

Esri Developer Summit

March 8–11, 2016 | Palm Springs, CA

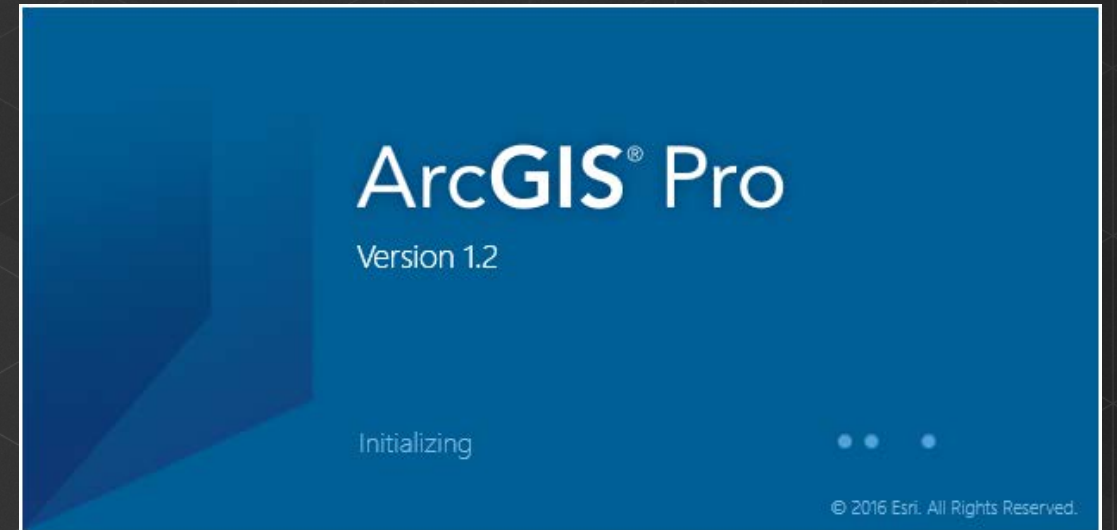


ArcGIS Pro SDK for .NET: UI Design and MVVM

Charlie Macleod, Wolf Kaiser

Important Customization Patterns for the Pro SDK

- **MVVM**
- **Hooking Pro Commands**
- **MVVM – Lists & Collections**
- **MVVM – Embedded Control**

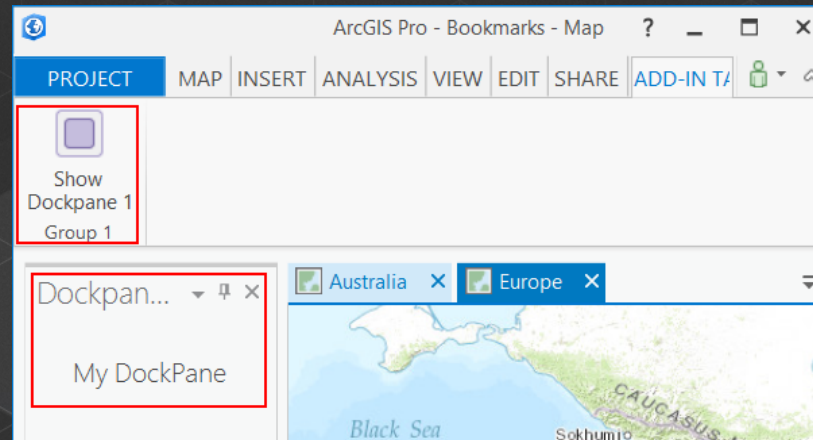


MVVM Pattern in Add-ins

- MVVM and variants are de-facto pattern for WPF UI implementations
- Many Frameworks available
 - Home grown or “hand rolled”
 - Commercial (free, purchased)
- Pro MVVM is built on top of ActiPro
(<http://www.actiprosoftware.com/products/controls/wpf>)
- The Basic Pattern is:
 - ViewModel declared in DAML and implemented in code
 - View referenced in DAML and implemented as WPF UserControl
 - Model is optional
- Note: To customize the Pro UI, you must use its MVVM Framework. Substitutes are not allowed.

