



Developing Windows Desktop Apps with ArcGIS Runtime SDK for Microsoft .NET Framework

Thad Tilton - @T_hadde_us

Apurva Goyal

Technical Workshop

Session overview

What will we cover?

- **What is the ArcGIS Runtime SDK for .NET?**
 - ArcGIS Runtime Core
 - ArcGIS Runtime APIs and SDKs
- **Get started**
 - Installation and system requirements
 - Documentation, tutorials, and samples
- **Display a map**
 - MapView and Map controls
 - Types of layers

Session overview (continued)

What will we cover?

- **Work with tasks**
 - Query
 - Geocode and Routing
- **Work offline**
 - Store features locally (offline geodatabase)
 - Make edits while disconnected
 - Sync changes to the server
- **Licensing**
 - Development and testing
 - Basic
 - Standard

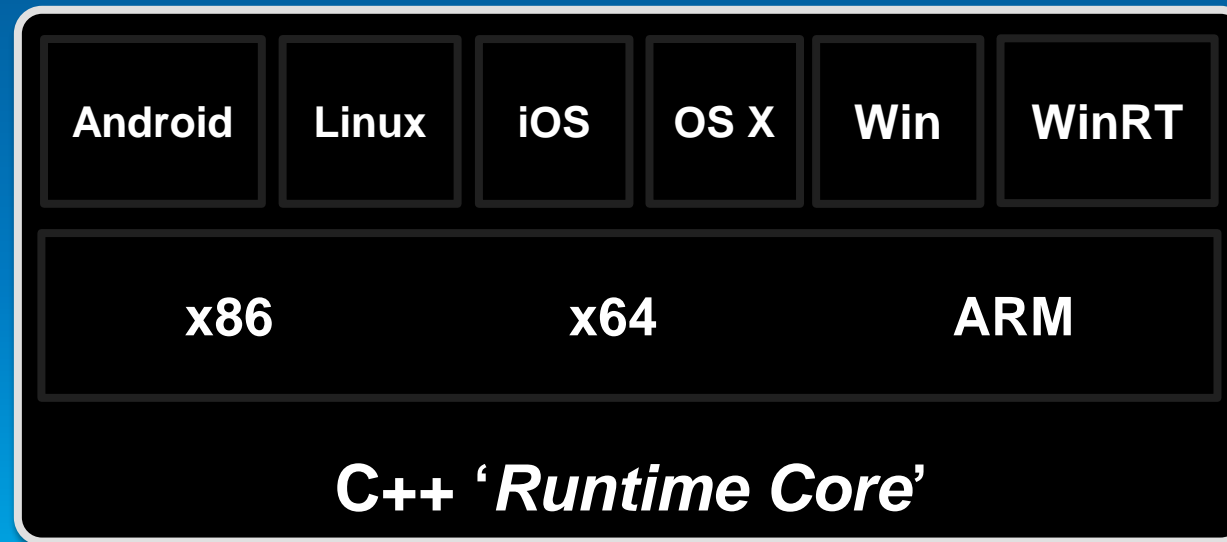
ArcGIS Runtime Overview

Thad Tilton



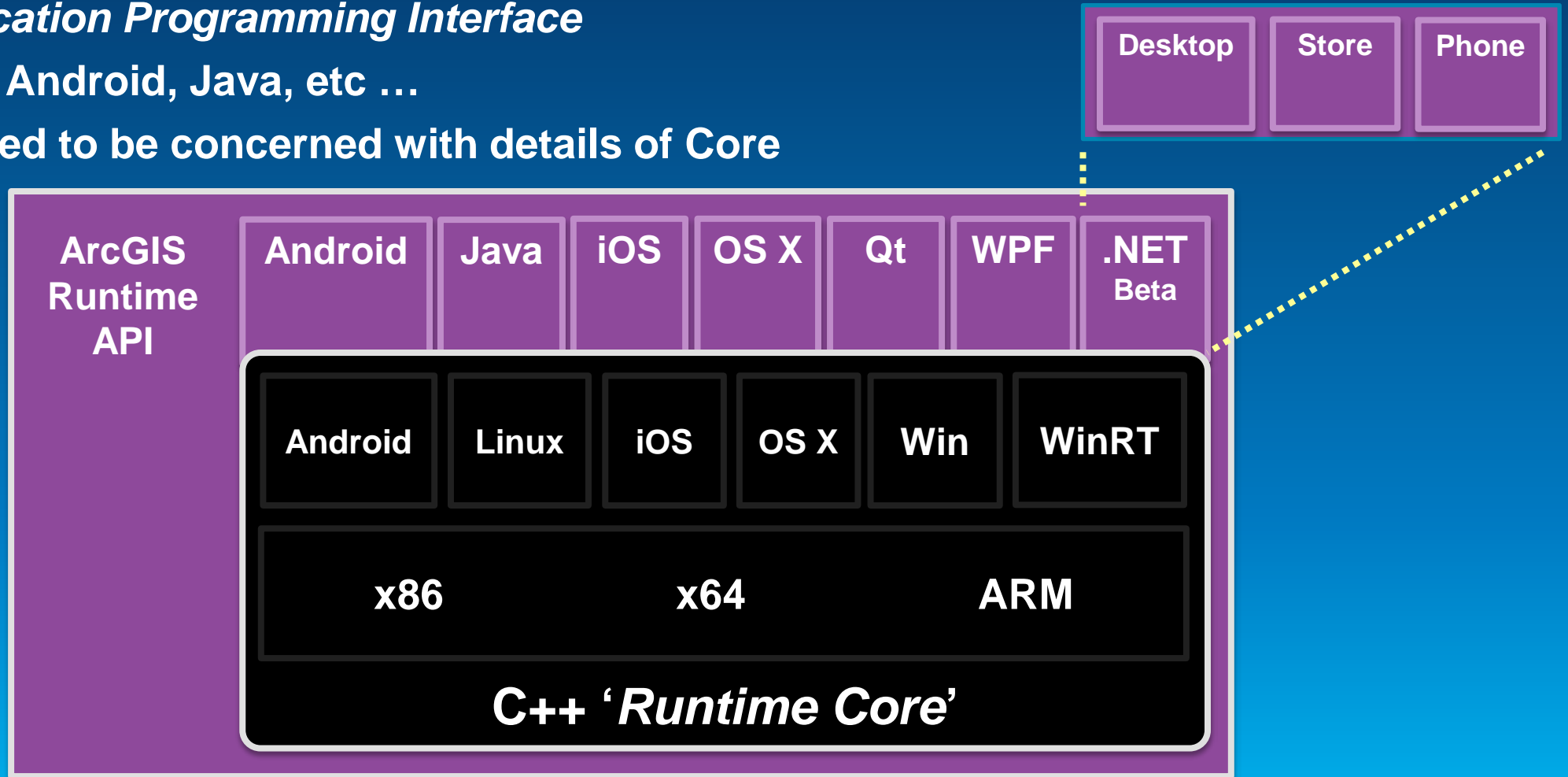
ArcGIS Runtime overview

- **Runtime Core (C++)**
 - Small footprint, high performance
 - Core functionality: Display, geometry, data access, ...
 - Compiled for multiple platforms and architectures



