

Esri Developer Summit

March 26–29, 2012 | Palm Springs, California

esri.com/events/devsummit



Advanced Development with ArcGIS API for Silverlight

Morten Nielsen

Rex Hansen

Jennifer Nery

A decorative graphic at the bottom of the slide. It features a curved orange border at the top. Below the border is a semi-transparent map of a landscape with green fields and blue water. Overlaid on the map is white text representing code snippets from the ArcGIS API for Silverlight. The code includes symbols like 'esri.symbol.SimpleLineSymbol', 'new dojo.Color([0,0,0,0.5])', and 'feature.setSymbol(polySymbolGreen)'.

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

Agenda

- Graphics data sources
- Printing
- Deep dive into image service demo
- Feature service editor tracking



Microsoft®
Silverlight™

Graphics Data Sources

Morten Nielsen

Adding graphics to GraphicsLayer

- `foreach(Graphic g in myGraphics)`
- `graphicsLayer.Graphics.Add(myGraphics);`

• or

read-only when GraphicsSource is used!



- `graphicsLayer.GraphicsSource = myGraphics;`

can be any IEnumerable<Graphic>



Adding custom data to a GraphicsLayer

```
public class MyData {  
    public double Longitude { get; set; }  
    public double Latitude { get; set; }  
}  
...  
foreach(MyData in dataList)  
    myGraphicsLayer.Graphics.Add(  
        new Graphic() {  
            new MapPoint(Longitude, Latitude)  
        }  
    );
```

...must simpler with the PointDataSource

```
<esri:GraphicsLayer>  
  <esri:GraphicsLayer.GraphicsSource>  
    <esri:PointDataSource  
      XCoordinateBinding="{Binding Longitude}"  
      YCoordinateBinding="{Binding Latitude}"  
      ItemsSource="{StaticResource dataList}" />  
  </esri:GraphicsLayer.GraphicsSource>  
</esri:GraphicsLayer>
```

Build your own custom datasource!

Morten Nielsen

Diving into the Image Editor Sample

Morten Nielsen

The screenshot displays the 'Image Service Editor' application. At the top, there are navigation links for 'home', 'map', 'catalog', and 'about'. Below this is a green header bar with the text 'Image Service Editor' and a version number '1.0.0.0'. The main workspace is divided into two map panes: 'Input' on the left and 'Reference' on the right. The 'Input' map shows a satellite-style image of a coastal area with several white circular markers and dashed lines indicating georeferencing points. The 'Reference' map shows a corresponding image with a different color scheme (blue and green) and similar markers. Below the maps, there are 'Accept' and 'Cancel' buttons. To the right of the 'Reference' map, there is a small thumbnail of the input image. Below the maps, there is a metadata table with the following data:

Field	Value
OBJECTID	4121
Name	latina 08-28-2005
MirPS	0
MaxPS	5000

Creating Map Overlays

Converting between screen and map coordinates:

```
Map.MapToScreen(MapPoint);
```

```
Map.ScreenToMap(ScreenPoint);
```

Convert screen points between visual elements:

```
map.TransformToVisual(elm2).Transform(...)
```

Use `Map.ExtentChanging` event to reposition elements.

UploadTask

- **Great for upload of large files.**
- **Provides progress, cancellation and deletion.**
- **Great for large geoprocessing input or raster uploads.**

