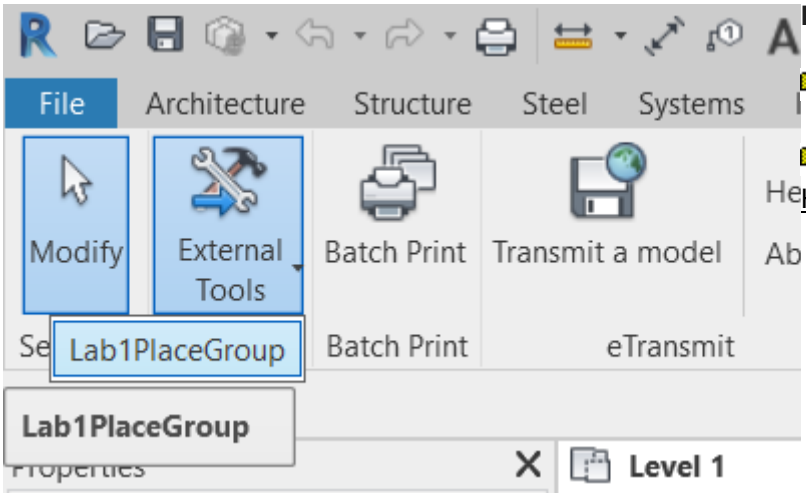
My First Plug-in Training

Lesson 1: The Basic Plug-in

In this lesson you will create your very first basic Autodesk Revit plug-in for copying groups selected by the user to a specified location.

Provide Feedback: Please provide feedback about this Revit Training or this lesson via email: myfirstplugin@autodesk.com



Lesson Downloads

lesson 1_revit_projects.zip

Sample Revit Project File: Hotel.rvt

View Now

Steps to Create Your First Plug-in

1. **Launch the Visual Studio development environment:**
   Open Visual Studio using the Windows **Start** menu, selecting **All Programs**, and then **Microsoft Visual Studio**.

Note: *Supported .NET version for Revit version is as follows:*

- Revit 2024/2023/2022/2021 - .NET 4.8

  You can find the developer guide for the versions here:

Revit 2024: https://help.autodesk.com/view/RVT/2024/ENU/?guid=Revit_API_Revit_API_Developers_Guide_html

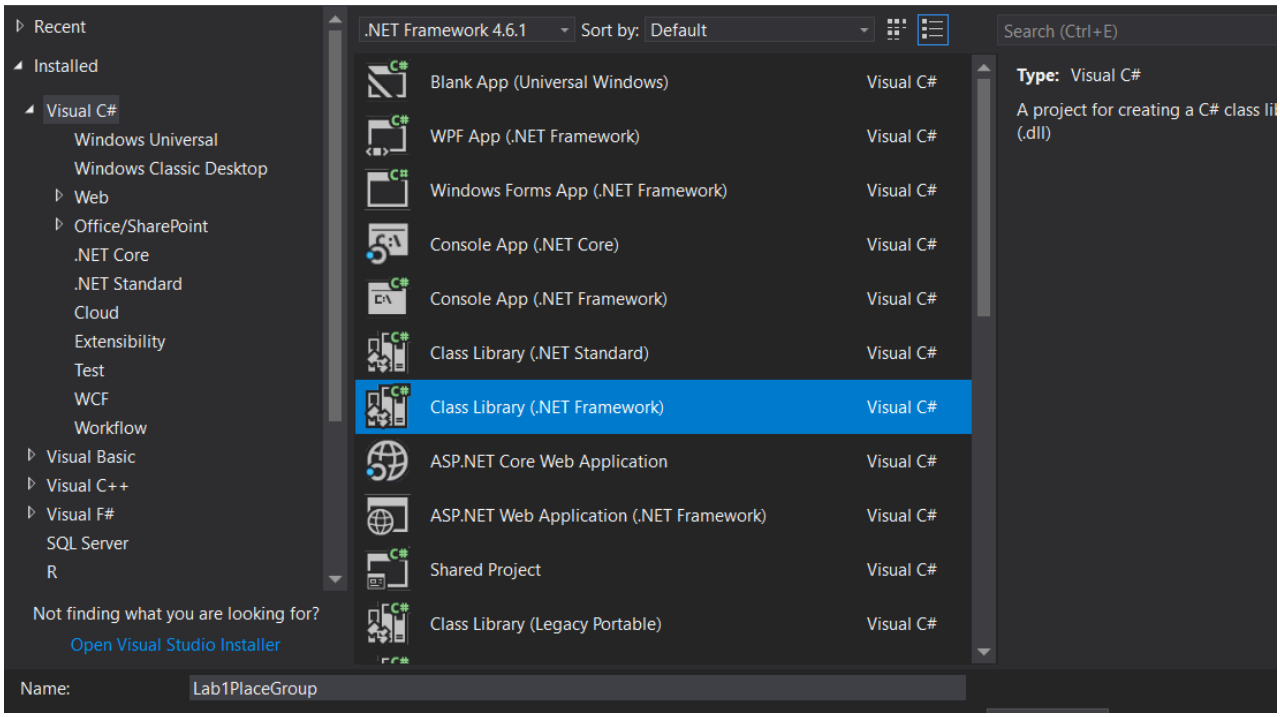
Revit 2023: https://help.autodesk.com/view/RVT/2023/ENU/?guid=Revit_API_Revit_API_Developers_Guide_html

