Esri International User Conference | San Diego, CA

Technical Workshops | July 11th 2011

Writing and Deploying Your First Applications for ArcGIS Engine

John Hauck and Ralf Gottschalk

Agenda

- Introduction to Engine
  - Controls
  - SDK and Requirements
- Runtime Binding and Licensing
- Working with Packages
- Adding Functionality to Engine
  - Custom Components
  - Using Geoprocessing in Engine
  - GraphicTracker
- ArcGIS Runtime

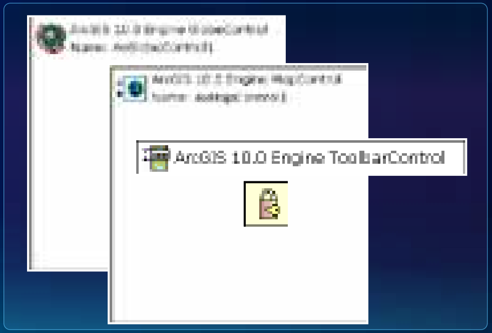
What is ArcGIS Engine?

What is ArcGIS Engine?

- Product used to build custom standalone GIS solutions
  - ArcObjects and ArcGIS Engine Controls
- Leverage the power of ArcGIS Desktop and Server
  - Use Desktop to author content
  - Consume Maps, Models, Geodatabases, Basemap Layers, Layer Packages, Query Layers, Services, and more…

What is ArcGIS Engine?

- Engine Controls
  - Components that can be added to windows forms
  - Add mapping and supporting GIS functionality to your custom application
  - 8 Controls
  - Over 200 built in commands
    - Editing, Add Data, Open Document, Select, Find, and more…


ArcGIS 10.0 Engine GlobeControl
Name: AxGlobeControl1
ArcGIS 10.0 Engine MapControl
Name: axMapControl1
ArcGIS 10.0 Engine ToolbarControl

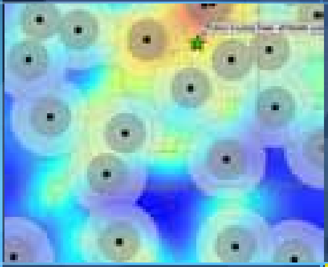
What is needed to Develop .NET Engine Applications?

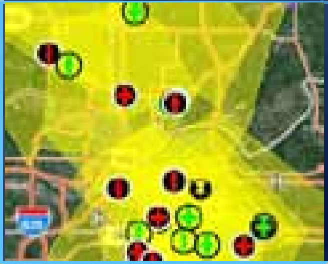
- Visual Studio 2008 sp 1
- Visual Studio 2010
- VS Express 2008
- .NET Framework - 3.5 sp 1
- ArcGIS Requirements
  - ArcGIS Engine Runtime
  - ArcObjects .NET SDK
  - ArcGIS Engine Developer Kit License

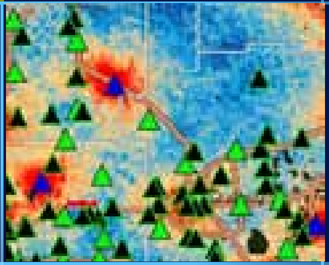
ArcGIS Engine Resources

- Visual Studio 2008 / 2010 IDE Integration
  - Tools to make development easier and faster
- ArcObjects .NET SDK
  - Walkthroughs, samples, code snippets, Object Model Diagrams, API reference
- ArcGIS Desktop Help
  - GIS and data concepts
- Resource Centers, Blogs, and Forums
  - http://resources.arcgis.com/
- Support Center
  - Technical Articles, white papers, downloads

ArcGIS Engine Controls and SDK Tools







Runtime Binding and Licensing

Runtime Binding

What is Runtime Binding?

