Getting Started with Customizing

Dynamo for the Non-Programmers Using C#

Marcello Sgambelluri SE

Director of Advanced Technology

John A. Martin Structural Engineers

AUTODESK. UNIVERSITY

About the speaker

Marcello Sgambelluri

Marcello is internationally recognized at one of the top BIM leaders and contributors to the education and implementation of BIM technology in the building industry. Marcello continually speaks at Autodesk University since 2012. Marcello has received the 1st place speaker award 5 times out of the last 6 years between 2012 thru 2017. Marcello received his Bachelors and Master's degrees in Civil Engineering and he is also a licensed Civil and Structural Engineer. Marcello has also started a media empire that includes:

Simply Complex Blog Site -

http://therevitcomplex.blogspot.com/

Simply Complex YouTube

https://www.youtube.com/channel/UC7IkO1Bc4PhFKAHEArmQ0jw/videos

Simply Complex Podcast -

http://simplycomplex.sharedcoordinates.com/

AEC Complex Comic -

https://www.aeccomplexcomic.com/

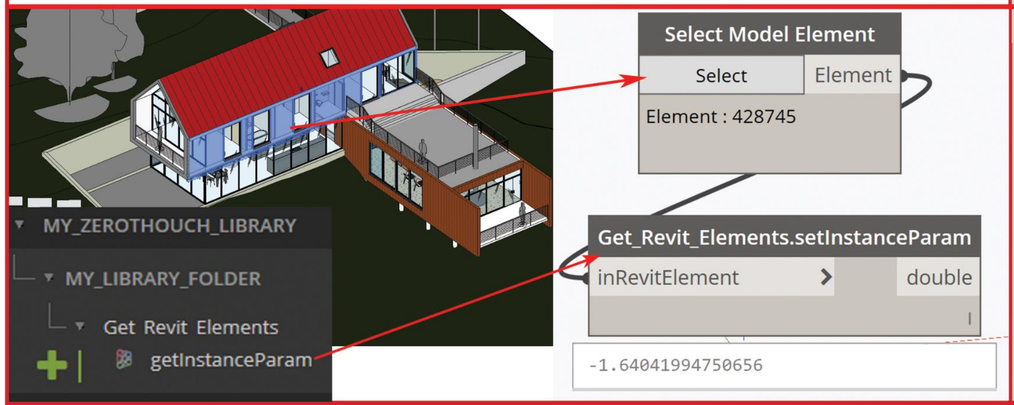
Dynamo Cheat sheet Anatomy

GET REVIT WALL BASE OFFSET PARAMETER VIA UNWRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Get_Revit_Elements
    {
        private Get_Revit_Elements()
        {
        }
        public static Double getInstanceParam
          (Revit.Elements.Element inRevitElement)
        {
            //unwrap
            Autodesk.Revit.DB.Element UnwrappedElement
            = inRevitElement.InternalElement;
            Double UnwrappedElementParameterValue
            = UnwrappedElement.get_Parameter
            (BuiltInParameter.WALL_BASE_OFFSET).AsDouble();
            return UnwrappedElementParameterValue;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

OPEN VISUAL STUDIO FOLDER “GET_REVIT_PARAMETER_START_START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE “GET_REVIT_PARAMETER_START_START.rvt”
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

Title Zone

Dynamo Code Zone

Geometry + Node Zone

Notes and Steps Zone

Pre Exercises

CREATE A NEW C# ZERO TOUCH LIBRARY IN DYNAMO PART 1

STEPS

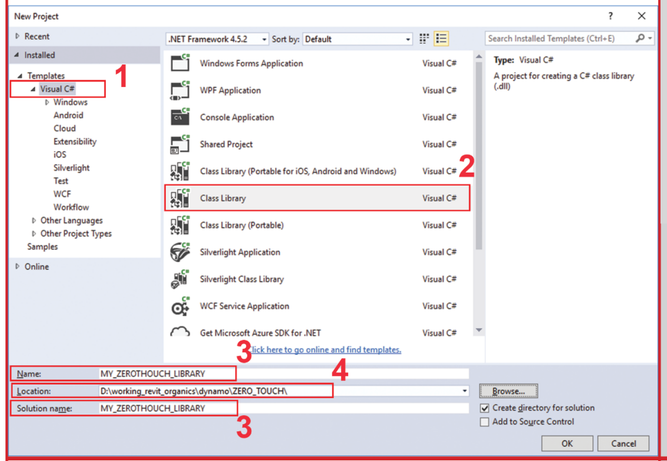
STEP 1:
OPEN VISUAL STUDIO AND START A NEW PROJECT SELECT C#

STEP 2:
SELECT THE CLASS LIBRARY. THE CLASS LIBRARY IS WHAT WILL "HOLD" THE DYNAMO NODES

STEP 3:
TYPE THE NAME OF THE PROJ-ECT. THE PROJECT NAME WILL BE THE NAME OF THE .DLL. THIS IS ALSO KNOWN IN ZERO TOUCH AS YOUR "LIBRARY". THIS "DLL" WILL BE THE FILE THAT CONTAINS ALL THE NODES.

STEP 4:
SAVE THE PROJECT "DLL" TO A FOLDER. THIS FOLDER WILL BE THE LOCATION WHERE ALL THE SOLUTION FILES WILL BE STORED SO CHOOSE THIS CAREFULLY.

VISUAL STUDIO START SCREEN

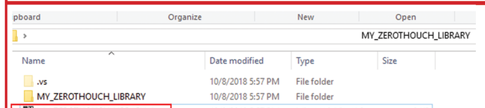


NOTES

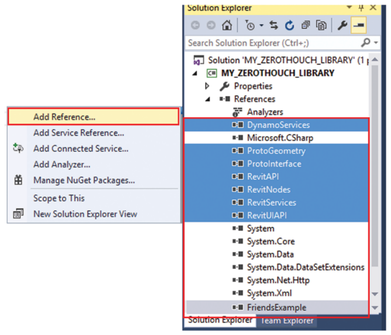
NOTES:
UNDERSTANDING THE STRUCTURE OF THE ZERO TOUCH PROJECT IS IMPORTANT AS IT WILL DETERMINE THE ORGANIZATION OF THE DYNAMO NODE LIBRARY. A LIBRARY IS A COLLECTION OF DYNAMO NODES.

CREATE A NEW C# ZERO TOUCH LIBRARY IN DYNAMO PART 2 REFERENCES

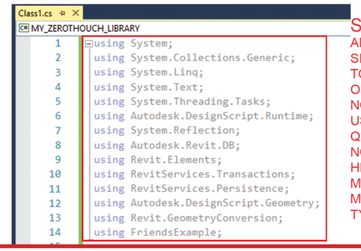
VISUAL STUDIO START SCREEN STEPS



STEP 1:
NAVIGATE TO THE FOLDER THAT THE PROJECT IS STORED AND CLICK ON THE SOLUTION FILE ".SLN



STEP 2:
RIGHT CLICK OVER THE REFERENCES IN THE SOLUTION EXPLORER IN VISUAL STUDIO AND ADD THE PROPER REFERENCES THAT ARE ".DLL" FORMAT. REFERENCES ARE ANY PRESET LIBRARY THAT THE CURRENT PRO-GRAM NEEDS TO USE TO RUN PROPERLY. KNOWING WHICH REFERENCES TO ADD IS NOT ALWAYS EASY. IN THIS CASE SINCE, THE REVIT DATABASE WILL BE ACCESSED THEN REVIT REFERENCES ASS SHOWN NEED TO BE ADDED ALSO THE "FRIENDSEXAMPLE" REFERENCES MUST BE ADDED AS WELL.
(ALSO SEE NOTES BELOW)



STEP 3:
ADD USING STATEMENTS BY SIMPLY TYPING THEM IN AT THE TOP OF THE SCREEN. TYPE THE ONES SHOWN.
NOTE:
USING STATMENTS ARE NOT RE-QUIRED TO CREATE A DYNAMO NODE HOWEVER THEY ARE VERY HELPFUL BECAUSE USING STATE-MENTS "SHORTEN" WHAT COM-MANDS THAT NEEDED TO BE TYPE IN THE CODE.