Revit 2022: https://help.autodesk.com/view/RVT/2022/ENU/?guid=Revit_API_Revit_API_Developers_Guide_html

Revit 2021: https://help.autodesk.com/view/RVT/2021/ENU/?guid=Revit_API_Revit_API_Developers_Guide_html

2. Create a class library project:

Inside Visual Studio, on the **File** menu, click **New Project**. In the **Installed Templates** tab in the left-hand window, click **Visual C#** .In the middle window, click **Class Library**.



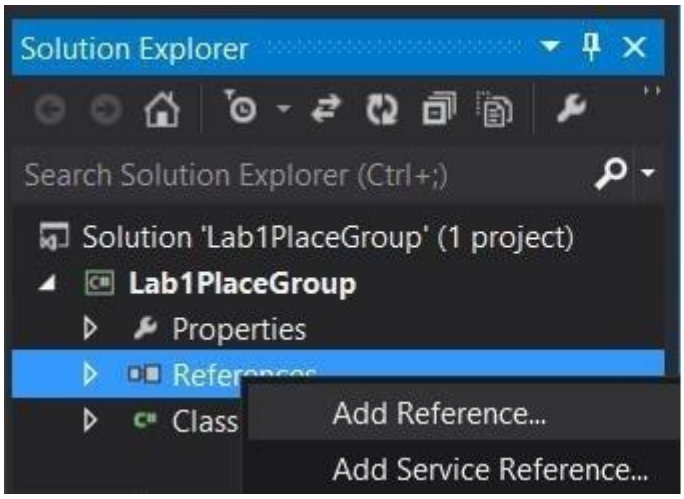
Enter **Lab1PlaceGroup** in the **Name** box. And then click **OK**

Visual Studio will create a default code project for you and display the code in the code window.

