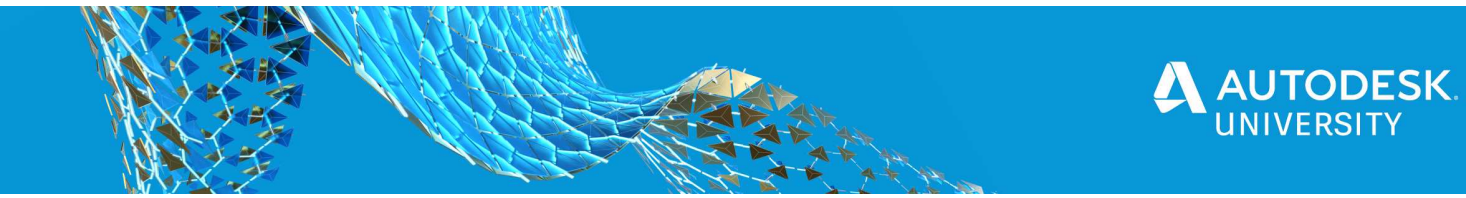


SD468556

Modeless Revit plugins with WPF

Gopinath Rajendran
DiRoots Ltd

Learning Objectives

- Learn why Revit API and C#?
- Learn the basics of WPF, general advantages and disadvantages
- Modeless applications and external events
- Learn how to implement WFP and modeless Dialog Windows

Description

Are you interested to learn more about modeless Revit plugins? Yes! Great, this class is for you. User experience and accessibility are becoming a trend while developing software. How do you achieve that? User experience can be achieved by using WPF instead of Windows form and accessibility can be achieved by using modeless dialogs. This class will present the advantage of using WPF and modeless dialog while developing your Revit plugins. User can see the actions and data flows between Revit and plugin. In this session, a demo of the workflow will be shown and it consists of multiple steps that depend on each other to show the features that WPF provides.

Speaker(s)

Gopinath Rajendran has a background in Computer Sciences. He is the Team Leader and one of the key developers involved in the popular free Revit plugins from DiRoots. He is highly involved in the implementation and training of developing custom software tools for Autodesk Revit for the past five years. His passion and commitment are focused on the development of tools that aims to automate tedious tasks with high accuracy and efficiency.

Revit API

.NET API which can be used to automate repetitive tasks, extend the core functionality of Revit. Revit .NET API allows you to program with any .NET compliant language including C#, VB.NET, and C++/CLI

WPF

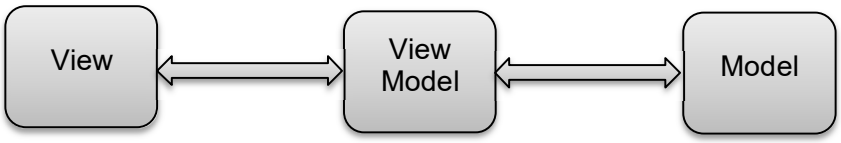
WPF (Windows Presentation Foundation), previously known as "Avalon", was initially released as part of .NET Framework 3.0 in 2006.

Advantages

- MVVM Pattern.
- Nested controls.
- Templates.

MVVM Pattern

MVVM (Model, View, ViewModel) pattern allows you to get a more clean separation of data and layout



Model – holds the data
ViewModel - acts as the link between the Model and View
View – User interface

Nested Controls

Allows creating control inside another control.

advanced_sample_family.rfa

```
<Button FontSize="15" HorizontalContentAlignment="Left" Command="{Binding Path= (???).ExpandCommand}" Cursor="Hand" >
  <StackPanel Orientation="Vertical" HorizontalAlignment="Left">
    <TextBlock Text="{Binding Path= (???).File.FileName}" Margin="[l:]0,[t:]-1,[r:]0,[b:]0" FontSize="15" HorizontalAlignment="Left" />
    <Label Background="{Binding Path= (???).File.BrushColor}" Width="100" HorizontalAlignment="Left" FontSize="13" Height="3" VerticalContentAlignment="Center" >
      <Label.Resources>
        <Style TargetType="{x:Type TypeName=Border}">
          <Setter Property="CornerRadius" Value="[all:]3" />
        </Style>
      </Label.Resources>
    </Label>
  </StackPanel>
</Button>
```

FIGURE 1: Nested controls

TextBlock and Label controls are nested inside in the button control which allows the control to display different values (Family name and color) at the same time with different properties.