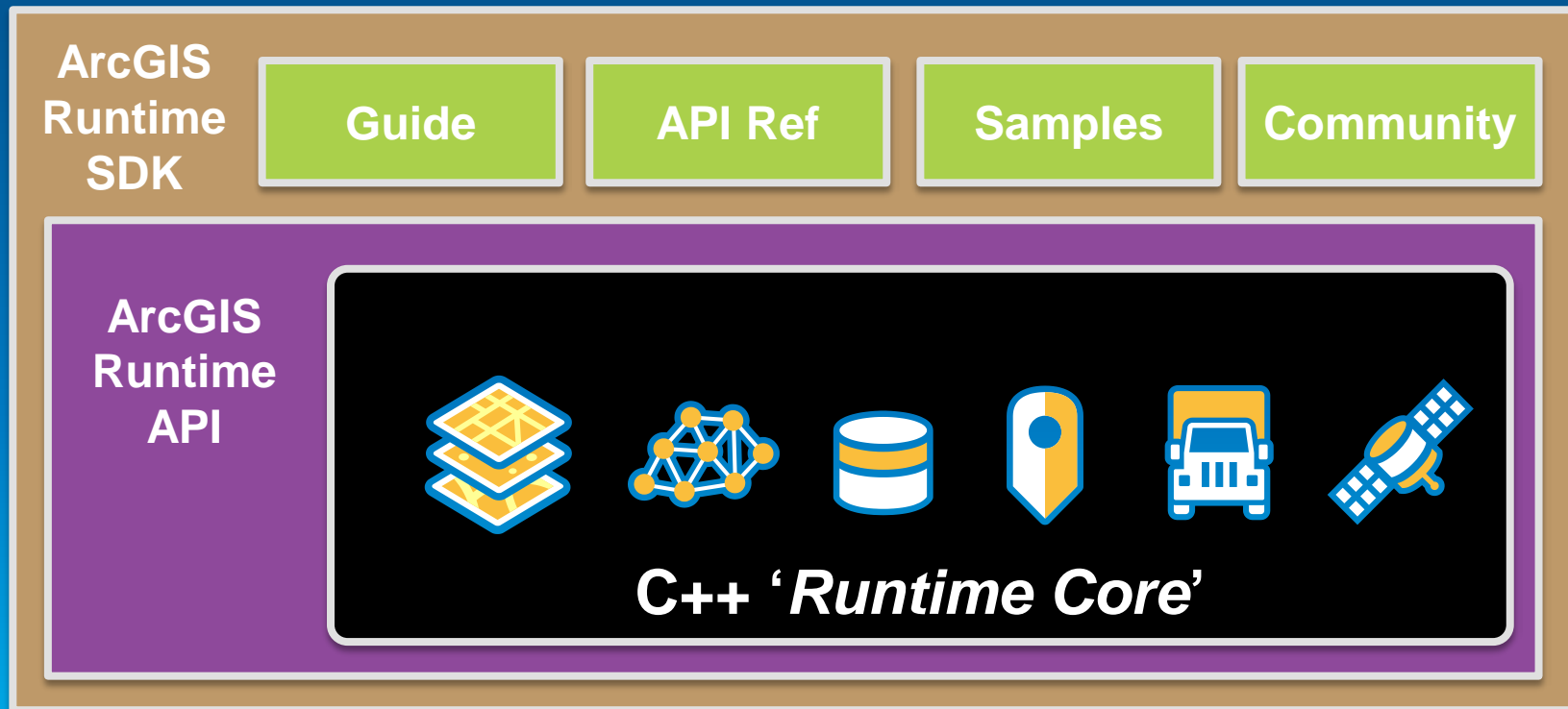
ArcGIS Runtime overview

- Access core functionality via a native API for each platform:
 - *Application Programming Interface*
 - .NET, Android, Java, etc ...
 - No need to be concerned with details of Core



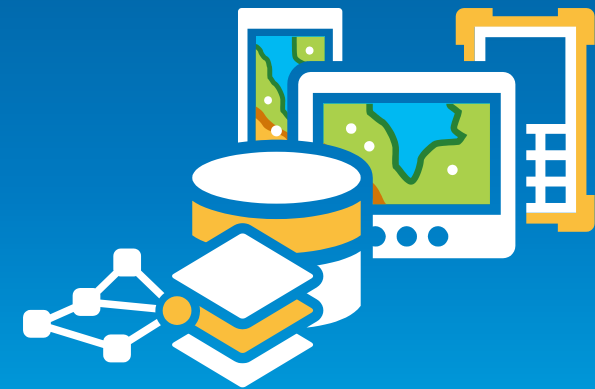
ArcGIS Runtime SDK

- **Software Development Kit** : tools for developers
 - Conceptual doc, API reference, samples, and the developer community
