MP2153 - Automate your Autodesk® Revit® Workflows: Let the API Do the Work

David Rushforth, PE, LEED AP

Electrical Engineer, Glumac

www.rushforthprojects.com

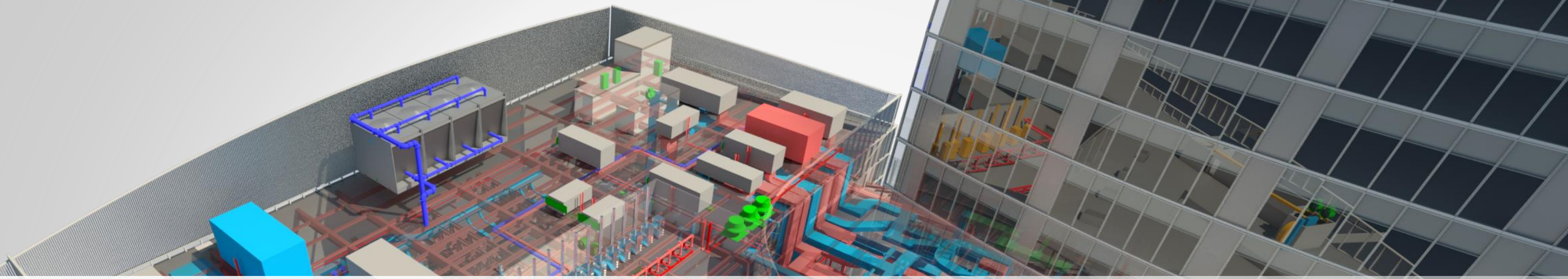
Class summary

This class will provide a **big picture** overview of how the API interacts with your projects as well as give specific training on using the API to automate your workflows. We will discuss actual time-saving solutions that are made possible with the API with a focus on giving participants the general understanding and resources needed to start their **own API projects**. We will discuss specific code samples of how to **extract information** from elements in linked models and some practical uses of the **raytrace** functions of the Revit API. The use of the API to automate **data manipulation** and the transfer of data from other sources will also be presented. **Example applications** will be shown and ideas for future development will be discussed.

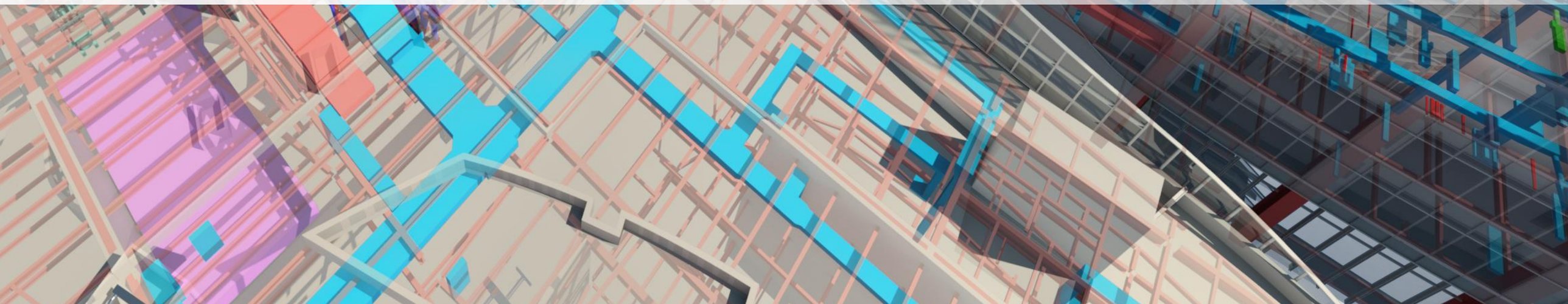
Key learning objectives

At the end of this class, you will be able to:

- Recognize the potential for the API to accelerate production in the real world
- Understand the API development process as a whole
- Use the API to query linked models for element information
- Identify methods of data import/export and model manipulation using the API



Expectations and Introduction



Class Expectations

- Have Questions?
  - Save discussion questions for the end. Clarifying questions are OK anytime.
- Experience Level
  - All levels from Beginner to Advanced.
- Goal
  - ~~Recognize the capabilities of the Revit API~~
  - Start using the API to automate your workflows NOW.

Class Expectations

- Have Questions?
  - Save discussion questions for the end. Clarifying questions are OK anytime.
- Experience Level
  - All levels from Beginner to Advanced.
- Goal
  - ~~Recognize the capabilities of the Revit API~~
  - Start using the API to automate your workflows NOW.

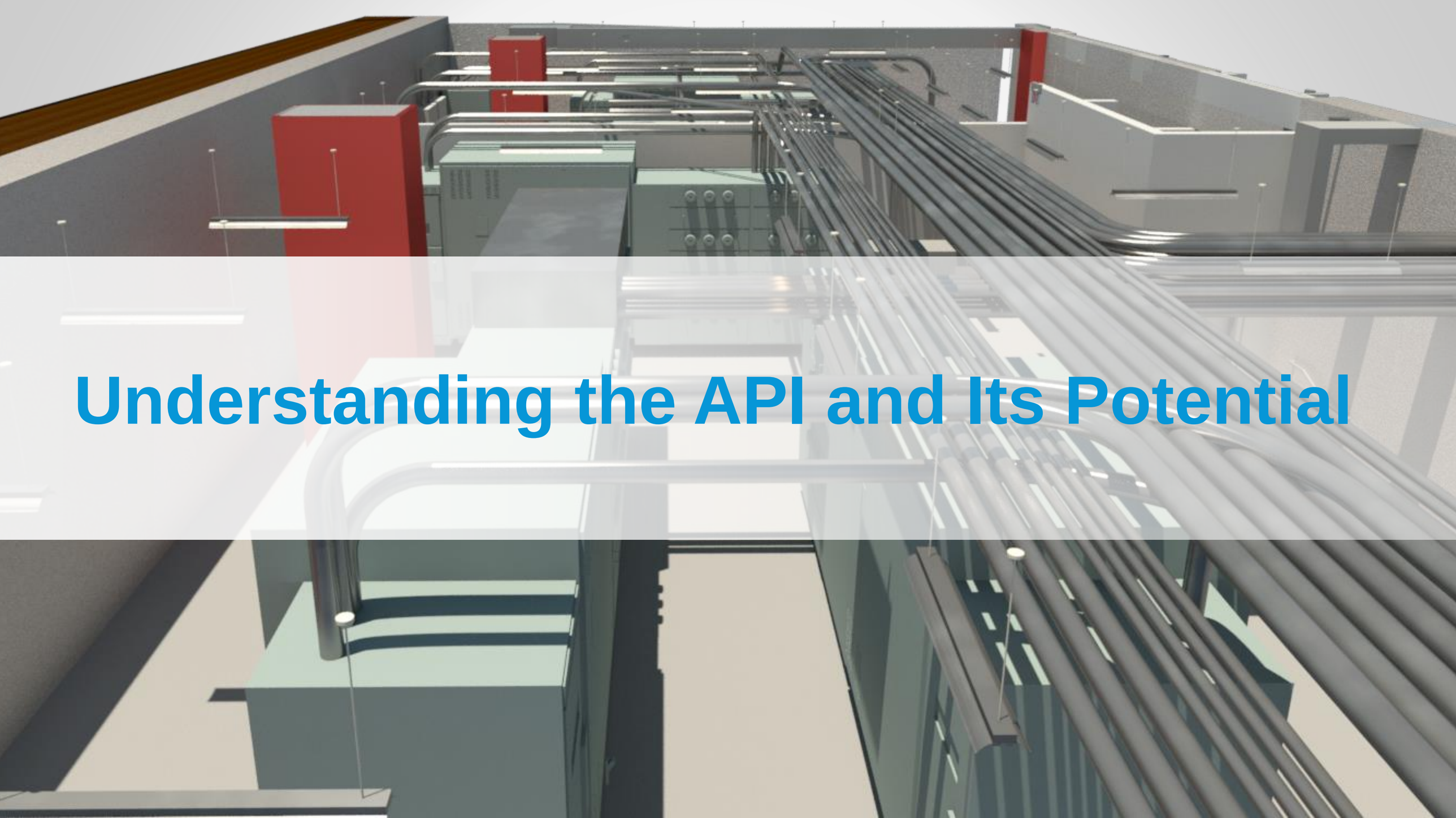
- WARNING!
  - The following content was prepared by an engineer, NOT a programmer. The samples of code shown may contain disturbing examples of poor programming practices and represent only one of many ways to accomplish the desired objective.