NOTES

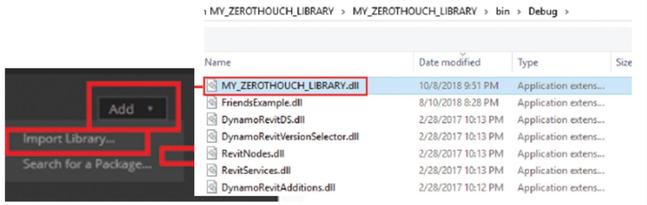
NOTES:
ADDING THE CORRECT REFERENCES AND FINDING THEM IS NOT ALWAYS VERY EASY. IT IS SOME-TIMES EASIER TO GET THE REFERENCES AND USING STATEMENTS FROM A PREVIOUS PROJECT. OPEN THE 'MY ZEROTOUCH LIBRARY' SAMPLE FILE TO START OR GET THESE REFERENCES.

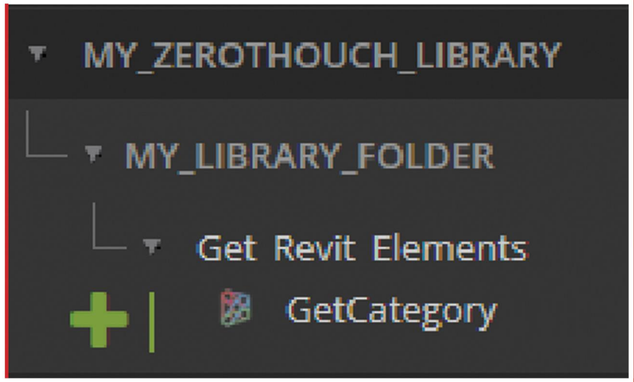
LOADING LIBRARY OR "DLL" INTO DYNAMO

LOADING DLL INTO DYNAMO

LOAD LIBRARY DLL

A. CLICK "ADD" + "IMPORT LIBRARY"
B. BROWSE TO FILE
"EXAMPLENAME\START\MY_ZEROTHOUCH_LIBRARY\bin\Debug\MY_ZEROTHOUCH_LIBRARY.DLL





NOTES

THIS IS ONLY ONE METHOD TO GET THE COMPILED "BUILD" DLL CREATED WITH ZERO TOUCH INTO DYNAMO. WITH THIS METHOD, AN NODE FROM THE LIBRARY "DLL" MUST BE PLACED ON THE CANVAS IN ORDER FOR THE DLL TO REMAIN. A "PACKAGE" COULD BE CREAT-ED AND LOADED INTO DYNAMO WITH THE DLL AND THE "ADD" DOES NOT NEED TO BE USED

Exercises

GET MARCELLO'S AGE ZERO TOUCH NODE W/ AND W/O INPUT

ZEROTOUCH CODE

```
using FriendsExample;

namespace MY_LIBRARY_FOLDER
{
    public class GETSIMPLEINFO
    {
        private GETSIMPLEINFO()
        {
        }

        public static int marcelloAgeNodeNoInput()
        {
            int marcelloAge = Friends.Marcello.Age;
            return marcelloAge;
        }

        public static int marcelloAgeNodeInput(int fudgefactor = 0)
        {
            int marcelloAge = Friends.Marcello.Age - fudgefactor;
            return marcelloAge;
        }
    }
}
```

DYNAMOLIBRARY + ZT NODES

STEPS

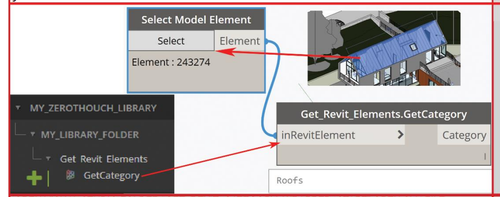
NOTES:
OPEN VISUAL STUDIO FOLDER "GET_MARCELLO_AGE_ZT_START" CLICK ON THE SLN FILE.
TYPE CODE AS SHOWN FOR BOTH WITH AND WITHOUT INPUT PORT CASE. BUILD THE SOLUTION.
OPEN DYNAMO VIA REVIT AND LOAD THE DLL FROM BIN DIR. OPEN "FINAL" FOLDER IF NEEDED.

GET REVIT CATEGORY FROM WRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{

    public class Get_Revit_Elements
    {

        private Get_Revit_Elements()
        {
        }
        //Revit.Elements means wrapped Revit Element
        //Autodesk.Revit.DB means Unwrapped Revit Element

         public static Revit.Elements.Category
         GetCategory(Revit.Elements.Element inRevitElement)
        {
            Revit.Elements.Category CATFROMINPUT
            = inRevitElement.GetCategory;
            return CATFROMINPUT;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

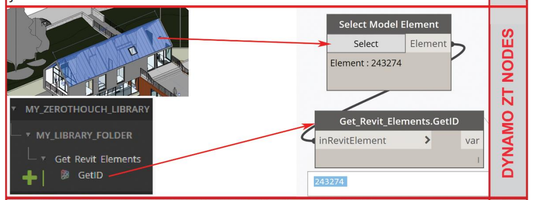
OPEN VISUAL STUDIO FOLDER "GET_REVIT_CATEGORY_WRAPPED_START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE "GET_REVIT_CATEGORY_WRAPPED_START.rvt"
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED.

GET REVIT ELEMENT ID FROM UNWRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Get_Revit_Elements
    {
        private Get_Revit_Elements()
        {
        }
        public static Autodesk.Revit.DB.ElementId GetID
            (Revit.Elements.Element inRevitElement)
        {
            //unwrap
            //Revit.Elements means wrapped Revit Element
            //Autodesk.Revit.DB means Unwrapped Revit Element
            Autodesk.Revit.DB.Element UnwrappedElement
                = inRevitElement.InternalElement;
            Autodesk.Revit.DB.ElementId UnwrappedElementID
                = UnwrappedElement.Id;

            return UnwrappedElementID;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

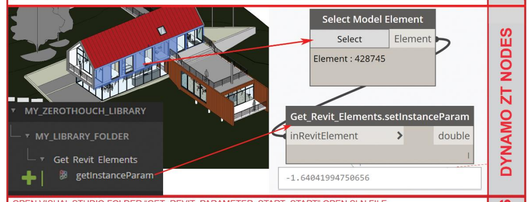
OPEN VISUAL STUDIO FOLDER "GET_REVIT_ID_UNWRAPPED_START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE "GET_REVIT_IS_UNWRAPPED_START.rvt"
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED.