 - **GitHub: Samples, Toolkit, Portal Viewer app**



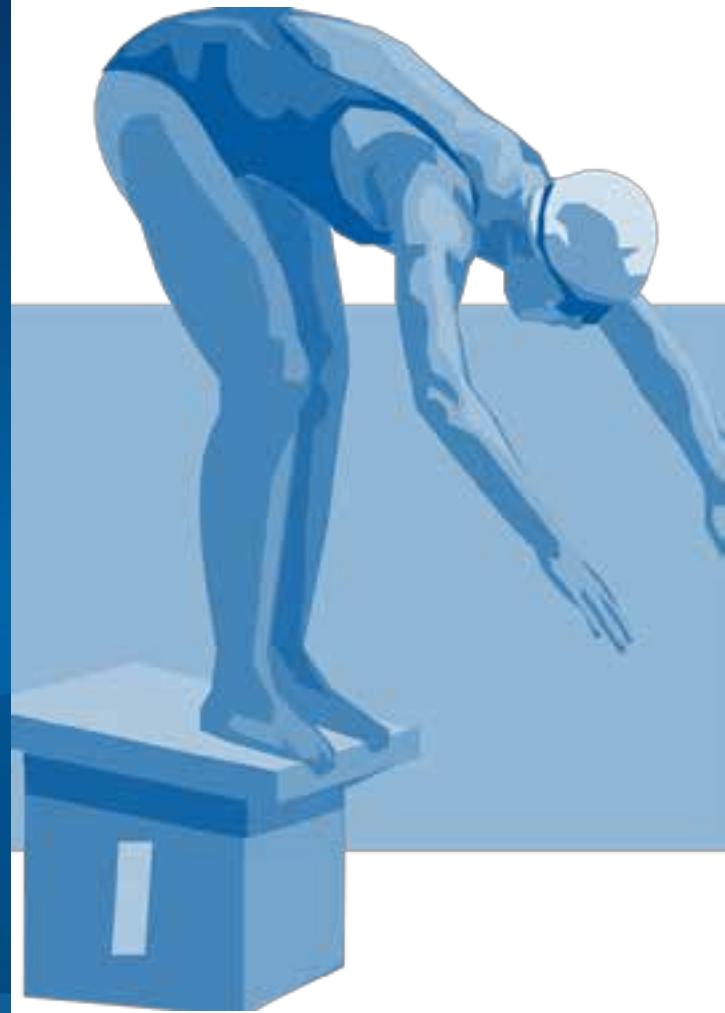
ArcGIS Runtime SDK Highlights

- Support for many platforms
- Runs natively on devices
- Built for performance
- Use data from ArcGIS for Portal, ArcGIS for Server, and ArcGIS Online
- Work offline with local basemaps and data
 - Editing and sync, Geocode, Routing
- Perform advanced geometric operations locally
- Task-based asynchronous pattern
- MVVM friendly
- Simplified licensing model