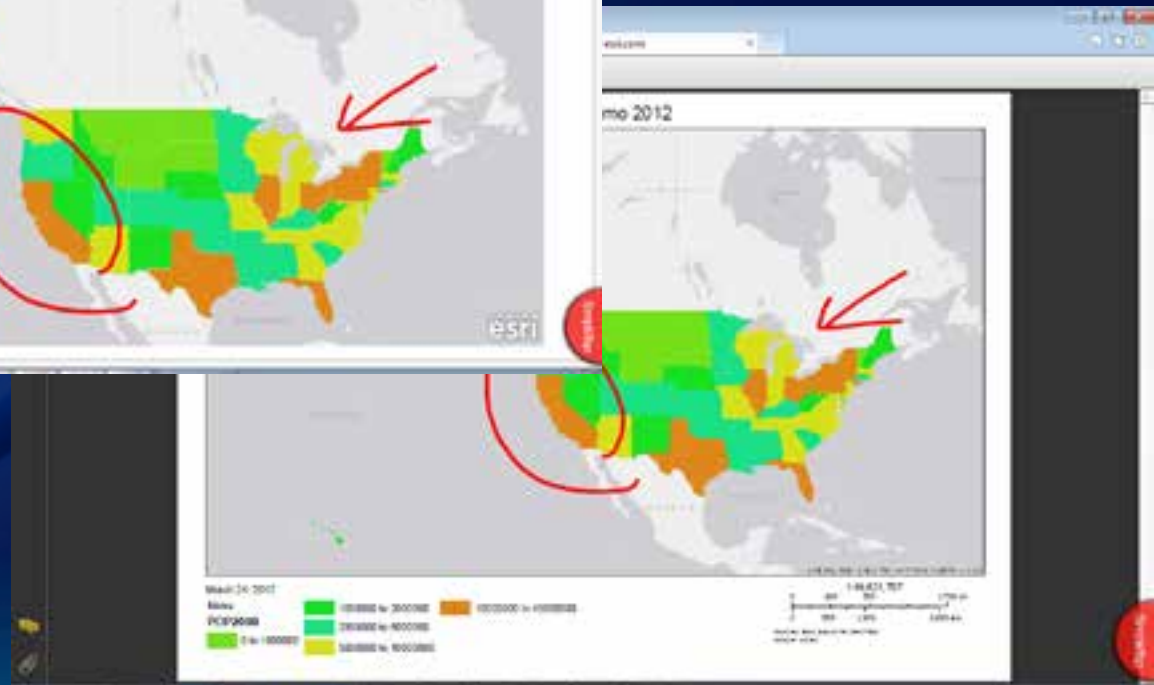
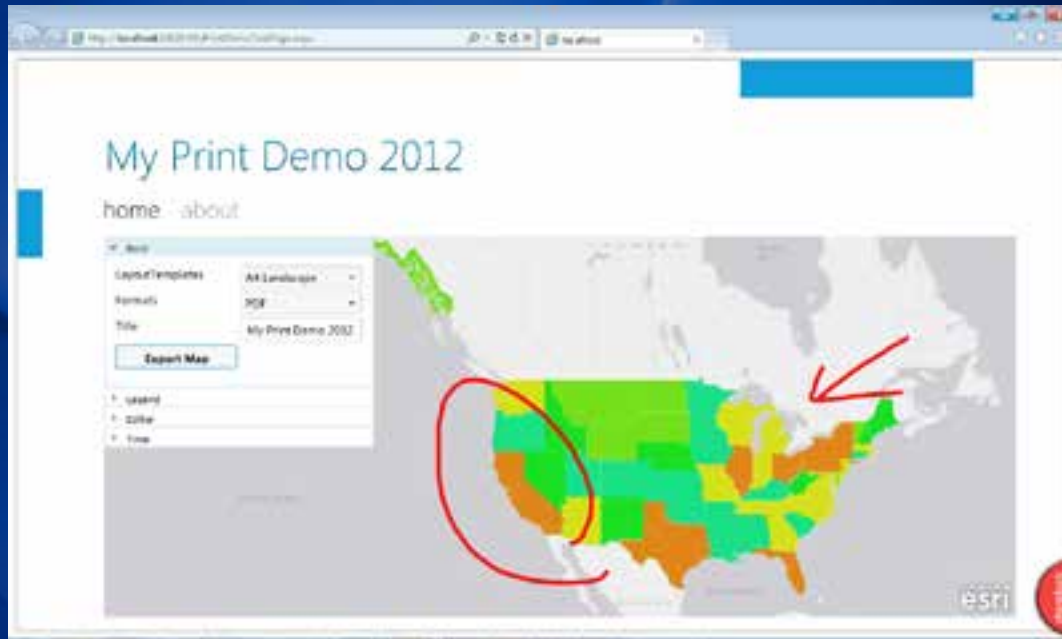
```
OpenFileDialog dialog = new OpenFileDialog() { Filter = "Image|.jpg" };
if (dialog.ShowDialog() == true) {
    var imageStream = dialog.File.OpenRead();
    var uploadTask = new ESRI.ArcGIS.Client.Tasks.UploadTask(serviceUrl);
    uploadTask.UploadCompleted += (s, e) => {
        imageStream.Close(); //close file stream
        string itemID = e.Result.Item.ItemID;
    };
    uploadTask.UploadProgress += (s, e) => {
        Dispatcher.BeginInvoke(() => {
            progressBar.Value = b.BytesWritten / (double)b.TotalBytes;
        });
    };
    uploadTask.UploadAsync(new UploadParameters() {
        FileStream = imageStream, FileName = dialog.FileName });
}
```