MVVM Pattern in Add-ins

- **Model-View-ViewModel (MVVM) Pattern used for many of the Framework elements**
 - Dockpane
 - Pane
 - Custom Control
 - Embeddable Control
 - Property Page



Demo: MVVM – Dockpane – Template



Hooking Commands

- Get any Pro control's ICommand and use it in your add-in using Framework's "GetPlugInWrapper" method.
- Following example shows how a Pro control's command can be added to your add-in button click method. (or anywhere else in your add-in).

```
// ArcGIS Pro's Create button control DAML ID.
var commandId = DAML.Button.esri_mapping_createBookmark;
// get the ICommand interface from the ArcGIS Pro Button
// using command's plug-in wrapper
// (note ArcGIS.Desktop.Core.ProApp can also be used)
var iCommand = FrameworkApplication.GetPlugInWrapper(commandId) as ICommand;
if (iCommand != null)
{
    // Let ArcGIS Pro do the work for us
    if (iCommand.CanExecute(null))
        iCommand.Execute(null);
}
```


Hooking Commands

- Adding a button to the Dockpane to run the 'Close ArcGIS Pro' Command

```
protected Dockpane1ViewModel()  
{  
    CloseCmd = FrameworkApplication.GetPlugInWrapper(DAML.Button.esri_core_exitApplicationButton)  
        as ICommand;  
}  
  
public ICommand CloseCmd { get; set; }
```

- Adding a button to the Dockpane with our 'custom' Zoom in behavior

```
protected Dockpane1ViewModel()  
{  
    ZoomInCmd = new RelayCommand(() => MappingModule.ActiveMapView.ZoomInFixedAsync(),  
        () => MappingModule.ActiveMapView != null);  
}  
  
public ICommand ZoomInCmd { get; set; }
```

- RelayCommand is an implementation of ICommand which lets you specify your own implementation of Execute and CanExecute

Demo: Hooking Pro Commands



MVVM – Dockpane

- Singleton
- Has no context association with the ribbon and no active tool
- Once created only hidden, never destroyed
- View Model derives from the DockPane Contract
- View is Custom User Control associated with DockPane declaratively
 - (Use of Model is optional and completely up to developer)

```
<dockPanes>  
  <dockPane id="custom_TOCDockPane" caption="My Contents" className="DockPaneViewModel"  
    dock="group" condition="esri_core_MapPane" dockWith="esri_core_ProjectDockPane">  
    <content className="DockPaneView" />  
  </dockPane>  
</dockPanes>
```

Multi-threading considerations

- ArcGIS Pro Framework's managed threading model:
 - Framework provides `QueuedTask` to guarantee that UI actions happen in a sensible order without corruption
- Updating UI collections from a worker thread
 - Locking is required when sharing objects across threads
- Recommended pattern for updating collections from a worker thread
 - Found in .Net `BindingOperations` helper class:
`BindingOperations.EnableCollectionSynchronization`

```
private readonly object _lockListOfBookmarks = new object();

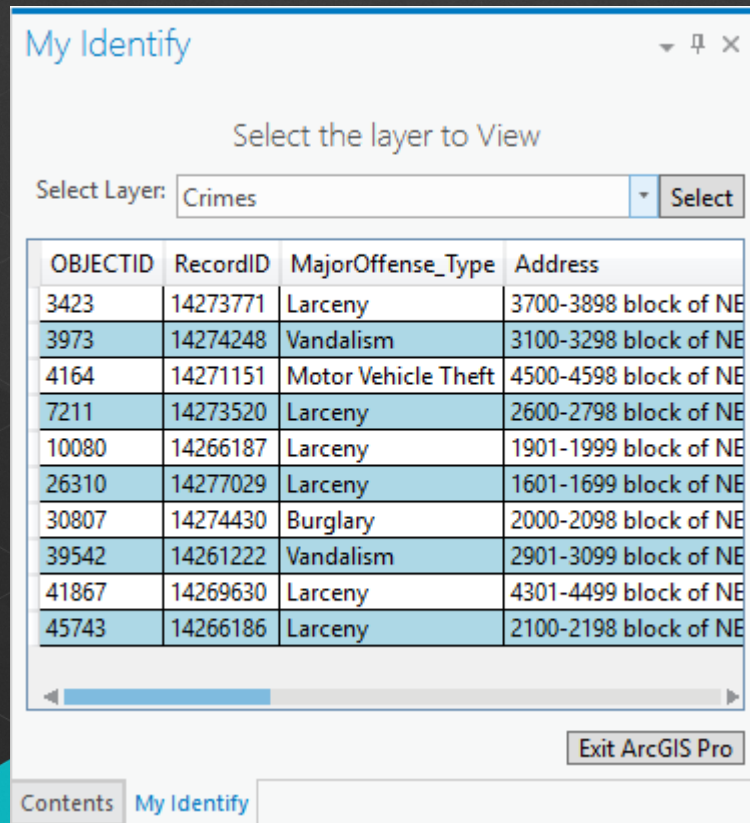
protected override Task InitializeAsync()
{
    BindingOperations.EnableCollectionSynchronization(ListOfBookmarks, _lockListOfBookmarks);
    GetBookmarkCollection();
    return base.InitializeAsync();
}
```