Get started

Thad Tilton



System Requirements for **Windows Desktop** development

- **Operating system**

- Windows 7
- Windows 8
- Windows 8.1



- **.Net framework**

- 4.5
- 4.5.1

- **IDE**

- Visual Studio 2013 (all editions)
- Visual Studio Express 2013 for Windows Desktop
- Visual Studio 2012 with Update 3 (all editions)
- Visual Studio Express 2012 for Windows Desktop with Update 3

For **Windows Store**

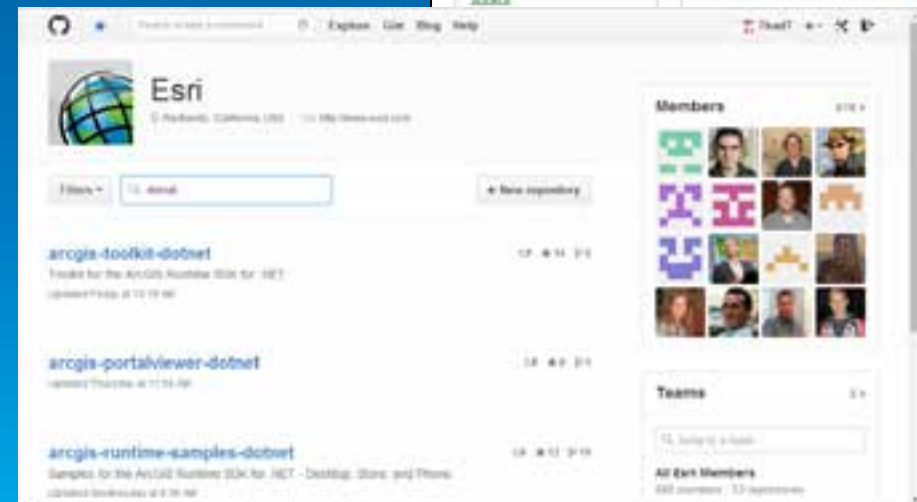
- Windows 8.1
- Visual Studio 2013 (or Express)

For **Windows Phone**

- Windows 8.1 (64 bit only)
- Windows Phone OS 8.1
- Visual Studio 2013 with Update 2 (or Express)

Where to start?

- **Developers site**
 - <https://developers.arcgis.com/net>
 - Check system requirements
- **Download and install the public beta**
 - <http://betacommunity.esri.com>
- **Code resources**
 - GitHub repos - <https://github.com/Esri>
 - Toolkit
 - Samples
- **Provide feedback**
 - Beta community (use forums, log issues)



Demo: Developer Resources

Thad Tilton



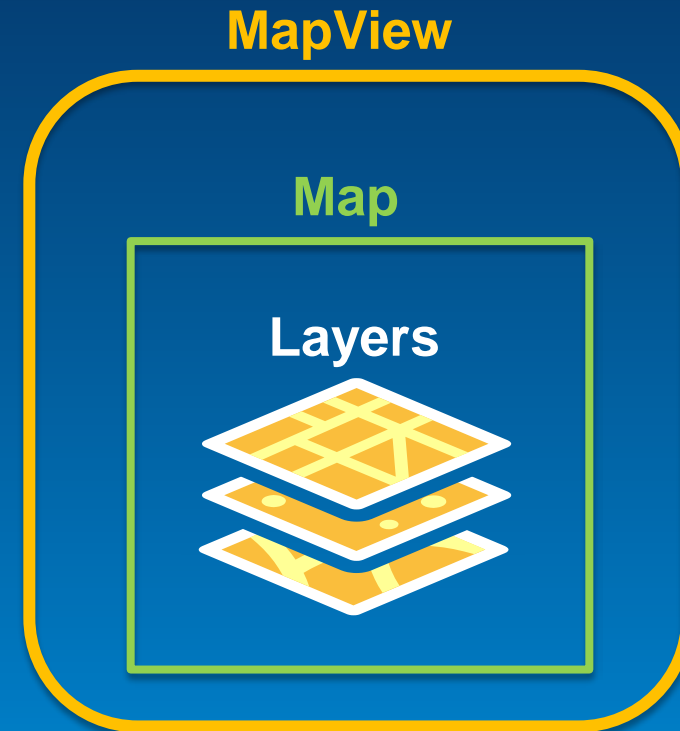
Display a map

Thad Tilton



Mapping classes

- **MapView (control)**
 - UI container for a single Map
 - Facilitates MVVM design
- **Map**
 - Container for a collection of layers
- **Layer**
 - Display geographic features
 - Various types



Define a map with XAML

```
<!-- Add XML Namespace declaration for ArcGIS Runtime -->  
xmlns:esri="http://schemas.esri.com/arcgis/runtime/2013"
```