GET REVIT WALL BASE OFFSET PARAMETER VIA UNWRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Get_Revit_Elements
    {
        private Get_Revit_Elements()
        {
        }
        public static Double getInstanceParam
          (Revit.Elements.Element inRevitElement)
        {
            //unwrap
            Autodesk.Revit.DB.Element UnwrappedElement
            = inRevitElement.InternalElement;
            Double UnwrappedElementParameterValue
            = UnwrappedElement.get_Parameter
            (BuiltInParameter.WALL_BASE_OFFSET).AsDouble();
            return UnwrappedElementParameterValue;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

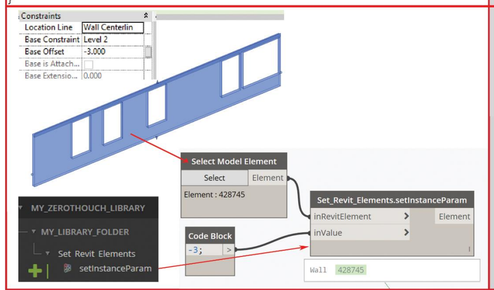
OPEN VISUAL STUDIO FOLDER "GET_REVIT_PARAMETER_START_START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE "GET_REVIT_PARAMETER_START_START.rvt"
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

SET REVIT WALL BASE OFFSET PARAMETER VIA UNWRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Set_Revit_Elements
    {
        private Set_Revit_Elements()
        {
        }

        public static Revit.Elements.Element setInstanceParam
            (Revit.Elements.Element inRevitElement, Double inValue)
        {
            //unwrap
            Autodesk.Revit.DB.Element UnwrappedElement = inRevitElement.InternalElement;
            //Start Transaction because changing the Revit DataBase
            //Get Current Document (standard stuff)
            Autodesk.Revit.DB.Document doc = DocumentManager.Instance.CurrentDBDocument;
            TransactionManager.Instance.EnsureInTransaction(doc);
            UnwrappedElement.get_Parameter(BuiltInParameter.WALL_BASE_OFFSET).Set(inValue);
            //End Transaction because changing the Revit DataBase
            TransactionManager.Instance.TransactionTaskDone();
            return inRevitElement;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

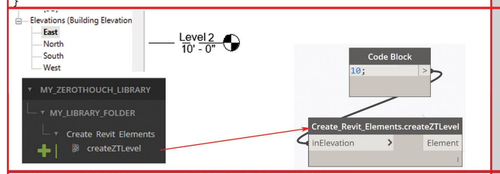
OPEN VISUAL STUDIO FOLDER "SET_REVIT_PARAMETER_START_START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE "SET_REVIT_PARAMETER_START_START.rvt"
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

CREATE REVIT LEVELS VIA ZT VIA TRANSACTIONS AND WRAPPING

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Create_Revit_Elements
    {
        private Create_Revit_Elements()
        {
        }

        public static Revit.Elements.Element
            createZTSomething(double inElevation)
        {
            //Get Current Document (standard stuff) +
            //Start Transaction because changing the Revit DataBase
            Autodesk.Revit.DB.Document doc =
                DocumentManager.Instance.CurrentDBDocument;
            TransactionManager.Instance.EnsureInTransaction(doc);

            //End Transaction because changing the Revit DataBase
            Autodesk.Revit.DB.Level newLevel =
                Autodesk.Revit.DB.Level.Create(doc, inElevation);
            TransactionManager.Instance.TransactionTaskDone();

            //wrapit! since Revit element generated in Code and sent to Dynamo
            Revit.Elements.Element wrappedLevel = newLevel.ToDSType(false);

            return wrappedLevel;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

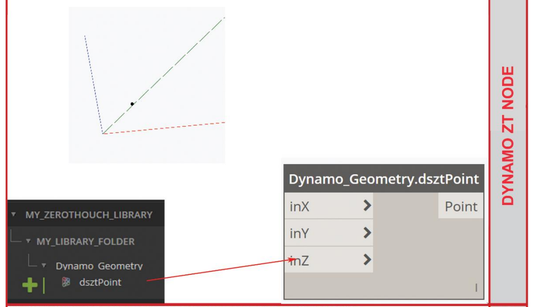
OPEN VISUAL STUDIO FOLDER "CREATE_REVIT_LEVEL...START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE "CREATE_REVIT_LEVEL...START.rvt"
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

CREATE A DYNAMO POINT VIA DEFAULT X,Y,Z =1

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Dynamo_Geometry
    {
        private Dynamo_Geometry()
        {
        }
        public static Autodesk.DesignScript.Geometry.Point dsztPoint
            (double inX = 1, double inY = 1, double inZ = 1)
        {
            //Autodesk.DesignScript.Geometry.Point  is a Dynamo point
            Autodesk.DesignScript.Geometry.Point dPt =
                Autodesk.DesignScript.Geometry.Point.ByCoordinates(inX, inY, inZ);
            return dPt;