3. Save the project:

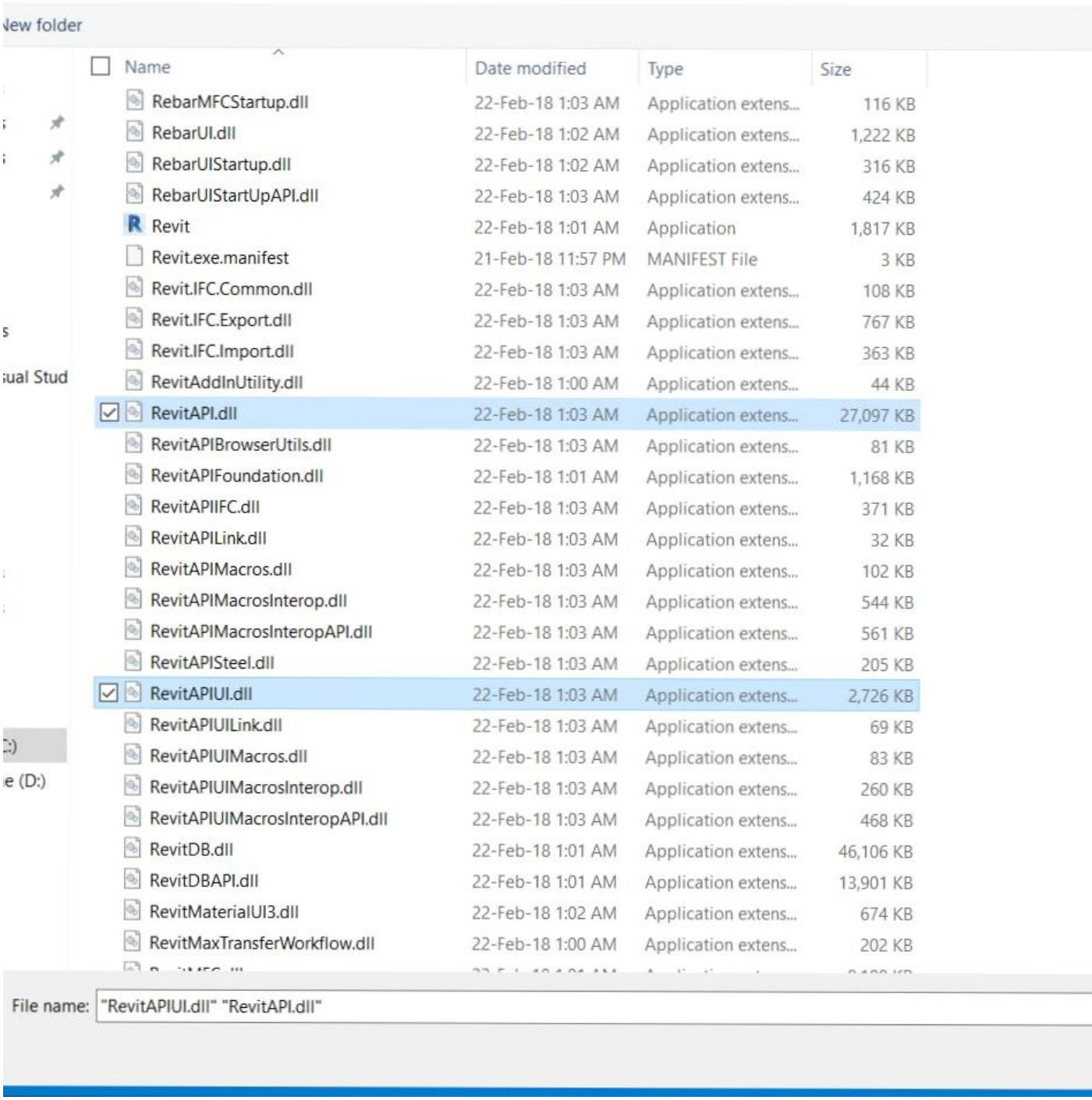
On the **File** menu, click **Save All**. In the display window type ***C:\test*** in the **Location** box, and then click **Save**.

4. **Add references:**
   In the **Solution Explorer** window on the right-hand side of the Visual Studio window, right-click **References** and click **Add Reference…**