Note: Syntax differs in Store/Phone apps

```
<!-- Use namespace prefix to access ArcGIS Runtime objects -->
```

```
<esri:MapView x:Name="MyMapView">  
  <esri:Map x:Name="MyMap">  
    <esri:ArcGISTiledMapServiceLayer  
      ServiceUri="http://services.arcgisonline.com/ArcGIS/rest/services/World_Street_Map/MapServer"  
      ID="Basemap" />  
    <esri:GraphicsLayer ID="Graphics"/>  
  </esri:Map>  
</esri:MapView>
```

Use x:Name to identify MapView or Map

Use ID to identify Layers

Create a map with code

Code behind

```
var theMap = new Esri.ArcGISRuntime.Controls.Map();

var uri = new Uri("http://services.arcgisonline.com/ArcGIS/rest/services/World_Street_Map/MapServer");
var basemap = new Esri.ArcGISRuntime.Layers.ArcGISTiledMapServiceLayer(uri);
basemap.ID = "Basemap";

var graphicsLayer = new Esri.ArcGISRuntime.Layers.GraphicsLayer();
graphicsLayer.ID = "Graphics";

theMap.Layers.Add(basemap);
theMap.Layers.Add(graphicsLayer);

this.MyMapView.Map = theMap;
```

```
<esri:MapView x:Name="MyMapView">
</esri:MapView>
```

XAML

Create a map with code (MVVM)

ViewModel

```
class MapViewModel
{
    public MapViewModel()
    {
        var map = new Map();
        var uri = new Uri("http://services.arcgisonline.com/...");
        var basemap = new ArcGISTiledMapServiceLayer(uri);
        basemap.ID = "Basemap";

        map.Layers.Add(basemap);

        ParcelMap = map;
    }

    public Map ParcelMap { get; private set; }
}
```

<esri:MapView Map="{Binding ParcelMap}">

</esri:MapView>

View

Several types of layers

- **Tiled layers**

- **ArcGISTiledMapServiceLayer**
- **ArcGISLocalTiledLayer**
- **BingLayer**
- **WebTiledLayer**
- **WmtsLayer**

- **Dynamic layers**

- **ArcGISDynamicMapServiceLayer**
- **DynamicMapServiceLayer**
- **FeatureLayer**
- **GraphicsLayer**



Tiled layers

- **Base map to provide context**
 - Data that don't change often
- **Cached for performance**
- **Find on ArcGIS Online**
 - <http://www.arcgis.com/features/maps/basemaps.html>



```
<esri:ArcGISTiledMapServiceLayer
  ServiceUri="http://services.arcgisonline.com/ArcGIS/rest/services/World_Street_Map/MapServer"
  ID="Basemap" />
```

Dynamic layers

- Used to display changing information
- Online or local data source
- Draw on top of base map layer(s)

```
<esri:FeatureLayer ID="MyFeatureLayer">  
  <esri:FeatureLayer.FeatureTable>  
    <esri:GeodatabaseFeatureServiceTable  
      ServiceUri="http://sampleserver3.arcgisonline.com/ArcGIS/rest/services/Fire/Sheep/FeatureServer/2" />  
    </esri:FeatureLayer.FeatureTable>  
  </esri:FeatureLayer>
```



Demo: Display a map

Apurva Goyal



Work with tasks

Thad Tilton



Tasks are used to perform specialized pieces of work

- **QueryTask:** use attribute or spatial criteria to find features
- **IdentifyTask:** get information about features at a location
- **FindTask:** find a text value across several layers and fields
- **LocatorTask:** geocode an address
- **RouteTask:** find the best path between points on a network
- **PrintTask:** output data to a specific device

- **Many others ...**



Tasks ...

- **Have corresponding Parameter and Result objects**
 - **FindTask – FindParameters, FindResult**
 - **IdentifyTask – IdentifyParameter, IdentifyResult**
 - **GenerateRendererTask – GenerateRendererParameter, GenerateRendererResult**
 - **QueryTask – Query, QueryResult**
- **Execute asynchronously**
 - **Keep UI responsive**
 - **Can be cancelled**
- **May have local and online versions**
 - **LocalLocatorTask / OnlineLocatorTask**

Consistent task workflow

1. Create Task object, point it to an appropriate resource (online or local)

```
var myTask = new SomeTask(new Uri("http://..."));
```

