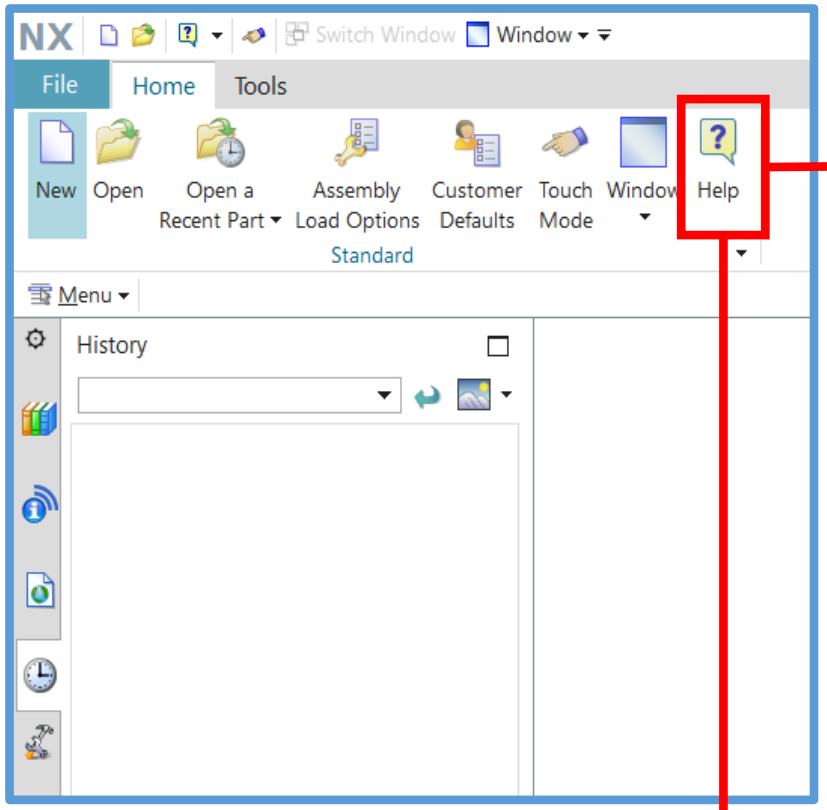
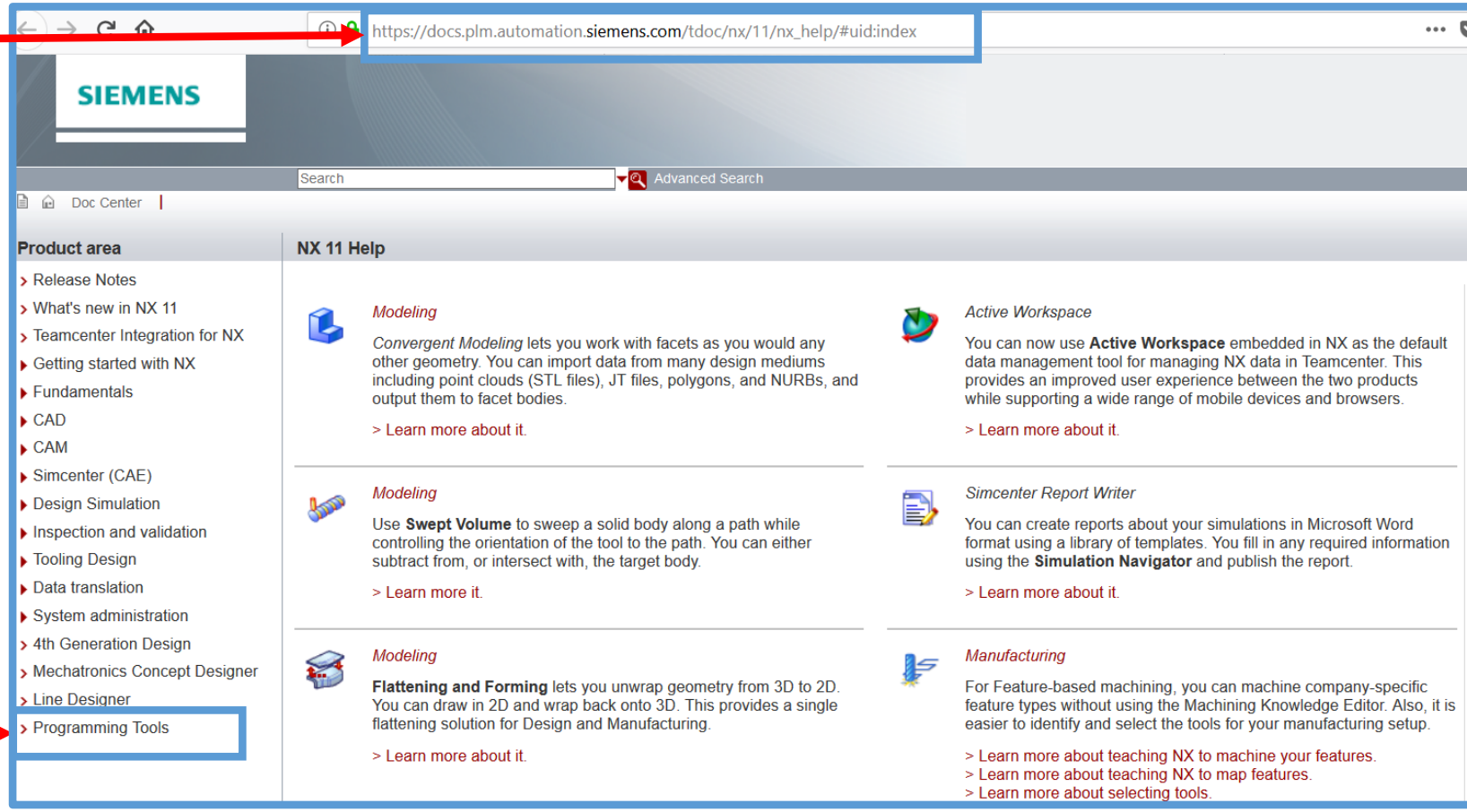
How to Open Programming help From NX

Step 1: Open NX ->Home Page ->Clip Help



This Page displays in your default web browser



Select ->Programming Tool from List

Note :If Pop up Blocked ,Allow Pop Up for once

Source :Siemens NX Help Documentation

Step 2:Access Programming tools

https://docs.plm.automation.siemens.com/data_services/resources/nx/11/nx_help/common/en_US/graphics/fileLibrary/nx/api_collection

Accessing the Programming Tools help

The Programming Tools help is an optional feature of the NX Help. You can access it from the Siemens website or a local install.

To view the Programming Tools help on the Siemens website, click here

If you can't access the Siemens website and have the Programming Tools help installed locally, click here

- If the local install link does not work, run setup.exe to launch the Install Shield Wizard and click on the icon for **NX API Manuals**. Be sure to install both the **NX On-line Help** and **NX API Manuals** in the same directory so that the links between the help collections work correctly.
- For additional information on installing the Programming Tools help, see the *NX Documentation Installation Instructions for Windows* (nxdocs_install.pdf) for your installed version.

Click the link to open the NX Programming tools

Source :Siemens NX Help Documentation

Step 3: Select Programming tool

https://docs.plm.automation.siemens.com/tdoc/nx/11/nx_api/#uid:index

SIEMENS

Search | Advanced Search

Doc Center

Product area

- NX Open
  - NX Open Getting Started Guide
  - NX Open Programmer's Guide
  - Open for C/C++
  - Open for Java
  - NX Open Python
  - Open for .NET
  - Open for GRIP
  - Open for CAE
- SNAP (Simple NX Application Programming)
- Knowledge Fusion Help and Best Practices
- Knowledge Fusion Language Reference
- Menuscript User's Guide
- Block UI Styler
- Pre-NX 6 UI Styler

Programming Tools

Click NX Open to Expand

NX Open

NX Open is a collection of APIs that allows you to create custom applications for NX through an open architecture using well-known programming languages (C/C++, Visual Basic, C#, Java, and Python). You can automate complex and repetitive tasks, integrate third party applications, and customize the NX interface in your preferred programming language.

