



# Programming Custom Tasks for ArcGIS Explorer

*Shelly Gill*  
*Rob Dunfey*

# Outline

- **What is ArcGIS Explorer**
- **Programming custom tasks**
  - Objects in the Explorer API
  - Software developer kit
- **Task Framework**
  - Creating custom tasks
  - Task conceptual steps and lifecycle
- **Typical customizations and issues**
  - Calling web services
  - Task deployment/security
- **Tips**

# What is ArcGIS Explorer?

- Free to deploy, lightweight, easy to use 3D desktop application
- A client for ArcGIS Server
- Easy way to deliver access to GIS content and capabilities, integrate, and utilize GIS services, geographic content, and other web services .

***More than an exploration tool,  
it's a way to deliver and  
publish ArcGIS capabilities  
to your users***



# ArcGIS Server 9.2 Software Development Kits

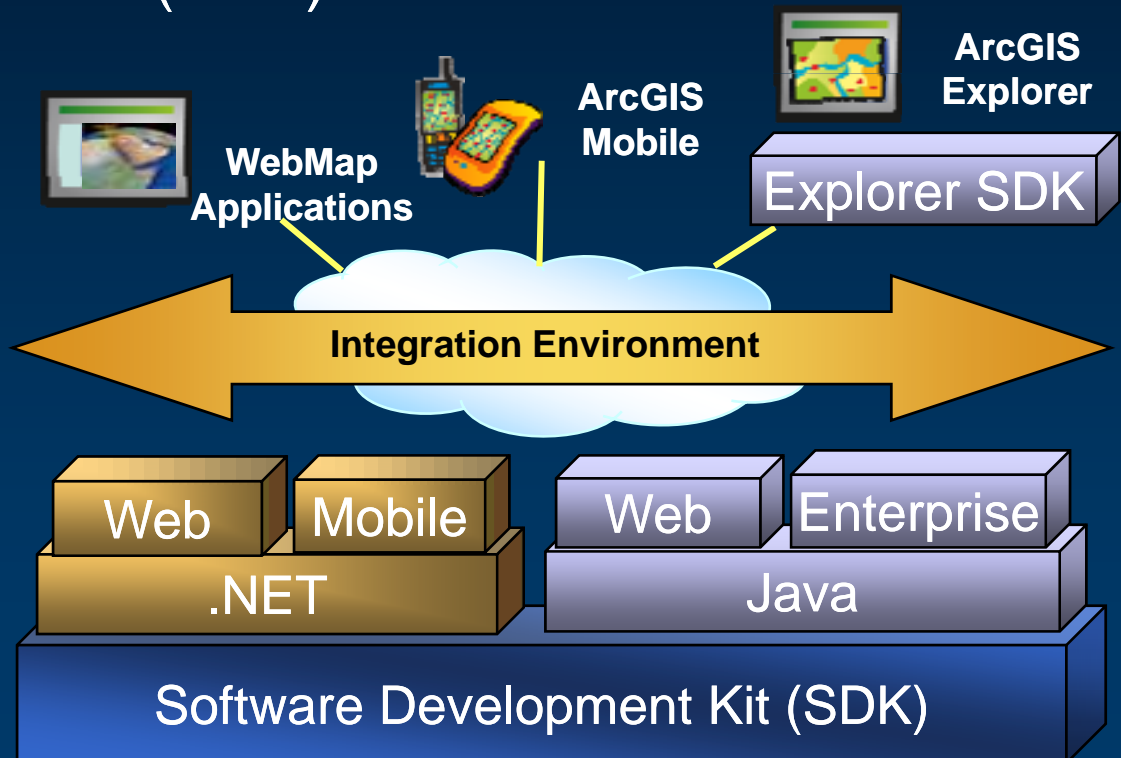
- Build web & enterprise geospatial applications and services
- Productivity boost with out-of-the-box IDE integration
- Software Development Kit (SDK) includes :

