AR16178

Dynamo – Everyone's Doing It – V1.1

Carl Storms – Senior Applications Expert
IMAGINiT Technologies

Learning Objectives

- Learn about Dynamo
- Find out some of the many workflows Dynamo can improve
- Gain a basic understanding of the Dynamo user interface
- Have fun

Description

This session is not for people who are heavy Dynamo software users. This session is for people who aren't programmers (visual or other), but from time to time may need to use Dynamo as part of their workflows. It's also for people who have heard of Dynamo, or visual programming, but never thought it was something they could do (but it totally is!). It's also for those who just don't think Dynamo can help them in their everyday workflows (don't be so sure). By the end of this session, you will have an introductory understanding of Dynamo. You will learn that Dynamo can be used to create geometry with parametric relationships and that this geometry can then be pushed into a Revit software project or family using the Revit software database. You will learn that Dynamo can do more than just geometry. It can also help with everyday tasks like creating grids and levels, adding some trees to your site, performing an energy analysis, or even changing your text from sentence case to all CAPS. This session features Revit and Dynamo

Your AU Expert:

Carl Storms

Drawing on his nearly 20 years of experience in architecture, engineering, and construction, **Carl Storms** provides a practical and well-rounded understanding of Building Information Modeling (BIM) to clients. He's worked in residential and commercial architecture, as well as in construction, and he has over 5 years of teaching experience at the collegiate and industry levels. Beyond that, he's also done a bit of sales and marketing, which aid him in providing the business case for BIM to clients as well as helping them make the most of their collaboration, coordination, and design tools and processes.

Through implementations, instruction, mentoring, seminar presentations, whitepapers and more, Carl assists clients with the adoption of design technology and BIM processes. As someone who truly enjoys the process of building information modeling, Carl spreads his love of all things BIM via Twitter @theBIMsider, on his Blog www.thebimsider.com , or on Facebook at www.facebook.com/theBIMsider

Session Handout and Presentation

Get the most current PDF version of this handout and presentation using the Dropbox link below: https://www.dropbox.com/sh/32u1u63ipyj0toq/AACt40yW1jKyxBCb-Nq9jQIsa?dl=0