2. Define task parameters

```
var taskParams = new SomeTaskParameter();
```

3. Execute the task asynchronously

```
var taskResult = await myTask.ExecuteAsync(taskParams);
```

4. Process the result

```
myGraphicsLayer.GraphicsSource = taskResult.FeatureSet.Features;
```

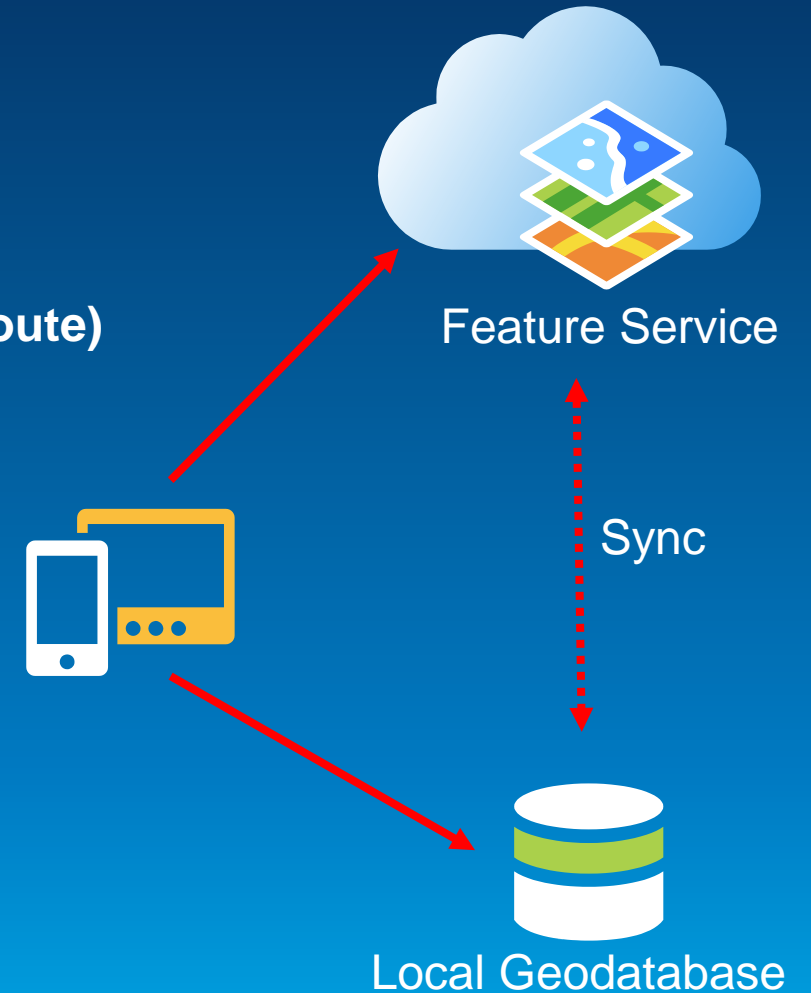
Work offline

Thad Tilton



Work with data while disconnected from the internet

- **Download online data sources for local access**
 - Tiled service as image cache
 - Feature service as geodatabase
 - Can also make functionality available offline (Geocode, Route)
- **Consistent user experience**
- **Consistent developer experience**
- **Synchronize data sources when connected**
 - Push edits made to local data
 - Pull newest version of online data



Going Offline – Tiled Map Services

- Use `ExportTileCacheTask` to take tiled map services offline
 - `GenerateTileCacheAsync` – creates a tile package (.tpk) or compact cache
 - `DownloadTileCacheAsync` – downloads a .tpk or compact cache
 - `GenerateTileCacheAndDownloadAsync` – does both in one call

```
var task = new ExportTileCacheTask(new Uri(onlineTiledLayer.ServiceUri));
var downloadResult = await task.GenerateTileCacheAndDownloadAsync(
    generateOptions,           // Options for cache generation
    downloadOptions,          // Options for cache download
    TimeSpan.FromSeconds(3),  // Frequency of status checks
    CancellationToken.None,    // Token to manage cancellation
    onGenerateUpdate,         // Callback for status updates during cache generation
    onDownloadUpdate);        // Callback for status updates during cache download
```

Going Offline – Feature Services

- Use `GeodatabaseSyncTask` to take feature services offline
 - `GenerateGeodatabaseAsync` – creates a geodatabase
 - Handle `onComplete` callback to save results when task completed
 - Check for exceptions

```
GeodatabaseSyncTask gdbTask = new GeodatabaseSyncTask(new Uri(featureServiceUrl));
var result = await gdbTask.GenerateGeodatabaseAsync(
    gdbParameters,           // Parameters for the operation
    onGenerateCompleted,    // Callback to handle operation completion
    TimeSpan.FromSeconds(3), // Interval to check status
    onGenerateProgress,     // Callback to handle operation status updates
    CancellationToken.None); // Token to handle cancellation
```

Generate geodatabase parameters

- **GenerateGeodatabaseParameters** object
 - Layers to include in the output
 - Geometry (polygon) for filtering features
 - Output spatial reference
 - Synchronization model: per layer or per geodatabase

```
var layerIDs = new List<int>{0,1,2};  
var extent = MyMapView.Extent;  
  
var gdbParameters = new GenerateGeodatabaseParameters(layerIDs, extent);  
gdbParameters.SyncModel = SyncModel.PerLayer;  
gdbParameters.OutSpatialReference = MyMapView.SpatialReference;
```

Download generated geodatabase

- Make request to `GeodatabaseStatusInfo.ResultUri`

```
private async void onGenerateCompleted(GeodatabaseStatusInfo statusInfo, Exception ex)
{
    // if unsuccessful, return
    if (ex != null) { return; }

    // read the generated geodatabase from the server
    var client = new ArcGISHttpClient();
    var gdbStream = client.GetOrPostAsync(statusInfo.ResultUri, null);

    // write geodatabase to local location
    await Task.Factory.StartNew(async () =>
    {
        using (var stream = System.IO.File.Create(_geodatabasePath))
        {
            await gdbStream.Result.Content.CopyToAsync(stream);
        }
    });
}
```