- .NET components

- Web ADF
- Mobile ADF
- ArcGIS Explorer SDK

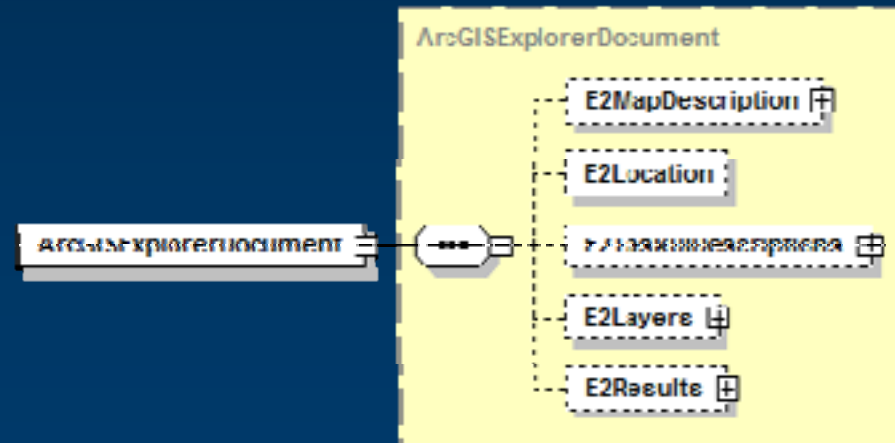
- Java components

- Web ADF
- Enterprise ADF



# NMF Files

- NMFs are how you share maps
  - Put on website or fileshare
- Can contain
  - Maps and settings
  - Layers
  - Results
  - Tasks
- XML
  - Published schema
  - Allows programmatic creation, editing





# NMF Files

*demonstration*

*What's in an NMF for developers?*

# Configuration

vs

# Customization

- **NMF**
  - Data
  - Effects (fog, clouds)
  - Tasks
  - Results
- **Add geoprocessing services as tasks**
- **Home Server**
  - Application behaviours
  - Skin – color scheme, logo, north arrow

- **Programming custom tasks against the .NET API**

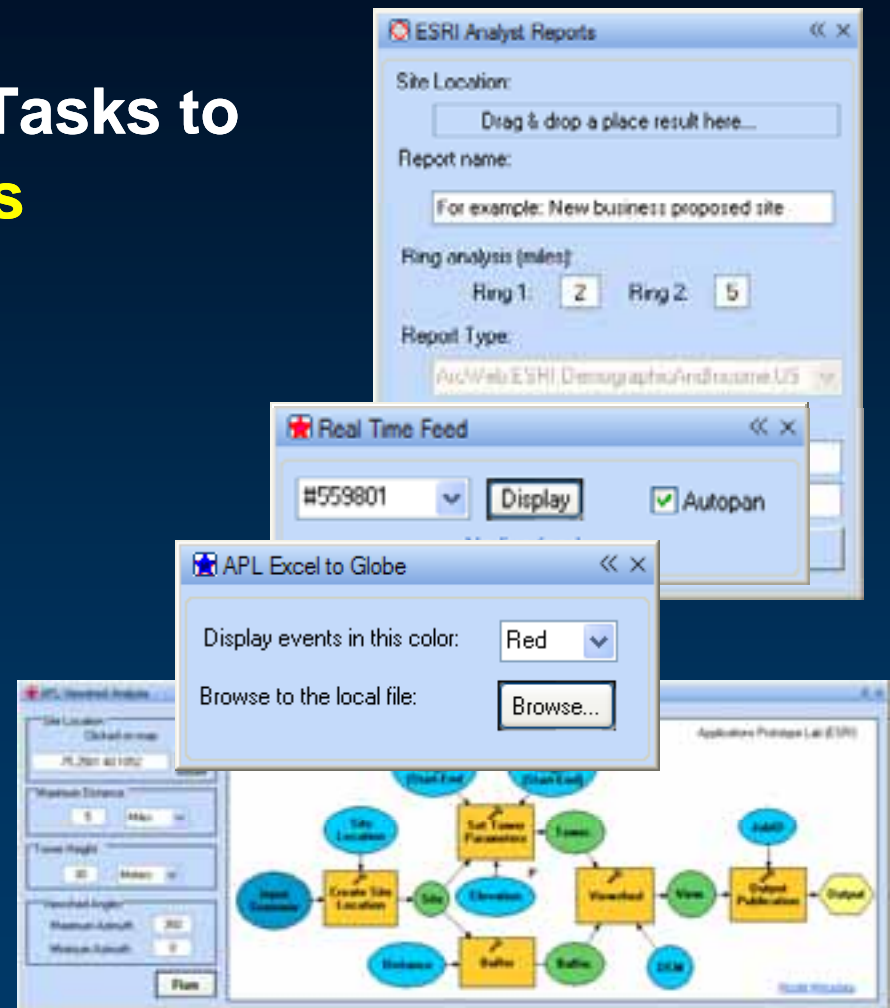
Discussed in "Getting the Most Out of the ArcGIS Explorer (Best Practices) " technical session.