        }
    }
}
```

DYNAMO ZT NODE



STEPS

OPEN VISUAL STUDIO FOLDER "CREATE_DYNAMO_POINT_START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.

OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

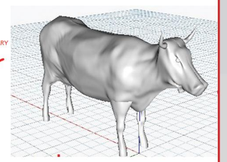
CREATE A COW IN DYNAMO WITH MULTIPLE OUTPUT PORTS

DYNAMO GEOMETRY AND CODE NOTES

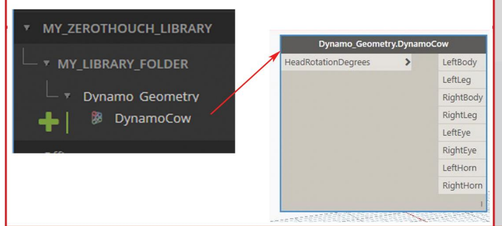
```
USE A MULTIRETURN TAG IF THERE IS
MORE THAN 1 OUTPUT PORT
//SETTING UP MULTIPLE RETURNS TAG
[MultiReturn(new[] { "LeftBody", "LeftLeg", "RightBody", "RightLeg",
    "LeftEye", "RightEye", "LeftHorn", "RightHorn" })]
.....
//RETURNING
Dictionary<string, object> OutInfo =
    new Dictionary<string, object>
{
    {"LeftBody",Surfacetorso},
    {"LeftLeg",Surfaceleg},
    {"RightBody",MirrorSurfacetorso},
    {"RightLeg",MirrorSurfaceleg},
    {"LeftEye",RotationEyeball},
    {"LeftHorn",RotationHorn},
    {"RightEye",MirrorRotationEyeball},
    {"RightHorn",MirrorRotateHorn},
};
return OutInfo;
```

CREATE A DICTONARY TO STORE MULTIPLE OUTPUTS

RETURN THE DICTIONARY



DYNAMO NODES



STEPS & NOTES

OPEN VISUAL STUDIO FOLDER "CREATE_DYNAMO_COW_FINAL" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

Additional Exercises

CREATE A NEW C# ZERO TOUCH LIBRARY IN DYNAMO PART 1

STEPS

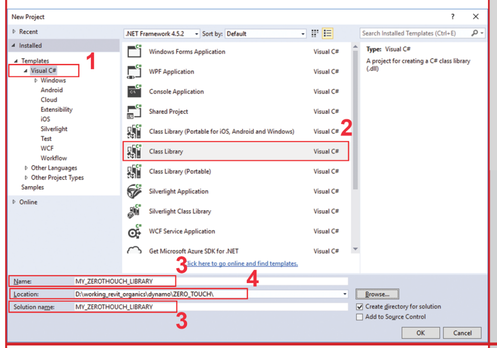
STEP 1:
OPEN VISUAL STUDIO AND START A NEW PROJECT
SELECT C#

STEP 2:
SELECT THE CLASS LIBRARY. THE CLASS LIBRARY IS WHAT WILL "HOLD" THE DYNAMO NODES

STEP 3:
TYPE THE NAME OF THE PROJECT. THE PROJECT NAME WILL BE THE NAME OF THE .DLL. THIS IS ALSO KNOWN IN ZERO TOUCH AS YOUR "LIBRARY". THIS "DLL" WILL BE THE FILE THAT CONTAINS ALL THE NODES.

STEP 4:
SAVE THE PROJECT "DLL" TO A FOLDER. THIS FOLDER WILL BE THE LOCATION WHERE ALL THE SOLUTION FILES WILL BE STORED SO CHOOSE THIS CAREFULLY.

VISUAL STUDIO START SCREEN

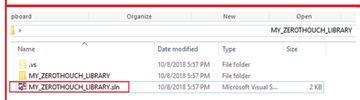


NOTES

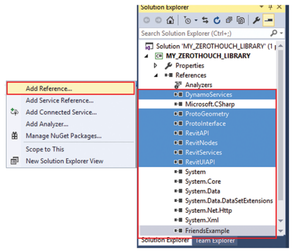
NOTES:
UNDERSTANDING THE STRUCTURE OF THE ZERO TOUCH PROJECT IS IMPORTANT AS IT WILL DETERMINE THE ORGANIZATION OF THE DYNAMO NODE LIBRARY. A LIBRARY IS A COLLECTION OF DYNAMO NODES.

CREATE A NEW C# ZERO TOUCH LIBRARY IN DYNAMO PART 2 REFERENCES

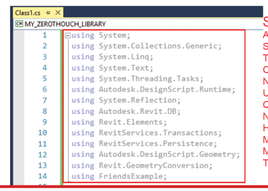
VISUAL STUDIO START SCREEN STEPS



STEP 1:
NAVIGATE TO THE FOLDER THAT THE PROJECT IS STORED AND CLICK ON THE SOLUTION FILE ".SLN



STEP 2:
RIGHT CLICK OVER THE REFERENCES IN THE SOLUTION EXPLORER IN VISUAL STUDIO AND ADD THE PROPER REFERENCES THAT ARE ".DLL" FORMAT. REFERENCES ARE ANY PRESET LIBRARY THAT THE CURRENT PROGRAM NEEDS TO USE TO RUN PROPERLY. KNOWING WHICH REFERENCES TO ADD IS NOT ALWAYS EASY. IN THIS CASE SINCE, THE REVIT DATABASE WILL BE ACCESSED THEN REVIT REFERENCES ASS SHOWN NEED TO BE ADDED ALSO THE "FRIENDSEXAMPLE" REFERENCES MUST BE ADDED AS WELL.
(ALSO SEE NOTES BELOW)



STEP 3:
ADD USING STATEMENTS BY SIMPLY TYPING THEM IN AT THE TOP OF THE SCREEN. TYPE THE ONES SHOWN.
NOTE:
USING STATMENTS ARE NOT REQUIRED TO CREATE A DYNAMO NODE HOWEVER THEY ARE VERY HELPFUL BECAUSE USING STATEMENTS "SHORTEN" WHAT COMMANDS THAT NEEDED TO BE TYPE IN THE CODE.

NOTES

NOTES:
ADDING THE CORRECT REFERENCES AND FINDING THEM IS NOT ALWAYS VERY EASY. IT IS SOMETIMES EASIER TO GET THE REFERENCES AND USING STATEMENTS FROM A PREVIOUS PROJECT. OPEN THE "MY ZEROTOUCH LIBRARY" SAMPLE FILE TO START OR GET THESE REFERENCES.

ANATOMY OF A SIMPLE ZERO TOUCH NODE NO INPUT

DESCRIPTIONS

1:
THE NAME OF THE ZERO TOUCH LIBRARY OR HIGHESET LEVEL FOLDER NAME (SHOWN IN THE DYNAMO BROSWER) IS THE NAME OF THE DLL

2:
THE NAME OF THE ZERO TOUCH LIBRARY SUB-FOLDER IS THE NAME OF THE NAMESPACE IN THE ZERO TOUCH CODE

3:
THE NAME OF THE ZERO TOUCH LIBRARY SUB-SUB-FOLDER IS THE NAME OF THE CLASS IN THE ZERO TOUCH CODE

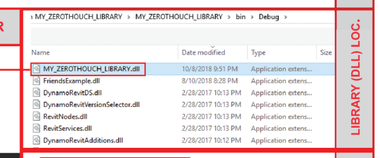
4:
THE NAME OF THE ZERO TOUCH NODE IS THE NAME OF THE METHOD IN THE ZERO TOUCH CODE

5:
THE NAME OF THE INPUT PORT IS THE NAME OF THE PARAMETER IN THE METHOD (BLANK IN THIS EXAMPLE)

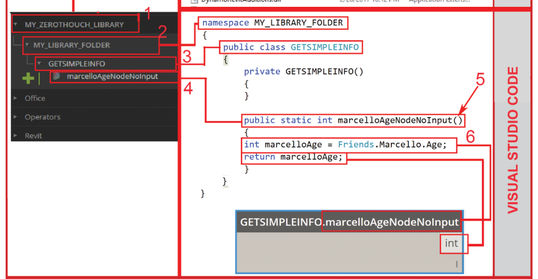
6:
THE NAME OF THE OUTPUT PORT IS THE NAME OF THE METHOD TYPE RETURNED (INT IN THIS EXAMPLE)
ALSO SEE NOTES BELOW.

LIBRARY (DLL) LOC.

DYNAMO BROWSER



VISUAL STUDIO CODE



NOTES

NOTES:
UNDERSTANDING THE STRUCTURE OF THE ZERO TOUCH PROJECT IS IMPORTANT AS IT WILL DETERMINE THE ORGANIZATION OF THE DYNAMO NODE LIBRARY. ALSO IF MULTIPLE OUTPUT IS REQUIRED THEN THE METHOD WILL NEED TO OUTPUT A "DICTIONARY" OF MULTIPLE VALUES.

ANATOMY OF A SIMPLE ZERO TOUCH NODE W/ INPUT

DESCRIPTIONS

1:
THE NAME OF THE ZERO TOUCH LIBRARY OR HIGHESET LEVEL FOLDER NAME (SHOWN IN THE DYNAMO BROSWER) IS THE NAME OF THE DLL

2:
THE NAME OF THE ZERO TOUCH LIBRARY SUB-FOLDER IS THE NAME OF THE NAMESPACE IN THE ZERO TOUCH CODE

3:
THE NAME OF THE ZERO TOUCH LIBRARY SUB-SUB-FOLDER IS THE NAME OF THE CLASS IN THE ZERO TOUCH CODE

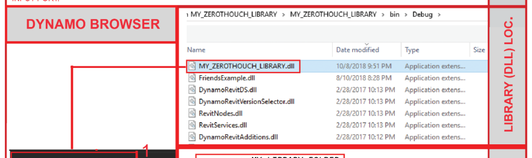
4:
THE NAME OF THE ZERO TOUCH NODE IS THE NAME OF THE METHOD IN THE ZERO TOUCH CODE

5:
THE NAME OF THE INPUT PORT IS THE NAME OF THE PARAMETER. ALSO IF PARAMETER (A METHOD IS "=" TO A VALUE, THAT VALUE IS THE DEFAULT VALUE IN THE INPUT PORT.

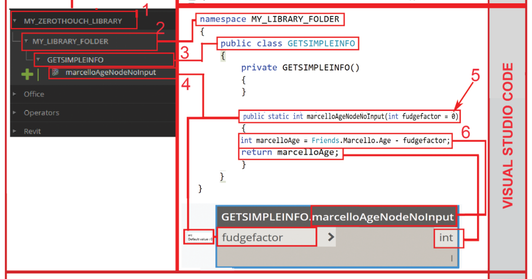
6:
THE NAME OF THE OUTPUT PORT IS THE NAME OF THE METHOD TYPE RETURNED (INT IN THIS EXAMPLE)
ALSO SEE NOTES BELOW.

LIBRARY (DLL) LOC.

DYNAMO BROWSER



VISUAL STUDIO CODE



NOTES

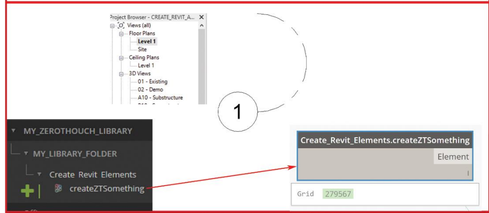
NOTES:
UNDERSTANDING THE STRUCTURE OF THE ZERO TOUCH PROJECT IS IMPORTANT AS IT WILL DETERMINE THE ORGANIZATION OF THE DYNAMO NODE LIBRARY. ALSO IF MULTIPLE OUTPUT IS REQUIRED THEN THE METHOD WILL NEED TO OUTPUT A "DICTIONARY" OF MULTIPLE VALUES.

CREATE REVIT ARC GRID VIA ZT VIA TRANSACTIONS AND WRAPPING

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Create_Revit_Elements
    {
        private Create_Revit_Elements()
        {
        }

        public static Revit.Elements.Element createZTSomething()
        {
            //Get Current Document (standard stuff) +
            //Start Transaction because changing the Revit DataBase
            Autodesk.Revit.DB.Document doc =
                DocumentManager.Instance.CurrentDBDocument;
            TransactionManager.Instance.EnsureInTransaction(doc);

            Autodesk.Revit.DB.XYZ p1 = new Autodesk.Revit.DB.XYZ();
            Autodesk.Revit.DB.XYZ p2 = new Autodesk.Revit.DB.XYZ(0, 10, 0);
            Autodesk.Revit.DB.XYZ p3 = new Autodesk.Revit.DB.XYZ(5, 5, 0);
            Autodesk.Revit.DB.Arc revitArc = Autodesk.Revit.DB.Arc.Create(p1, p2, p3);
            Autodesk.Revit.DB.Grid newGrid = Autodesk.Revit.DB.Grid.Create(doc, revitArc);
            //End Transaction because changing the Revit DataBase
                    TransactionManager.Instance.TransactionTaskDone();
            //wrapit
            Revit.Elements.Element wrappedGrid = newGrid.ToDSType(false);
            return wrappedGrid;
        }
    }
}
```