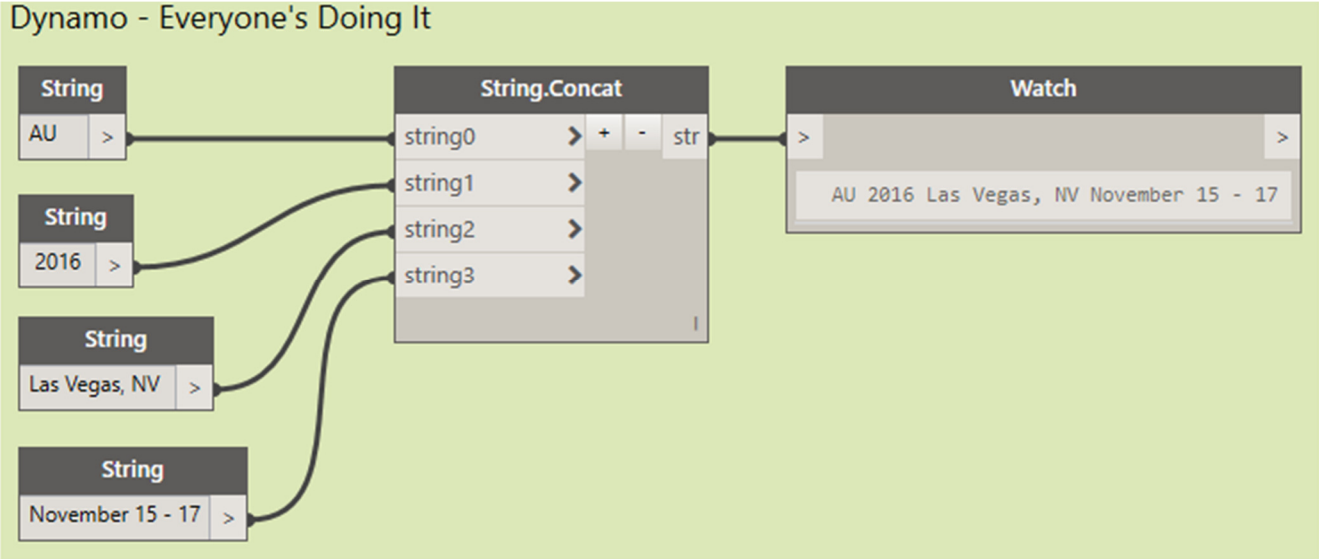
Table of Contents

What is Visual Programing

Visual Programing lets users create programs by manipulating program elements graphically rather than by specifying them textually.

Is Dynamo Visual Programing?

Yes. Dynamo is a general purpose graphical algorithm editor that can be used to extend the parametric functionality of other applications, including Autodesk Revit. Dynamo is developing at a faster rate than many tools in architecture, engineering, and construction (AEC).



- Other Visual Programing Options
  - LabVIEW
  - Lego Mindstorms (kids)
  - Scratch (kids)
  - App Inventor
  - Grasshopper (works with Rhino)
  - FLUX (works with Dynamo, Excel, Grasshopper, Revit, SketchUp)

What is Dynamo?

"Open-source Dynamo is a visual programming extension for Autodesk® Revit that allows you to manipulate data, sculpt geometry, explore design options, automate processes, and create links between multiple applications."

This is the description of Dynamo right from the horse's mouth, ok the Dynamo website.

Here is another definition of Dynamo from the Dynamo Primer

"Dynamo is, quite literally, what you make it. Working with Dynamo may include using the application, either in connection with other Autodesk software or not, engaging a Visual Programming process, or participating in a broad community of users and contributors."

But really Dynamo started because of the question: Wouldn't it be cool if we could make this technology talk to this technology.

- Dynamo is 100% Open Source Code!
- Dynamo can interface with other programs using their API (Application Programing Interface)
- At its core, Dynamo is a logic and mathematics processing engine

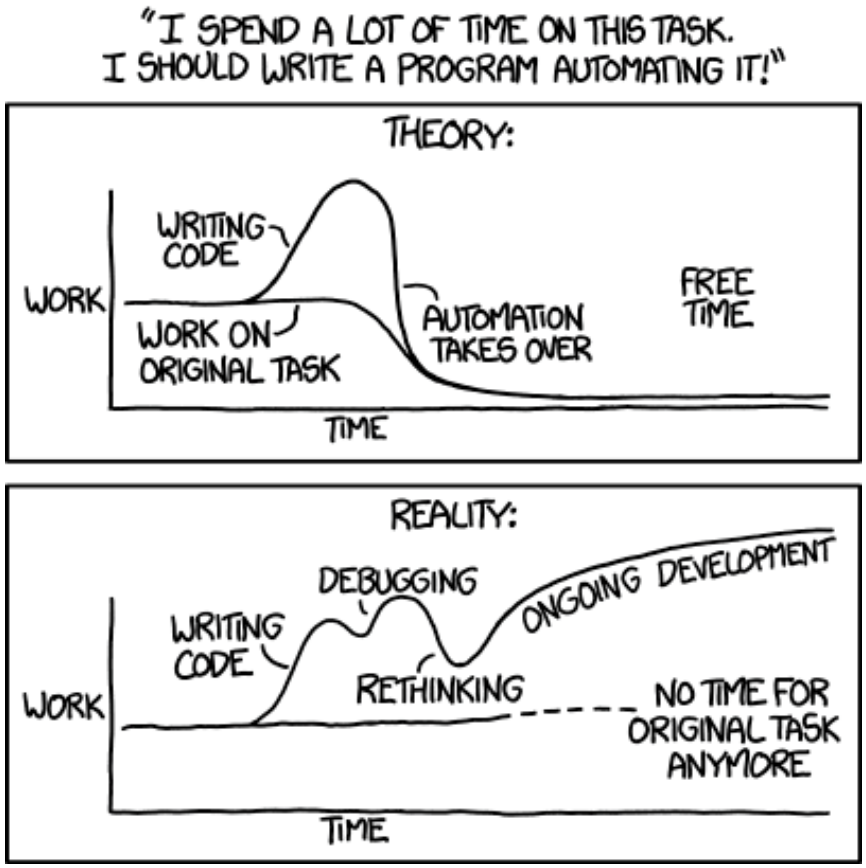
Now that you know what Dynamo is, how do you work with Dynamo? Simply put working with Dynamo is writing "Algorithms".