- At ArcGIS 10 each product has it’s own runtime
  - Products have separate install locations
  - Service pack products separately
  - Uninstall service packs
- ArcObjects must be pointed to a runtime to work
  - Before any other ArcObjects calls
  - Required for all standalone applications
  - Engine applications can bind to either Desktop or Engine Runtimes

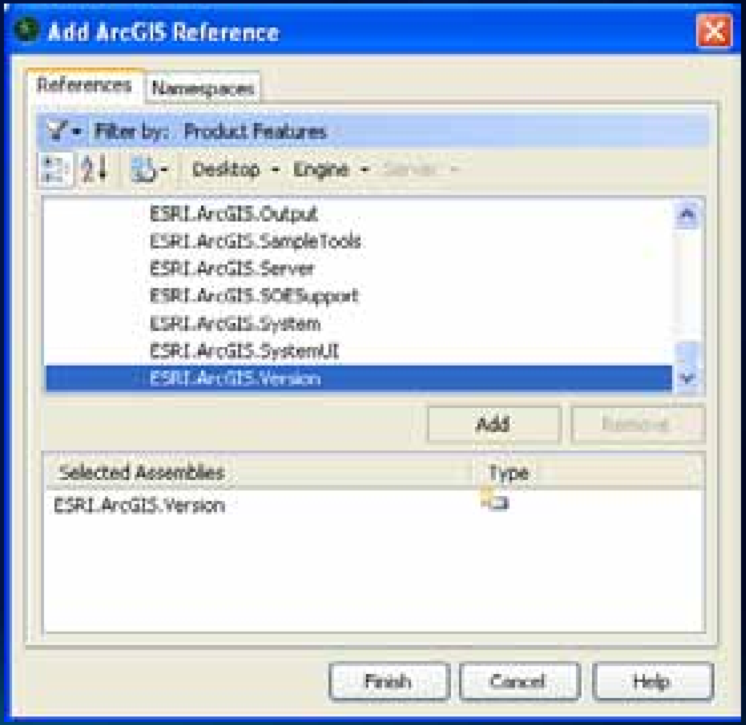
Runtime Binding

How to bind to a runtime

- Bind using the **RuntimeManager** static class

- Add reference to:

  ESRI.ArcGIS.Version



Filter by: Product Features

ESRI.ArcGIS.Output
ESRI.ArcGIS.SampleTools
ESRI.ArcGIS.Server
ESRI.ArcGIS.SOESupport
ESRI.ArcGIS.System
ESRI.ArcGIS.SystemUI
ESRI.ArcGIS.Version

Selected Assemblies	Type
ESRI.ArcGIS.Version	

Runtime Binding

How to bind to a runtime

- Add the following code, before any other ArcObjects calls:

```
ESRI.ArcGIS.RuntimeManager.Bind(ESRI.ArcGIS.ProductCode.Engine);
```

- Preferably in the main method or in the application events
- Tip:
  - The Assembly is called ESRI.ArcGIS.Version
  - The Namespace is ESRI.ArcGIS
  - Bind method returns a Boolean that you can use to handle binding errors