# Agenda, next...

- **What is ArcGIS Explorer**
- **Programming custom tasks**
  - Objects in the Explorer API
  - Software developer kit
- **Task Framework**
- **Typical customizations and issues**
- **Tips**

# When do you create tasks?

- Developers Create Custom Tasks to **connect to new web services**
- Can also
  - Automate existing tasks
  - Refine existing user interfaces
  - Perform local operations



# Defining and designing tasks

- **Tasks perform a function, by**
  - Asking the question
  - Displaying the answer (results window)
- **Tasks are lightweight**
  - Less code on the client
  - Use servers to get their job done

# ArcGIS Explorer Object Model

- Concise
- Application classes
  - What you can do to control / interact with the application
- Task Framework
  - How you incorporate your tasks into the Application
- Task framework - **only** customization point in Explorer

# ArcGIS Explorer API

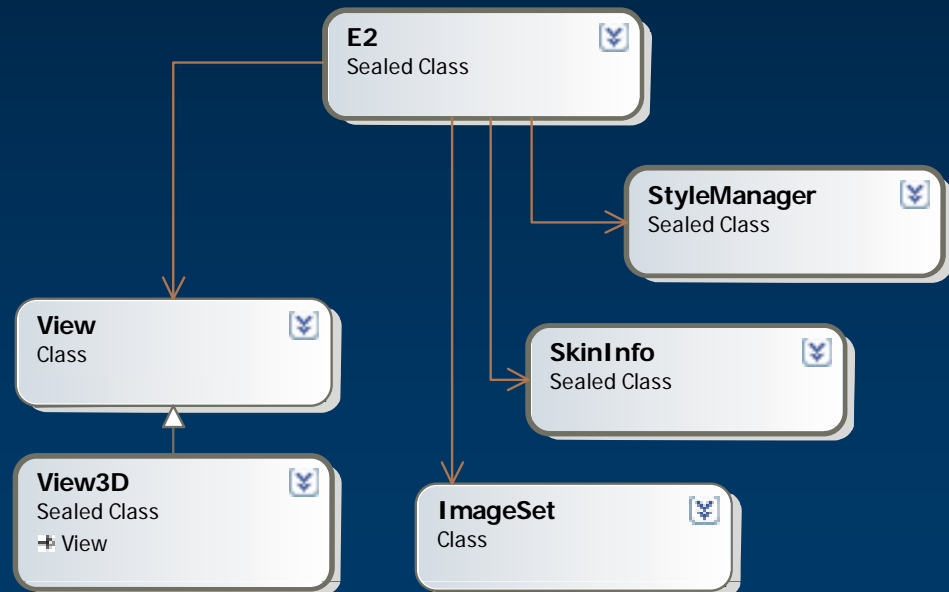
- **Object model exposed in managed API layer**
- **ESRI.ArcGIS.E2API namespace/assembly**
  - Not GAC'd (allows non-admin install)
- **Managed task loading framework**
- **Each task in own AppDomain**
  - All AppDomains unloaded on document close
  - Implications for multiple-task assemblies
- **Task framework is multi-threaded**
  - Many tasks executing simultaneously
  - Responsive user interface
  - Framework design avoids third party thread programming

# E2API Classes

- **Application and View classes**
- **Layer classes**
- **Task framework classes**
- **Result classes**
- **Geometry classes**
- **Feature classes**