Demo Custom Identify Window



Matching ArcGIS Pro Look & Feel

- Check “ProGuide ArcGIS Pro Styles” for available styles, fonts, etc.
- Example buttons:



My Identify

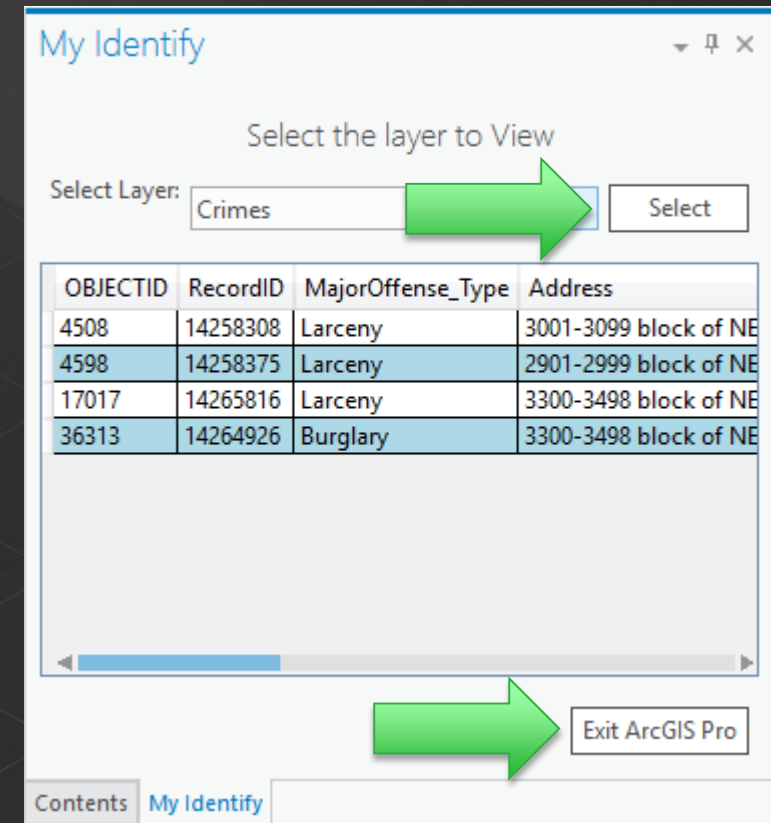
Select the layer to View

Select Layer: Crimes Select

OBJECTID	RecordID	MajorOffense_Type	Address
3423	14273771	Larceny	3700-3898 block of NE
3973	14274248	Vandalism	3100-3298 block of NE
4164	14271151	Motor Vehicle Theft	4500-4598 block of NE
7211	14273520	Larceny	2600-2798 block of NE
10080	14266187	Larceny	1901-1999 block of NE
26310	14277029	Larceny	1601-1699 block of NE
30807	14274430	Burglary	2000-2098 block of NE
39542	14261222	Vandalism	2901-3099 block of NE
41867	14269630	Larceny	4301-4499 block of NE
45743	14266186	Larceny	2100-2198 block of NE