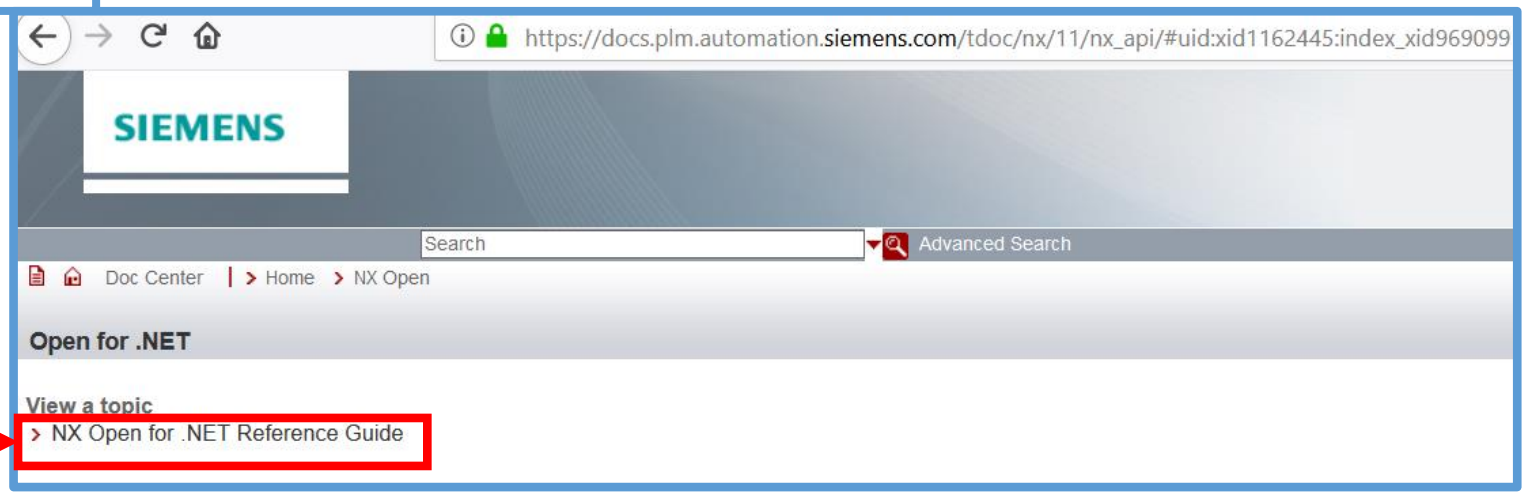
```
lw = s.ListingWindow
lw.Open()
lw.WriteLine('Get Meshes')
theFeModel = basePart.BaseFtModel
meshList = theFeModel.MeshManager.GetMeshes()
lw.WriteLine("Number of meshes = {}".format(len

for theMesh in meshList:
    meshName = theMesh.Name
    collectorName = theMesh.MeshCollector.Name
    lw.WriteLine("Mesh {0} belongs to Collector
        meshName, collectorName))
```

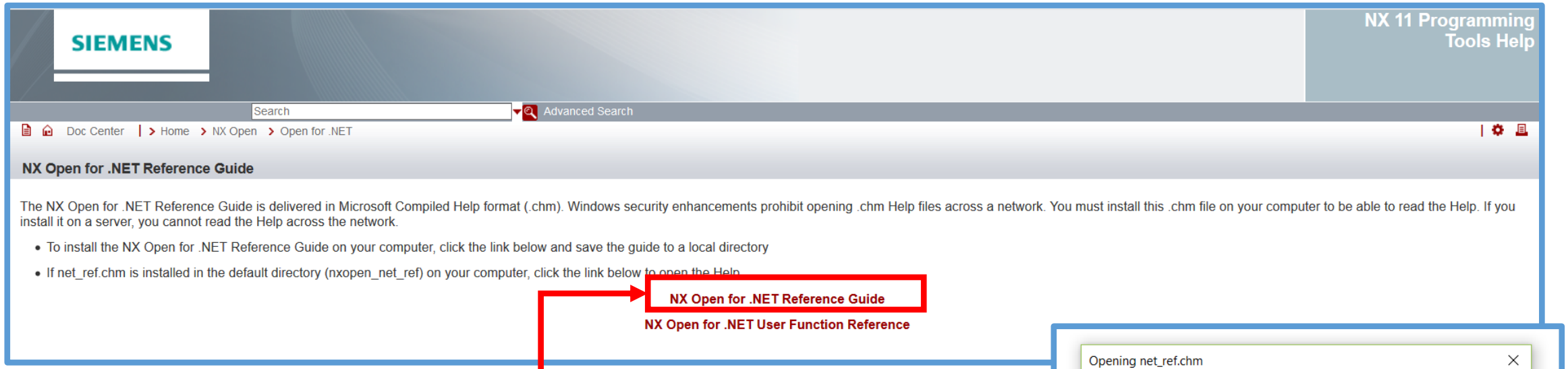
Click Open for .NET

Source :Siemens NX Help Documentation

Step 4:Download NX Open .Net Reference Guide

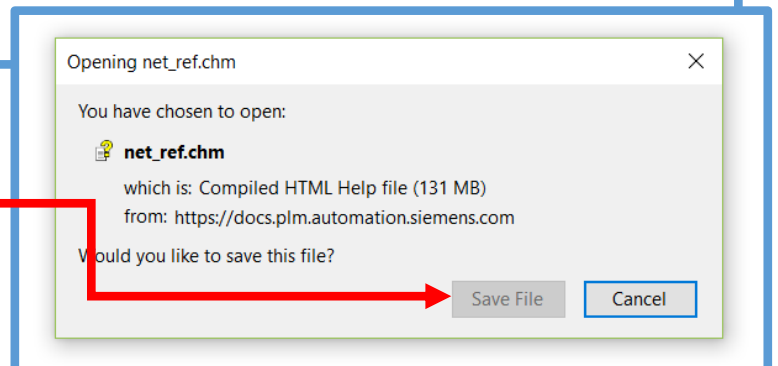
Click Open for .NET Reference Guide





Select NX Open for .NET Reference Guide

Save This File



Source :Siemens NX Help Documentation

Step 5:Open net_ref.chm file from downloads

NX NXOpen .Net API Reference

Hide | Locate | Back | Forward | Stop | Refresh | Home | Print | Options

Contents | Index | Search | Favorites

- Namespaces
- CAE Solver Language documentation
- User Function Reference
- Mapping UF to NXOpen