Runtime Binding

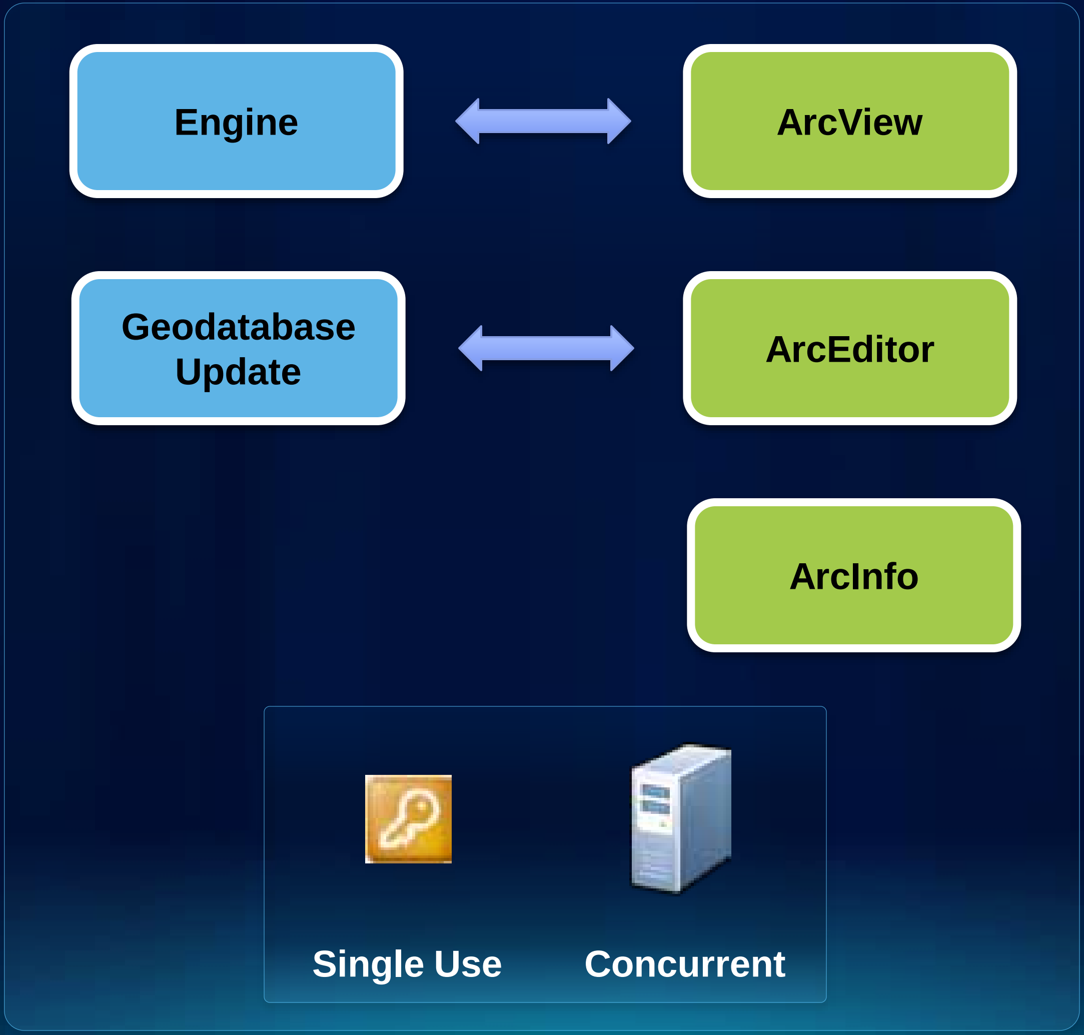
ESRI.ArcGIS.Version Additional Functionality and Uses

- RuntimeManager.BindLicense
  - Bind and License with one method
- RuntimeCollection
  - Identify Installed Runtimes
- RuntimeInfo
  - Identify the Path, Product, & Version
- ActiveRuntime
  - Currently bound runtime

Licensing

- Engine applications must check out a license at runtime
  - Either Desktop or Engine License
  - Product Licenses are checked out for the life of the application
  - Extensions can be checked out and returned as needed
- At ArcGIS 10 Engine concurrent Engine licenses are available
- Licensing is not the same as binding
  - Binding specifies the runtime
  - Licensing specifies the product functionality

Licensing



Engine

ArcView

Geodatabase Update

ArcEditor

ArcInfo

Single Use

Concurrent

Binding and Licensing

Binding and Deployment

- ArcGIS Engine is not required on the target machine
  - Engine applications work with either a Desktop or Engine Runtime
- Can mix and match Licensing and runtimes for flexible Engine applications
  - Leverage existing Desktop runtime and licenses on client’s machine

Additional Tips

64 Bit Support

- ArcGIS Engine is a 32 bit application
  - Run as a 32 bit applications on a 64 Bit OS
  - Set platform to x86 in Visual Studio Configuration Manager
    - Default is “Any CPU”
- At Version 10 ArcGIS applications are Large Address Aware
  - On 64 Bit OS 32 bit processes can take up to 4 gigabytes of RAM if available
- When compiling VS 2010 applications on a x64 machine
  - Follow KB 37879