DYNAMO ZT NODES

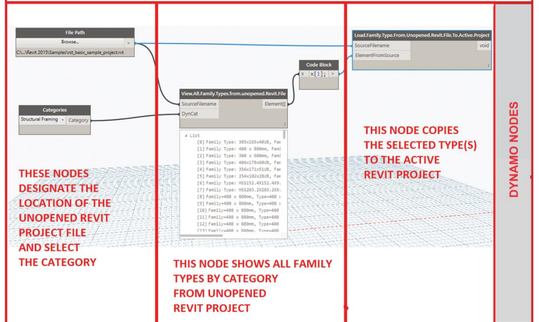


STEPS

OPEN VISUAL STUDIO FOLDER "CREATE_REVIT_ARC_GRID..START" OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE "CREATE_REVIT_ARC_GRID..START.rvt"
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN "FINAL" FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

COPY TYPES FROM UNOPENED REVIT PROJECT TO ACTIVE REVIT PROJECT USING ZT DYNAMO NODES

DYNAMO NODES



CODE AND REVIT GEOMETRY

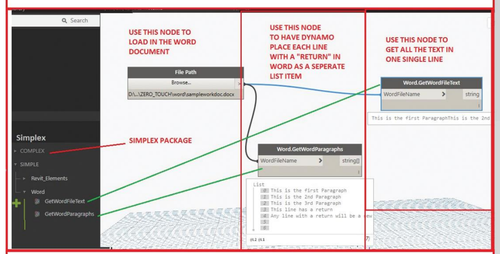
```
public static List<Revit.Elements.Element> FindAllOfCategoryInFile(string SourceFilename, Revit.Elements.Category DynCat)
{
    Autodesk.Revit.DB.Document doc = DocumentManager.Instance.CurrentDBDocument;
    Autodesk.Revit.UI.UIApplication uiapp = DocumentManager.Instance.CurrentUIApplication;
    //TransactionManager.Instance.EnsureInTransaction(doc);

    Autodesk.Revit.DB.Document familyDoc = uiapp.Application.OpenDocumentFile(SourceFilename);

    Autodesk.Revit.DB.Category FoundCat = GetRevitCategory(DynCat);

    BuiltInCategory BIcat = (BuiltInCategory)DynCat.Id;

    var Elements = new FilteredElementCollector(FamilyDoc)
    .OfCategory(BIcat);

    List<Revit.Elements.Element> AllElements = new List<Revit.Elements.Element>();

    foreach (Autodesk.Revit.DB.Element TempEle in Elements)
    {
        Revit.Elements.Element WrappedEle = TempEle.ToDSType(false);
        AllElements.Add(WrappedEle);

    }

    return AllElements;
}
```

STEPS & NOTES

STEP 1: OPEN THE REVIT SAMPLE FILE "rst_advanced_sample_project.rvt"
STEP 2: START AND NEW DYNAMO FILE AND ADD NODE SHOWN
STEP 3: SELECT THE SAMPLE FILE " "rst_advanced_sample_project.rvt" FROM THE "PATH" NODE
NOTE: CUSTOM NODES COULD BE FOUND IN THE "SIMPLEX" DYNAMO PACKAGE

READ TEXT FROM MS WORD DOCUMENTS (.DOCX)

DYNAMO NODES



MS WORD

This is the first Paragraph¶
This is the 2nd Paragraph¶
This is the 3rd Paragraph¶
This line has a return¶
Any line with a return will be a new item in a list¶
¶

Note: This workflow was intended for non-complex formated word documents

STEPS & NOTES

STEP 1: OPEN THE REVIT SAMPLE FILE "rst_advanced_sample_project.rvt"
STEP 2: START AND NEW DYNAMO FILE AND ADD NODE SHOWN
STEP 3: SELECT THE SAMPLE FILE "READ_TEXT_FROM_WORD.DOCX" FROM THE "PATH" NODE
NOTE: CUSTOM NODES COULD BE FOUND IN THE "SIMPLEX" DYNAMO PACKAGE

GET ALL FRAME NAMES IN ETABS USING DYNAMO USING ETABS API AND ZERO TOUCH C#

ZERO TOUCH C# API ETABS CODE

```
public static ETABS2016.cSapModel GetNameList()
{
    SapModelObject; UseExistingInstance();

    ETABS2016.cOAPI myETABSObject = null;
    //attach to a running instance of ETABS

    myETABSObject =
    //create SapModel object
                = myETABSObject.SapModel;
    int NumberNames = 0;
    string[] MyName = new string[0];
        SapModelObject.cFrameObj.GetNameList(ref NumberNames, ref MyName);
    Dictionary<string, object> OutVars = new Dictionary<string, object>();

    return SapModelObject
}
```

CREATES A DICTIONARY FOR OUTPUT PORTS
STARTS THE METHOD NOTE: NAME OF METHOD IS NAME OF NODE
(ZERO TOUCH IS A SPECIFIC FORMAT)
(sets initial value place holder values)
METHOD GETS FRAME NAMES
OUTPUTS VALUES TO OUTPUT PORTS
RETURNS VARIABLES

ETABS API MANUAL

cFrameObj.GetNameList Method

Retrieves the names of all defined frame objects.

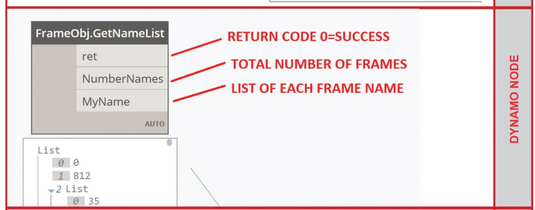
Namespace: ETABS2016
Assembly: ETABS2016 (in ETABS2016.dll) Version: 16.0.0.0 (16.0.0.0)

C#

```
int GetNameList(
    ref int NumberNames,
    ref string[] MyName
)
```

INPUT IF THERE
STRING NAME
OUTPUT

DYNAMO NODE

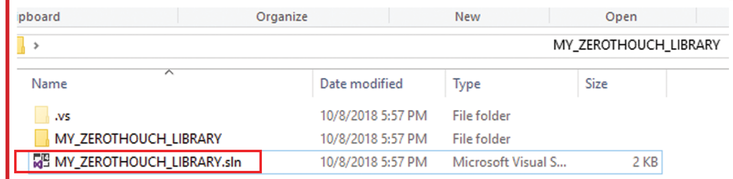


STEPS & NOTES

STEP 1: OPEN ETABS AND OPEN DYNAMO FOR REVIT OPEN ETABS FILE "ETABS_SAMPLE_START"
STEP 2: COMPILE THE CODE AND ADD THE .DLL TO DYNAMO
NOTE: "REF" IN THE ETABS API MANUAL MEANS OUTPUT REST MEAN INPUT