Exit ArcGIS Pro

Contents My Identify



My Identify

Select the layer to View

Select Layer: Crimes Select

OBJECTID	RecordID	MajorOffense_Type	Address
4508	14258308	Larceny	3001-3099 block of NE
4598	14258375	Larceny	2901-2999 block of NE
17017	14265816	Larceny	3300-3498 block of NE
36313	14264926	Burglary	3300-3498 block of NE

Exit ArcGIS Pro

Contents My Identify

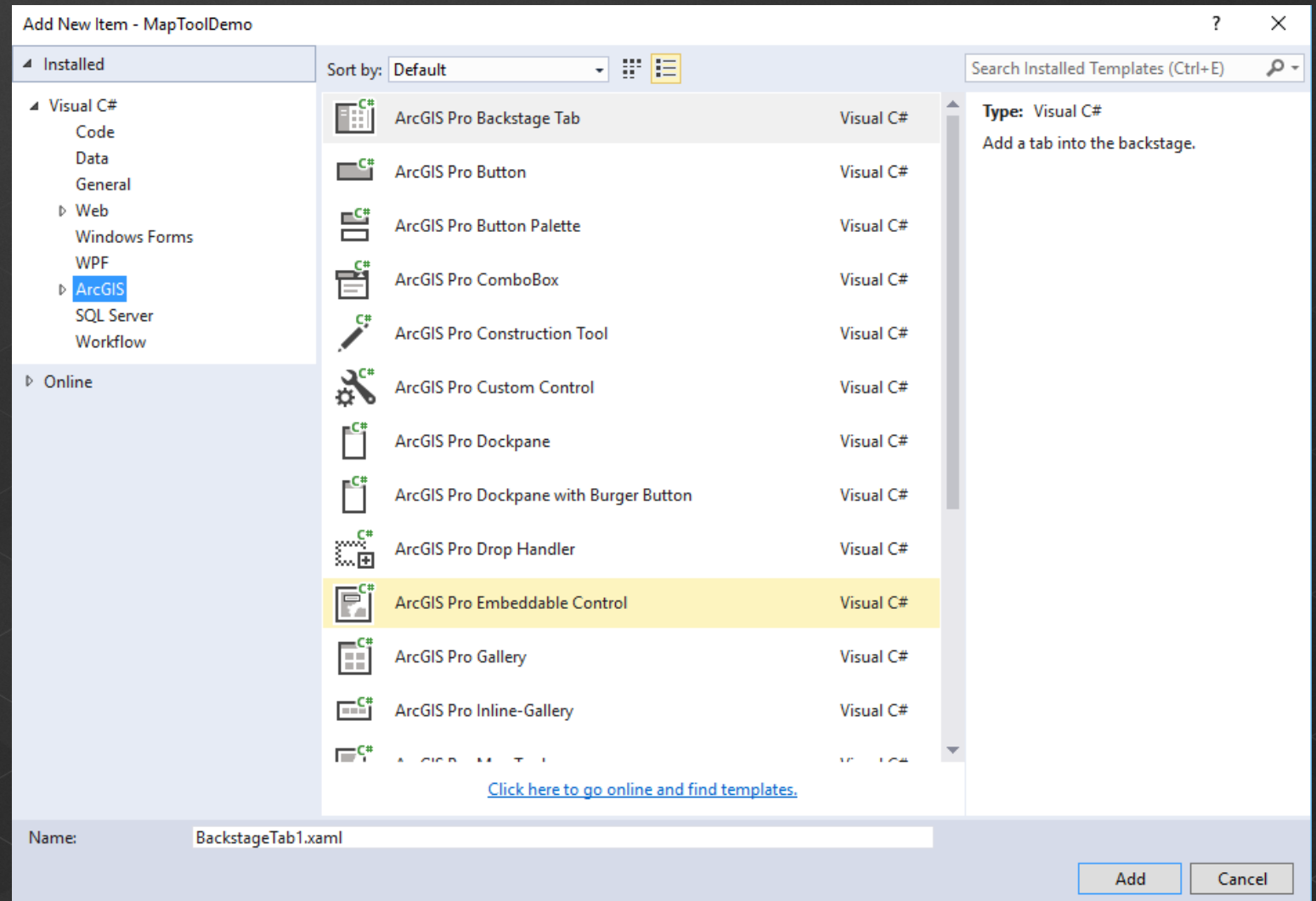
Embeddable Control

- Adds a Custom UI Component for use with MapTools
 - Uses MVVM
 - Linked with a MapTool
 - When the Tool is activated the Control becomes visible over the Active MapView
 - When the Tool is deactivated the Control is hidden
 - Think “popup”

Embeddable Control

New for 1.2

- Add to the Config.daml via the ArcGIS Pro SDK template



Embeddable Control

- Template updates the “esri_embeddableControls” category in Config.daml
 - DAML follows same Pro MVVM pattern

```
<categories>
  <updateCategory refID="esri_embeddableControls">
    <insertComponent id="MapToolDemo_EmbeddableControl"
      className="EmbeddableControl1ViewModel">
      <content className="EmbeddableControl1View" />
    </insertComponent>
  </updateCategory>
</categories>
```

Embeddable Control

- Associate a Tool with the Control via the Tool's `OverlayControlID`
- Access the View Model via `this.OverlayEmbeddableControl`

```
internal class MySelectionTool : MapTool
{
    private EmbeddableControl1ViewModel _vm = null;
    public MySelectionTool() {
        this.OverlayControlID = "MapToolDemo_EmbeddableControl1";
    }
    protected override Task OnToolActivateAsync(bool active) {
        if (_vm == null) {
            _vm = this.OverlayEmbeddableControl as EmbeddableControl1ViewModel;