Namespaces

Namespaces

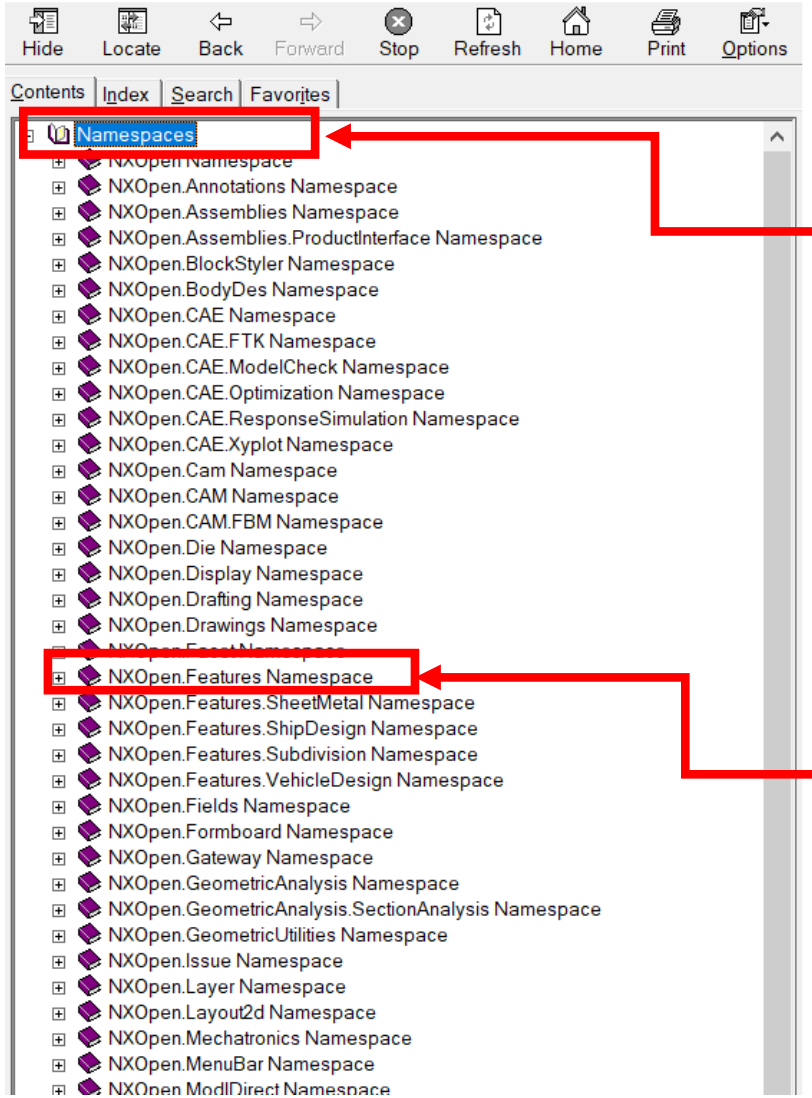
Namespaces

Namespace	Description
NXOpen	Provides classes and interfaces for the NX Open Common API.
NXOpen.Annotations	Provides classes and interfaces relating to Dimensions, Notes, Symbols, PMI, and other Annotations.
NXOpen.Assemblies	Provides classes and interfaces for Assemblies.
NXOpen.Assemblies.ProductInterface	Provides classes and interfaces for Assembly Product Interfaces.
NXOpen.BlockStyler	Provides classes and interfaces for the Block Styler.
NXOpen.BodyDes	Provides classes and interfaces for Body Design.
NXOpen.CAE	Provides classes and interfaces to Advanced Simulation and Design Simulation.
NXOpen.CAE.FTK	Provides classes and interfaces to Function Toolkit.
NXOpen.CAE.ModelCheck	Provides classes and interfaces relating to Model Check.
NXOpen.CAE.Optimization	Provides classes and interfaces relating to Optimization.
NXOpen.CAE.ResponseSimulation	Provides classes and interfaces for Response Simulation.
NXOpen.CAE.Xyplot	Provides classes and interfaces for XY Graphing.
NXOpen.Cam	Provides classes and interfaces for Manufacturing.
NXOpen.CAM	Provides classes and interfaces for Manufacturing.
NXOpen.CAM.FBM	Provides classes and interfaces relating to Feature Based Machining.
NXOpen.Die	Provides classes and interfaces relating to Die Engineering and Die Design.
NXOpen.Display	Provides classes and interfaces for display.
NXOpen.Drafting	Provides classes and interfaces for the Drafting.

Source :Siemens NX Help Documentation

Step 6: How To use this Guide ?

Example : How to Create a Solid Block in NX using this .net Reference Guide



Click Namespace and expand

Click NxOpen.Features.Namespace and expand

Source :Siemens NX Help Documentation

Step 7: Open the BlockFeatureBuilder Class

This Page Explains What are the Properties and Methods required to create a Block

Contents | Index | Search | Favorites

- AssociativeArcBuilder.StartReference Enumeration
- AssociativeArcBuilder.Types Enumeration
- AssociativeLine Class
- AssociativeLineBuilder Class
- AssociativeLineBuilder.EndOption Enumeration
- AssociativeLineBuilder.EndReference Enumeration
- AssociativeLineBuilder.StartOption Enumeration
- AssociativeLineBuilder.StartReference Enumeration
- AssociativityType Enumeration
- AutomotiveCollection Class
- BaseFeatureCollection Class
- Bead Class
- Bend Class
- BendTaper Class
- BlendCorner Class
- BlendCornerBuilder Class
- BlendCornerBuilder.ContinuityTypes Enumeration
- BlendPocket Class
- BlendPocketBuilder Class
- BlendPocketBuilder.AngledWallSolutions Enumeration
- BlendPocketBuilder.ToolTypes Enumeration
- Block Class
- BlockFeatureBuilder Class
  - BooleanOption Property
  - BooleanType Property
  - GetOrientation Method (xAxis, yAxis)
  - Height Property
  - Length Property
  - Origin Property
  - OriginPoint Property
  - ParentAssociativity Property
  - PointFromOrigin Property
  - SetBooleanOperationAndTarget Method (booleanOperation, targetBc
  - SetHeight Method (height)
  - SetLength Method (length)
  - SetOrientation Method (xAxis, yAxis)
  - SetOriginAndLengths Method (originPoint, lengthExpression, widthExp
  - SetTwoDiagonalPoints Method (originPoint, cornerPoint)
  - SetTwoPointsAndHeight Method (originPoint, cornerPoint, heightExpre
  - SetWidth Method (width)
  - Target Property
  - Type Property
  - Width Property
- BlockFeatureBuilder.Types Enumeration
- BodyFeature Class
- BooleanBuilder Class
- BooleanFeature Class
- BoundedPlane Class
- BoundedPlaneBuilder Class

BlockFeatureBuilder Class

Namespaces ► NXOpen.Features ► **BlockFeatureBuilder**

Represents a block feature builder.

Declaration Syntax

C# | Visual Basic | Visual C++

```
public class BlockFeatureBuilder : FeatureBuilder
```

Members

All Members | Methods | Properties

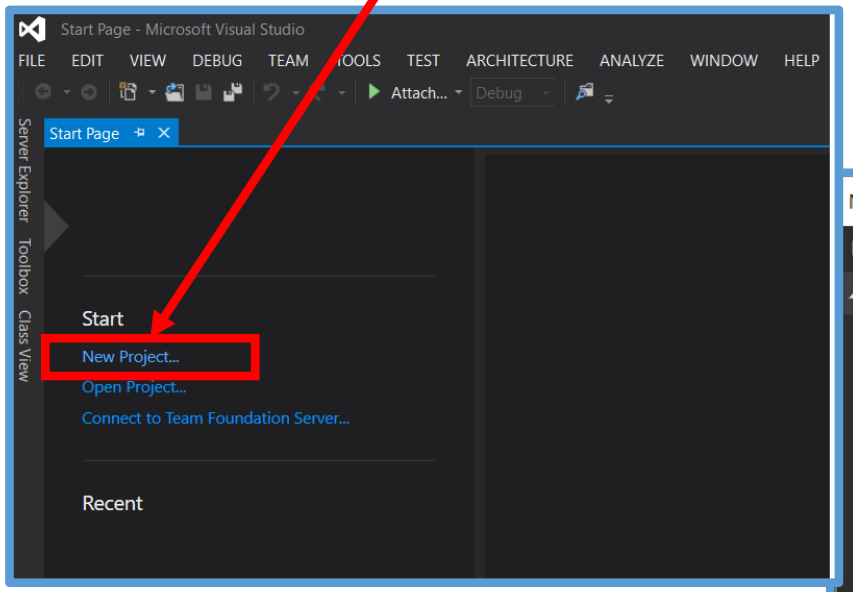
☑ Public ☑ Protected ☑ Instance ☑ Static ☑ Declared ☑ Inherited

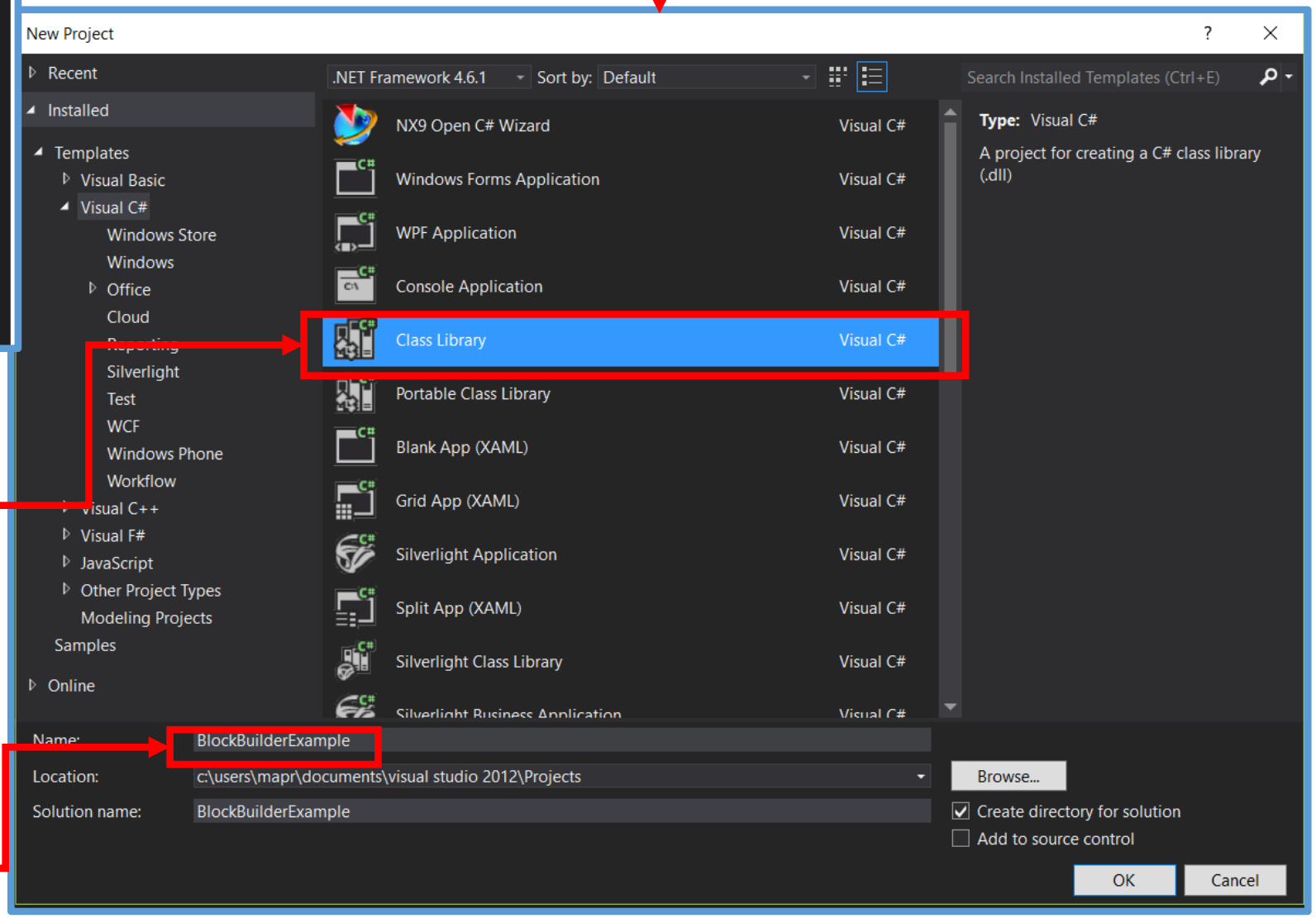
Icon	Member	Description
	AsyncProcessMessage(IMessage, IMessageSink)	Asynchronously processes the given message. (Inherited from NXRemotableObject.)
	BooleanOption	Returns the boolean option
	BooleanType	Returns or sets the boolean operation for the block
	Commit()	Commits any edits that have been applied to the builder. (Inherited from Builder.)
	CommitFeature()	Commits the feature parameters and creates the feature (Inherited from FeatureBuilder.)
	CreateObjRef(Type)	Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject.)
	Destroy()	Deletes the builder, and cleans up any objects created by the builder. (Inherited from Builder.)
	Equals(Object)	Determines whether the specified Object is equal to the current Object. (Inherited from Object.)
	GetCommittedObjects()	For builders that create more than one object, this method returns the objects that are created by commit.