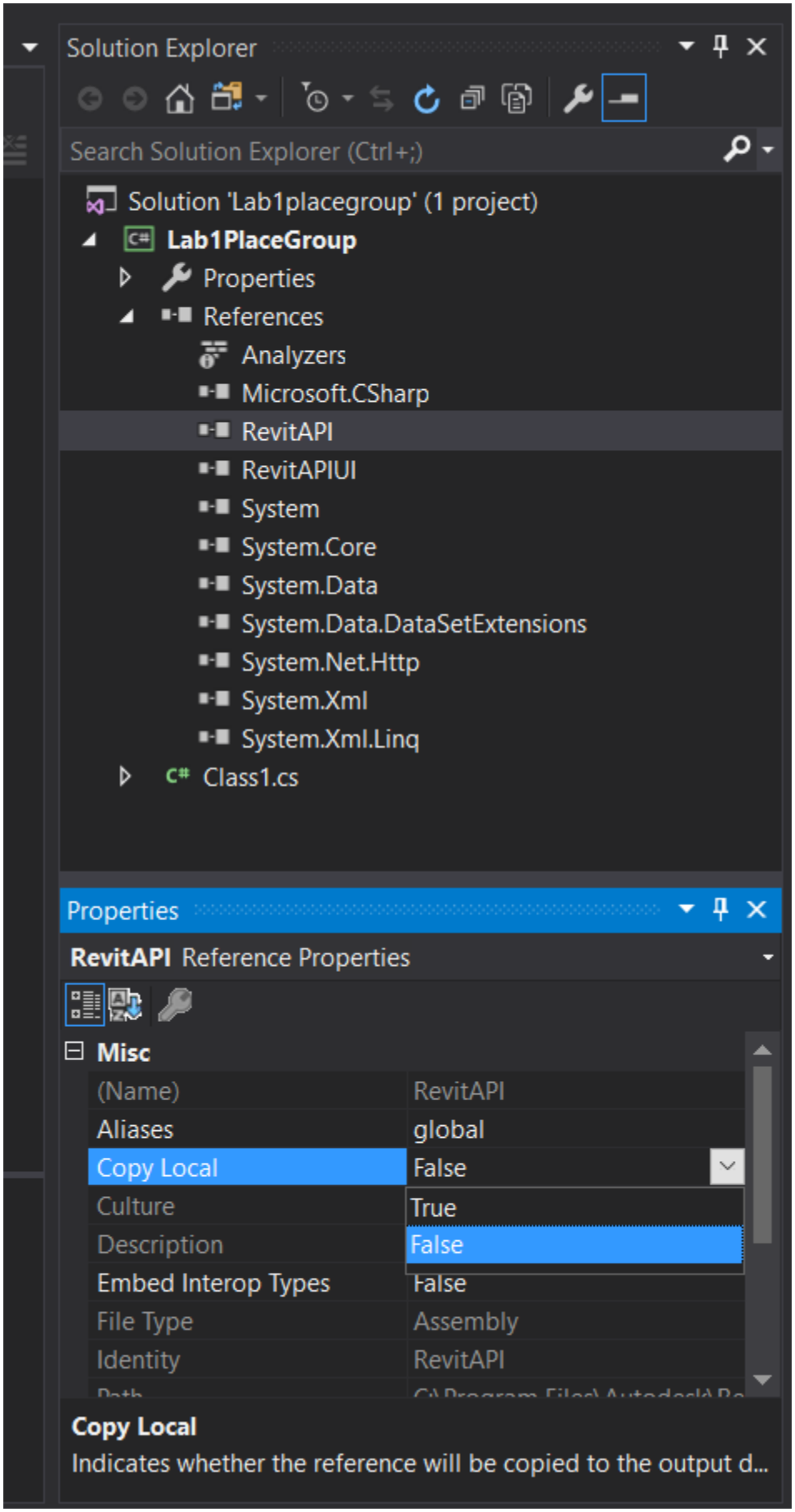


5. Click the **Browse** tab and in the **Add Reference** dialog and browse to the Revit product installation sub-folder. (The sub-folder path depends on where you have installed Revit 20xx. The default path is *C:\Program Files\Autodesk\Revit 20xx**).

** The path may vary depending on the flavor of Autodesk Revit you are using*



You will add two reference files from this folder. Select **RevitAPI.dll**, hold the Ctrl key and select **RevitAPIUI.dll**, and then click **OK**. Now the two interface DLL files are referenced in your project. All the Revit APIs are exposed by these interface files and your project can use all of those available APIs from them.


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Lab1placegroup' (1 project)
Lab1PlaceGroup
Properties
References
Analyzers
Microsoft.CSharp
RevitAPI
RevitAPIUI
System
System.Core
System.Data
System.Data.DataSetExtensions
System.Net.Http
System.Xml
System.Xml.Linq
Class1.cs
Properties
RevitAPI Reference Properties
Misc
(Name)
RevitAPI
Aliases
global
Copy Local
False
Culture
Description
Embed Interop Types
File Type
Assembly
Identity
RevitAPI
True
False
Copy Local
Indicates whether the reference will be copied to the output d...

6. Set the referenced files' Copy Local property value

In the **Solution Explorer** window you saw in step 5,
click **RevitAPI** under **Reference** node. In the **Properties** window,
click **Copy Local** property, and then click the drop-down list, select **False**.
Repeat the same steps to change **RevitAPIUI**'s **Copy Local** property value to **False**.