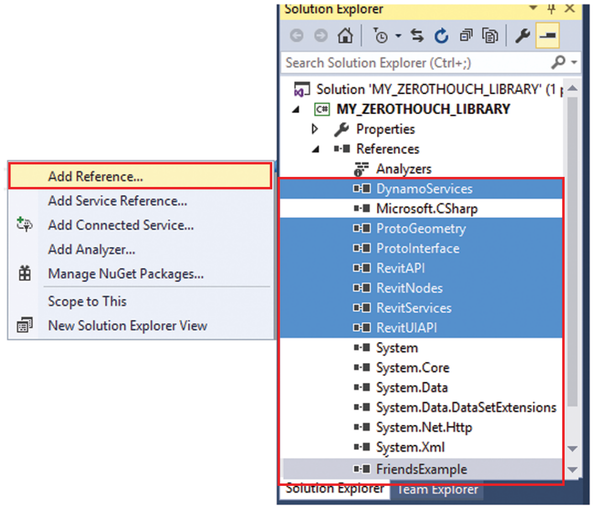
Pre-Exercise

CREATE A NEW C# ZERO TOUCH LIBRARY IN DYNAMO PART 2 REFERENCES



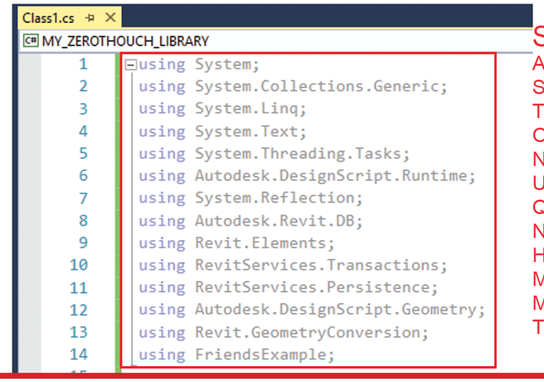
STEP 1:

NAVIGATE TO THE FOLDER THAT THE PROJECT IS STORED AND CLICK ON THE SOLUTION FILE ".SLN



STEP 2:

RIGHT CLICK OVER THE REFERENCES IN THE SOLUTION EXPLORER IN VISUAL STUDIO AND ADD THE PROPER REFERENCES THAT ARE ".DLL" FORMAT. REFERENCES ARE ANY PRESET LIBRARY THAT THE CURRENT PRO-GRAM NEEDS TO USE TO RUN PROPERLY. KNOWING WHICH REFERENCES TO ADD IS NOT ALWAYS EASY. IN THIS CASE SINCE, THE REVIT DATABASE WILL BE ACCESSED THEN REVIT REFERENCES ASS SHOWN NEED TO BE ADDED ALSO THE "FRIENDSEXAMPLE" REFERENCES MUST BE ADDED AS WELL.
(ALSO SEE NOTES BELOW)



STEP 3:

ADD USING STATEMENTS BY SIMPLY TYPING THEM IN AT THE TOP OF THE SCREEN. TYPE THE ONES SHOWN.
NOTE:
USING STATMENTS ARE NOT RE-QUIRED TO CREATE A DYNAMO NODE HOWEVER THEY ARE VERY HELPFUL BECAUSE USING STATE-MENTS "SHORTEN" WHAT COM-MANDS THAT NEEDED TO BE TYPE IN THE CODE.

VISUAL STUDIO START SCREEN STEPS

NOTES:
ADDING THE CORRECT REFERENCES AND FINDING THEM IS NOT ALWAYS VERY EASY. IT IS SOME-TIMES EASIER TO GET THE REFERENCES AND USING STATEMENTS FROM A PREVIOUS PROJECT. OPEN THE "MY ZEROTOUCH LIBRARY' SAMPLE FILE TO START OR GET THESE REFERENCES.

NOTES

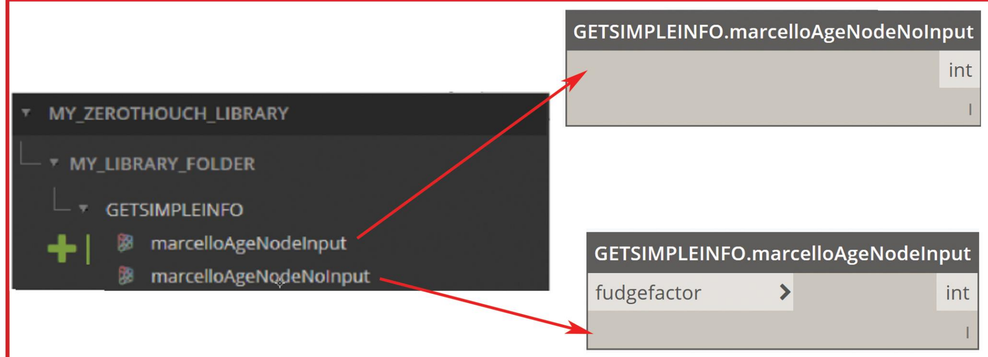
Exercise 1

GET MARCELLO'S AGE
ZERO TOUCH NODE W/ AND W/O INPUT

ZEROTOUCH CODE

```
using FriendsExample;

namespace MY_LIBRARY_FOLDER
{
    public class GETSIMPLEINFO
    {
        private GETSIMPLEINFO()
        {
        }

        public static int marcelloAgeNodeNoInput()
        {
            int marcelloAge = Friends.Marcello.Age;
            return marcelloAge;
        }

        public static int marcelloAgeNodeInput(int fudgefactor = 0)
        {
            int marcelloAge = Friends.Marcello.Age - fudgefactor;
            return marcelloAge;
        }
    }
}
```

DYNAMOLIBRARY + ZT NODES



STEPS

NOTES:
OPEN VISUAL STUDIO FOLDER "GET_MARCELLO_AGE_ZT_START" CLICK ON THE SLN FILE.
TYPE CODE AS SHOWN FOR BOTH WITH AND WITHOUT INPUT PORT CASE. BUILD THE SOLUTION.
OPEN DYNAMO VIA REVIT AND LOAD THE DLL FROM BIN DIR. OPEN "FINAL" FOLDER IF NEEDED.

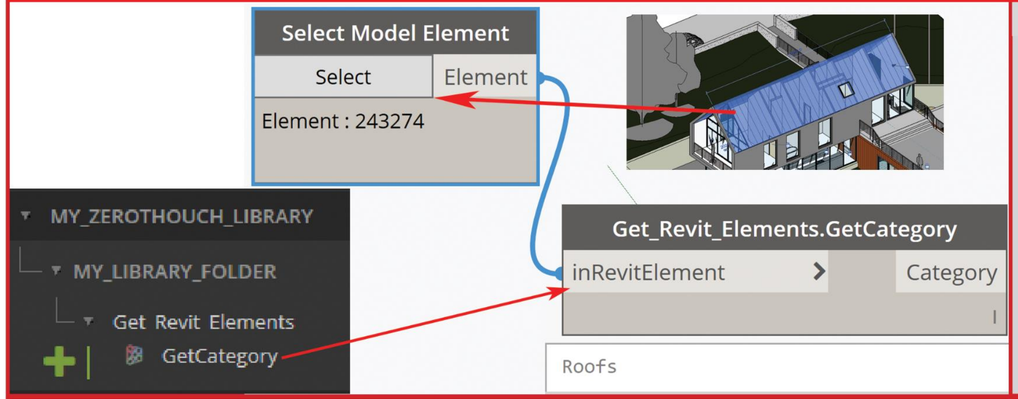
Exercise 2

GET REVIT CATEGORY FROM WRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{

    public class Get_Revit_Elements
    {

        private Get_Revit_Elements()
        {
        }
        //Revit.Elements means wrapped Revit Element
        //Autodesk.Revit.DB means Unwrapped Revit Element

         public static Revit.Elements.Category
         GetCategory(Revit.Elements.Element inRevitElement)
        {
            Revit.Elements.Category CATFROMINPUT
            = inRevitElement.GetCategory;
            return CATFROMINPUT;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

OPEN VISUAL STUDIO FOLDER “GET_REVIT_CATEGORY_WRAPPED_START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE “GET_REVIT_CATEGORY_WRAPPED_START.rvt”
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED.