About the Presenter

- Electrical Engineer, Glumac
- Owner/Software Developer, RushForth Projects
- Native to Las Vegas
- BS Electrical Engineering, BYU
- MS Electrical Engineering, UNLV
- Programming experience:
  - Always looking for ways to make my work easier
  - Revit add-ins since 2008

Discussion Sections

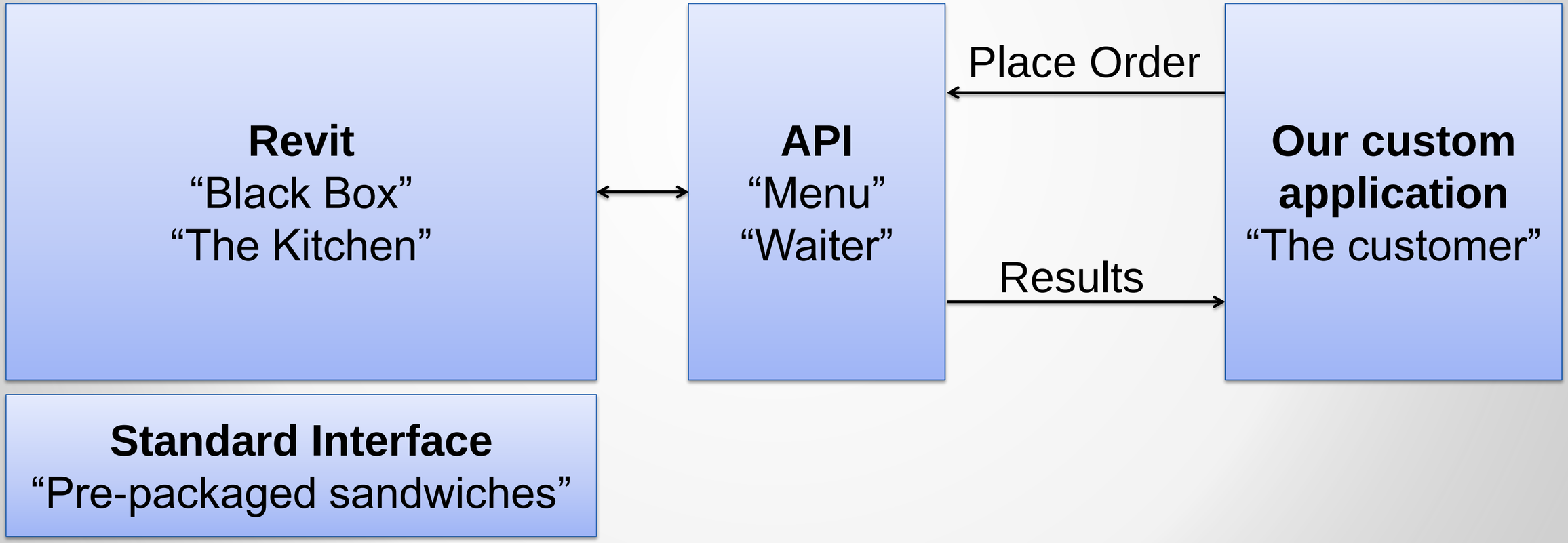
- Understanding the API and Its Potential
- API Development Process and Resources Overview
- Live Macro Demonstrations
- API to Query Linked Model Data/Ray Trace
- Data Import/Export and Model Manipulation



Understanding the API and Its Potential

What is an API?

- “Application Programming Interface”
- List of commands/functions you can give Revit
- The MENU to the restaurant’s kitchen



What is a DLL file?

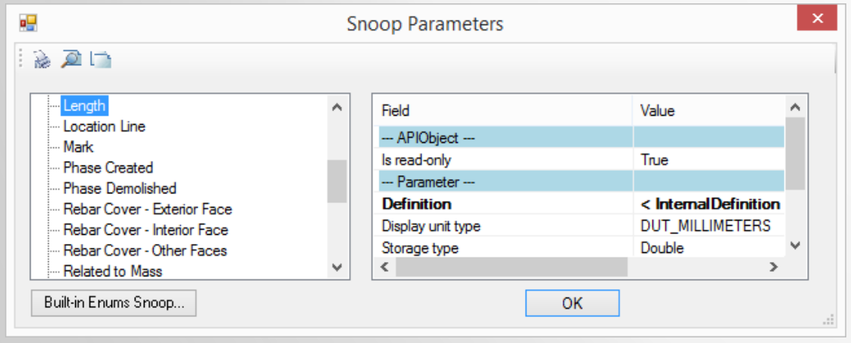
- “Dynamic Link Library”
- Code and Resources
  - Like an .exe, but without a set starting point
- Can be accessed by other programs if they know how to ask
  - Link to your programming project
  - Get access to the DLL’s functions
- Like a restaurant with kitchen and menu
- Reference another DLL in your project to add its restaurant and menu to your imaginary ‘food court’

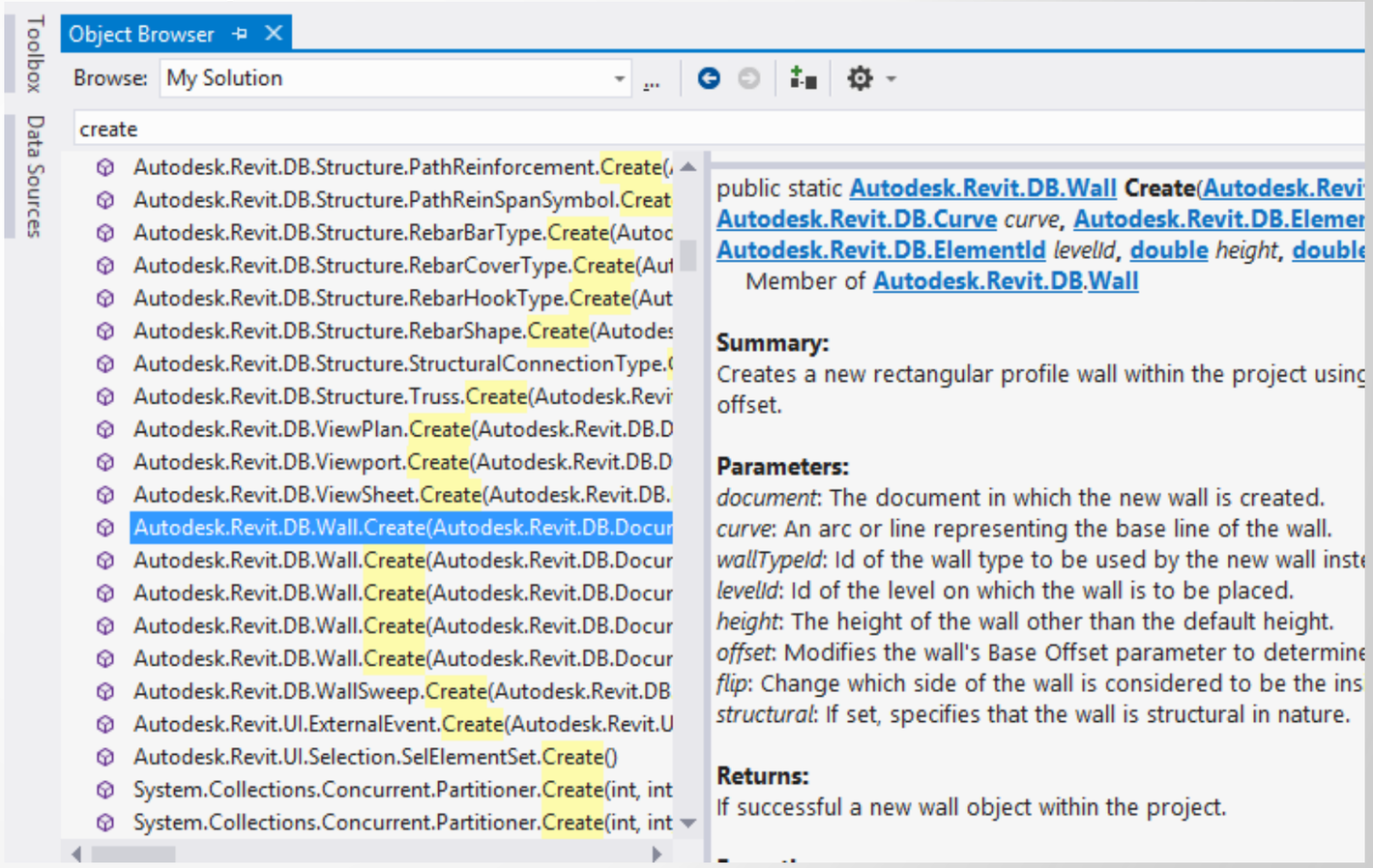
What is the .NET Framework?