Z & M Support

Morten Nielsen

Printing

Jennifer Nery



using ESRI.ArcGIS.Client.Printing

```
var printTask = new PrintTask("http://taskUrl");
printTask.ExecuteCompleted += (s, e) =>
{
    if (e.Error == null)
        HtmlPage.Window.Navigate(e.PrintResult.Url, "_blank");
};
printTask.ExecuteAsync(new PrintParameters(MyMap));
or
printTask.StatusUpdated += (s, e) => {...};
printTask.JobCompleted += (s, e) => {...};
printTask.SubmitJob(...);
```

```
<StackPanel x:Name="PrintPanel"/>
  <ComboBox x:Name="Templates"
            ItemsSource="{Binding LayoutTemplates}"/>
  <ComboBox x:Name="Formats"
            ItemsSource="{Binding Formats}"/>
  <TextBox x:Name="MapTitle" Text="Print Demo"/>
</StackPanel>
```

```
printTask.GetServiceInfoCompleted += (s, e) =>
{
    PrintPanel.DataContext = e.ServiceInfo;
};
printTask.GetServiceInfoAsync();
```

```
var printParams= new PrintParameters(MyMap);
printParams.LayoutTemplate = (string) Templates.SelectedItem;
printParams.Format = (string) Formats.SelectedItem;
printParams.LayoutOptions = new LayoutOptions()
{
    Title = MapTitle.Text,
    LegendOptions = new LegendOptions() {...},
    ScaleBarOptions = new ScaleBarOptions() {...},
    ...
};
printParams.MapOptions = new MapOptions() {...};
printParams.ExportOptions = new ExportOptions() {...}
printTask.ExecuteAsync(printParams);
```


PrintTask: Supported Features in v3.0

- **Get service information (choice list, type).**
- **Support server synchronous/asynchronous type.**
- **Ability to cancel print job.**
- **Print options: map, export, layout.**
- **10.1 Capabilities: DynamicLayer/Source, GdbVersion.**
- **Wrap-around.**
- **Token-secured services.**

PrintTask: Supported Layers in v3.0

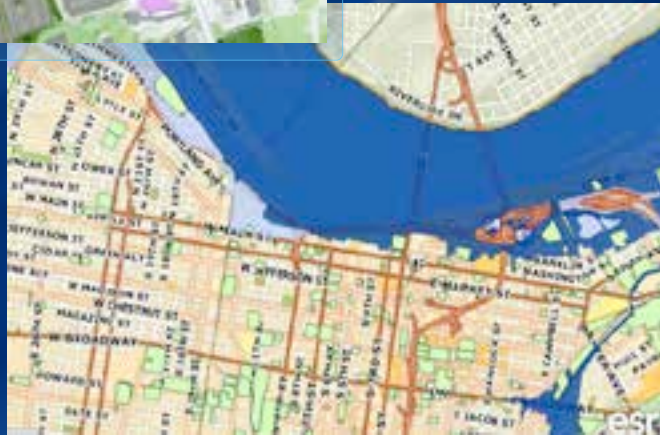
- ArcGISTiledMapServiceLayer
- ArcGISDynamicMapServiceLayer
- ArcGISImageServiceLayer
- FeatureLayer (service, featureCollection)
- GraphicsLayer
- GroupLayer
- Bing.TileLayer
- KmlLayer
- WmsLayer
- WmtsLayer
- OpenStreetMapLayer