UAC and Engine Applications

- User Account Control (UAC) on Windows
  - UAC restricts access certain parts of the system
    - Program Files directory
    - Parts of the registry
- Can be changed in the application Manifest
  - Enables application to be run as an administrator

Disabling Windows Aero Themes

- Esri recommends disabling Windows Aero themes in Engine Applications
- KB 38465
- Use P/Invoke
  - Load the Desktop Windows library if it exists
    - Remember XP does not support Aero
  - Disable Aero for the life of the application

Authoring Content for Engine

Working with Packages

What are packages?

- Packages are a single file that contains a map or layer[s] and supporting data
  - Also can contain references to SDE data
- Easy to share
  - Single file
- ArcGIS 10 supports
  - Layer Packages (.lpk)
  - Map Packages (.mpk)

Working with Packages

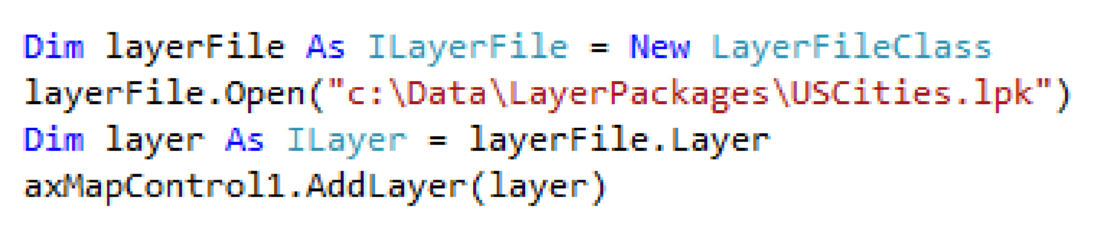
Using Packages in Engine

- At Engine 10 sp 1
  - Programmatically consume packages
- IMapDocument.Open
  - Map Packages
  - Layer Packages
  - Web Maps
- ILayerFile.Open
  - Layer Packages
- Just point to the path of the package

Working with Packages

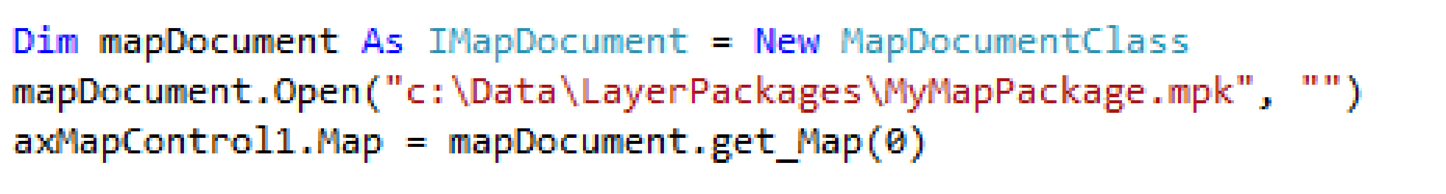
Using Packages in Engine

- Opening Layer Packages



```
Dim layerFile As ILayerFile = New LayerFileClass
layerFile.Open("c:\Data\LayerPackages\USCities.lpk")
Dim layer As ILayer = layerFile.Layer
axMapControl1.AddLayer(layer)
```

- Opening Map Packages



```
Dim mapDocument As IMapDocument = New MapDocumentClass
mapDocument.Open("c:\Data\LayerPackages\MyMapPackage.mpk", "")
axMapControl1.Map = mapDocument.get_Map(0)
```