- Part of the Microsoft Windows API
- Library of functions and preset dialogs

So what do I have access to manipulate in Revit?

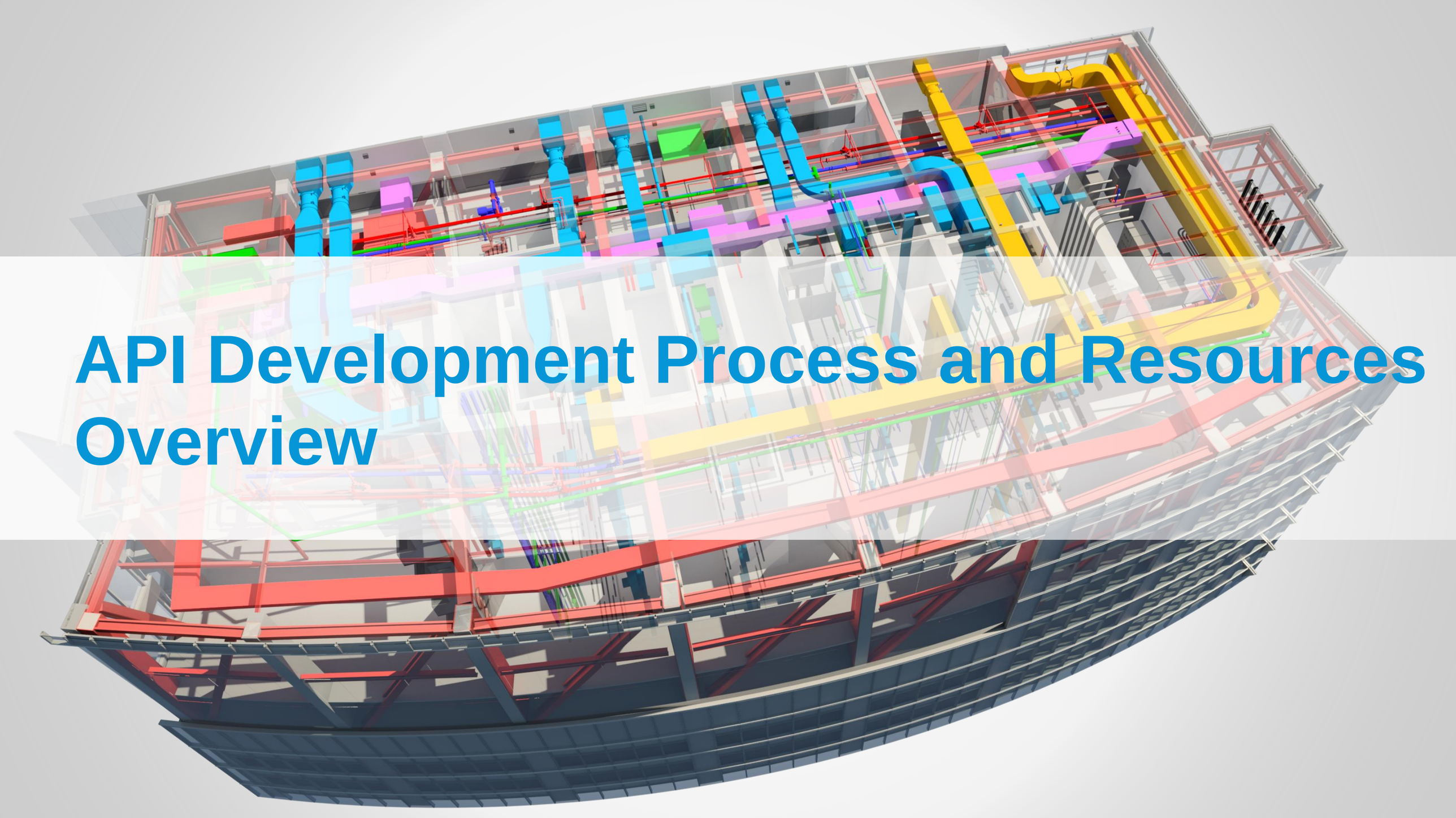
- Modify properties or parameter data
- Create elements, schedules, families or parameters
- Monitor changes/activity
- Events and dialogs
- Revit SDK
  - Instructions/Help file
  - Examples
  - Revit Lookup
    - Read-only parameter?
- Object Browser





Which programming language?

- You have access to the same kitchen even if the menu is in more than one language
- C#
- VB.NET
- Ruby
- Python
- IDE – Integrated Development Environment (Microsoft Visual Studio)



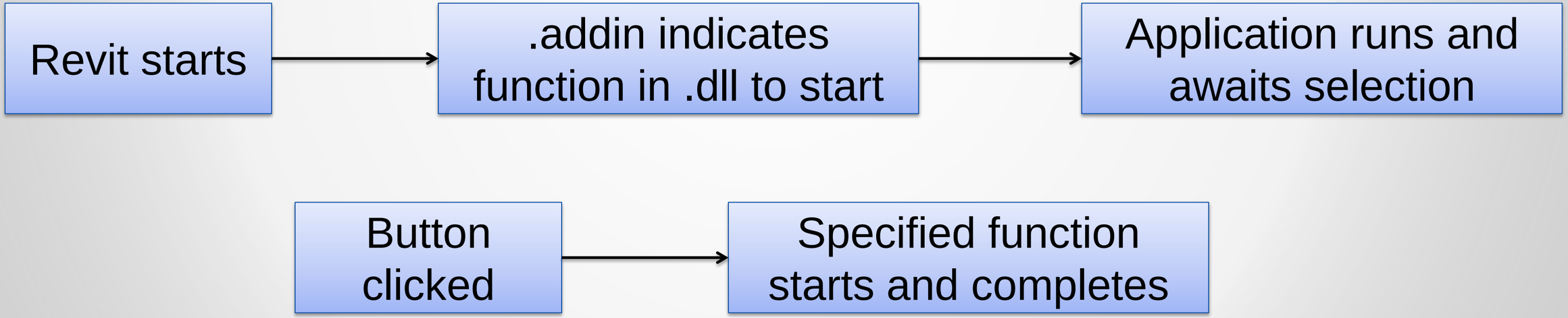
API Development Process and Resources Overview

The API and your Revit Project

- Macro
  - A command that you manually start and stop
  - Stored in the project or on a single computer
- External Command
  - Has a set start and stop point
  - Compiled in a DLL file (easy to share)
  - Example: A user clicks a button to start the command. The command performs its function, like create a 3D view from a selected region, then stops.
- External Application
  - Starts with Revit, ends with Revit
  - Continuously running
  - Compiled in a DLL file
  - Example: Ribbon toolbar starts when Revit starts and is visible until Revit closes

The API and your Revit Project

- Letting Revit know how to start your application or command
  - .addin text file
    - Revit looks for these on startup
    - Contains location of the .dll and name of function to start
  - Use API functions to create a new tab or new buttons in the Revit interface
    - See example in Revit SDK
    - Button click calls another function. Specify path to .dll and which function to start (can be in same or different .dll)



