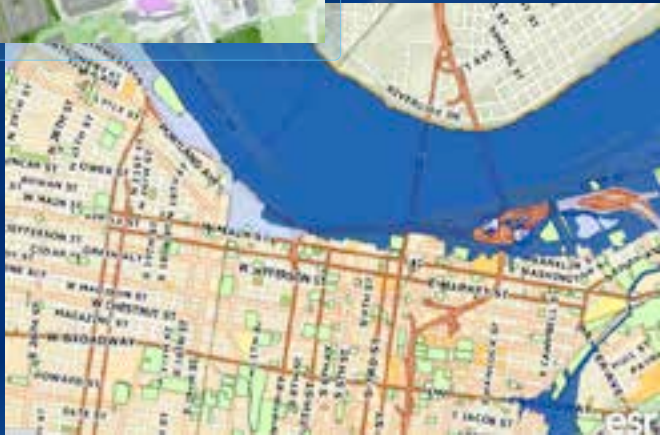
- Map Display
- Basemap
- Operational



# Graphics layers



- **Graphics layers for display of temporary features**
  - Vehicles, events, query results, user interaction
- **Manipulate geometry**
- **Set attributes**
- **Symbol class**
- **Graphics are in memory on client**
  - Performance dependent on number and complexity of features



# Graphics Layer

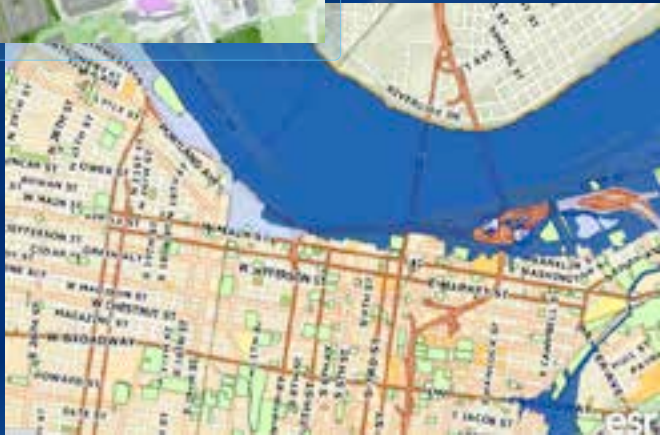
Elise Acheson

- Graphics Layer

# GPS Layer

- **Computers today are more mobile than ever**
- **Location of the Device is important**
- **Solutions are taking advantage of this**
- **GPSLayer**
  - **Displays Data from a GPSWatcher**
  - **Change the look and feel of the symbol used to display position**
  - **Support NMEA Sentences**





# GPS Layer

Elise Acheson

- GPS Layer

# Military Message Processing

- **Military messages**
  - Take military messages from the “wire”
  - Using Military Symbology (Tactical Graphics and Force Elements)
  - Insert, Update, Delete – directly into the map
  - Supports Mil2525C format
  - Final -



# Interacting with the Map

```
esri.symbol.SimpleLineSymbol,
new dojo.Color([0,0,0,0.5]),
polySymbol.setColor(new dojo.Color([0,0,0,0.5]));
feature.setSymbol(polySymbol);
} else if(f == 1) {
var polySymbolGreen = new esri.symbol.SimpleLineSymbol(
polySymbolGreen.setColor(new dojo.Color([0,0,0,0.5]), 1));
polySymbolGreen.setSymbol(polySymbolGreen);
} else if(f == 2) {
var polySymbolBlue = new esri.symbol.SimpleLineSymbol(
polySymbolBlue.setColor(new dojo.Color([0,0,0,0.5]), 1));
polySymbolBlue.setSymbol(polySymbolBlue);
}
```

# MapOverlay

- **JMap contains the Map**
- **Control MouseInteraction**
  - Capture Mouse events happening on the Map Control
  - For example: Customized tool or special behavior
- **Display non-geographical components**
  - Paint on top of the MapControl
  - For example: Company Logo or Copyright Statement
- **Com.esri.map.MapOverlay**

# MapOverlay – Mouse Methods

- **Extent MapOverlay**

1. **Import com.esri.map.MapOverlay;**
2. **private class MouseMoveOverlay extends MapOverlay {  
    @Override  
    public void onMouseMoved(MouseEvent arg0){....}**
3. **map.addMapOverlay(new MouseMoveOverlay());**

- **Override Mouse methods**

- **Clicked, Dragged, Pressed, Released, Moved, MouseWheelMoved**
- **Beware multiple MapOverlays!**

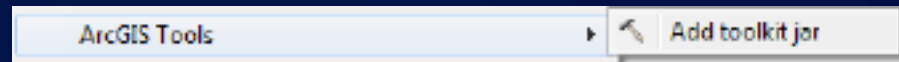


# Specialized Overlay Classes

- **Toolkit Class Overlays**
  - **HitTestOverlay**
  - **InfoPopupOverlay**
  - **NavigatorOverlay**
  - **ScaleBarOverlay**
  - **FeatureEditOverlay**
  - **FeatureCreateOverlay**
- **All these implement MapOverlay**