7. Add the code:

Double click **Class1.cs** in the **Solution Explorer** window to show the code-editing window. Delete everything in this window and then type the following C# code. To get the full experience of developing with Visual Studio – including the use of features such as IntelliSense – we recommend you type the code from this guide rather than copying and pasting it. That said, if constrained for time you can also copy and paste into the Visual Studio code window: although this reduces the experience you gain from working with the code directly.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Autodesk.Revit.ApplicationServices;
using Autodesk.Revit.Attributes;
using Autodesk.Revit.DB;
using Autodesk.Revit.UI;
using Autodesk.Revit.UI.Selection;
using Autodesk.Revit.DB.Architecture;
namespace Lab1PlaceGroup
{
    [Transaction(TransactionMode.Manual)]
    [Regeneration(RegenerationOption.Manual)]
    public class Class1 : IExternalCommand
    {
        public Result Execute(ExternalCommandData commandData, ref
string message, ElementSet elements)
        {
            //Get application and document objects
            UIApplication uiapp = commandData.Application;
            Document doc = uiapp.ActiveUIDocument.Document;

            //Define a reference Object to accept the pick result
            Reference pickedref = null;

            //Pick a group
            Selection sel = uiapp.ActiveUIDocument.Selection;
            pickedref = sel.PickObject(ObjectType.Element, "Please
select a group");
            Element elem = doc.GetElement(pickedref);
            Group group = elem as Group;
```

```
            //Pick point
            XYZ point = sel.PickPoint("Please pick a point to place
group");

            //Place the group
            Transaction trans = new Transaction(doc);
            trans.Start("Lab");
            doc.Create.PlaceGroup(point, group.GroupType);
            trans.Commit();

            return Result.Succeeded;
        }
    }
}
```

Don't worry about the details of the code for now, you'll come back to this shortly in the next couple of lessons.

8. **Save the file:**

On the **File** menu, click **Save All**.

9. **Change the .NET Framework**

In the **Solution Explorer** window on the right hand side of Visual Studio window, right-click on Lab1placegroup and select Properties.

In the application option, set target framework to .NET framework 4.8.

10. **Build the project:**

The code you have written is in human readable form. To make the code readable by a computer, you will need to translate it or “build” it.

Inside Visual Studio, in the **Build** menu, click **Build Solution** to compile and build your plug-in. **Build Success** message shows in status bar of the Visual Studio window if the code is successfully built.

```
Lab1PlaceGroup                                  Lab1PlaceGroup.Class1
10  using Autodesk.Revit.UI.Selection;
11
12  namespace Lab1PlaceGroup
13  {
14      [Transaction(TransactionMode.Manual)]
15      [Regeneration(RegenerationOption.Manual)]
            0 references
16      public class Class1 : IExternalCommand
17      {
                0 references
18          public Result Execute(ExternalCommandData commandData, ref string message, ElementSet elements)
19          {
20              //Get application and documnet objects
21              UIApplication uiapp = commandData.Application;
22              Document doc = uiapp.ActiveUIDocument.Document;
23
24              //Define a reference Object to accept the pick result
25              Reference pickedref = null;
26
27              //Pick a group
28              Selection sel = uiapp.ActiveUIDocument.Selection;
29              pickedref = sel.PickObject(ObjectType.Element, "Please select a group");
30              Element elem = doc.GetElement(pickedref);
31              Group group = elem as Group;
32
33              //Pick point
34              XYZ point = sel.PickPoint("Please pick a point to place group");
35
36              //Place the group
37              Transaction trans = new Transaction(doc);
38              trans.Start("Lab");
39              doc.Create.PlaceGroup(point, group.GroupType);
40              trans.Commit();
41
42              return Result.Succeeded;
43          }
44      }
45  }
```

That’s it! You have just written your first plug-in for Autodesk Revit.

Before you actually work with the plug-in in Revit, you will need to do one more step, which is to write an AddIn manifest.

Writing an AddIn Manifest

An AddIn manifest is a file located in a specific location checked by Revit when the application starts. The manifest includes information used by Revit to load and run the plug-in.

1. **Add the manifest code:**
   Start Notepad.exe from the Windows Start menu. Copy and paste the following plug-in load settings to the Notepad editor.

   Note: In the add-in file, you will see a <AddInId>GUID </AddInId> tag.It is a GUID that represents the id of this particular application. AddInId must be unique for a given session of Revit.

Autodesk recommends you generate a unique GUID for each registered application or command.