Demo: Take data offline

Apurva Goyal



Offline Editing: Editing Geometry Locally

- Use GeometryEngine static class

```
var reshapedGeometry = GeometryEngine.Reshape(geometryToBeReShaped, reshaperLine);
```

- Use Editor control

```
var editGeometry = await MyMapView.Editor.EditGeometryAsync(geometryToBeEdited);
```

```
var newGeometry = await MyMapView.Editor.RequestShapeAsync(DrawShape.Polygon);
```

GeometryEngine methods

- Area
- Buffer
- Clip
- Contains
- Crosses
- Cut
- DistanceFromGeometry
- Extend
- Generalize
- GeodesicArea
- GeodesicBuffer
- GeodesicLength
- Intersection
- Intersects
- NearestCoordinateInGeometry
- NearestVertexInGeometry
- Overlaps
- Project
- Relate
- Simplify
- Touches
- Union
- And others ...

Demo: Edit features

Apurva Goyal



Offline Editing: Synchronize changes

- Check if local data has changes

```
var gdbFeatureSvcTable = _parcelsFeatureLayer.FeatureTable as GeodatabaseFeatureTable;  
var adds = gdbFeatureSvcTable.AddedFeaturesCount;  
var updates = gdbFeatureSvcTable.UpdatedFeaturesCount;  
var deletes = gdbFeatureSvcTable.DeletedFeaturesCount;  
  
if (adds > 0 || updates > 0 || deletes > 0)  
    DoSync();
```

- Create sync task and parameters

```
var syncTask = new GeodatabaseSyncTask( new Uri("http://...") );
```

```
var syncParams = new SyncGeodatabaseParameters();  
sp.SyncDirection = SyncDirection.Bidirectional;
```

```
SyncDirection.Bidirectional  
SyncDirection.Download  
SyncDirection.Upload
```

Offline Editing: Synchronize changes

- Use `GeodatabaseSyncTask.SyncGeodatabaseAsync`
 - Push updates from the client and download changes from the service
 - Only changes (deltas) are downloaded/uploaded

```
var gdbTask = new GeodatabaseSyncTask(new Uri(featureServiceUrl));
var result = await gdbTask.SyncGeodatabaseAsync(
    syncParameters, // Operation parameters
    featureLayer.FeatureTable.Geodatabase, // Geodatabase to sync
    onSyncCompleted, // Callback for completion of sync operation
    onUploadCompleted, // Callback for changes being uploaded to server
    TimeSpan.FromSeconds(3), // Frequency of sync status checks
    onStatusUpdate, // Callback for status updates
    CancellationToken.None); // Token for handling cancellation
```