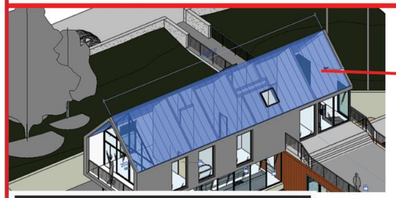
Exercise 3

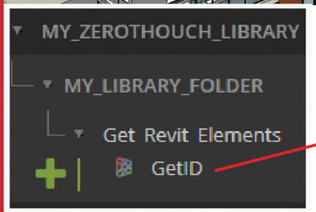
GET REVIT ELEMENT ID FROM UNWRAPPED DYNAMO ELEMENT

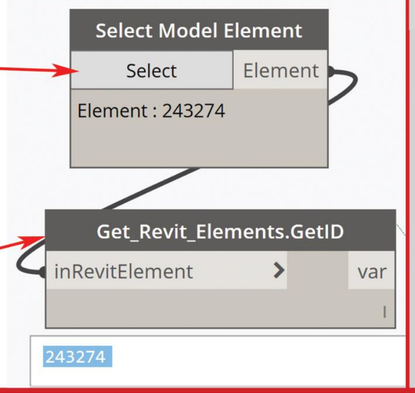
ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{

    public class Get_Revit_Elements
    {

        private Get_Revit_Elements()
        {
        }
        public static Autodesk.Revit.DB.ElementId GetID
            (Revit.Elements.Element inRevitElement)
        {
            //unwrap
            //Revit.Elements means wrapped Revit Element
            //Autodesk.Revit.DB means Unwrapped Revit Element
            Autodesk.Revit.DB.Element UnwrappedElement
                = inRevitElement.InternalElement;
            Autodesk.Revit.DB.ElementId UnwrappedElementID
                = UnwrappedElement.Id;

            return UnwrappedElementID;
        }
    }
}
```

DYNAMO ZT NODES







STEPS

OPEN VISUAL STUDIO FOLDER “GET_REVIT_ID_UNWRAPPED_START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE “GET_REVIT_IS_UNWRAPPED_START.rvt”
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED.

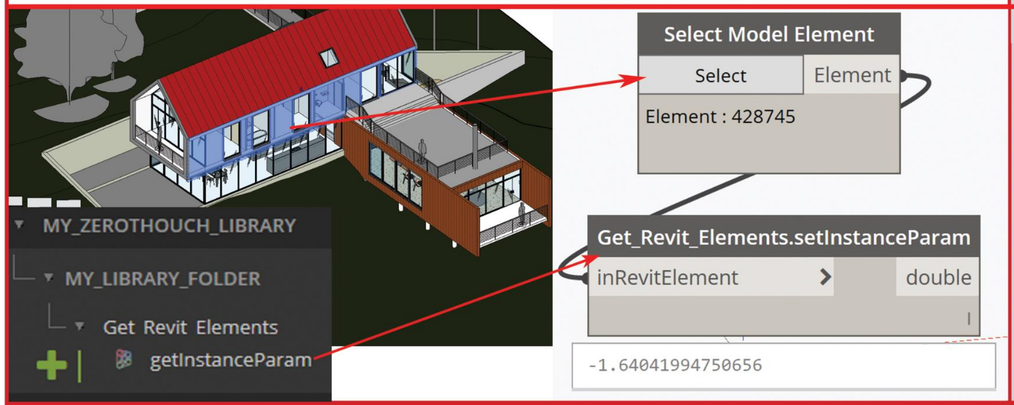
Exercise 4

GET REVIT WALL BASE OFFSET PARAMETER VIA UNWRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Get_Revit_Elements
    {
        private Get_Revit_Elements()
        {
        }
        public static Double getInstanceParam
          (Revit.Elements.Element inRevitElement)
        {
            //unwrap
            Autodesk.Revit.DB.Element UnwrappedElement
            = inRevitElement.InternalElement;
            Double UnwrappedElementParameterValue
            = UnwrappedElement.get_Parameter
            (BuiltInParameter.WALL_BASE_OFFSET).AsDouble();
            return UnwrappedElementParameterValue;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

OPEN VISUAL STUDIO FOLDER “GET_REVIT_PARAMETER_START_START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE “GET_REVIT_PARAMETER_START_START.rvt”
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

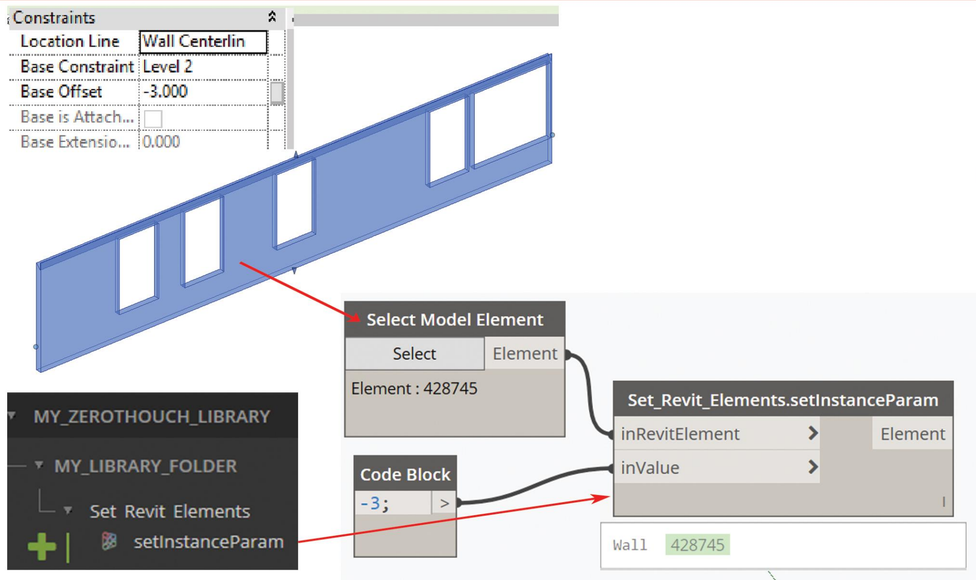
Exercise 5

SET REVIT WALL BASE OFFSET PARAMETER VIA UNWRAPPED DYNAMO ELEMENT

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Set_Revit_Elements
    {
        private Set_Revit_Elements()
        {
        }

        public static Revit.Elements.Element setInstanceParam
            (Revit.Elements.Element inRevitElement, Double inValue)
        {
            //unwrap
            Autodesk.Revit.DB.Element UnwrappedElement = inRevitElement.InternalElement;
            //Start Transaction because changing the Revit DataBase
            //Get Current Document (standard stuff)
            Autodesk.Revit.DB.Document doc = DocumentManager.Instance.CurrentDBDocument;
            TransactionManager.Instance.EnsureInTransaction(doc);
            UnwrappedElement.get_Parameter(BuiltInParameter.WALL_BASE_OFFSET).Set(inValue);
            //End Transaction because changing the Revit DataBase
            TransactionManager.Instance.TransactionTaskDone();
            return inRevitElement;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

OPEN VISUAL STUDIO FOLDER “SET_REVIT_PARAMETER_START_START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE “SET_REVIT_PARAMETER_START_START.rvt”
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

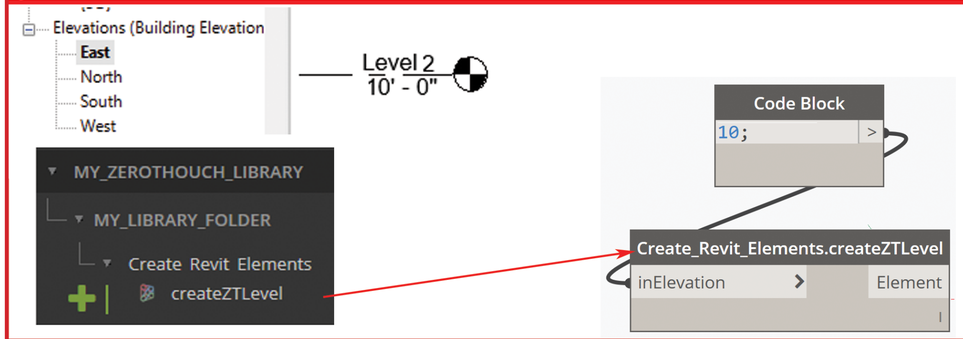
Exercise 6

CREATE REVIT LEVELS VIA ZT
VIA TRANSACTIONS AND WRAPPING

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Create_Revit_Elements
    {
        private Create_Revit_Elements()
        {
        }

        public static Revit.Elements.Element
            createZTSomething(double inElevation)
        {
            //Get Current Document (standard stuff) +
            //Start Transaction because changing the Revit DataBase
            Autodesk.Revit.DB.Document doc =
                DocumentManager.Instance.CurrentDBDocument;
            TransactionManager.Instance.EnsureInTransaction(doc);

            //End Transaction because changing the Revit DataBase
            Autodesk.Revit.DB.Level newLevel =
                Autodesk.Revit.DB.Level.Create(doc, inElevation);
            TransactionManager.Instance.TransactionTaskDone();

            //wrapit! since Revit element generated in Code and sent to Dynamo
            Revit.Elements.Element wrappedLevel = newLevel.ToDSType(false);

            return wrappedLevel;
        }
    }
}
```