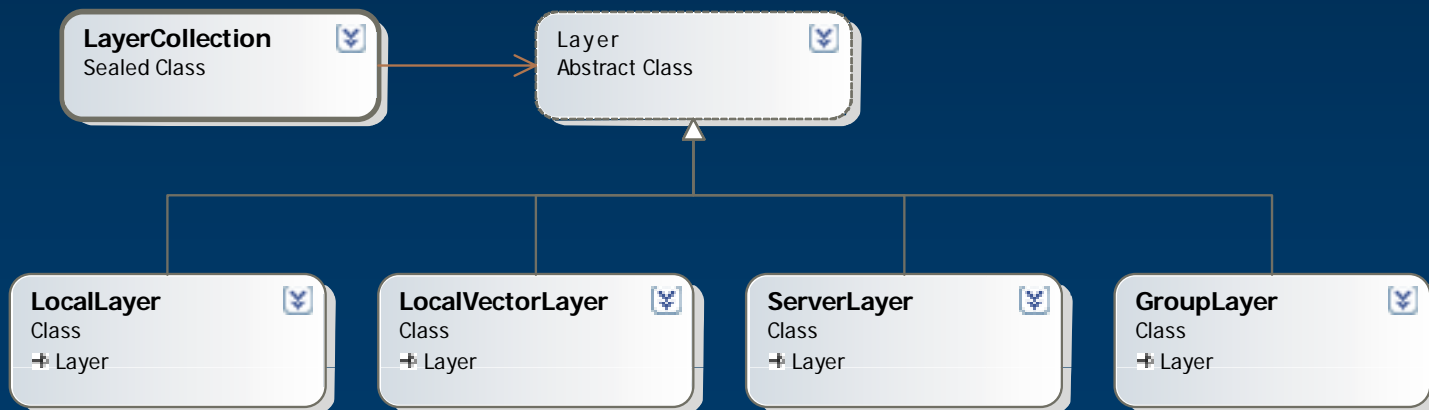
# E2API application classes

- E2 – the application
- StyleManager – style files, symbol names
- SkinInfo – skin colors / fonts
- ImageSet - icons
- View - superclass
- View3D – the 3D globe



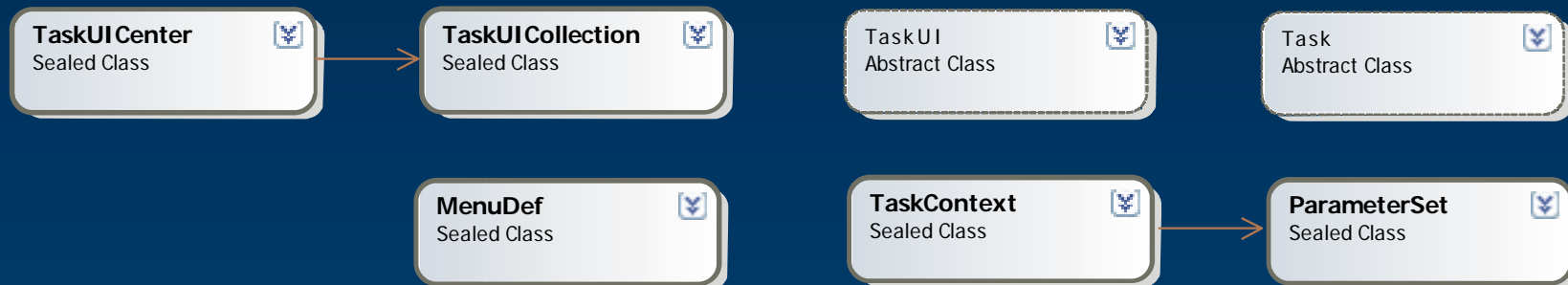
# E2API Layer classes

- **LayerCollection**
- **Layer** - superclass, abstract
- **LocalLayer** - raster, KML
- **LocalVectorLayer** - shapefile, FGDB
- **ServerLayer** - globe / map service, IMS, WMS
- **GroupLayer** - contain any layers