Source :Siemens NX Help Documentation

Step 8: Create a New Project In Visual Studio

It will Pop up the Below Window.
Select Visual C#

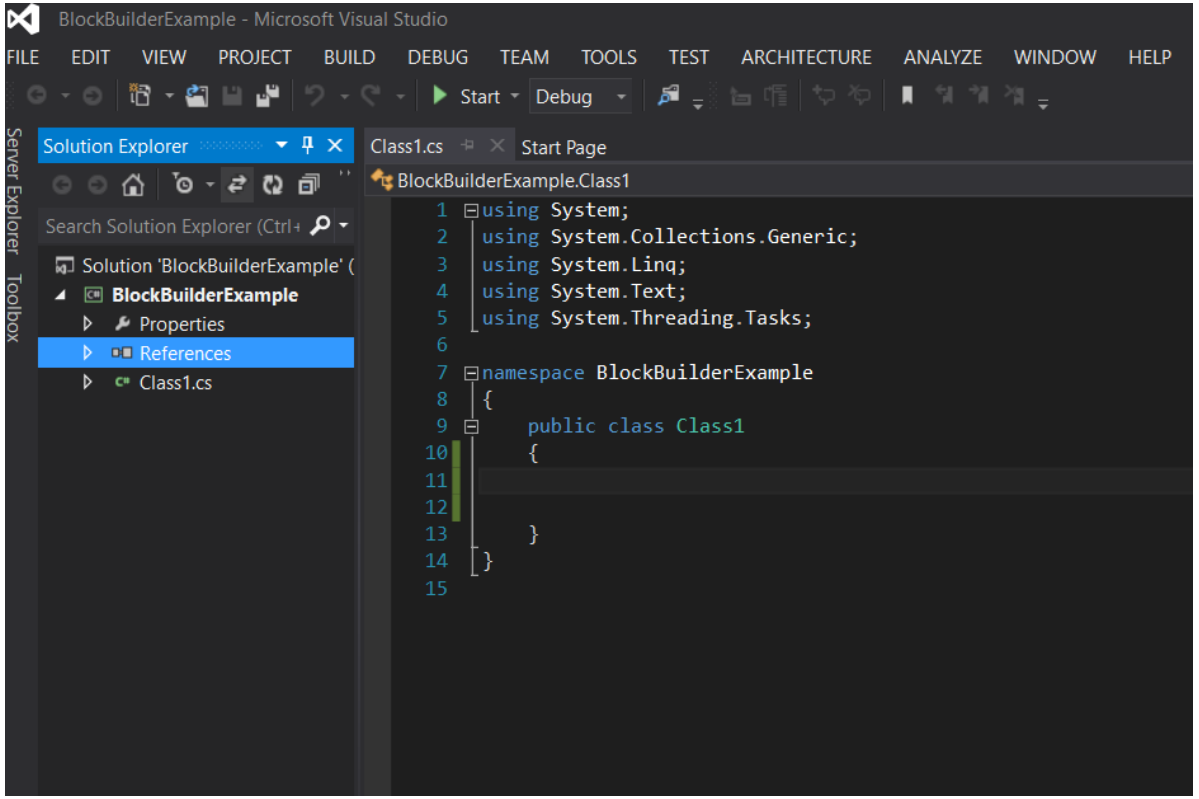




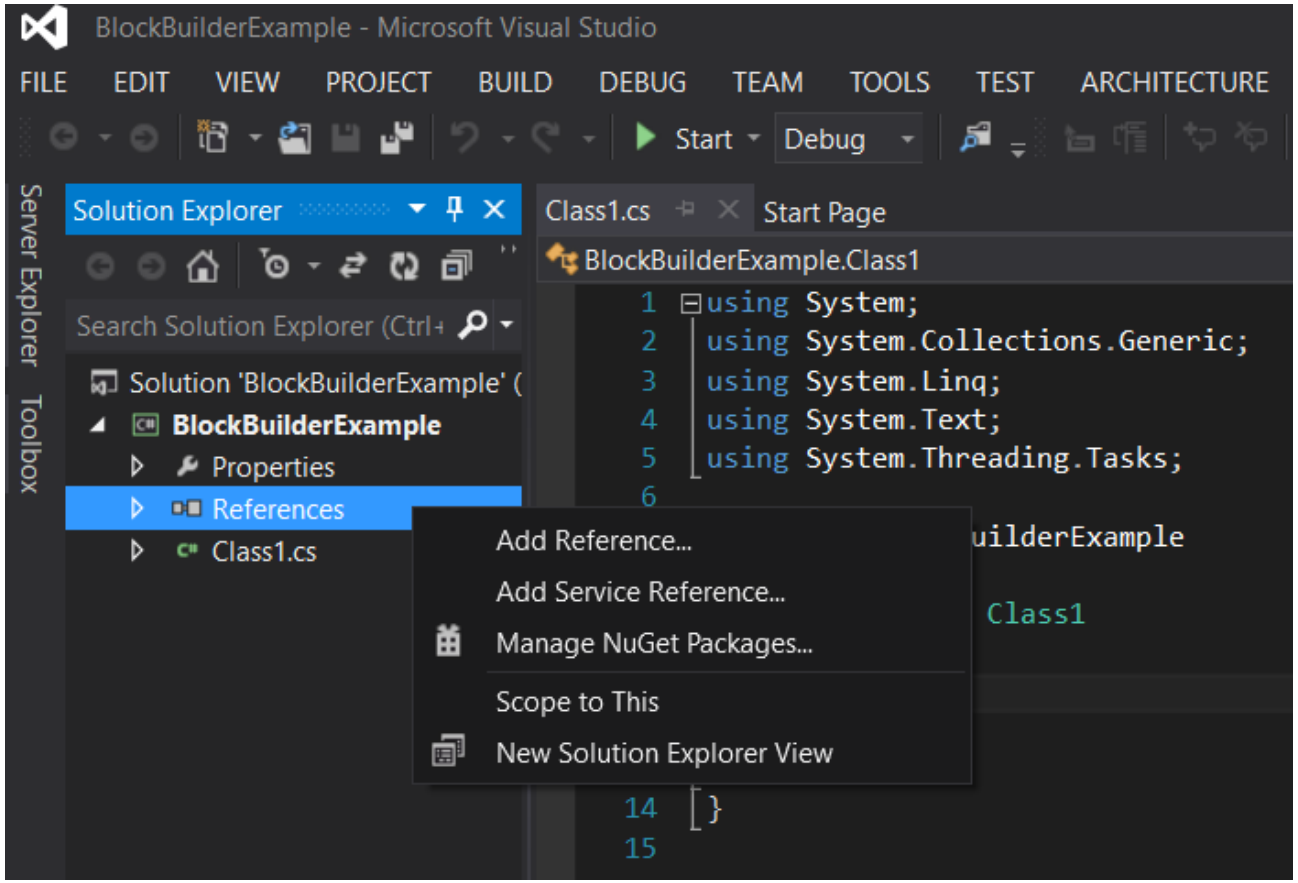
Select Class Library from list

Give a Solution Name and hit OK

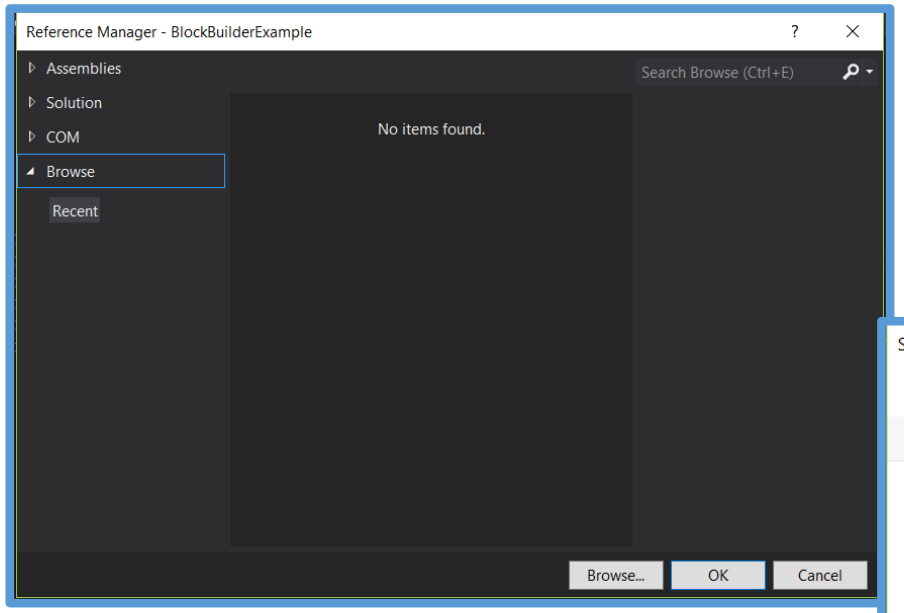