- Algorithms are a procedure, process, or a formula that's followed exactly in order to solve a problem.
- Writing these algorithms within Dynamo is a lot like writing a recipe.
    - Ingredients (inputs)
    - List of instructions
    - Outcome (output)

Why Dynamo?

For many, using the same tools over and over again to complete various mundane tasks, proves to be inefficient and tedious. These repetitive tasks can take a lot of time for something that could be simple and direct. So why do we use Dynamo, well the answer is because...

- Because every building system should be able to talk to every other system
- Because Revit parametric families can't talk to other Revit families
- Because Revit shouldn't just be used to make drawings
- Because the power to do crazy stuff is waiting to be tapped
    - The most growth in Revit over the last few years has been with the API
- Because design and analysis should happen at the same time, in the same model
- Because computational design is not just sexy geometry
    - Should also be about process automation



Source: http://xkcd.com/1319/

Dynamo, not just for Revit anymore

While most people are familiar with Dynamo as an add-in for Revit, it is able to interact with many more programs that just Revit. Below is just a partial list of what's out there, and the list is being added to all the time.

- **BumbleBee** = Provides enhanced Excel to Dynamo functions
- **DynaWorks** = Dynamo in Navisworks
- **Illustrator** = Work with swf files in Dynamo
- **Lunchbox** = Package of utilities that make shortcuts for computational design
- **Mantis Shrimp** = Dynamo for Rhino and works with Grasshopper
- **Optimo** = Multi-objective optimization
- **Rhynamo** = Manage complex geometry (does not need Rhino on your computer)
- **Slingshot!** = Not so sexy data manager plugin (write SQL in Revit)
- **Solar Analysis for Dynamo** = The name says it all
- **Unfold** = Make stuff flat

How to get Started in Dynamo

The best way to get started with Dynamo is to just simply start. Begin by opening it up and exploring the interface, maybe check out some of the samples that come with Dynamo. Explore the node library and start to learn where things are and what they might mean. Of course keep in mind that things might not mean what they appear to mean. You should also start small, going too big too fast may become overwhelming and turn you off of Dynamo.

- Check out the getting started videos from DynamoBIM.org/learn
- Read "The Dynamo Primer" the unofficial online user's manual
- Check out the "Dynamo Dictionary" a searchable database for Dynamo functionality.
- Use and play with the out of the box stuff that comes with Revit/Dynamo
    - It can also be easier if you have a problem that you want to solve
- You can also find/give answers on the DynamoBIM.org Forum
    - https://forum.dynamobim.com/

Understanding Dynamo

Here is a quick list of things you should be aware of before you get started with Dynamo.