Templates

Templates allow customizing the visual behavior and visual appearance of a control.

Window with Template:
In WPF window can be customized using a template.

ResourceDictionary is a repository for XAML resources, such as styles. FamilyManager contains two ResourceDictionary called **Images.xaml** and **LocalStyle.xaml**

Window Style:
Styles are used to changing appearance to a set of controls.

- 'Key' is a unique name of style.
- 'TargetType' is used to specify the target control type.
- In the setter, property means what behavior you want to change.
- In the setter, value means what will be the value of behavior.

  Locating ResourceDictionary In Visual Studio DiRoots.Public.FamilyManager=>UI=> Styles=>LocalStyle.xaml

```
<Style x:Key="CustomWindowStyle" TargetType="{x:Type TypeName=Window}">
  <Setter Property="FontFamily" Value="Raleway"/>
  <Setter Property="WindowStyle" Value="None"/>
  <Setter Property="AllowsTransparency" Value="True"/>
  <Setter Property="UseLayoutRounding" Value="True" />
  <Setter Property="Background" Value="#EBEBEB"/>
  <Setter Property="BorderBrush" Value="Gray"/>
  <Setter Property="BorderThickness" Value="[l:]0 [t:]0 [r:]0 [b:]0"/>
  <Setter Property="Foreground" Value="#E5AF42"/>
  <Setter Property="WindowChrome.WindowChrome">
    <Setter.Value>
      <WindowChrome CaptionHeight="0" ResizeBorderThickness="[all:]7" />
    </Setter.Value>
  </Setter>
  <Setter Property="Template" Value="{StaticResource ResourceKey=CustomWindowTemplate}"/>
</Style>
```

Other Advantages

- XAML makes it easy to create and edit your GUI and allows the work to be split between a designer (XAML) and a programmer (C#, VB.NET, etc.)
- Uses hardware acceleration for drawing the GUI, for better performance

Modeless applications

Modeless application, which stays on the screen but permits other user activities.

External Events

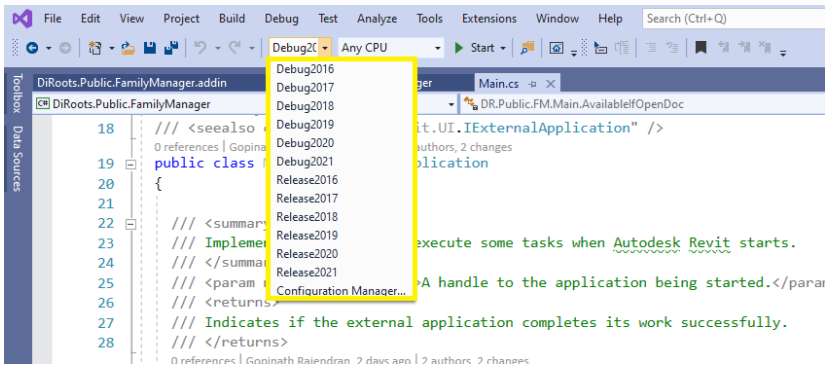
The Revit API provides an External Events framework to accommodate the use of modeless dialogs.

To implement a modeless dialog using the External Events framework, follow these steps:

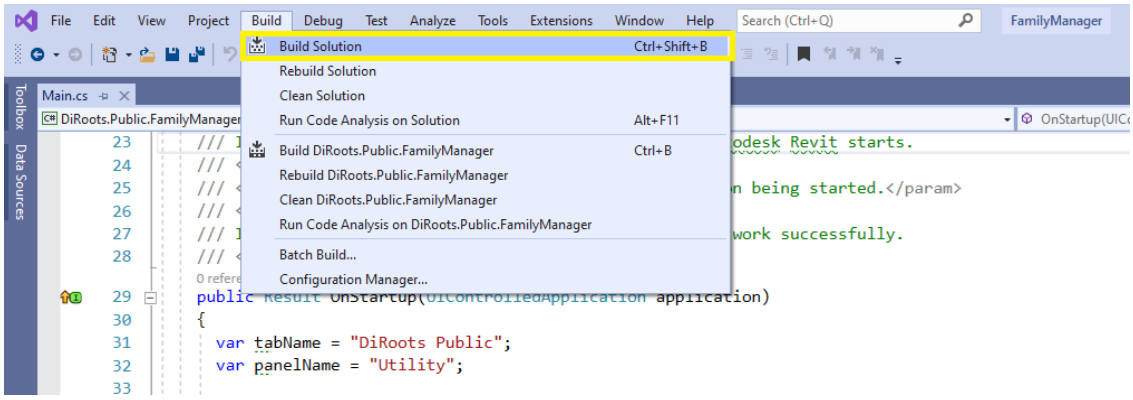
1. Implement an external event handler by deriving from the IExternalEventHandler interface
2. Create an ExternalEvent using the static ExternalEvent.Create() method
3. When an event occurs in the modeless dialog where a Revit action needs to be taken, call ExternalEvent.Raise()
4. Revit will call the implementation of the IExternalEventHandler.Execute() method when there is an available Idling time cycle.

How to run the application

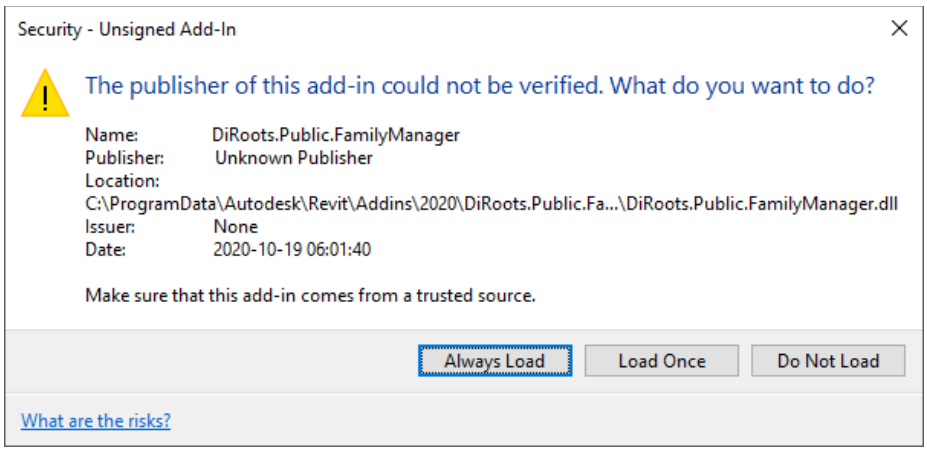
1. Change the configuration based on the Revit version available on the machine.



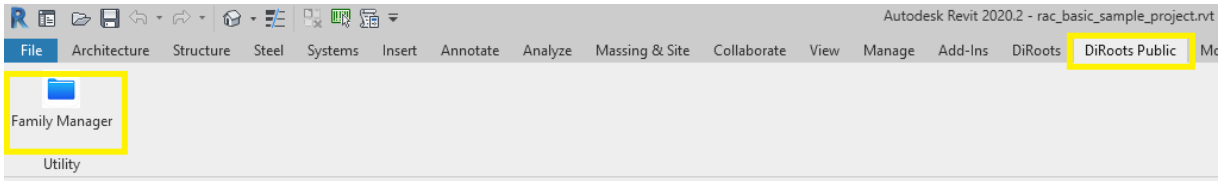
2. Build the solution Build=>Build Solution to copy libraries and .addin files to following path %programdata%\Autodesk\Revit\Addin\2020\ DiRoots.Public.FamilyManager



3. Open the Revit and select the **Always Load** option from the below dialog.



4. In Autodesk Revit, from the **DiRoots Public** ribbon tab click **Family Manager** as shown in Figure.



5. In the **Family Manager** dialog box, click **+Repository** to add the repository.