Step 9: Default Visual Studio Project Window



Right Click References ->Select Add Reference



Step 10: Click Browse to Add Reference Files

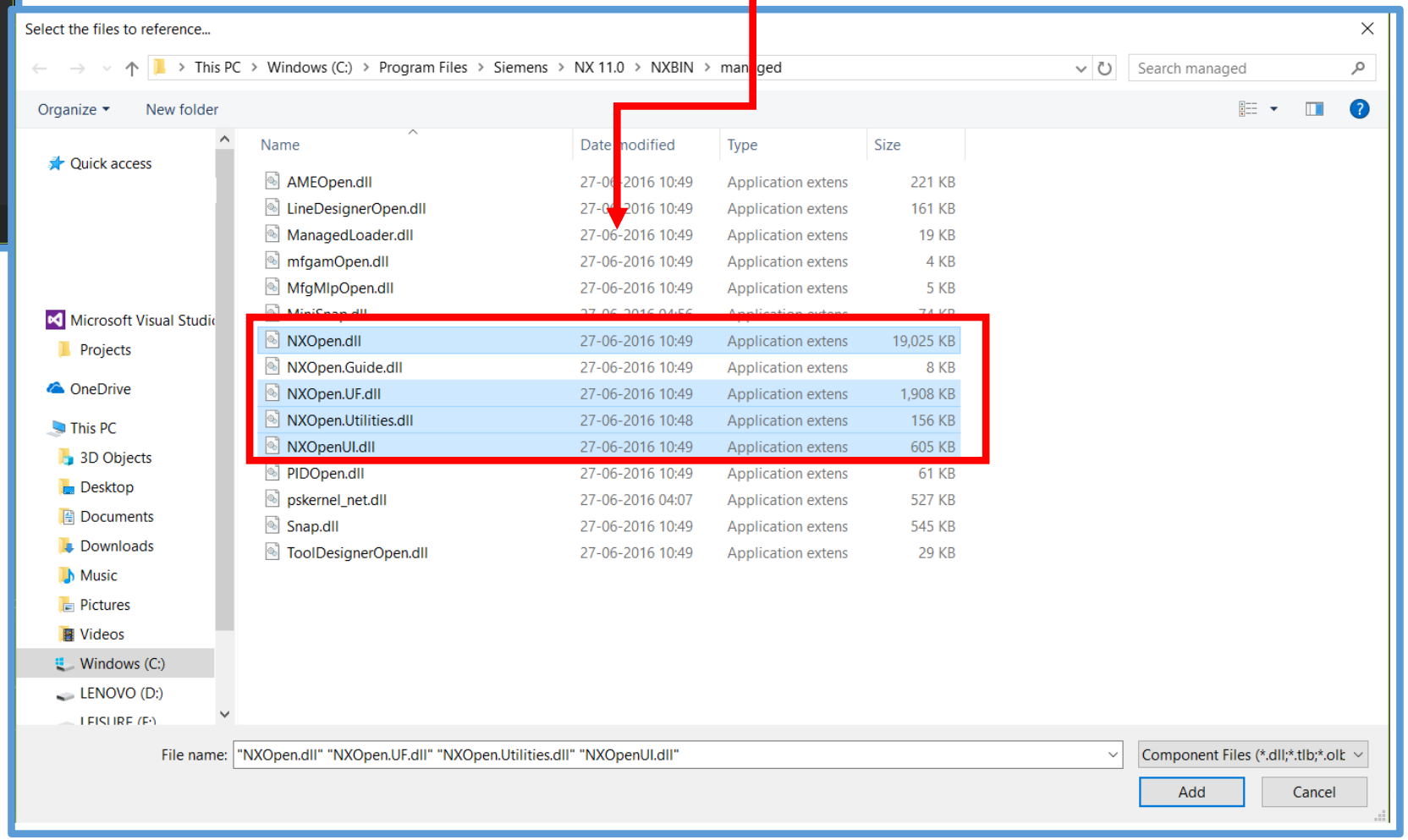


Select the 4 dll Files From below Folder

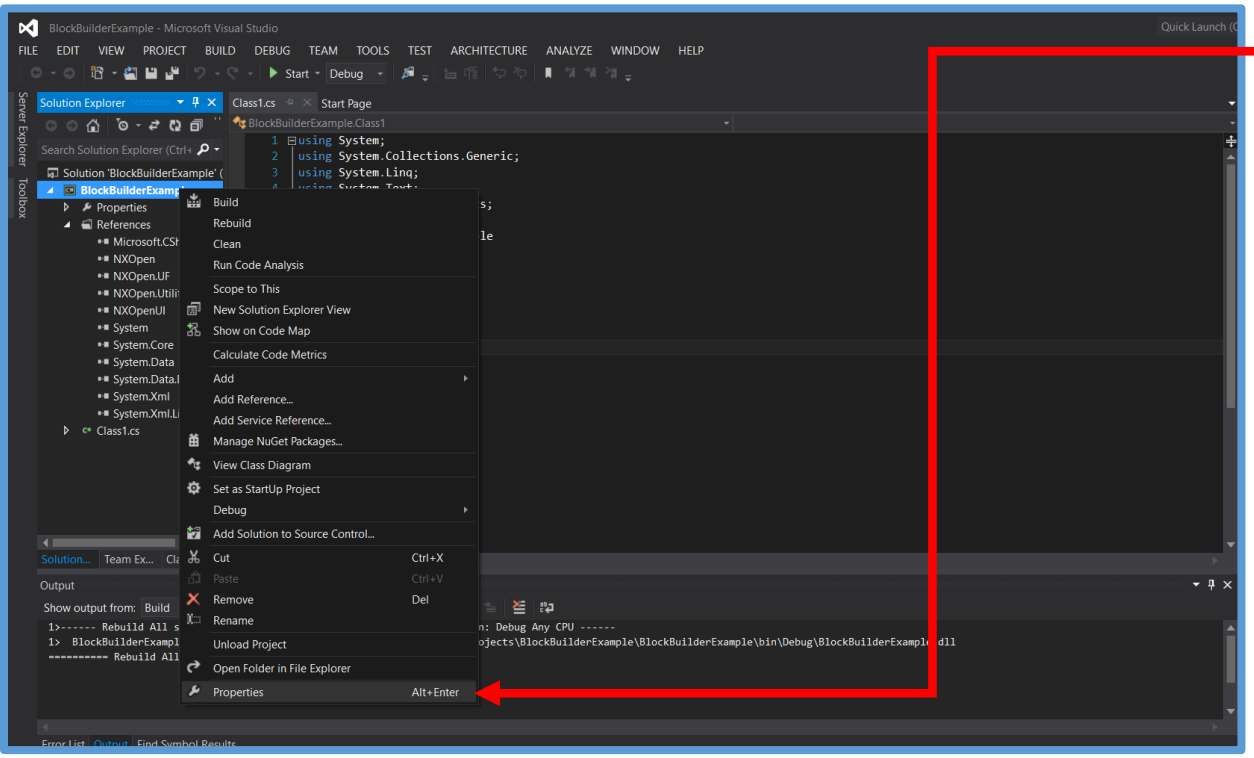
C:\Program Files\Siemens\NX (version)\NXBIN\managed