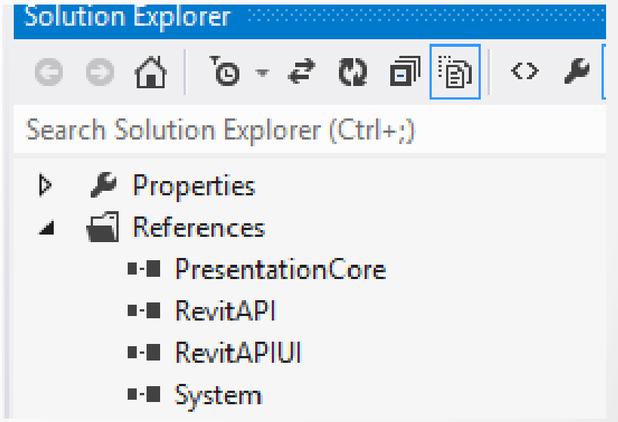
Getting Started

- Software to install
  - Visual Studio (Express versions are free)
  - Download Revit SDK (not required, but gives examples and documentation)

Getting Started: Understanding Visual Studio

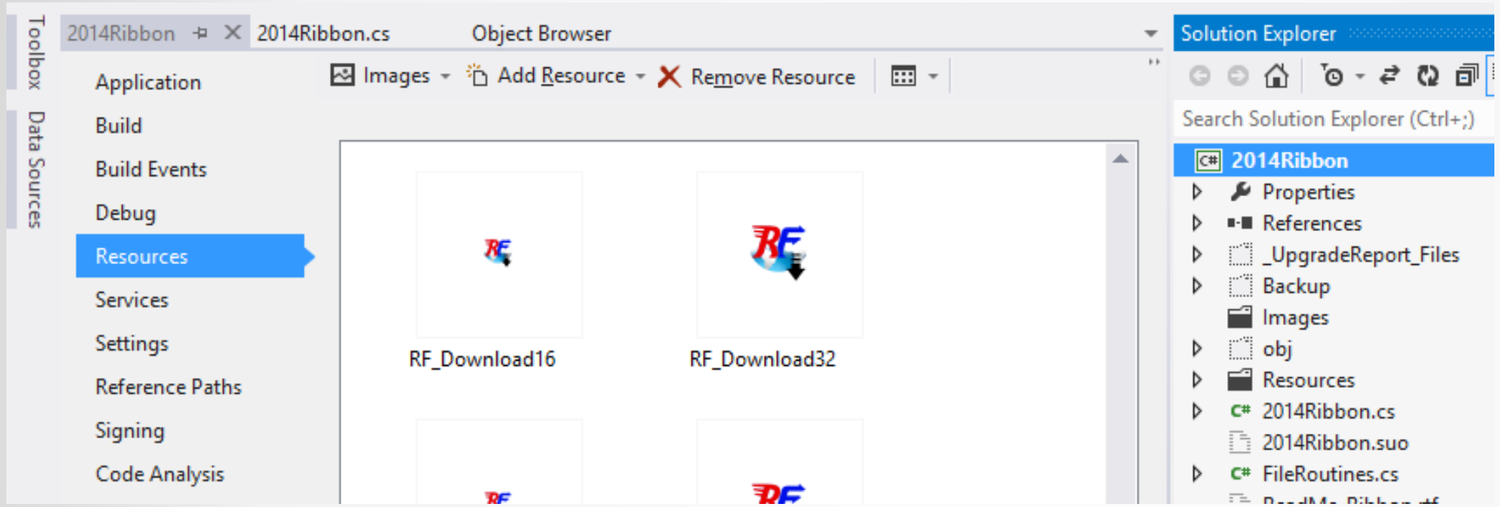
- Namespace, class name, function
- “Imports”(VB.NET)/”using”(C#) vs entire path
  - Autodesk.Revit.DB.View
- References
  - Get access to the restaurant’s menu
  - How to add (Usually don’t “Copy local”)
  - How to use

```
using System;
using Autodesk.Revit;
using System.IO;
```