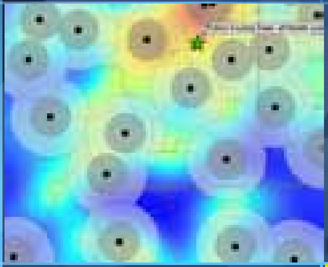
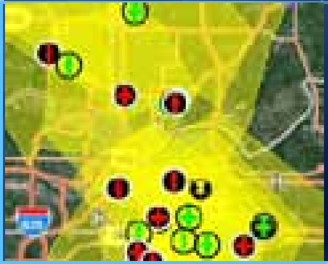
Working with Packages

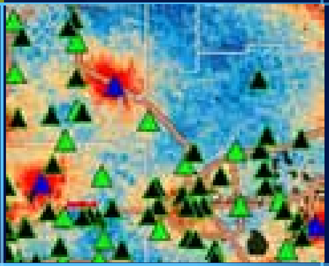
Using Online Content

- Consume data on ArcGIS Online
  - Pass in a URL with the id as the filename
- http://www.arcgis.com/home/item.html?id=224ee2a012154bbf84bcc5b04ea35fb5
- URL to ArcGIS Online Data
  - Point to the item.pkinfo file online

http://www.arcgis.com/sharing/content/items/224ee2a012154bbf84bcc5b04ea35fb5/item.pkinfo

Working with Packages



Working with Packages

Why use Packages in Engine?

- Provides a mechanism to easily deploy maps and data with your Engine solution
  - Simple deployment single file
  - Easy to update data off cycle
- Data can be uploaded to ArcGIS Online
  - Once a packages is downloaded it can be used locally
  - Use ArcGIS Online groups to manage access to data

Adding Functionality to Engine

Adding Functionality to Engine

- Leverage the existing commands and tools included in the Engine SDK
  - On a Toolbar Control
  - Programmatically
- Build your own components
  - Such as Commands, Tools, Extensions, Custom Layers, etc…
  - Specific to your Engine application
  - Generic for all Engine applications
  - Work in both Engine and Desktop applications

Adding Functionality to Engine

Using Geoprocessing in Engine

- Why use geoprocessing in Engine?
  - Why reinvent the wheel?
  - ArcGIS comes with hundreds of Geoprocessing tools
  - Developed by specialists in their field

- Provides a framework to author Model and Script tools in ArcGIS Desktop
  - Consume these tools in Engine

Using Geoprocessing in Engine

- Running system tools

```
Imports ESRI.ArcGIS.Geoprocessor
‘System Toolboxes have their own Assembly
Imports ESRI.ArcGIS.AnalysisTools

Dim gp As Geoprocessor = New Geoprocessor

‘Create the clip tool
Dim clipTool As Clip = New Clip
clipTool.in_features = “C:\Data\Test.gdb\InFeatures”
clipTool.clip_features = “C:\Data\Test.gdb\ClipFeatures”
clipTool.out_feature_class = “C:\Data\Test.gdb\ResultFeatures”

‘Execute the Tool
gp.Execute(clipTool, Nothing)
```

Using Geoprocessing in Engine

- Running custom tools

```
Imports ESRI.ArcGIS.Geoprocessor

Dim gp As Geoprocessor = New Geoprocessor

‘Add the toolbox
gp.AddToolbox(“C:\Data\MyToolbox.tbx”)

‘Populate the paramenters
Dim parameters As IVariantArray = New VarArray
parameters.Add(“C:\Data\Test.gdb\InFeatures”)
parameters.Add(“C:\Data\Test.gdb\ProcessFeatures”)
parameters.Add(“C:\Data\Test.gdb\ResultsFeatures”)

‘Execute the Tool
gp.Execute(“MyTool”, parameters, Nothing)
```

Background Geoprocessing

What is Background Geoprocessing

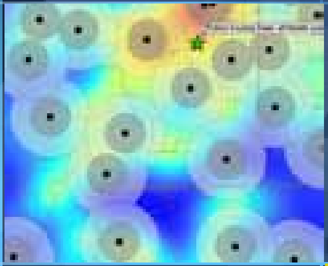
- Framework to allow Geoprocessing tools to execute in a separate processes
  - Great alternative to multi-threaded application
- Allows User Interface to remain responsive while processing