Example:

C:\Program Files\Siemens\NX 11.0\NXBIN\managed

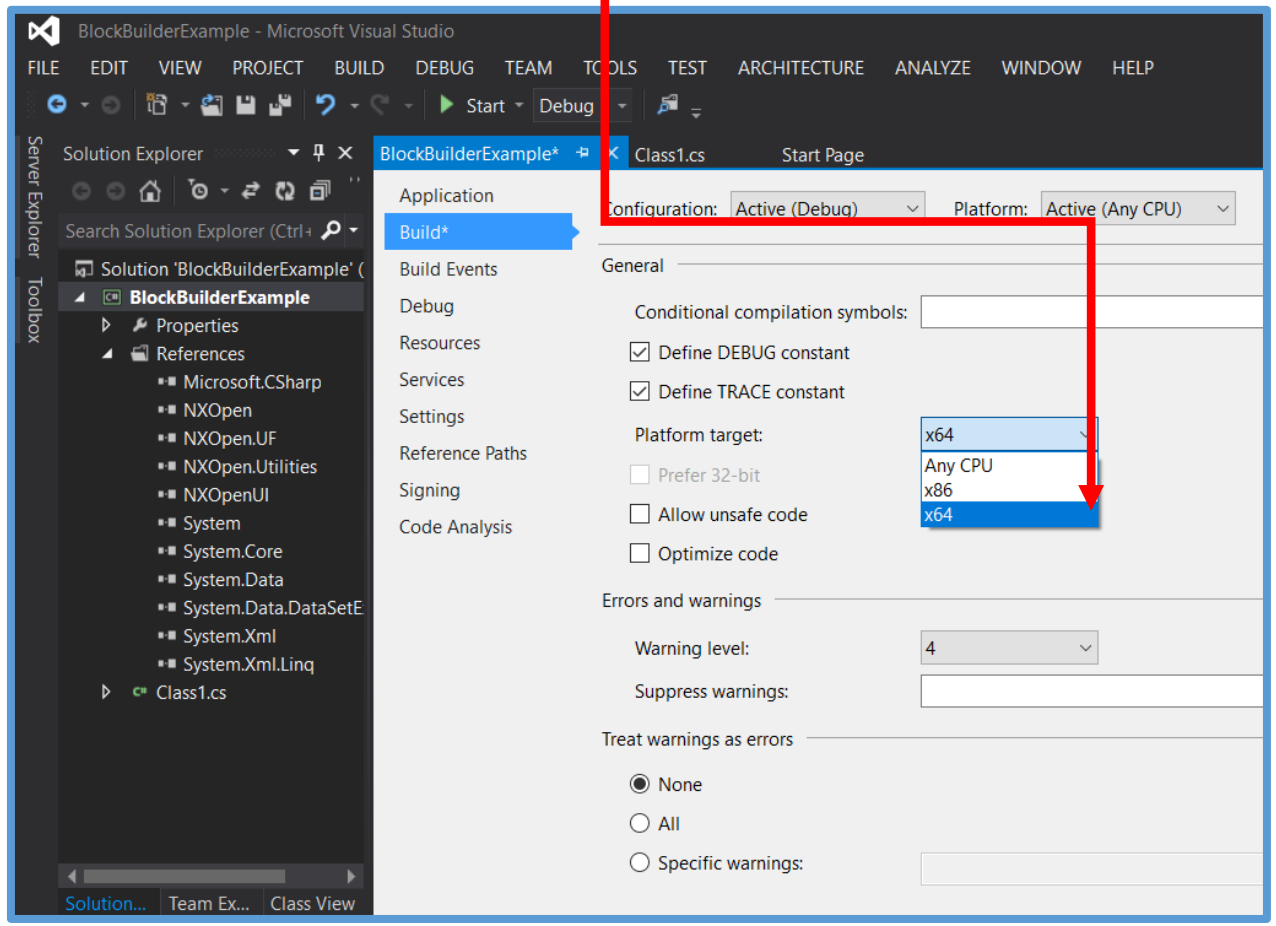


Step 11: Change Properties of Solution



Right Click the Solution and Select Properties

Go To Build->Change Platform Target to X64



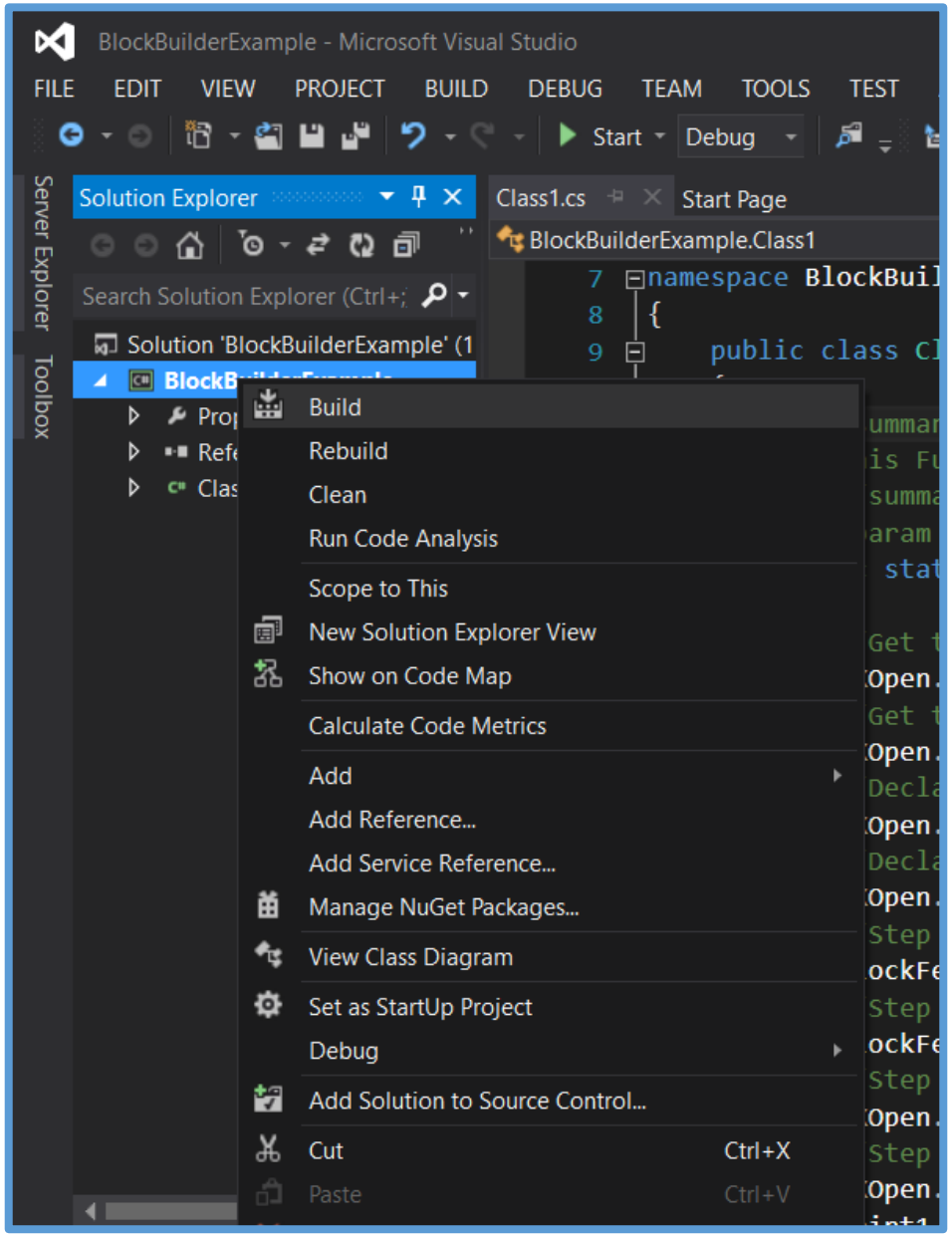
Step 12: Write the Program To Create a Block