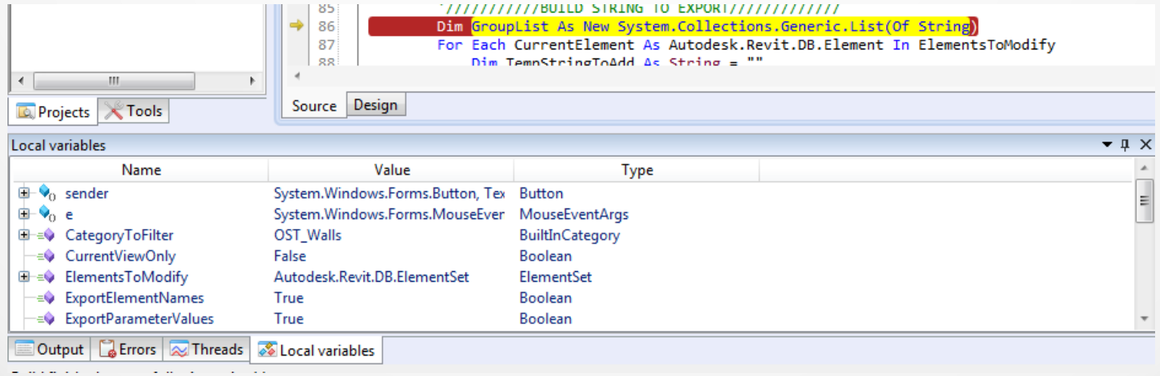


Getting Started: Understanding Visual Studio

- Resources

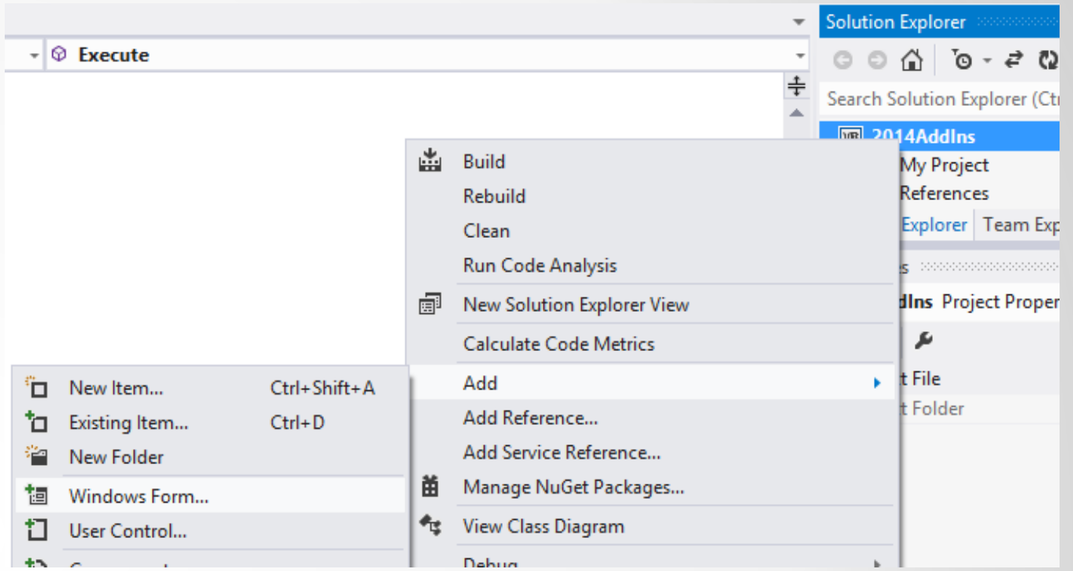


- Locals Window



Getting Started: Creating a simple form

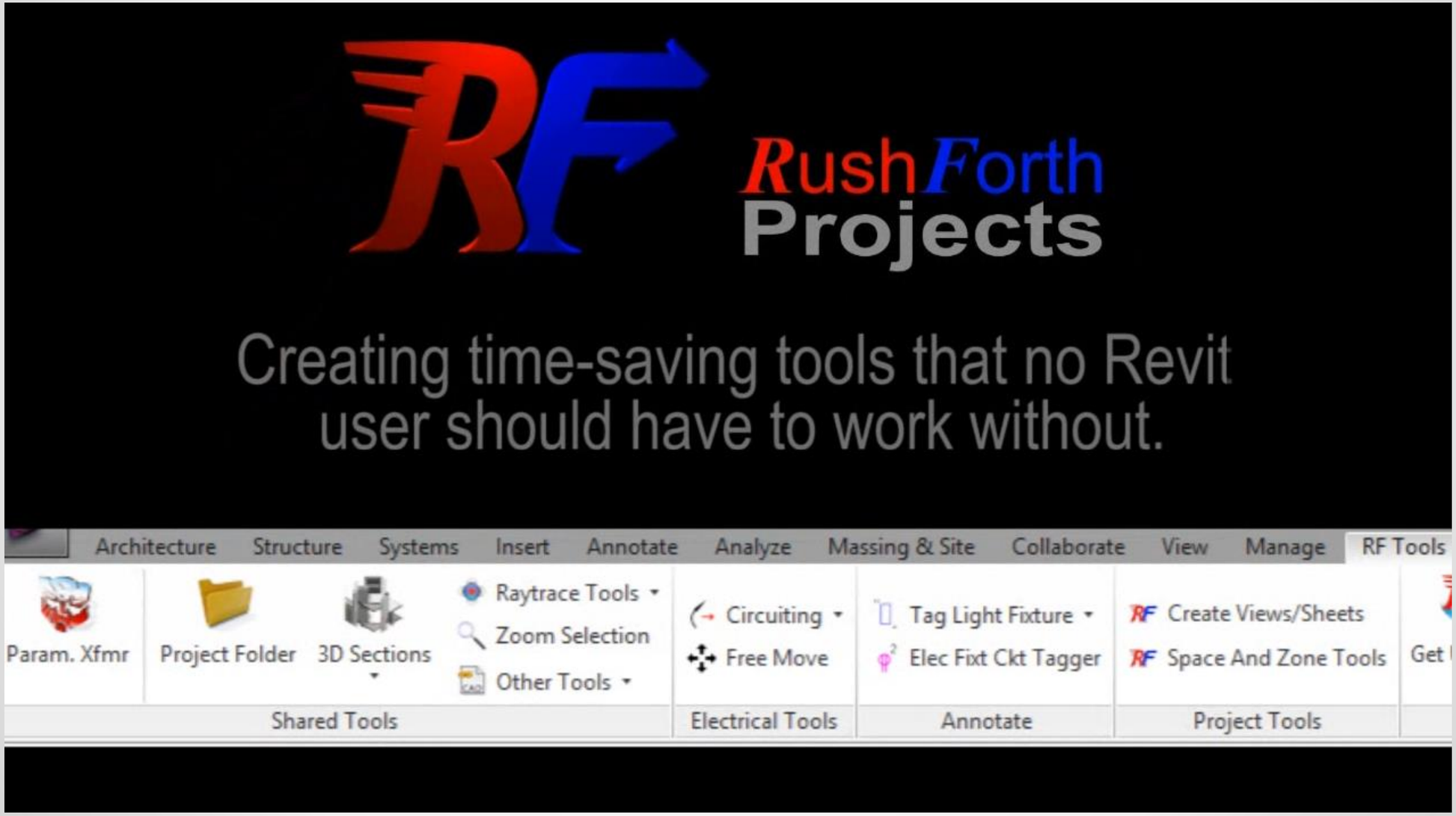
- Add new form to your solution



- In calling function
  - Dim NewFormToDisplay As New FormIJustCreated
  - NewFormToDisplay.ShowDialog(MyRevitData)
- In new form
  - Private RevitDataToUse As ExternalCommandData
  - Public Sub New(ByVal IncomingRevitData as ExternalCommandData)
  - RevitDataToUse = IncomingRevitData
  - InitializeComponent()
  - End Sub

Live Macro Demonstrations

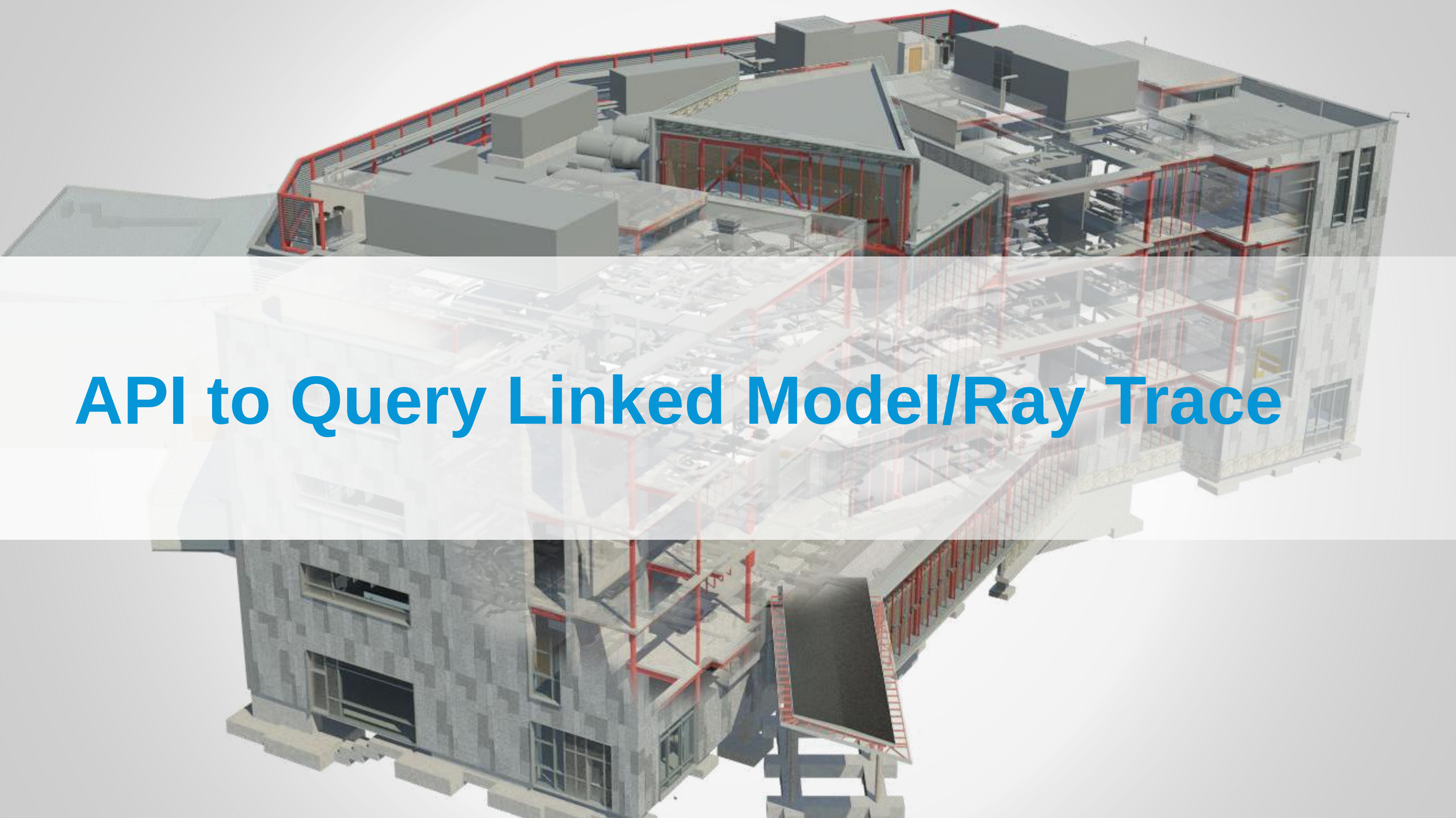
Example Applications Video



Example Applications Video

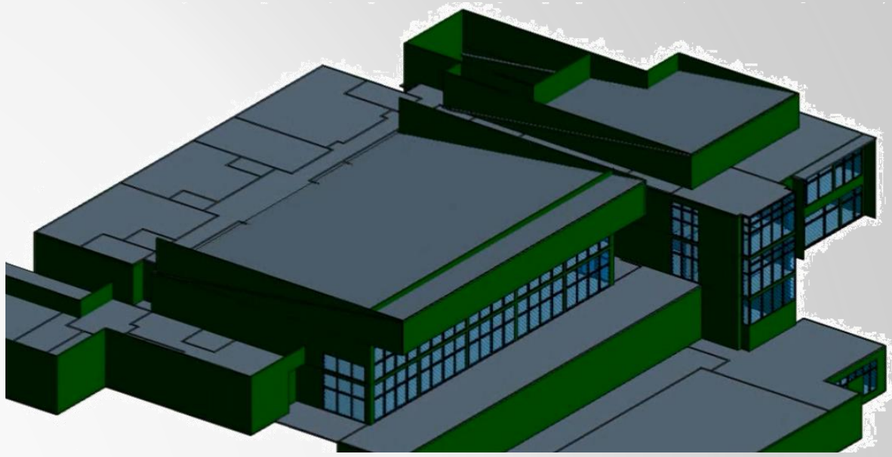
Sample Code

- WARNING!
  - Programming code is known to be an addictive substance.
  - Possible side effects include loss of sleep, loss of appetite, and loss of friends.