        }
        return base.OnToolActivateAsync(active);
    }
}

internal class EmbeddableControl1ViewModel : EmbeddableControl {
    public OverlayControlViewModel(XElement options) : base(options) {
    }
}
```


Embeddable Control

- By default, position is at Upper Left of MapView
- To enable moving disable hit testing on a portion of your UserControl
 - In other words, simply let the parent handle the MouseMove

```
<UserControl x:Class="...">
  <Grid>
    <Border Background="{DynamicResource Esri_Gray100}"
            BorderBrush="{DynamicResource Esri_Gray125}"
            BorderThickness="1"
            IsHitTestVisible="False">
      <TextBlock Text="{Binding ClickText}"
                HorizontalAlignment="Center"
                VerticalAlignment="Center"
                Margin="4"/>
    </Border>
  </Grid>
</UserControl>
```

ArcGIS Pro SDK for .NET Sessions

Tue March 8: 4:00 PM – 5:00 PM Primrose A (Palm Springs Convention Center)

ArcGIS Pro SDK for .NET: Programming Patterns

Tue March 8: 5:30 PM – 6:30 PM Primrose A (Palm Springs Convention Center)

ArcGIS Pro SDK for .NET: Editing and Geodatabase Integration

Wed March 9: 10:30 AM – 11:30 AM Catalina/Madera (Renaissance Hotel)

ArcGIS Pro SDK for .NET: Integration with ArcGIS Online

Wed March 9: 10:30 AM – 11:30 AM Mesquite C (Palm Springs Convention Center)

ArcGIS Pro SDK for .NET: Solution Based Configuration of ArcGIS (at 1.3)

Wed March 9: 4:00 PM – 5:00 PM Primrose B (Palm Springs Convention Center)

ArcGIS Pro SDK for .NET: UI Design and MVVM

Thu March 10: 9:00 AM – 10:00 AM Primrose C/D (Palm Springs Convention Center)

ArcGIS Pro SDK for .NET: Animation and Map Exploration