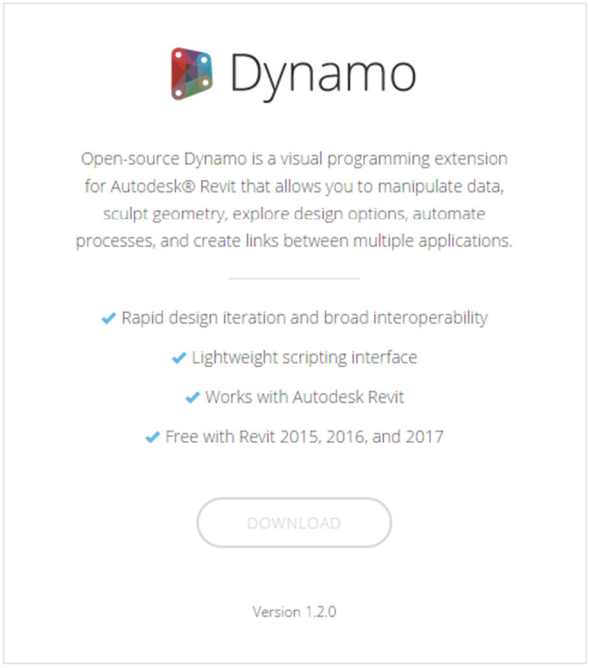
- Dynamo changes so quickly that it has daily builds released
    - Daily builds are always changing and may not be stable
- Dynamo releases stable builds about every 2-3 months
    - Current stable build is: 1.2.0 (as of October 2016)
    - The most current stable downloads can be found here
- Dynamo is a free program from Autodesk
    - There is a paid version called Dynamo Studio – subscription based
- The Dynamo add-in for Revit, allows Dynamo to access the Revit API (works with 2015-2017)
    - The latest Dynamo add-in can be downloaded here
        - In 2016 the add-in comes pre-installed in Revit under the "Add-ins" tab
        - In 2017 the add-in comes pre-installed in Revit under the "Manage" tab

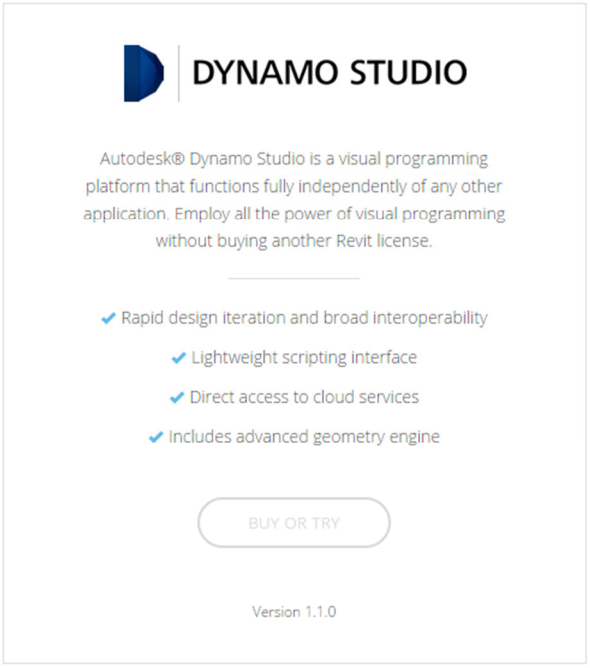
NOTE: After Dynamo 1.2, Dynamo will discontinue installation of Dynamo for Revit 2015.

Installing Dynamo

You can Download the latest version of Dynamo from the Dynamo website under "Download"

- Install options
  - Full installation will install all components
    - Core, Training Files, and all versions available
  - Compact installation will install all components EXCEPT training files
  - Custom install will allow you to select the components you wish to install

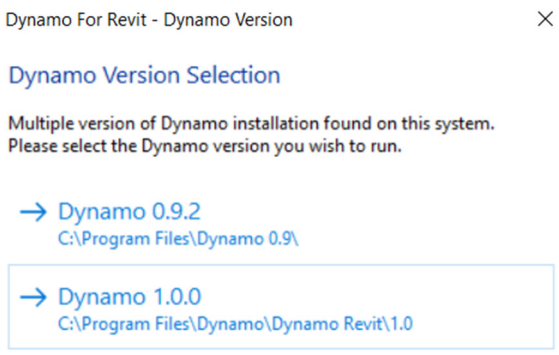




Opening Dynamo

Dynamo can be opened in two ways, as a Revit add-in or as a standalone product.