API to Query Linked Model/Ray Trace

Linked Model



- Find Linked Documents from Document List
  - Dim ListOfLinkedDocs As New Autodesk.Revit.DB.DocumentSet
  - Dim currentLink As Element
  - Dim LinkCollector As New FilteredElementCollector(MyDoc)
  - Dim LinkedElems As IList(Of Element) = LinkCollector.OfCategory(BuiltInCategory.OST_RvtLinks).OfClass(GetType(RevitLinkType)).ToElements()
  - For Each currentLink In LinkedElems
  - Dim linkType As RevitLinkType = TryCast(currentLink, RevitLinkType)
  - For Each CurrentDoc As Document In MyApp.Application.Documents
  - If CurrentDoc.Title.Equals(linkType.Name) Then
  - ListOfLinkedDocs.Insert(CurrentDoc)
  - Exit For
  - End If
  - Next
  - Next

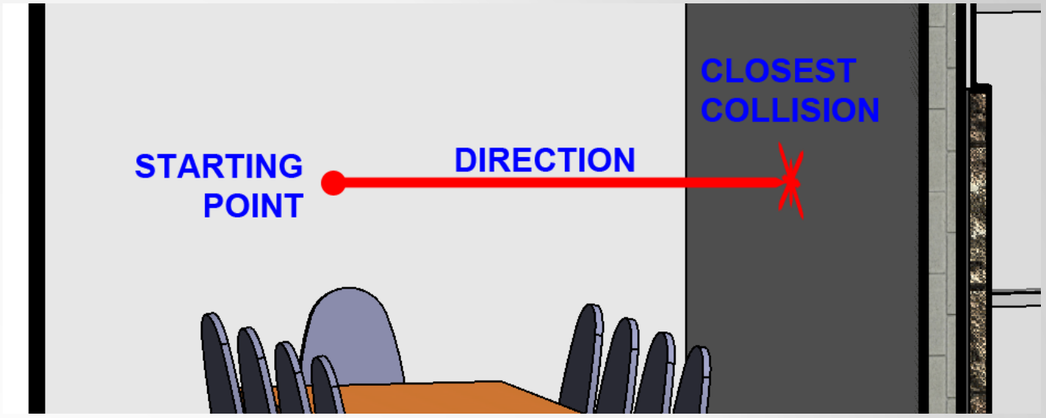
Color Key

Supporting code

Key lines of code

Comments

Ray Trace (Ray Casting)



- Ray trace to find location of intersection
  - '[SET UP A SOURCE POINT AND A DIRECTION]
  - Dim FoundCollisions As System.Collections.Ilist
  - FoundCollisions = MyLinkedDoc.FindReferencesWithContextByDirection(OriginalLocation, RayDirection, My3Dview)
  - 'NOTE: REFERENCEINTERSECTOR CLASS IN 2013 API
  - '[FILTER COLLISIONS BY CATEGORY, THEN FIND CLOSEST]
  - For j = 0 To FoundCollisions.Count - 1
    - Dim tempReferenceWithContext As ReferenceWithContext = TryCast(FoundCollisions .Item(j), ReferenceWithContext)
    - Dim TempReference As Reference= tempReferenceWithContext.GetReference
    - NewLocation = TempReference.GlobalPoint
    - …

Color Key

Supporting code

Key lines of code

Comments

Ray Trace (Ray Casting)

- Perimeter of room/space
  - Dim TempBoundaryOptions As New SpatialElementBoundaryOptions
  - myBoundaryArray = TempSpace.GetBoundarySegments(TempBoundaryOptions)
- Create element on wall vs linked model
  - If MyDoc.PathName = MyLinkedDoc.PathName Then ‘In this model
  - fi = MyDoc.Create.NewFamilyInstance(TempReference , NewLocation, MyReferenceDirection, MyFamilySymbol) ‘MyReferenceDirection = MyNormalRotated90
  - Else 'IF LINKED MODEL: CREATE REFERENCE PLANE
  - Dim MyReferencePlane As ReferencePlane
  - Dim PlaneFreeEnd As XYZ = NewLocation - MyReferenceDirection
  - Dim PlaneCutVector As XYZ = XYZ.BasisZ
  - MyReferencePlane = MyDoc.Create.NewReferencePlane(NewLocation, PlaneFreeEnd, PlaneCutVector, MyDoc.ActiveView)
  - fi = MyDoc.Create.NewFamilyInstance(MyReferencePlane.Reference, NewLocation, MyReferenceDirection, MyFamilySymbol)
  - MyDoc.Delete(MyReferencePlane.Id)
  - End if