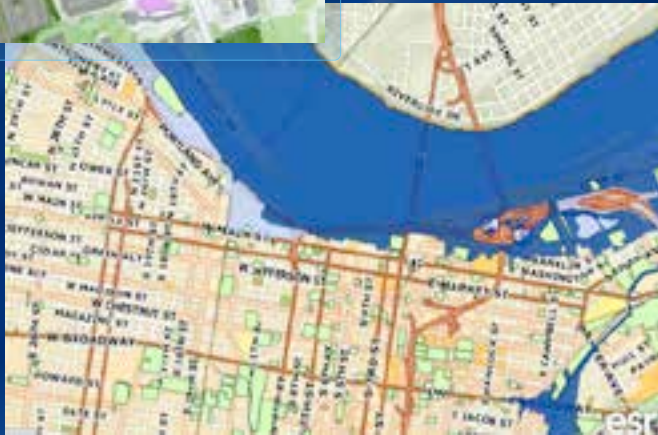
# Toolkit

- **Com.esri.toolkit**
  - **.attachments**
  - **editing**
  - **Infopopups**
  - **overlays**
- **Toolkit**
  - **ArcGIS\_Runtime\_Java\_Toolkit.jar**
- **Documentation**
  - **ArcGIS\_Runtime\_Java\_Toolkit-javadoc.jar**
- **Source Code**
  - **ArcGIS\_Runtime\_Java\_Toolkit-sources.jar**
- 



# Interacting with the Map

Ralf Gottschalk



- Toolkit
- Scalebar/InfoPopup
- Add Logo to MapOverlay
- Click on the Map

# Deployment

```
esri.symbol.SimpleLineSymbol,
new dojo.Color([0,0,0]),
polySymbol(feature.setSymbol(
} else if(f == 1) {
var polySymbolGreen =
polySymbolGreen.setOutline(
symbol.SimpleLineSymbol(esri.symbol.
Color([0,0,0,0.5]), 1));
polySymbolGreen.setSymbol(polySymbolGreen);
} else if(f == 2) {
var polyBlue = new esri.symbol.SimpleLineSymbol(
polyBlue.setOutline(new
esri.symbol.SimpleLineSymbol(
new dojo.Color([0,0,255,0.5]), 1));
```

# Deployment

- **Aims of Deployment**
  - **Easy to deploy**
  - **Small as possible**
  - **Side by Side deployments on the same machine**
  - **Simple Licensing**
  - **No License timeout**
  - **No registration**

# Deploying Your Application

1. **License your application**
2. **Obtain ArcGIS Runtime components**
3. **Create Deployment**

# 1. License for Deployment

- **Basic**
  - Full client to ArcGIS Server services
  - Local Tile Packages & GPS Support
- **Standard**
  - Local Map, Geoprocessing and Locator Packages
  - Geodatabase Editing & Routing
- **Extensions**
  - Spatial Analyst, 3D Analyst, Network Analyst
- **Determine the Type and Number of Licenses**
- **Purchase Runtime Licenses**

# 1. License for Deployment

- Enable Licenses using Software Authorization Wizard
- Use License Viewer to get license string



- Call `ArcGISRuntime.setLicense("runtimestandard,.....")`



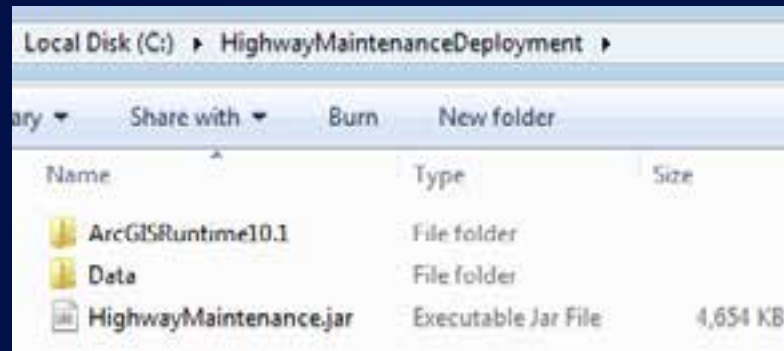
## 2. Obtain ArcGIS Runtime components

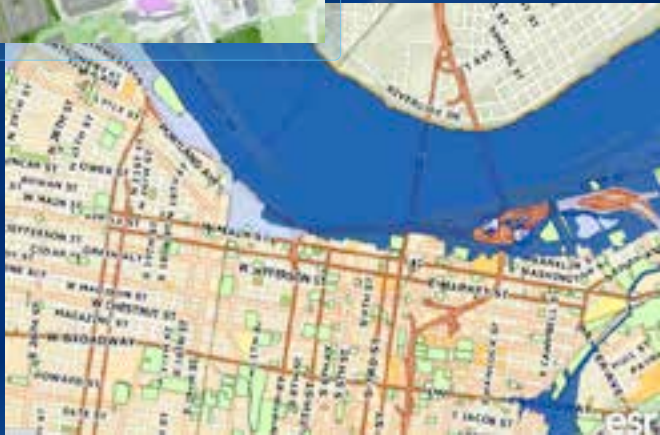
- Deploy only what you need



### 3. Create Deployment

- Create application – Runnable jar
- Place it alongside ArcGIS Runtime folder
- Add any other resources





# Deployment

Ralf Gottschalk

- Licensing
- Deploying an Application

# Deployment

- **Simple Deployment**
- **Deployment is self contained**
  - **Licensed**
  - **ArcGIS Runtime components**
- **Side by Side deployments**
- **Simple Uninstall**

# Benefits

- **Solid foundation for developers moving forwards**
  - Pure JavaSE API
  - Native 32 and 64 bit code execution
  - Utilizes hardware (Cores, CPUS, GPUs)
  - Asynchronous programming pattern
  
- **Simplified Deployment**
  - No install required
  - Deploy only needed components
  - Side-by-Side deployment
  - Independent of other ArcGIS installs



esri