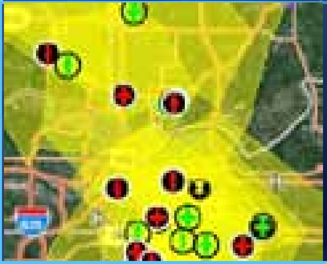
Background Geoprocessing

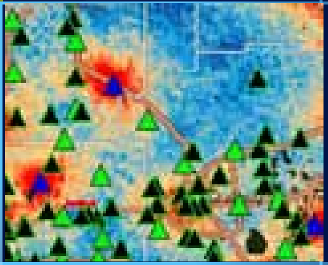
What is Background Geoprocessing

- Execute tool using ExecuteAsync method on the Geoprocessor object
- Wire in the events
  - ToolExecuted
    - Must handle to know when tools completes
  - ProgressChanged
  - MessagesCreated
  - ToolExecuting
- Run system tools, models, and script tools.

Using Geoprocessing in Engine







Geoprocessing

Tips for working with the Geoprocessor

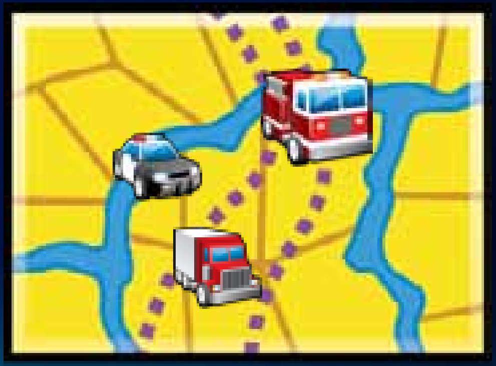
- Tips:
  - Set OverwriteOutput = True
  - Tools require different license levels or extensions
  - Desktop help is your friend
    - Provides extensive documentation on how to run and interpret the results of geoprocessing tools
  - Understand GP Messaging
  - Learn about the Result Object

GraphicTracker

GraphicTracker

What is the GraphicTracker?

- Simple API
  - Add, remove, update, and move graphics
  - Pass in a geometry and symbol
    - Works with Points, Lines, and Polygons
- Same API for Map, Globe, and Dynamic Display
  - GraphicTracker manages the display
- All objects passed in ByValue
  - Objects managed by the GraphicTracker

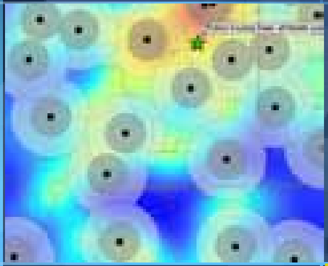


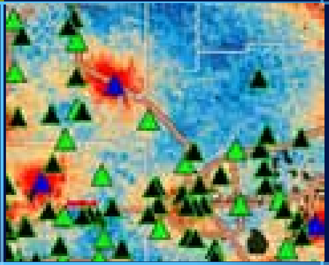
GraphicTracker

Tips when using the GraphicTracker

- Pause with IGraphicTracker.SuspendUpdate
  - For adding groups of items
- IGraphicTracker.Add method returns an integer to reference the graphic
  - Store this integer into a table for easy reference to use later
    - GraphicTracker Ids may not be sequential
- Use multiple GraphicTrackers
  - Separate GraphicTrackers for points, lines, and polygons