Color Key

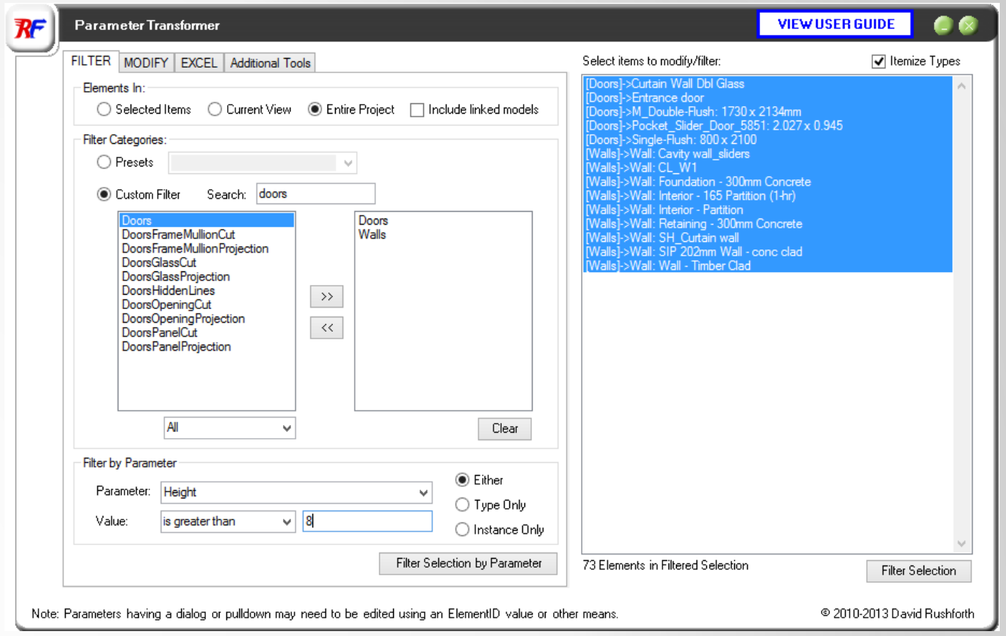
Supporting code

Key lines of code

Comments

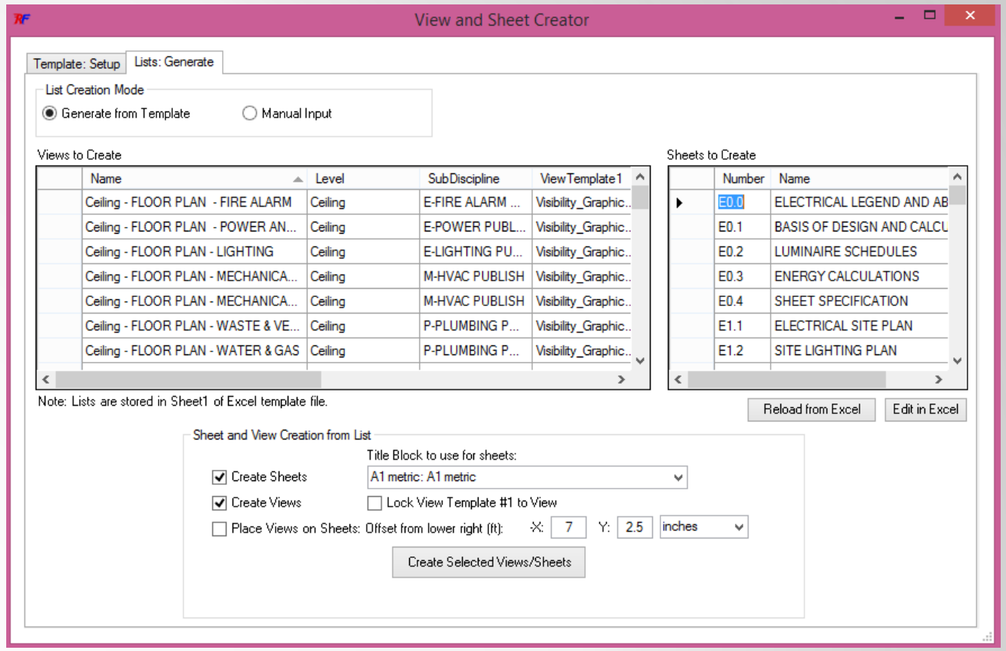
Model Manipulation and Data Import/Export

Parameter Value Manipulation



- Get and set parameter values (‘for each’ loop)
  - currentTypeParam.Set(ParamValueToSet)
  - Select Case ParameterToReturn.StorageType
  - ‘[Remember ConvertFromAPI and ConvertToAPI]
  - CurrentParamValue = ParameterToReturn.AsString()
- Element filters
  - Dim OtherElementCollector As New Autodesk.Revit.DB.FilteredElementCollector(MyDoc)
  - Dim MyOtherFilter As New ElementCategoryFilter(BuiltInCategory.OST_ElectricalCircuit)
  - Dim MyOrFilter As New ElementCategoryFilter(BuiltInCategory.OST_ElectricalInternalCircuits)
  - Dim NewPreCombinedFilter As New LogicalOrFilter(MyOtherFilter, MyOrFilter)
  - Dim MyInvertedFilter As New ElementClassFilter(GetType(Autodesk.Revit.DB.ElementType), True) 'don't include element types
  - Dim NewCombinedFilter As New LogicalAndFilter(NewPreCombinedFilter, MyInvertedFilter) ‘MULTI-STAGE FILTER EXAMPLE
  - FinalElementResult = OtherElementCollector.WherePasses(NewCombinedFilter).ToElements

Sheet and View Creation



- Create sheet
  - Dim CurrentSheetToCreate As ViewSheet = MyDoc.Create.NewViewSheet(TitleBlockForSheet)
    'ITERATE TITLBLOCKS PRIOR
  - CurrentSheetToCreate.Name = NewSheetToCreateName
  - CurrentSheetToCreate.SheetNumber = NewSheetToCreateNumber
- Create view
  - CurrentViewToCreate = ViewPlan.Create(MyDoc, CurrentViewType.Id, CurrentLevel.Id)
  - CurrentViewToCreate.Name = CurrentViewName 'ITERATE LEVELS PRIOR

Color Key

Supporting code

Key lines of code

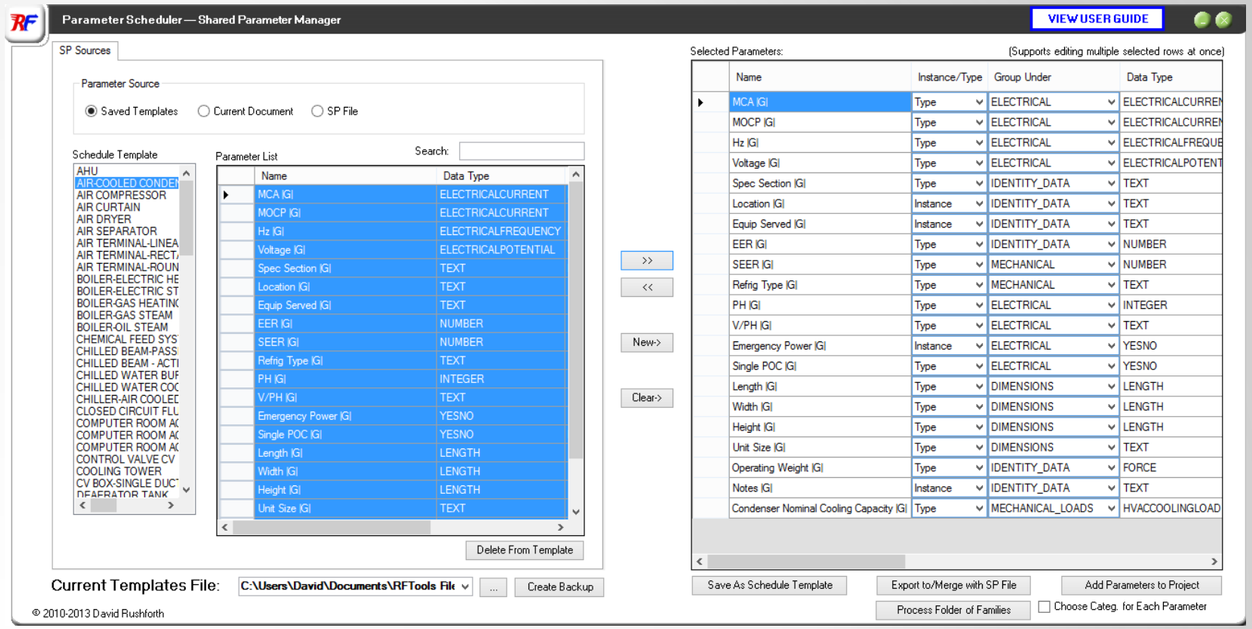
Comments

View Templates

- Apply View Template
  - If TypeOf elem Is Autodesk.Revit.DB.View Then ‘ITERATE VIEWS AND CHECK FOR NAME MATCHING TEMPLATE
  - Dim MyTempview As Autodesk.Revit.DB.View = TryCast(elem, Autodesk.Revit.DB.View)
  - If MyTempview.IsTemplate() = True and MyTempview.Name = TemplateName Then
  - CurrentViewTemplate = MyTempview
  - End If
  - End If
  - If LockToView = True Then
  - CurrentViewToProcess.ViewTemplateId = CurrentViewTemplate.Id
  - Else
  - CurrentViewToProcess.ApplyViewTemplateParameters(CurrentViewTemplate)
  - End If

Color Key
Supporting code
Key lines of code
Comments

Family Parameters



- Create parameter in the family editor
  - For Each group As DefinitionGroup In mysharedFile.Groups
  - For Each SharedParamDef As ExternalDefinition In group.Definitions
  - If SharedParamDef.Name = ParametersToCreate(i).ParameterToCreateName Then
  - ‘[DETERMINE GROUP AND WHETHER INSTANCE OR TYPE]
  - MyFamilyDoc.FamilyManager.AddParameter(SharedParamDef, MyParamGroup, IsInstanceParameter)
  - End If
  - Next
  - Next

Color Key

Supporting code

Key lines of code

Comments

Excel

- Excel connection (refer to 2010 class for Access, SQL, Excel, Text, etc.)
  - Dim strXlsFile As String
  - strXlsFile = "C:\Temp\test.xls"
  - Dim mExcelApplication As New Microsoft.Office.Interop.Excel.Application
  - Dim mExcelWorkbook As Microsoft.Office.Interop.Excel.Workbook = mExcelApplication.Workbooks.Open(strXlsFile)
  - Dim mExcelWorksheet As Microsoft.Office.Interop.Excel.Worksheet = mExcelWorkbook.Worksheets(1)
  - Dim readValue As String
  - **readValue = mExcelWorksheet.Range("A1").Value**
  - **mExcelWorksheet.Range("A2").Value = “New Value"**
  - mExcelWorkbook.Windows(1).Visible = True
  - mExcelWorkbook.Save()
  - mExcelWorkbook = Nothing
  - mExcelApplication.Quit()
  - mExcelApplication = Nothing