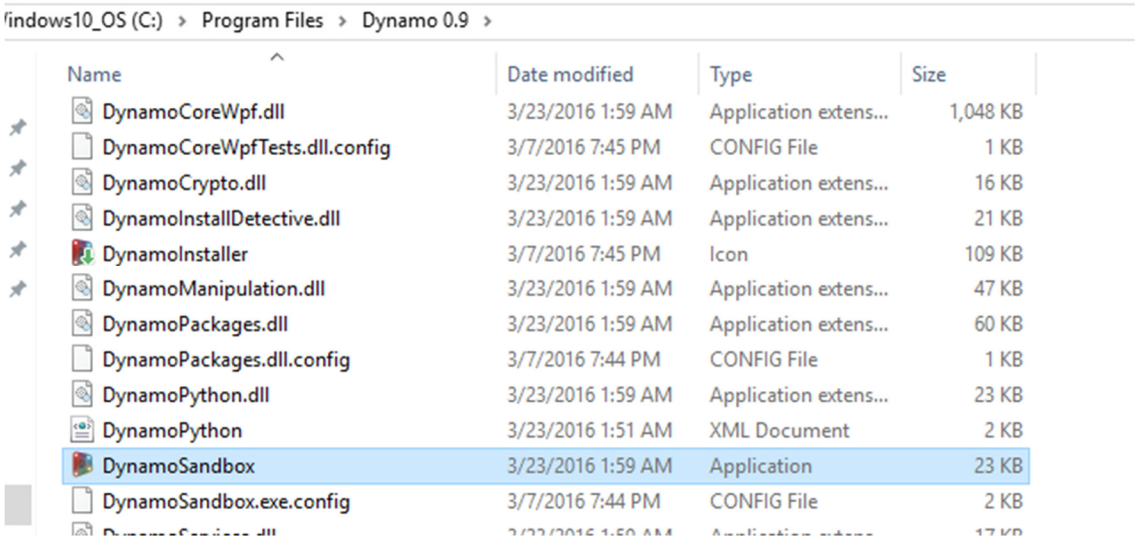
- Dynamo can be opened directly from the Revit Add-Ins Tab, or the Manage Tab in Revit 2017
  - If you have multiple versions of Dynamo you will get a dialog box asking you to choose a version



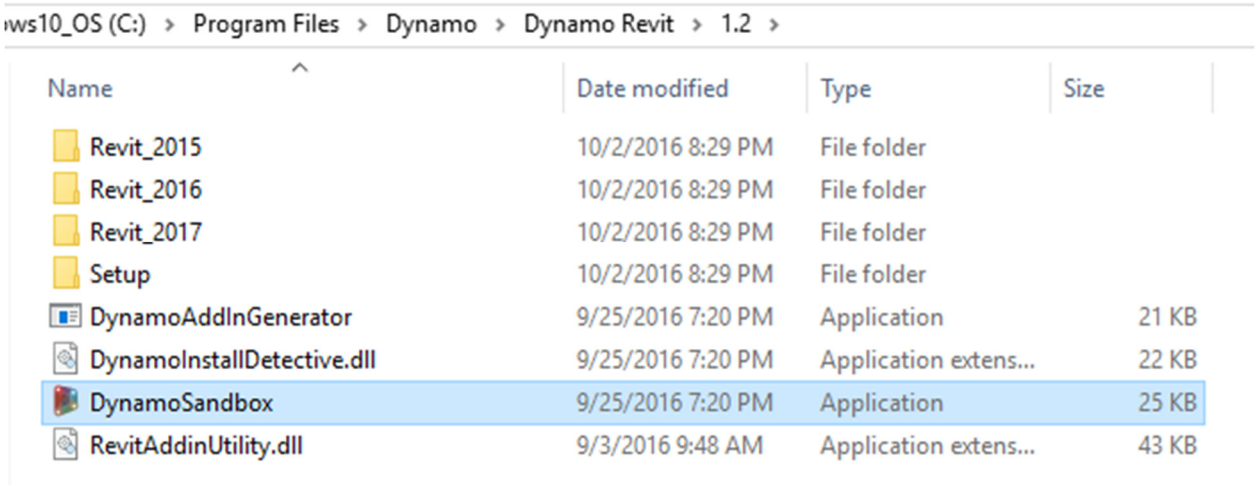
- When opened in Revit, Dynamo will automatically be associated (connected) to that Revit model or family you have open