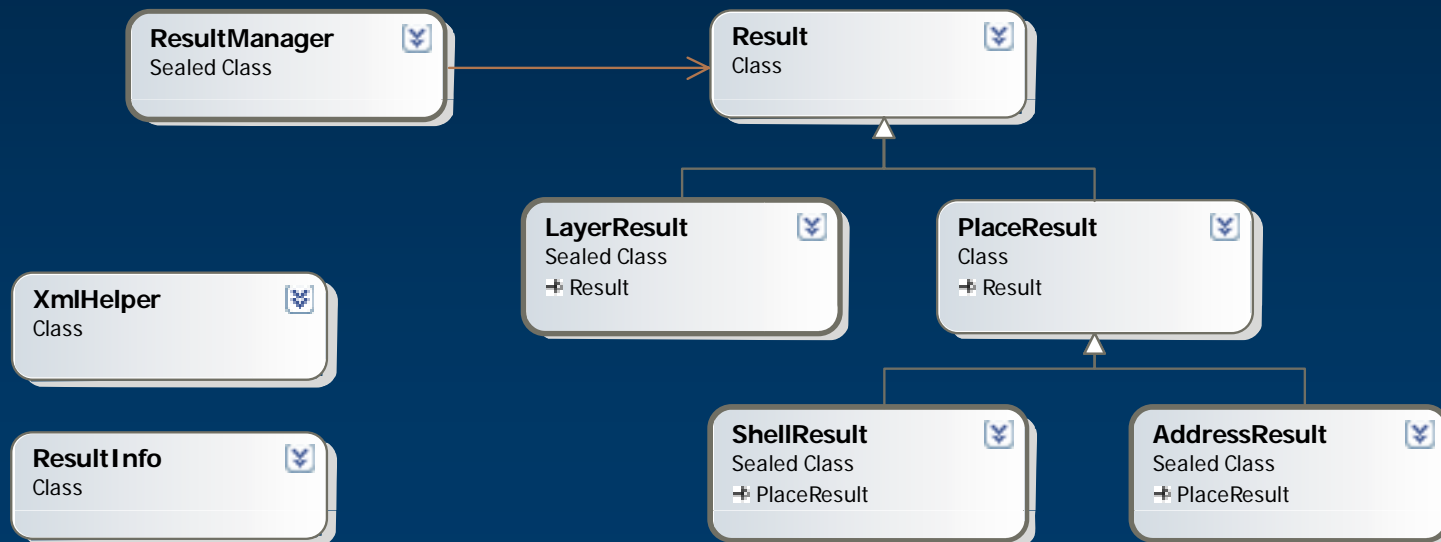
# E2API Task framework classes

- **TaskManager, TaskUICollection** - access to tasks
- **TaskUI, Task** - abstract
- **TaskContext** - holds info about a 'run' of a task
- **ParameterSet** - pass data between Task and TaskUI
- **MenuDef** - define menus



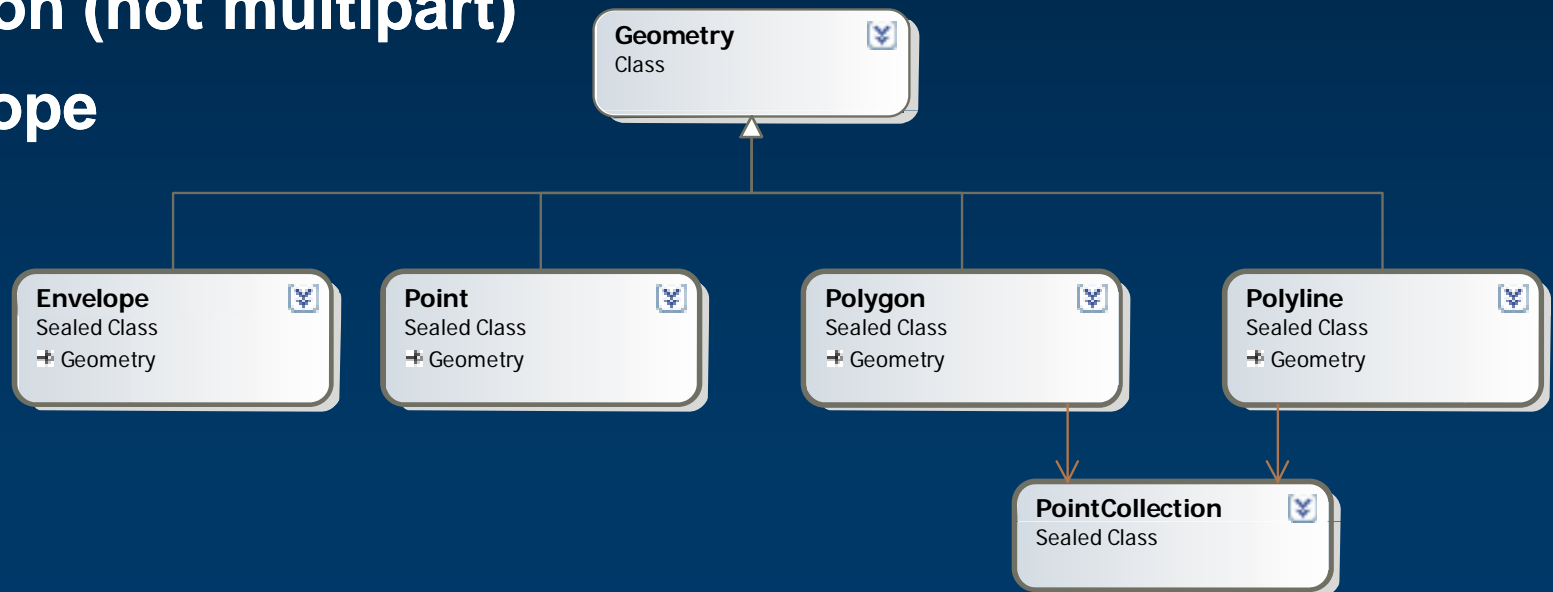
# E2API Results classes

- **ResultManager**
- **Result** - superclass
- **PlaceResult** - location
- **AddressResult** - address
- **LayerResult** - link to layer
- **ShellResult** - ShellExecute
- **ResultInfo** - used in menus
- **XmlHelper** - drag and drop