```
<?xml version="1.0" encoding="utf-8"?>
<RevitAddIns>
 <AddIn Type="Command">
       <Name>Lab1PlaceGroup</Name>
       <FullClassName>Lab1PlaceGroup.Class1</FullClassName>
       <Text>Lab1PlaceGroup</Text>
       <Description>Places the Group at Particular Point</Description>
       <VisibilityMode>AlwaysVisible</VisibilityMode>

<Assembly>C:\test\Lab1PlaceGroup\Lab1PlaceGroup\bin\Debug\Lab1placeGrou
p.dll</Assembly>
       <AddInId>502fe383-2648-4e98-adf8-5e6047f9dc34</AddInId>
    <VendorId>ADSK</VendorId>
    <VendorDescription>Autodesk, Inc,
www.autodesk.com</VendorDescription>
 </AddIn>
</RevitAddIns>
```

Depending on what version you are using you may need to change the path here to match your Lab1PlaceGroup.dll location on your computer: *C:\test\Lab1PlaceGroup\Lab1PlaceGroup\bin\Release\Lab1PlaceGroup.dll*

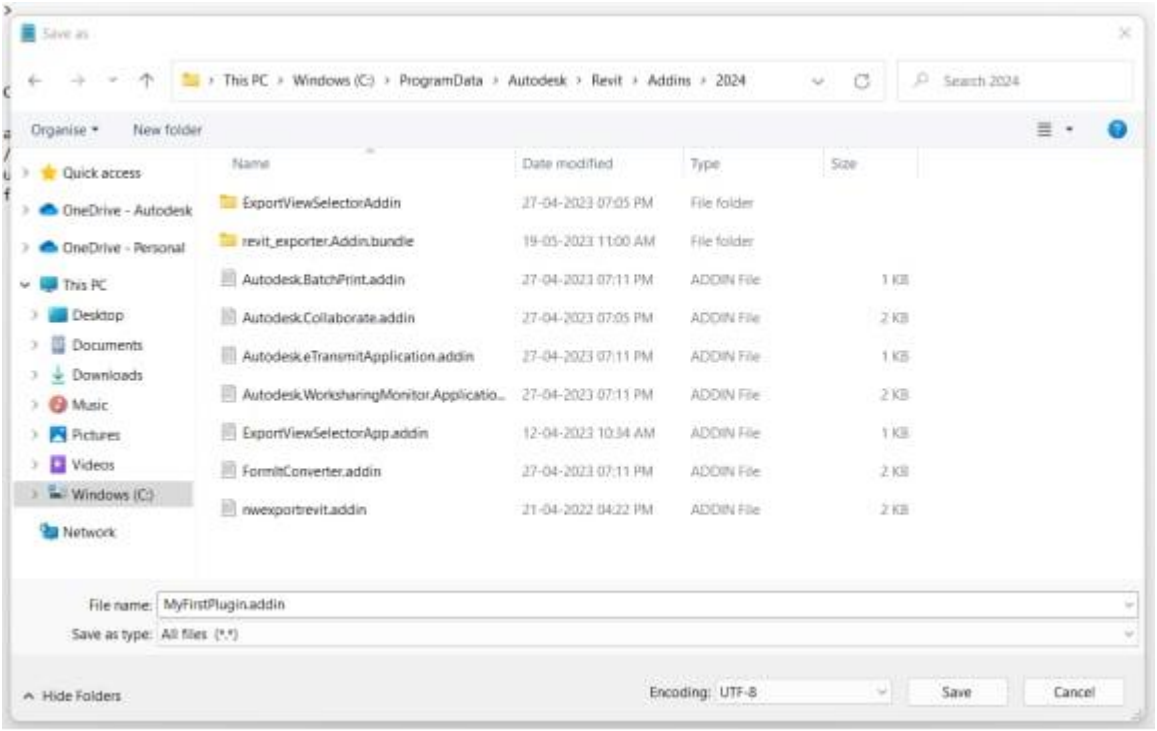
2. **Save the file:**

On **Notepad's File** menu, click Enter **MyFirstPlugin.addin** in the **File name** box. Change **Save as type** to the **All Files** option (the file name is up to you; however, the file extension must be ".addin"). Browse to the following subfolder, and then click the **Save** button.

Under Windows 10

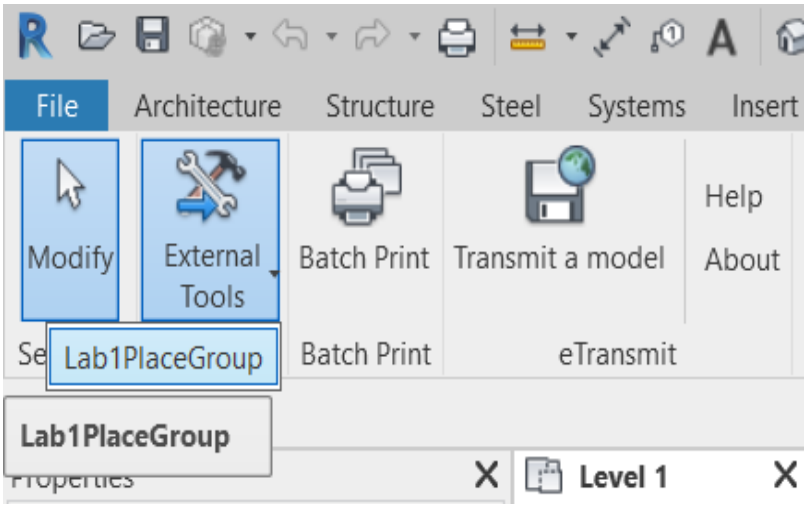
- *C:\ProgramData\Autodesk\Revit\Addins\20xx* (The ProgramData folder is hidden by default)

For example, here is the setting in Save As dialog in Windows 10 for Revit 20xx.



3. **Load your plug-in into Revit and allow the plug-in to communicate with Revit:**

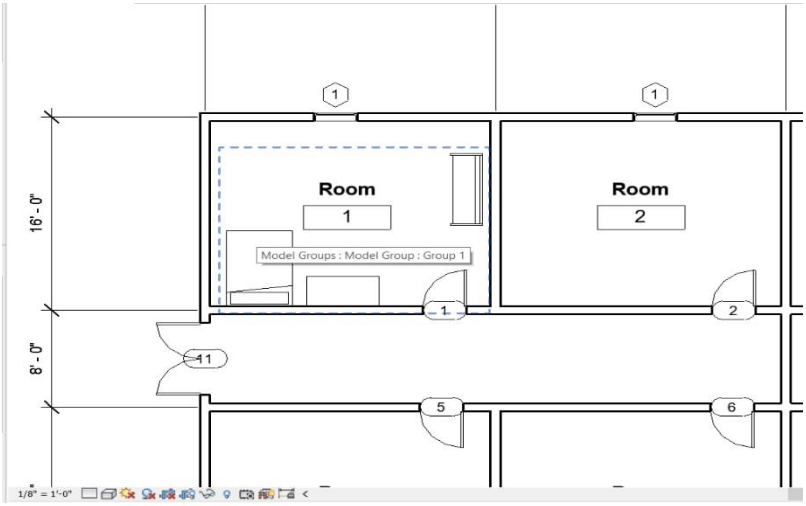
Inside Revit on the **Add-Ins** ribbon tab, click the **External Tools** drop-down list, then click **Lab1PlaceGroup**. This will start your plug-in.



4. **Work with the plug-in:**

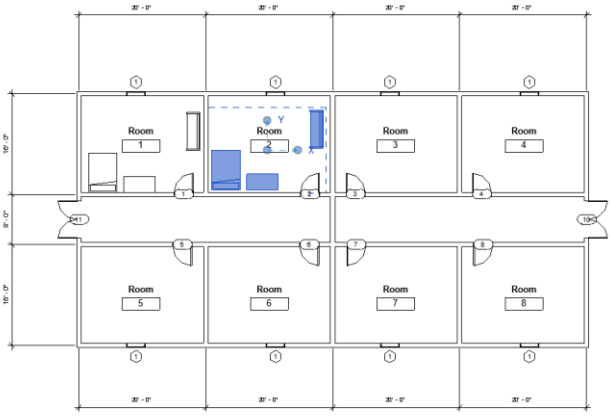
