**Revit API Intro Labs**

**Lab 7 – Shared Parameter**

Created by A. Nagy, May 2011  
Updated by DevTech AEC WG  
Last modified: 3/9/2015

**<VB.NET>**VB.NET Version**</VB.NET>**

**Objective:** In this lab, we will learn how to create shared parameters. We’ll learn how to:

* Create shared parameter group
* Create shared parameter definition
* Bind it to a specific category – e.g. doors

The following is the breakdown of step by step instructions in this lab:

1. Define a New External Command
2. Create some helper functions
3. Create shared parameter for doors
4. Create per document parameter
5. Summary
6. **Define A New External Command**

We’ll add another external command to the current project.

* 1. Add a new file and define another external command to your project. Let’s name them as follows:
* File name: **7\_SharedParameter.vb (or .cs)**
* Command class name: **SharedParameter**

**Required Namespaces:**

In addition to the name spaces you have used, add the name space:

* System.IO

Also declare some constants that hold the properties of the parameter we are creating.

**<VB.NET>**  
 Const kSharedParamsGroupAPI As String = "API Parameters"

Const kSharedParamsDefFireRating As String = "API FireRating"

Const kSharedParamsPath As String = "C:\temp\SharedParams.txt"

Public Function Execute(ByVal commandData As ExternalCommandData, ByRef message As String, ByVal elements As ElementSet) \_

As Result \_

Implements IExternalCommand.Execute

Dim uidoc As UIDocument = commandData.Application.ActiveUIDocument

Dim app As Application = commandData.Application.Application

Dim doc As Document = uidoc.Document

**</VB.NET>**

1. **Create some helper functions**

First we need a function that gets back or if not already available then creates a new shared parameters file

**<VB.NET>**   
 Public Shared Function GetSharedParamsFile(ByVal app As Application) \_

As DefinitionFile

' Get current shared params file name

Dim sharedParamsFileName As String

Try

sharedParamsFileName = app.SharedParametersFilename

Catch ex As Exception

TaskDialog.Show("Get shared params file", \_

"No shared params file set:" + ex.Message)

Return Nothing

End Try

If 0 = sharedParamsFileName.Length Or \_

Not System.IO.File.Exists(sharedParamsFileName) Then

Dim stream As StreamWriter

stream = New StreamWriter(kSharedParamsPath)

stream.Close()

app.SharedParametersFilename = kSharedParamsPath

sharedParamsFileName = app.SharedParametersFilename

End If

' Get the current file object and return it

Dim sharedParametersFile As DefinitionFile

Try

sharedParametersFile = app.OpenSharedParameterFile()

Catch ex As Exception

TaskDialog.Show("Get shared params file", \_

"Cannnot open shared params file:" + ex.Message)

sharedParametersFile = Nothing

End Try

Return sharedParametersFile

End Function

**</VB.NET>**

We need a function that creates if a specific parameter groups already exists, and if not, then creates it and passes it back to us.

**<VB.NET>**

Public Shared Function GetOrCreateSharedParamsGroup( \_

ByVal sharedParametersFile As DefinitionFile, ByVal groupName As String) \_

As DefinitionGroup

Dim g As DefinitionGroup = sharedParametersFile.Groups.Item(groupName)

If g Is Nothing Then

Try

g = sharedParametersFile.Groups.Create(groupName)

Catch generatedExceptionName As Exception

g = Nothing

End Try

End If

Return g

End Function

**</VB.NET>**

Then we need a function that checks if a given parameter definition already exists, and if not, then creates it and passes it back to us

**<VB.NET>**

Public Shared Function GetOrCreateSharedParamsDefinition( \_

ByVal defGroup As DefinitionGroup, \_

ByVal defType As ParameterType, \_

ByVal defName As String, \_

ByVal visible As Boolean) As Definition

Dim definition As Definition = defGroup.Definitions.Item(defName)

If definition Is Nothing Then

Try

definition = defGroup.Definitions.Create(defName, defType, visible)

Catch generatedExceptionName As Exception

definition = Nothing

End Try

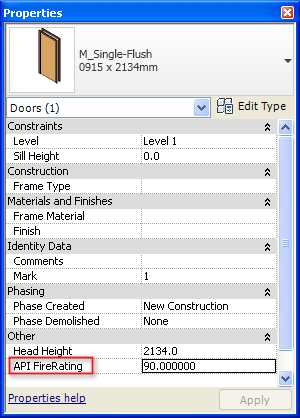
End If

Return definition

End Function

**</VB.NET>**

1. **Create shared parameter for doors**

****

All we need in order to create a shared parameter for doors is the shared parameters file, a parameter definition group, the parameter definition itself and then binding it to the doors category.