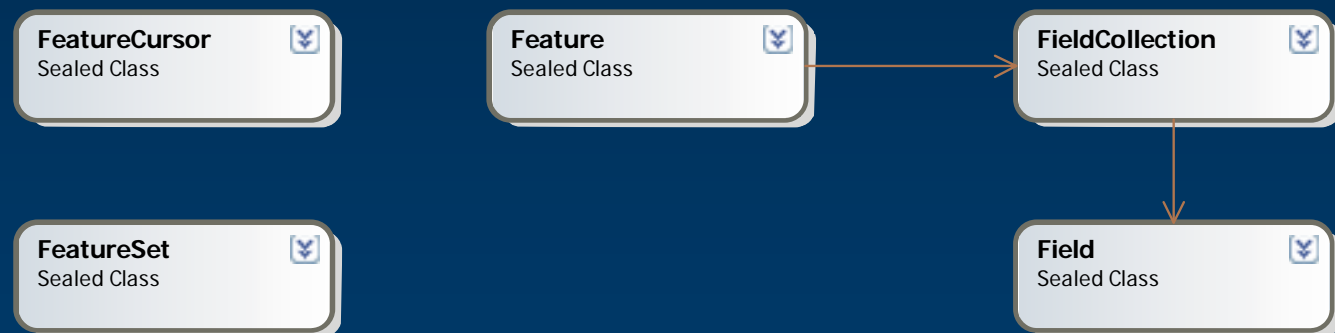
# E2API Geometry classes

- **Geometry** - superclass (+ used for unsupported types)
- **PointCollection** - not a geometry, holds poly vertices
- **Point**
- **Polyline** (not multipart)
- **Polygon** (not multipart)
- **Envelope**



# E2API Feature classes

- Note - applicable only to local vector layers
- FeatureSet - from QueryFeatures - all features
- FeatureCursor - from SearchFeatures - one at a time
- FieldCollection
- Field
- Feature

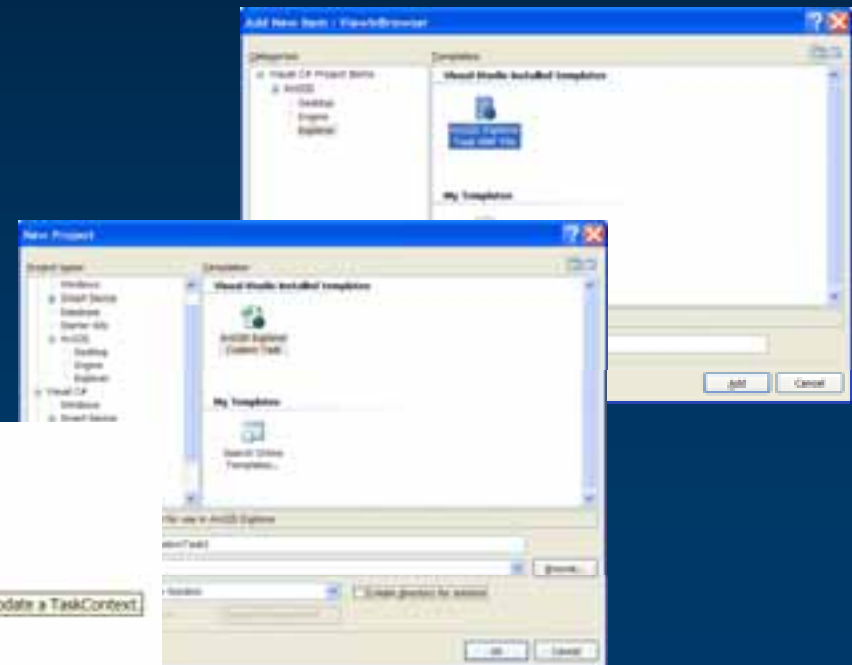


# The Software Developer Kit

- Free SDK
  - VS including Express editions
- Resources to create custom tasks
  - Conceptual documentation
  - Component help
  - Object model diagram
  - Visual Studio 2005 templates

```
class ViewInBrowserTask : Task
{
    public override void Execute(TaskContext taskContext)
    {
        Insert Snippet: <code>InsertSnippet</code>
        PARAMETERSET: PARAMS
        string url = param
        Process.Start(url);
    }
}
```