Demo: Synchronize data

Apurva Goyal



Runtime licensing

Thad Tilton



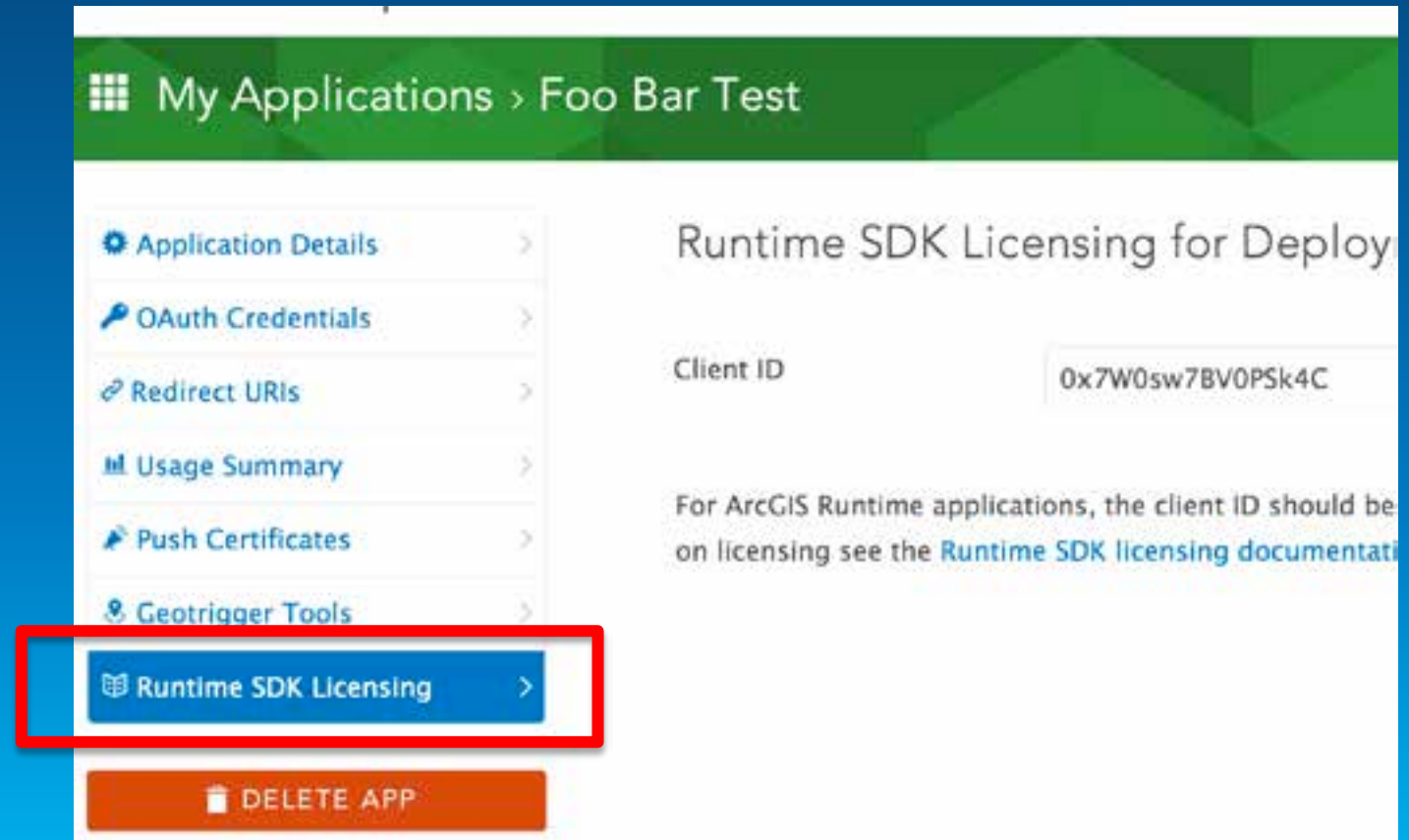
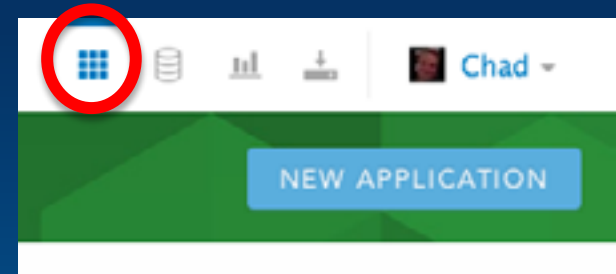
License levels and functionality

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">• Local locators (geocoding)• Local routing• Local geodatabase editing• Local geodatabase sync operations• Local server *

* For those SDKs that support it

How to license your app at the basic level

- <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 programmatically set your ClientID



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

Summary

- **Use ArcGIS Runtime SDK for .NET to develop for ...**
 - Windows Desktop, Windows Store apps, and Windows Phone
- **Key functionality**
 - Map visualization and query
 - Analysis, geocoding, routing, etc.
 - Offline capabilities
 - Editing and synchronization
- **Resources**
 - developers.arcgis.com/net – documentation, tutorials, API reference, forum
 - github.com/esri – code repositories for API toolkits, samples, full applications

ArcGIS Runtime SDK sessions Wednesday

Session Name	Time	Location
ArcGIS Runtime SDK for Qt: Tips and Tricks	9:30am – 10:00am	Developer Island (demo theatre)
Building .NET Apps with ArcGIS Runtime SDK: Tips and Tricks	11:30am – 12:00pm	Developer Island (demo theatre)
Offline Routing and Geocoding in ArcGIS Runtime SDK	3:00pm – 3:30pm	General Theater 2 (demo theatre)
Developing Windows Desktop Apps with ArcGIS Runtime SDK for .NET	8:30am – 9:45am	Room 09

ArcGIS Runtime SDK sessions Thursday

Session Name	Time	Location
Create your own Android App Tools Using ArcGIS Runtime SDKs	9:30am – 10:00am	Developer Island (demo theatre)
Dive Deep into the Performance of the ArcGIS Runtime SDKs Core Display Architecture	10:30am – 11:00am	Developer Island (demo theatre)
10 Things you Didn't Know You Can Do with ArcGIS Runtime SDK for iOS	11:30am – 12:00pm	Developer Island (demo theatre)
Animating Thousands of Graphics and Features with ArcGIS Runtime SDK for Java SE	12:30pm – 1:00pm	Developer Island (demo theatre)
Developing Mobile Apps with ArcGIS Runtime SDK for .NET	10:15am – 11:30am	Room 05 A
ArcGIS Runtime SDKs: The Road Ahead	1:30pm – 2:45pm	Room 07 A/B

- Questions?

?

有问题吗？

¿Preguntas?

?

Spørgsmål?

প্রশ্ন?

質問？

প্রশ্ন?

Mga Katanungan?

الأسئلة؟

Thank you...

- Please fill out the session survey:

Offering ID: 1347

Online – www.esri.com/ucsessionsurveys

Paper – fill out and put in drop box





Understanding our world.