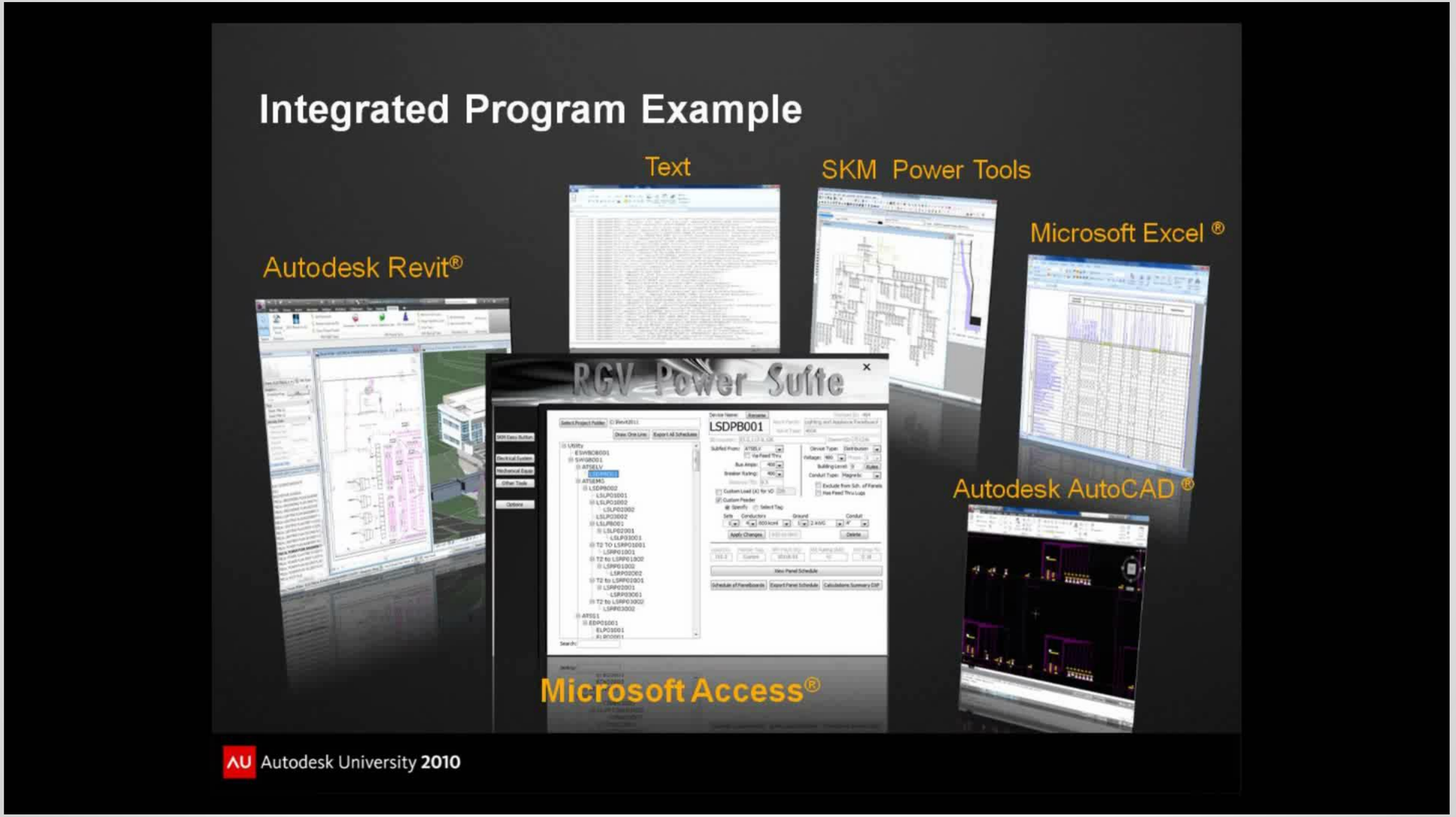
Color Key

Supporting code

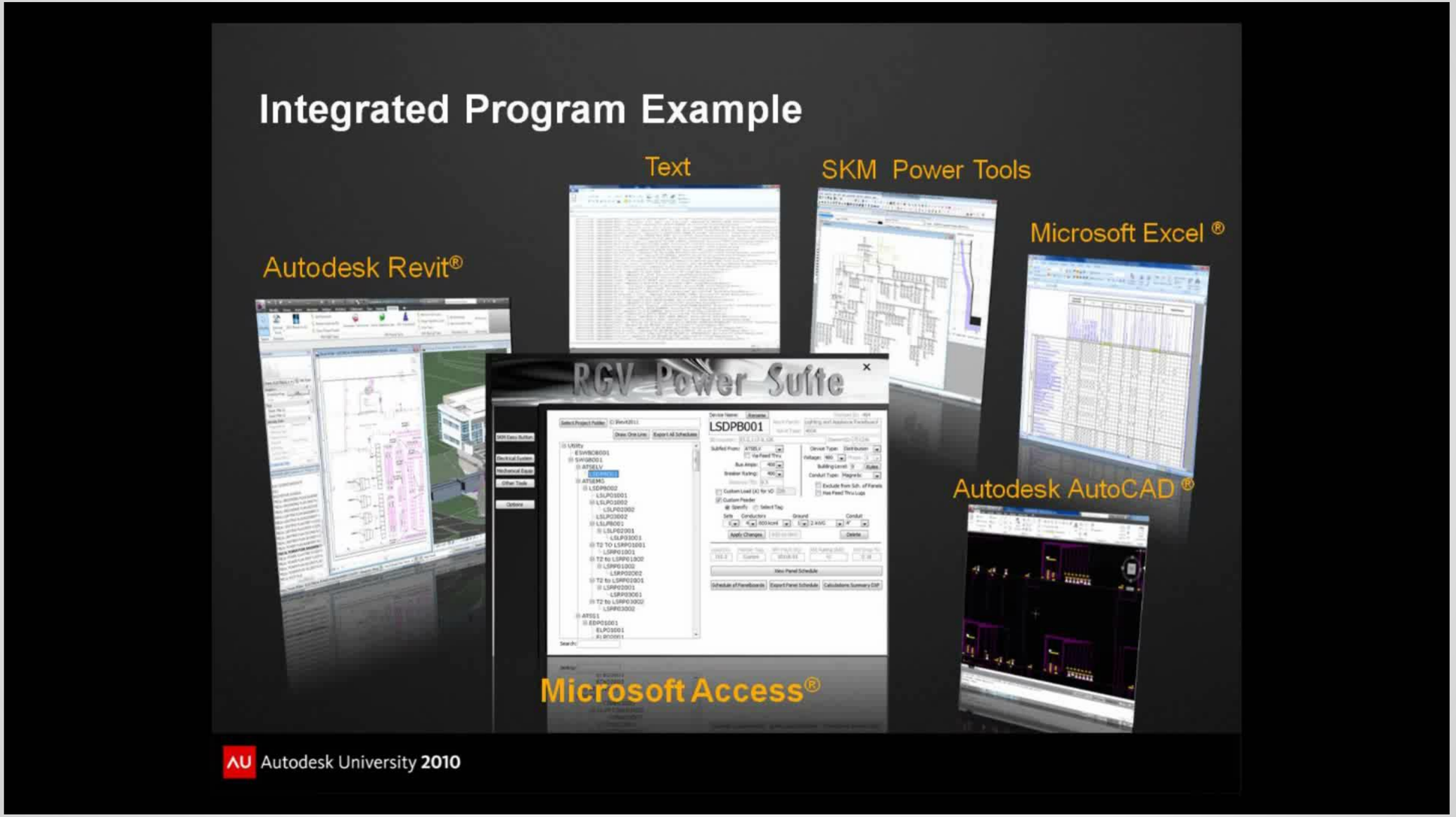
Key lines of code

Comments

Integration with External Calculations Video



Integration with External Calculations Video



Development Considerations

- Functionality
- Universality
- Aesthetics
- Deployment/Distribution/Licensing

Ideas for Future Development

- Ray trace element copier
- Automatic schedule creation from templates
- Anti-gravity ray trace
- Element linking tools
- Live connection to facilities metering data
- Device placement checks: Highlight items not at ceiling height/wall and choose to move to ceiling/wall

THANK YOU!