Insert Snippet: <code>InsertSnippet</code>	Create a ParameterSet	ParameterSet()
	Create a PlaceResult	PlaceResult()
	Create a ShellResult	ShellResult()
	Iterate the layers in the CurrentView	IterateLayers()





# The ArcGIS Explorer SDK for .NET

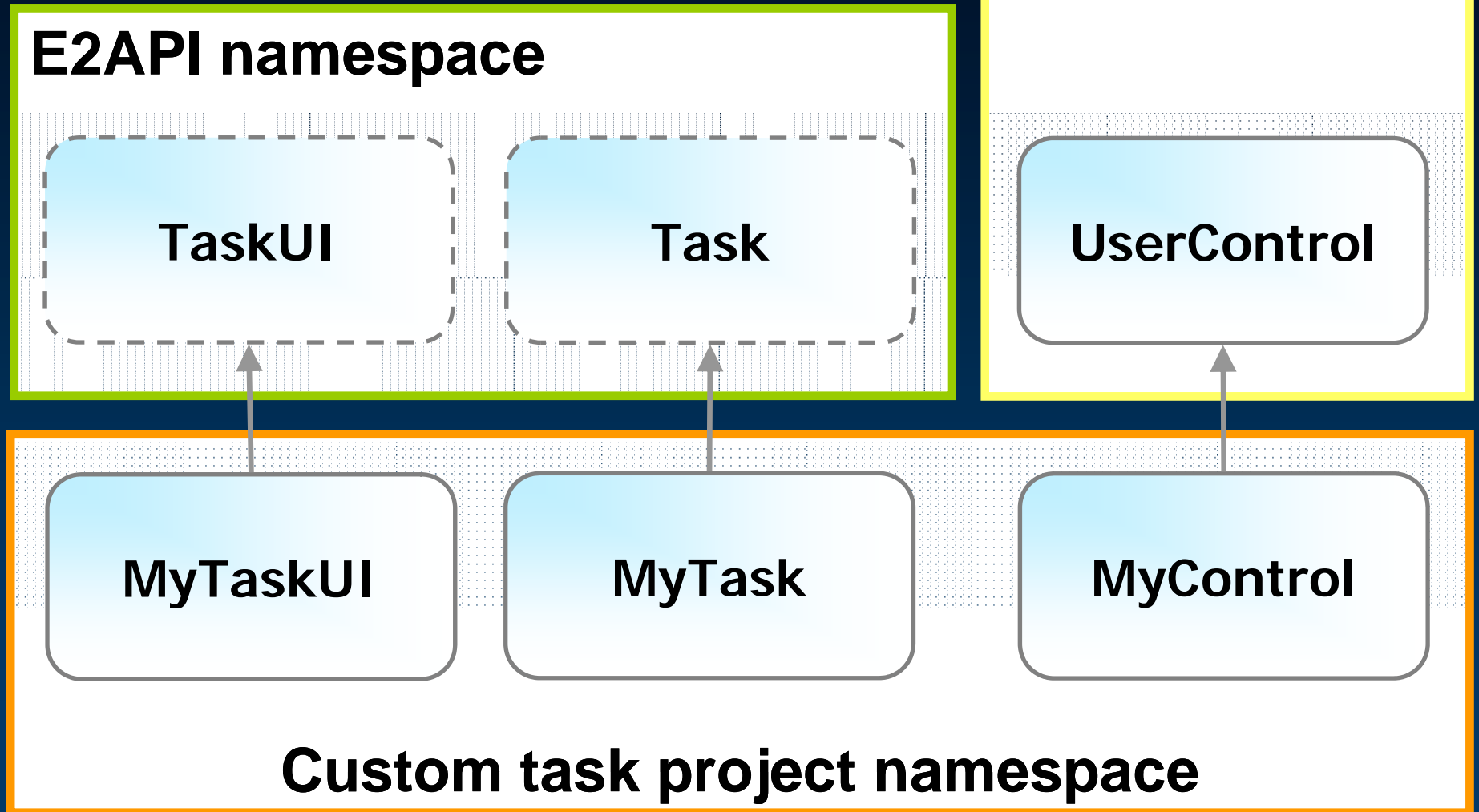
## *demonstration*

*What's in the SDK?*

# Agenda, next...

- **What is ArcGIS Explorer**
- **Programming custom tasks**
- **Task Framework**
  - Creating and programming custom tasks
  - Task conceptual steps and lifecycle
- **Typical customizations and issues**
- **Tips**