DYNAMO ZT NODES



STEPS

OPEN VISUAL STUDIO FOLDER “CREATE_REVIT_LEVEL...START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN REVIT FILE “CREATE_REVIT_LEVEL...START.rvt”
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

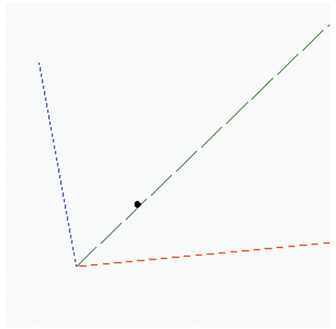
Exercise 7

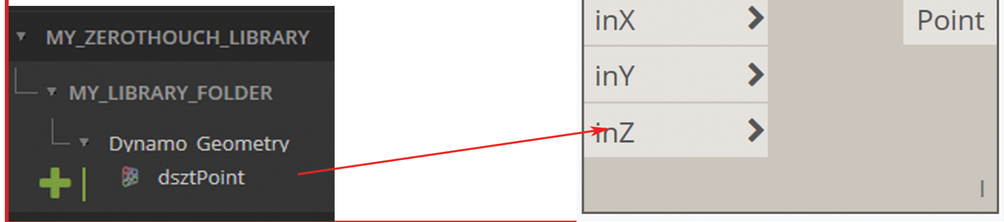
CREATE A DYNAMO POINT VIA DEFAULT X,Y,Z =1

ZEROTOUCH CODE

```
namespace MY_LIBRARY_FOLDER
{
    public class Dynamo_Geometry
    {
        private Dynamo_Geometry()
        {
        }
        public static Autodesk.DesignScript.Geometry.Point dsztPoint
            (double inX = 1, double inY = 1, double inZ = 1)
        {
            //Autodesk.DesignScript.Geometry.Point  is a Dynamo point
            Autodesk.DesignScript.Geometry.Point dPt =
                Autodesk.DesignScript.Geometry.Point.ByCoordinates(inX, inY, inZ);
            return dPt;

        }
    }
}
```

DYNAMO ZT NODE





STEPS

OPEN VISUAL STUDIO FOLDER “CREATE_DYNAMO_POINT_START” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.

OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

Exercise 8

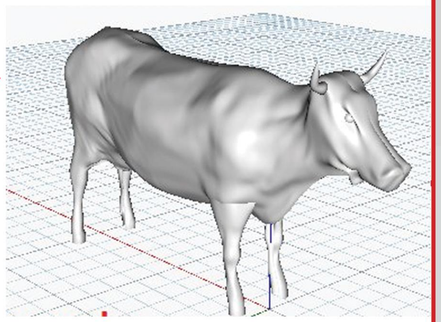
CREATE A COW IN DYNAMO WITH MULTIPLE OUTPUT PORTS

DYNAMO GEOMETRY AND CODE NOTES

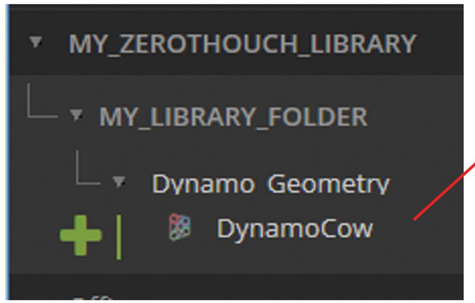
```
                                        USE A MULTIRETURN TAG IF THERE IS
                                        MORE THAN 1 OUTPUT  PORT
//SETTING UP MULTIPLE RETURNS TAG

[MultiReturn(new[] { "LeftBody", "LeftLeg", "RightBody", "RightLeg",
    "LeftEye", "RightEye", "LeftHorn", "RightHorn" })]

.....................................

 //RETURNING
 Dictionary<string, object> OutInfo =
      new Dictionary<string, object>
 {

     {"LeftBody",Surfacetorso},
     {"LeftLeg",Surfaceleg},
     {"RightBody",MirrorSurfacetorso},
     {"RightLeg",MirrorSurfaceleg},
     {"LeftEye",RotationEyeball},
     {"LeftHorn",RotationHorn},
     {"RightEye",MirrorRotationEyeball},
     {"RightHorn",MirrorRotateHorn},


 };

      return OutInfo;
```

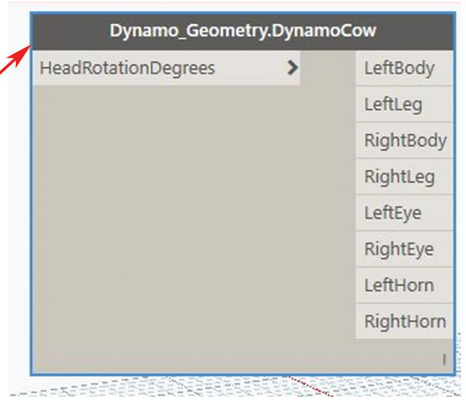
CREATE A DICTONARY TO STORE MULTIPLE OUTPUTS

RETURN THE DICTIONARY



DYNAMO NODES





STEPS & NOTES

OPEN VISUAL STUDIO FOLDER “CREATE_DYNAMO_COW_FINAL” OPEN SLN FILE.
TYPE CODE AS SHOWN. BUILD THE SOLUTION.
OPEN DYNAMO AND START A NEW FILE, LOAD THE DLL FROM BIN FOLDER
ADD NODES AS SHOWN. OPEN “FINAL” FILE FOLDER IF NEEDED. NOTE VALUE IS ALWAYS IN DECIMAL FEET

AUTODESK®

Make anything.