Read the Lines with Comments and Understand the Logic

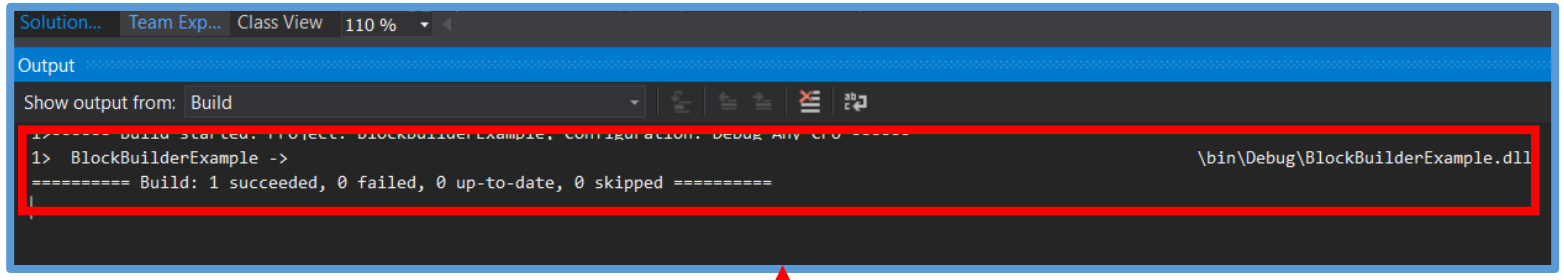
BlockBuilderExample.Class1

```
7   namespace BlockBuilderExample
8   {
9       public class Class1
10      {
11          /// <summary>
12          /// This Function Creates a Block In NX using C# .net Program
13          /// </summary>
14          /// <param name="args"></param>
15          public static void Main(string[] args)
16          {
17              //Get the Current NX Open Session
18              NXOpen.Session theSession = NXOpen.Session.GetSession();
19              //Get the Current Work part from NX Session
20              NXOpen.Part workPart = theSession.Parts.Work;
21              //Declare and initialize feature Class Object
22              NXOpen.Features.Feature nullNXOpen_Features_Feature = null;
23              //Declare  BlockFeatureBuilder Class Object
24              NXOpen.Features.BlockFeatureBuilder blockFeatureBuilder=null;
25              //Step 1-initialize object for blockFeature
26              blockFeatureBuilder = workPart.Features.CreateBlockFeatureBuilder(nullNXOpen_Features_Feature);
27              //Step 2-Set  BooleanOption
28              blockFeatureBuilder.BooleanOption.Type = NXOpen.GeometricUtilities.BooleanOperation.BooleanType.Create;
29              //Step 3 -Create Point for Origin
30              NXOpen.Point3d coordinates1 = new NXOpen.Point3d(0.0, 0.0, 0.0);//This Is Value
31              //Step 4-This is Object for Point.It should be Created from Point3d coordinates.
32              NXOpen.Point point1 = null;
33              point1 = workPart.Points.CreatePoint(coordinates1);
34              //Step 5-BlockFeatureBuilder Type
35              blockFeatureBuilder.Type = NXOpen.Features.BlockFeatureBuilder.Types.OriginAndEdgeLengths;
36              //Step 6-Set OriginPoint for blockFeature
37              blockFeatureBuilder.OriginPoint = point1;
38              //Step 7-SetOriginAndLengths for blockFeature
39              blockFeatureBuilder.SetOriginAndLengths(coordinates1, "100", "100", "100");
40              //Step 8-Now all Inputs are set.So we Can Commit Feature to get the output
41              NXOpen.Features.Feature feature1;//feature Object -Output
42              //Commits the feature parameters and creates the feature
43              feature1 = blockFeatureBuilder.CommitFeature();
44              //Step 9-Finally Destroy the builder.
45              //Deletes the builder, and cleans up any objects created by the builder
46              blockFeatureBuilder.Destroy();
47
48          }
49          //Unload the DLL after Execution
50          //This Function Important otherwise you could not able to complile
51          // this program if you make any change with NX Session in open.
52          public static int GetUnloadOption(string dummy)
53          {
54              return (int)NXOpen.Session.LibraryUnloadOption.Immediately;
55          }
56
```

Step 13: Build the Solution



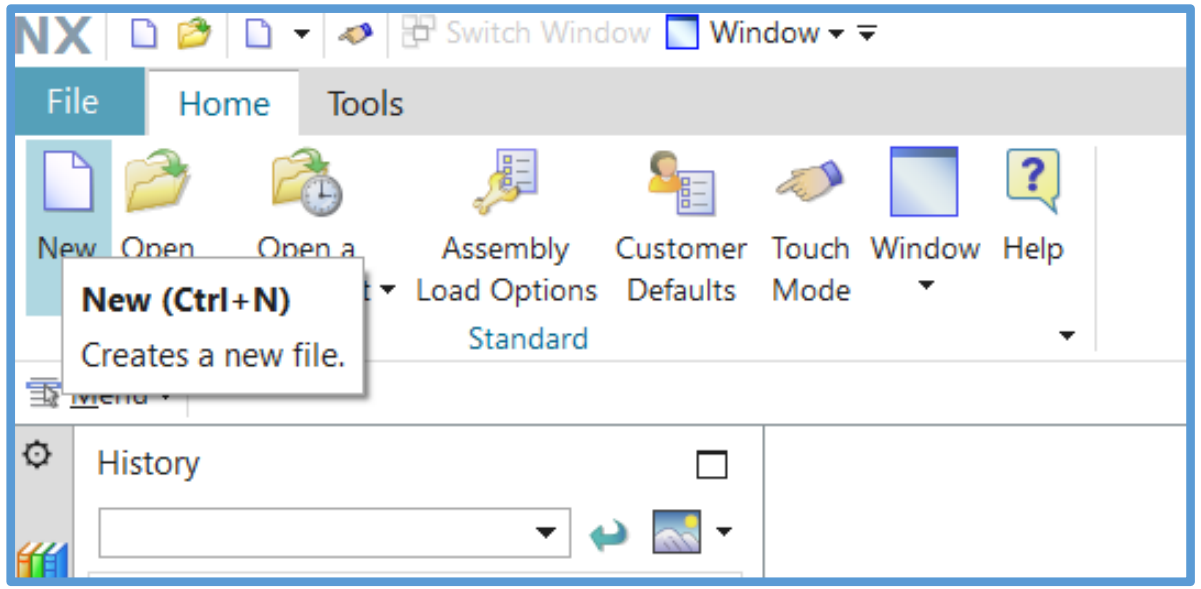
If succeeded ,Below window will be shown at the Bottom



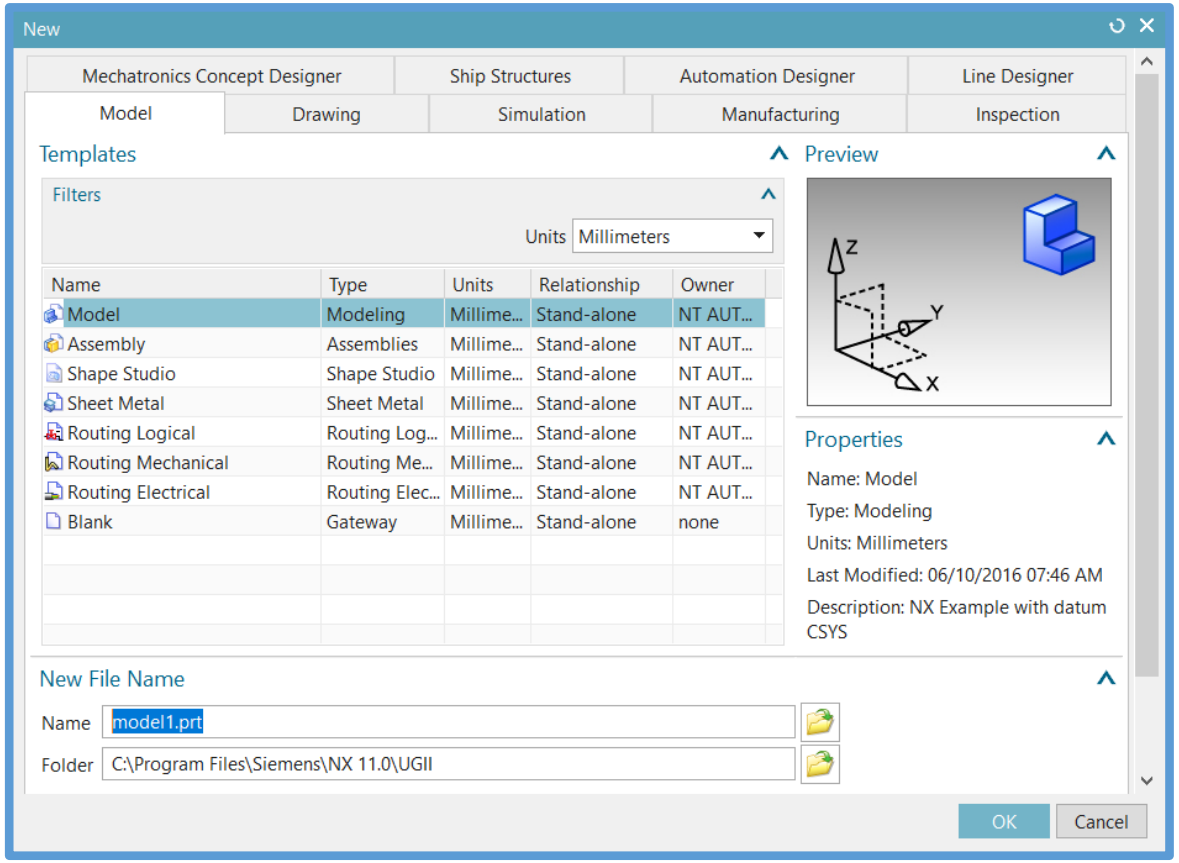
Path of your output file.
In this case it is BlockBuilderExample.dll