You can open Dynamo without opening Revit by looking for the "Dynamo Sandbox" version within the program folder.

- Program Files > Dynamo > Dynamo 0.9 (for versions 0.9.2 and earlier)



- Program Files > Dynamo > Dynamo Revit > 1. (for Versions 1.0 and higher)

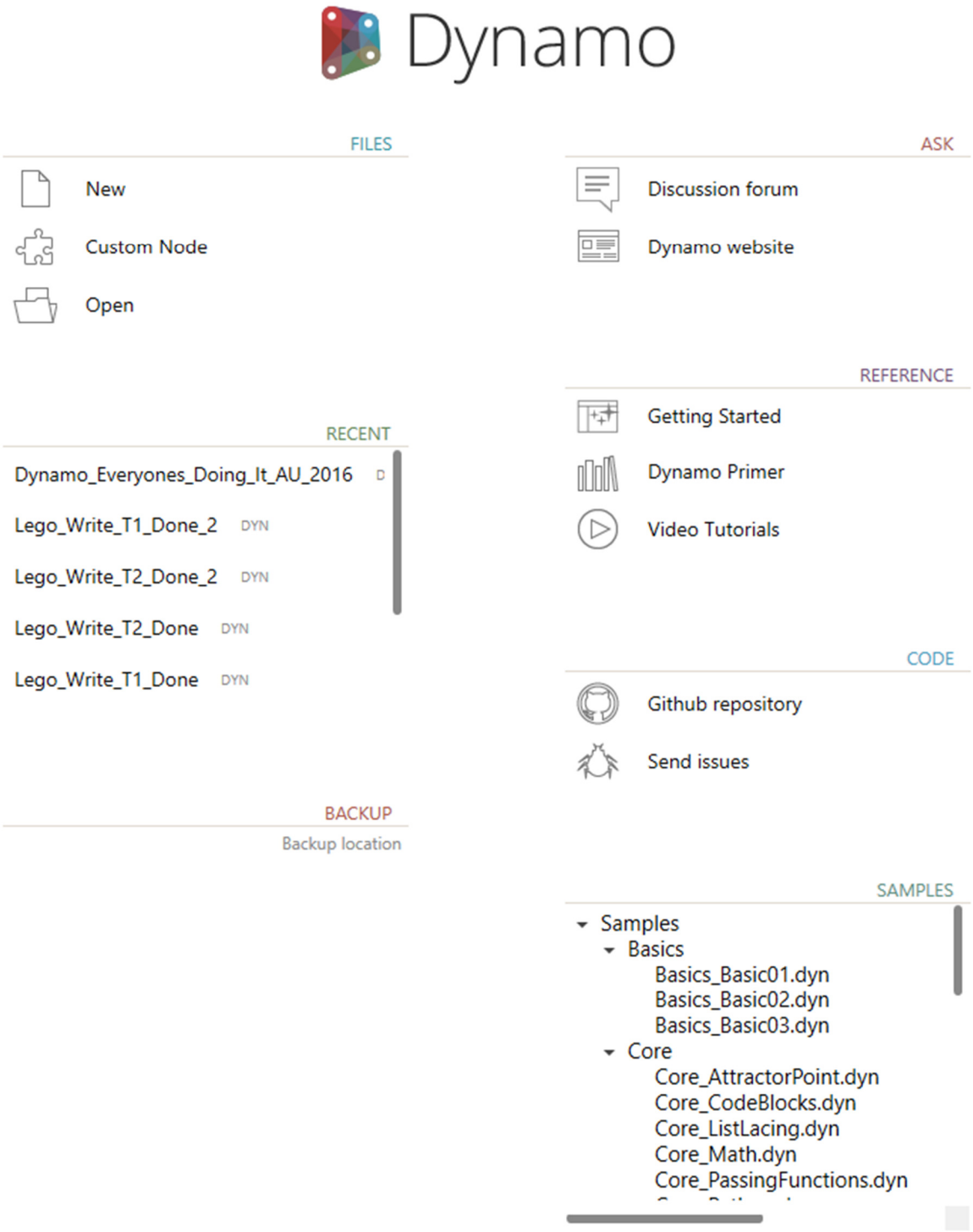


Using the Start Page

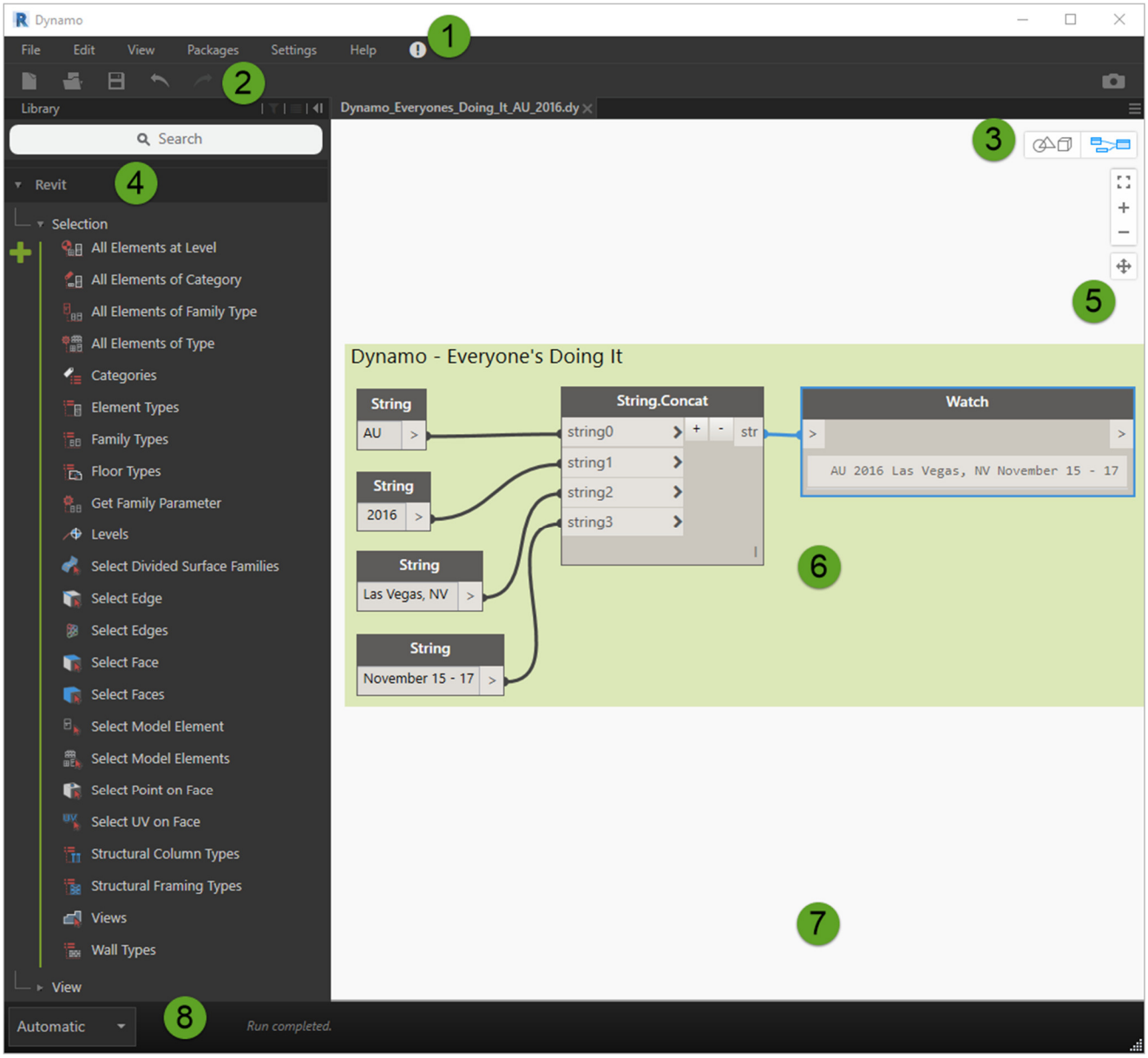
When you launch Dynamo there is a Start Page that opens before you begin working on your graph (file).