GraphicTracker Demo





GraphicTracker

Performance Considerations

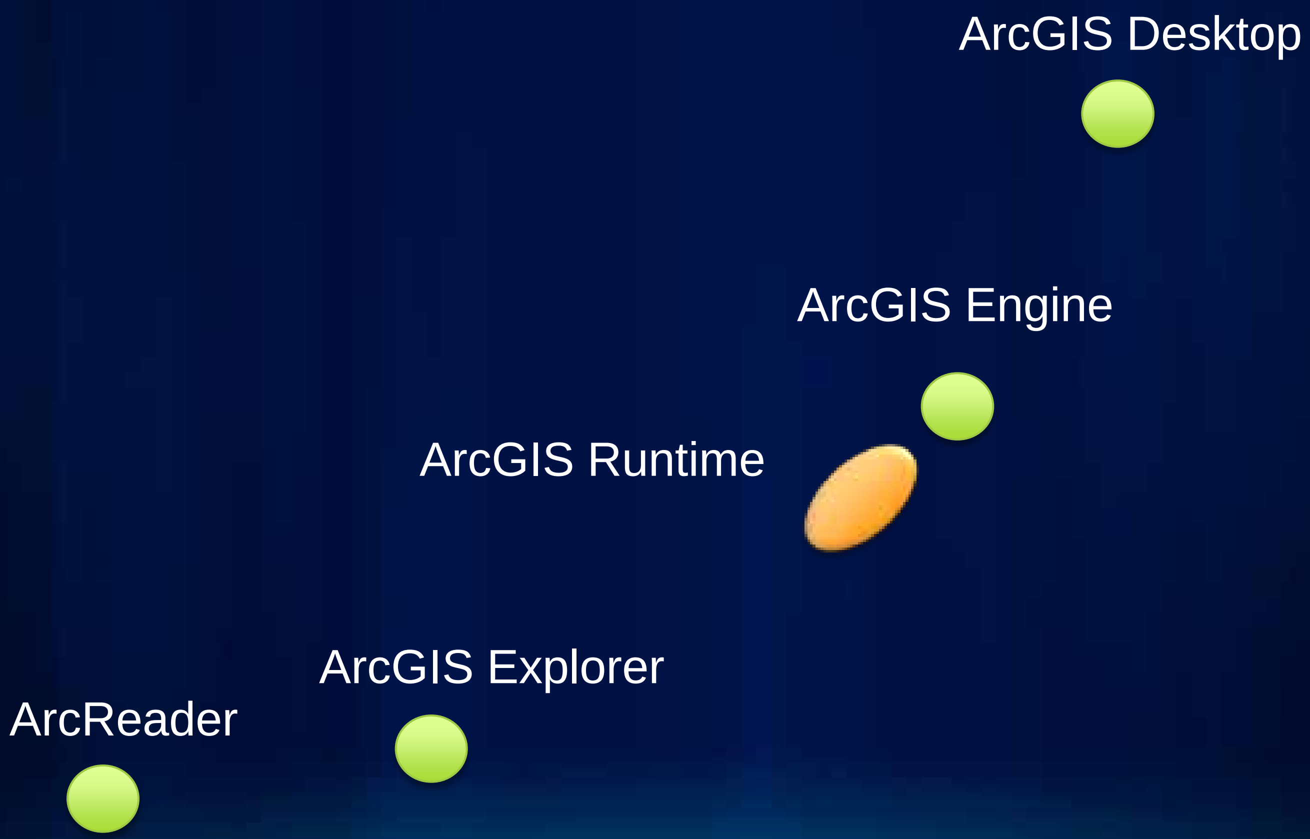
- How many graphics does the GraphicTracker support?
  - Number of graphics
  - Complexity of the graphics and symbols
  - Complexity of your map
  - Update interval for moving graphics
  - Using labels

ArcGIS Runtime

ArcGIS Runtime for Windows and Linux

- New at 10.1 – Lightweight GIS Developer Solution
- XCopy deployable
- Program against it using WPF, Java, and C++ Qt APIs
- Not a replacement for Engine
  - Maybe be an option depending on your workflow
  - Certain workflows still will only be possible in Engine

ArcGIS Runtime, how does it relate to Engine?



ArcGIS Desktop

ArcGIS Engine

ArcGIS Runtime

ArcGIS Explorer

ArcReader

Questions?

- Please don’t forget to fill out online surveys
- www.esri.com/sessionevals

esri