Step 14: open NX and Create file

Create a new file in NX as Below



Assign file name for model



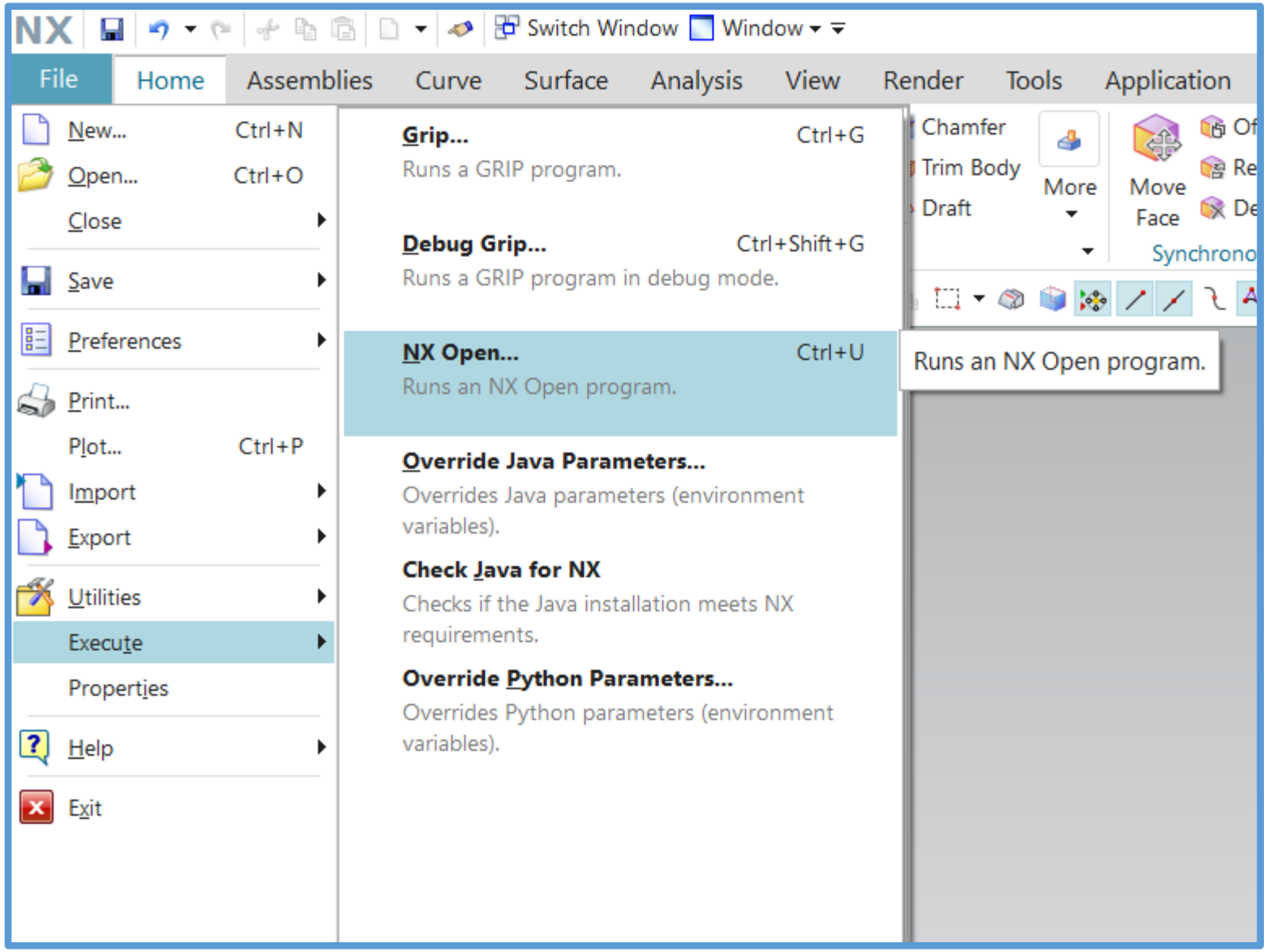
Step 15: Execute NX Open File

Method 1:

Go to File->Execute->Select NX Open

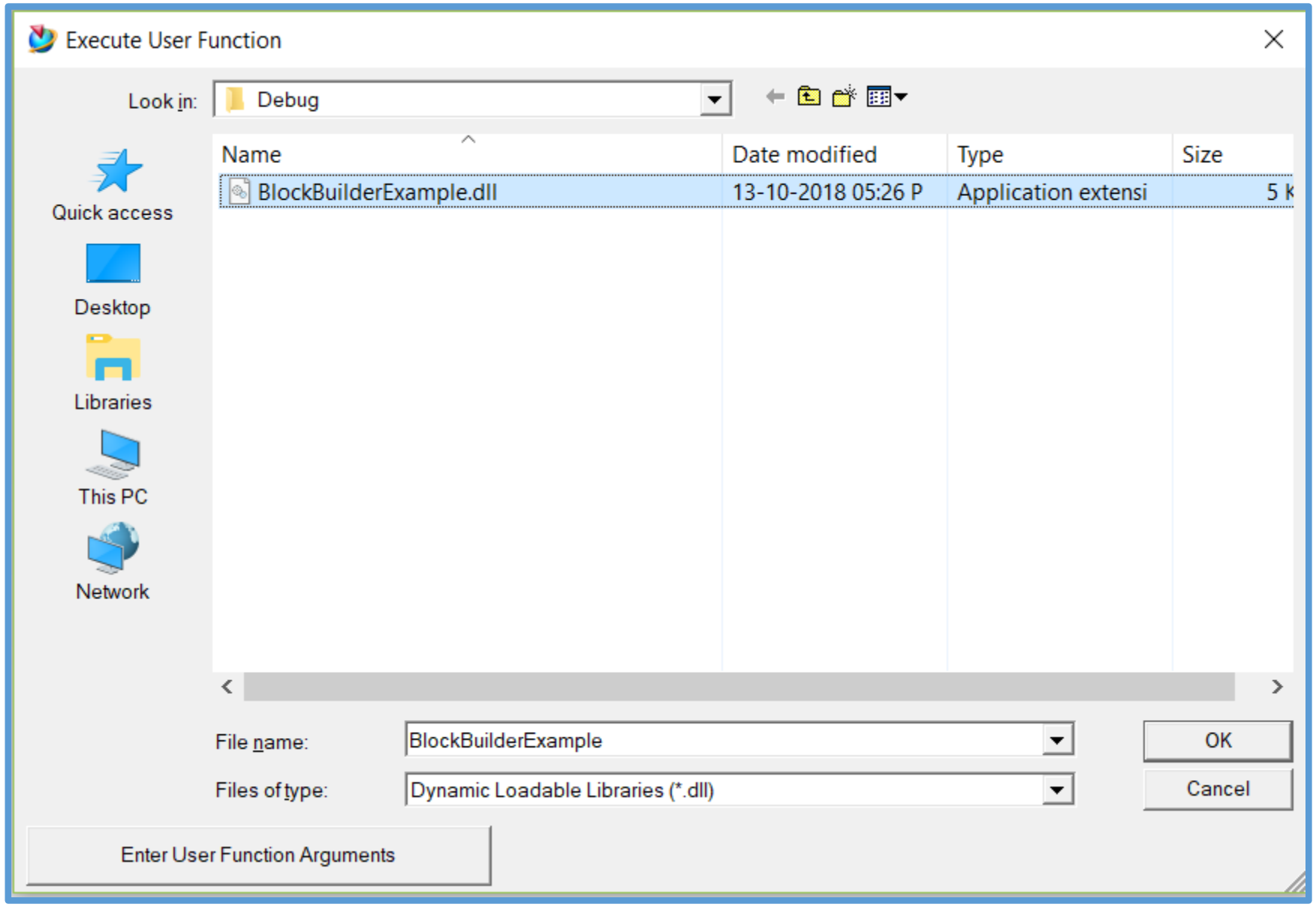
Method 2:

Cntrl+U



Step 16: Select your application

Go to the location mentioned In Step 13 and Select the Application ->Click OK to Run



Step 17: Final Output

A block Created as per the Code written in visual studio