# Custom task architecture





# Getting started

## *demonstration*

### *Creating your first task*

# How a task works 1: conceptual operation

1. TaskUI is instantiated
2. TaskUI gathers the inputs
3. TaskUI creates a TaskContext and populates it with inputs
4. TaskUI tells application to process TaskContext
5. Application creates Task and executes it – task execution happens in a separate thread
6. Task performs operation and creates Results
7. Results displayed in application

## How a task works 2: Task and TaskUI communication

- **TaskUI and Task do not communicate directly**
- **TaskContext**
  - Represents a single execution of a task
  - Agile
  - Regulates information flow
  - Also regulates data types...
- **Information directly back to E2 via Results in the TaskContext**
- **Information passed explicitly**
  - UpdateParameters and UpdateResults
  - Call from both Task and TaskUI





# TaskContext status

*demonstration*

*Setting status messages*

# Agenda, next...

- **What is ArcGIS Explorer**
- **Programming custom tasks**
- **Task Framework**
- **Typical customizations and issues**
  - calling web services
  - task deployment/security
- **Other tips**

# Integrating web services

- **This is the aim of custom task framework**
- **.NET language ideal for programming against web services**
  - SOAP
  - REST also simple to program against
- **Service outputting geometries from ArcGIS Server must be converted manually to E2 geometries**
  - (Requested feature)
- **Geoprocessing services must output a map service to show the result**
  - functionality here differs from the Explorer UI

# Integrating web services into custom tasks

*demonstration*

**1 Calling existing tasks**

**2 Calling a web service directly**

# Task Deployment

- **Explorer can automatically download tasks**
  - DownloadLocation element of TaskDescription in NMF
  - Can reference zipfiles which will be unzipped
  - Can create separate setup program if required
- **Non-admin deployment**
  - installed to profile ( implies per-user)
- **Can have multiple versions of same task installed on machine at once**
  - even in same map - but best practice to avoid this

# Deploying custom tasks

*demonstration*

*Auto-download and setup*

# Task Security

- **Explorer can automatically download and run binaries from the internet**
- **Allows non-admin deployment**
- **You get a warning in the application**
- **Tasks run in Local Machine security context**
- **A Home Server can prevent the ability to download code from**
  - **Internet**
  - **Intranet**
  - **Trusted Sites (IE)**

# Tips

- **Problems in tasks loading / showing up:**
  - Use MS Fuslogvw (Fusion Log Viewer)
  - DebugLog setting:
    - Key: H\_C\_U\Software\ESRI\E2\Settings
    - Value: DebugLog
    - Data: 1 (DWORD)
    - Outputs log to ArcGIS Explorer folder in profile
  - ESRI.ArcGIS.E2API.dll reference - CopyLocal and SpecificVersion set to False
  - 'Deployment Troubleshooting' topic in help

## Tips 2

- **Drawing of lots of results takes time**
  - Set visible to False to begin
- **Search and Query - local vector layers only!**
- **No Area function**
  - More complex issue than first appears...
- **"Hidden" layers**
  - Always check `LayerCollection.GetLayer(i) Layer != null`
- **Override OnIdle (no event model)**
- **Result.Description can be HTML - very powerful**

# Future releases and updates to the API

- Will not be updating the API at every release
- Plan to keep backwards compatibility
- Some releases will not be forwards compatible
- All compatibility rules will be hard-coded inside Explorer in future releases
  - Messages to user if task is incompatible with current installation

# In Summary

- **Entry point to API is custom Task/TaskUI**
- **Simple API on client**
- **Task Framework is key**
- **Make maximum use of server side power and functionality**
- **Automatic downloading of tasks – straightforward deployment, think security**



# Microsoft Keynote

Wednesday 6:00pm  
Oasis 2

Eddie Amos

Senior Director, Developer & Platform Evangelism

# Question and Answers

*Meet the ArcGIS Explorer Team:  
Community Center, Thursday, 10:30am*

*Please remember to fill out your surveys*

# *Thankyou!*

# Requirements

- **Windows XP, 2000 (SP3 or higher), or Server 2003**
- **.NET Framework 2.0**
- **Visual Studio 2005**
  - Express versions will be supported for final
- **ArcGIS Explorer runtime (application)**
  - Installed as Administrator
  - Setup will install Windows Installer 3.1 and MSXML 4 SP2
- **ArcGIS Explorer SDK for .NET**
  - Installed as Administrator