Move the cursor over **Room1** in the Revit building model. When the cursor is hovering over the furniture group, its bounding box should be highlighted as per the below picture, with a tooltip showing **Model Groups : Model Group : Group 1**. Click to select this furniture group. (Note: when highlighted the room looks very similar to the group. Please carefully select

the group according to the message in the tooltip. If the room is selected, you will not see the expected result after the following step.)



NOTE: Make sure the TOOLTIP message is ModelGroups, then click to pick.

5. Pick a point in another room, for example in **Room 2**. You should see the group copied to this location. The center of the new group is the point you selected.



Congratulations! You have just written your first plug-in for Autodesk Revit. You will be reviewing the code in detail in Lesson 3.

Before you move on to the next lessons, let us go back to some of the things we skipped over earlier, starting with basics concept about programming, and the benefits it can bring to your day-to-day work.

Additional Topics

Introduction to Programming

The C# code you have just executed to copy a group is only 30 lines long. Here you see a small amount of code working in a similar way to the internal Revit command, Create Similar. Software programming allows you to capture the logic of a particular functionality once and then reap the benefits over and over again, every time you want to perform this functionality.

What is Programming?

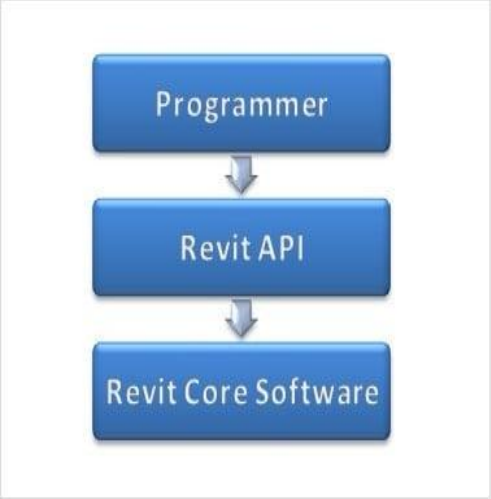
A simple answer to this question is: Computer programming is the process of creating a sequence of instructions to tell the computer to do something. You can look at your program as a sequence of instructions. During the course of the upcoming lessons, you will look at the various lines and blocks of code and look at them all in the context of being instructions for a computer.