First get the shared parameters file and make sure that our parameters group exists

**<VB.NET>**

' Get the current shared params definition file

Dim sharedParamsFile As DefinitionFile = GetSharedParamsFile(app)

If sharedParamsFile Is Nothing Then

message = "Error getting the shared params file."

Return Result.Failed

End If

' Get or create the shared params group

Dim sharedParamsGroup As DefinitionGroup = \_

GetOrCreateSharedParamsGroup(sharedParamsFile, kSharedParamsGroupAPI)

If sharedParamsGroup Is Nothing Then

message = "Error getting the shared params group."

Return Result.Failed

End If

**</VB.NET>**

Then get back the category we need (doors) and create our shared parameter definition

**<VB.NET>**

Dim cat As Category = \_

doc.Settings.Categories.Item(BuiltInCategory.OST\_Doors)

' Visibility of the new parameter:

' Category.AllowsBoundParameters property indicates if a category can

' have shared or project parameters. If it is false, it may not be bound

' to shared parameters using the BindingMap.

' Please note that non-user-visible

' parameters can still be bound to these categories.

Dim visible As Boolean = cat.AllowsBoundParameters

' Get or create the shared params definition

Dim fireRatingParamDef As Definition = \_

GetOrCreateSharedParamsDefinition( \_

sharedParamsGroup, ParameterType.Number, \_

kSharedParamsDefFireRating, visible)

If fireRatingParamDef Is Nothing Then

message = "Error in creating shared parameter."

Return Result.Failed

End If

**</VB.NET>**

Once it’s created we can bind it to the doors category and we’re done.

**<VB.NET>**

' Create the category set for binding and add the category

' we are interested in, doors or walls or whatever:

Dim catSet As CategorySet = app.Create.NewCategorySet()

Try

catSet.Insert(cat)

Catch generatedExceptionName As Exception

message = String.Format( \_

"Error adding '{0}' category to parameters binding set.", cat.Name)

Return Result.Failed

End Try

' Bind the param

Try

Dim binding As Binding = app.Create.NewInstanceBinding(catSet)

' We could check if already bound, but looks like Insert will

' just ignore it in such case

doc.ParameterBindings.Insert(fireRatingParamDef, binding)

Catch ex As Exception

message = ex.Message

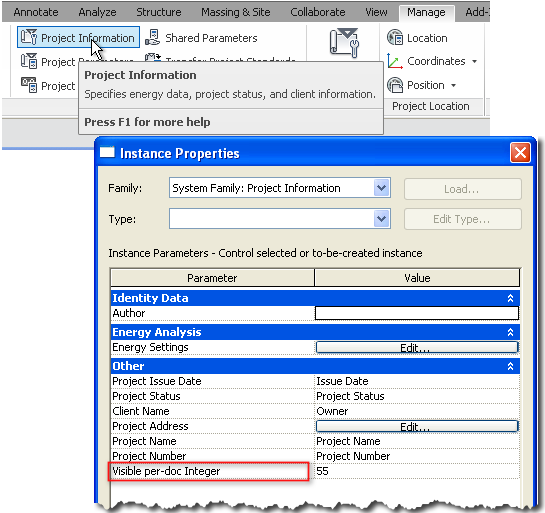
Return Result.Failed

End Try

**</VB.NET>**

Now, when selecting a door in the user interface we should be able to see the parameter listed in the properties palette – as was shown above.

1. **Create per document parameter**

****

Now add a new command that creates a visible and an invisible per document parameter. By per document parameter we mean a parameter that is added to an element that has only a single instance in any project, and that is the Project Information element. The only difference in this exercise from the previous one is that this time you need to bind the created parameter definition to the BuiltInCategory.OST\_ProjectInformation category.

Also, find the Project Information element in the project using FilteredElementCollector or get it directly through property Document.ProjectInformation and then set its newly created parameter’s value to something of your choice.

1. **Summary**

In this lab, we learned how to create shared parameters. We’ve learned how to:

* Create shared parameter group
* Create shared parameter definition
* Bind it to a specific category – e.g. doors

Autodesk Developer Network