PrintTask: Known Limitations in v3.0

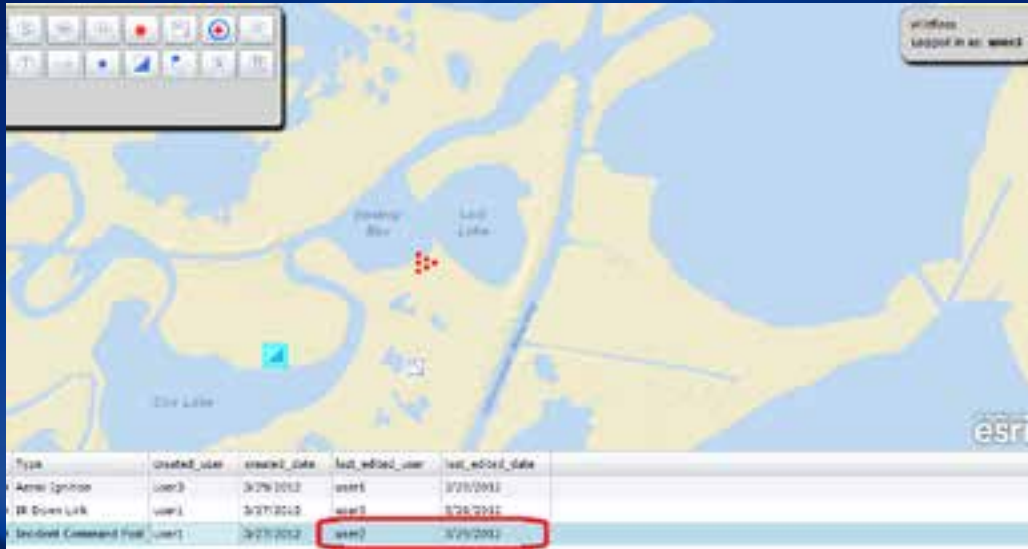
- **Temporal renderer (except MAP_ONLY)**
- **GraphicsLayer at least 1 attribute**
- **Custom symbols/renderers**
- **Clustering**
- **KmlLayer does not support visibleFolders**
- **GraphicsLayer with mixed geometries**
- **FeatureLayer selection/geometry require ObjectID**

Editor Tracking

Rex Hansen



Feature last edited by another user

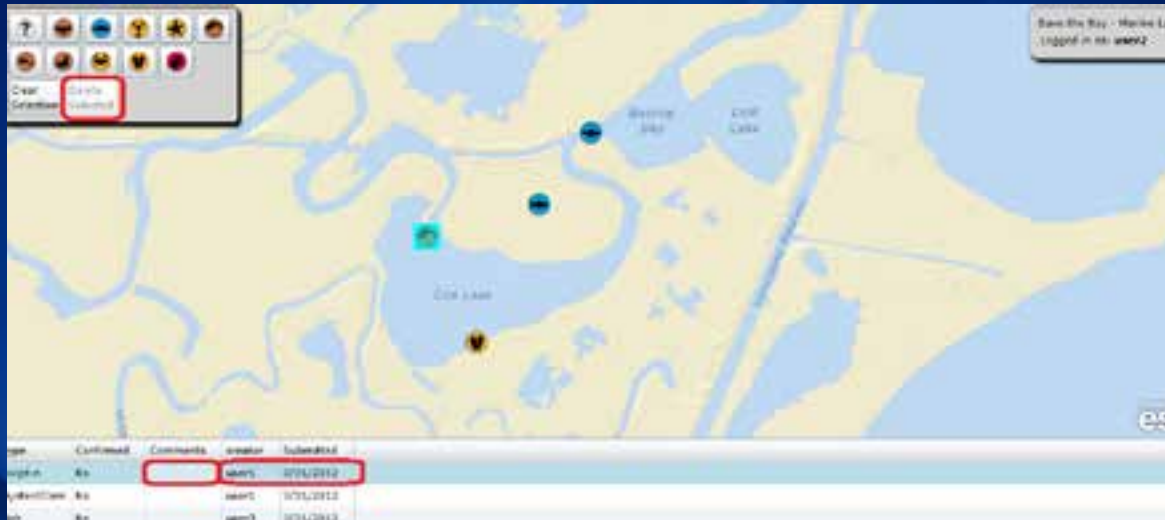


Editor Tracking



Feature moved, last edited user and date updated

Can't update or delete, don't own the feature



Ownership



Created feature, can update and delete

Feature service – Editor tracking

- **FeatureLayer.EditUserName**
 - Client side validation
 - **FeatureLayer.IsUpdateAllowed(graphic)**
 - Check if feature can be updated
-
- **FeatureLayerInfo.EditFieldsInfo**
 - Owner, editor, date
 - **FeatureLayerInfo.OwnershipBasedAccessControl**
 - If others can update, delete
 - **FeatureLayerInfo.AllowGeometryUpdates**
 - Attribute only editing

Related session

- **Supporting High-Quality Printing in Web Applications with ArcGIS 10.1 for Server**
- **Craig Williams, Mohammed Hoque**
- **Thursday, 1:30pm-2:45pm**
- **Primrose C/D**



esri