If you were to explain what computers are to a young child, you might say: a computer is a tool which follows instructions you provide. Programming is one way of giving instructions to the computer. Internally, a computer sees these instructions encoded as a series of numbers (also called machine code). The human-readable instructions you saw at the beginning of this lesson is called source code and the computer converts these instructions to machine code which it can then read and execute. A sequence of such instructions (or code), written to perform a specific task, is called a program and a collection of such programs and related data is called a software. Autodesk Revit is one such software product.

Source code can be written in different languages, just as humans use different languages to communicate between ourselves. The language you will be using in this guide is called C# (pronounced “C-Sharp”).

What is an API?

API is the acronym for Application Programming Interface: the way a software programmer can communicate with a software product. For instance, the Revit API is the way programmers can work with Revit, and it establishes what functionality a software programmer can use within Revit. Such as the Revit API allows you to write instructions for Revit to execute one after the other.



What is a Plug-in?

A software plug-in is a type of program module (or file) that adds functionality to a software product, usually in the form of a command automating a task or some customization of the product's behavior. When you talk about a plug-in for Revit – and you will also hear the term Add-In used for this product – we mean a module containing code that makes use of the Revit API. Revit loads such plug-ins and uses them to adjust its behavior under certain conditions, such as when a particular command is executed by the user of the plug-in.

My First Revit Plug-in Overview >> Lesson 1 >> Lesson 2 >> Lesson 3>> Lesson 4 >> Lesson 5 >> Lesson 6 >> Lesson 7 >> Lesson 8