- The start page basically has 2 sides
  - The Left side provides tools to create new files and open existing files
  - The Right side provides links to help files and other online references
- The Discussion Forum is a great place to learn and have questions answered by the Dynamo community

- To return to the start page from an open file select the “Help” tab, then select “Display Start Page”
    - Be sure to save your work before returning to the start page.



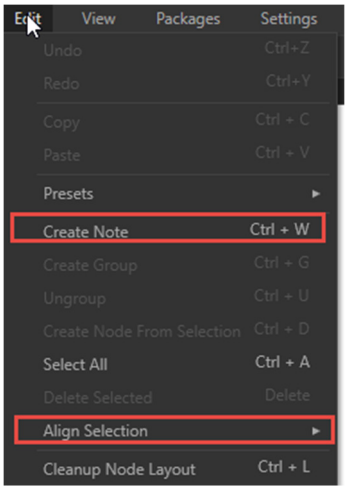
Explaining the User Interface



1. **Menu Bar** = Dropdown menus that give you access to basic Dynamo functions
2. **Toolbar** = Quick access to key Dynamo functions
3. **Workspace Toggle** = Toggle between geometry view and graph view
4. **Library =** Where the nodes are kept
5. **Zoom Options** = Zoom extends, Zoom in/out and Pan
6. **A Graph =** A Dynamo program or file
7. **Workspace =** The canvas for what you create in Dynamo
8. **Execution Bar =** This is where you run you Dynamo programs (Auto, Manual, or Periodic)

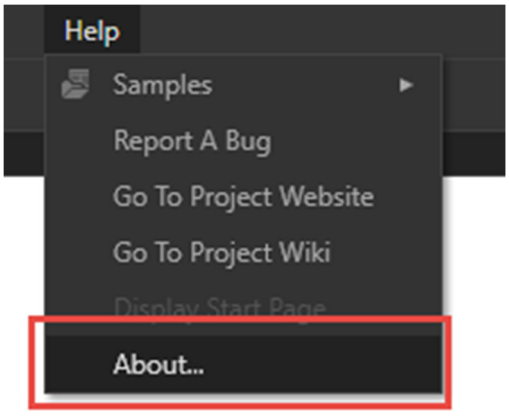
There are some useful settings under the “Edit” tab on the Menu Bar.

- Create Note:
  - This allows you to create a note on the graph (canvas).
  - The keyboard shortcut is: Ctrl + W
- Align Selection:
  - This allows you to align and organize your nodes in your graph (canvas)
  - This can also be accessed via a right-click menu within the graph (canvas)



If you are asking questions about Dynamo (like on the Dynamo Forum) be sure to include the version of Dynamo you are using.

- You can find your version under the Help tab > under About…



Nodes, Groups and Organizing your Definition

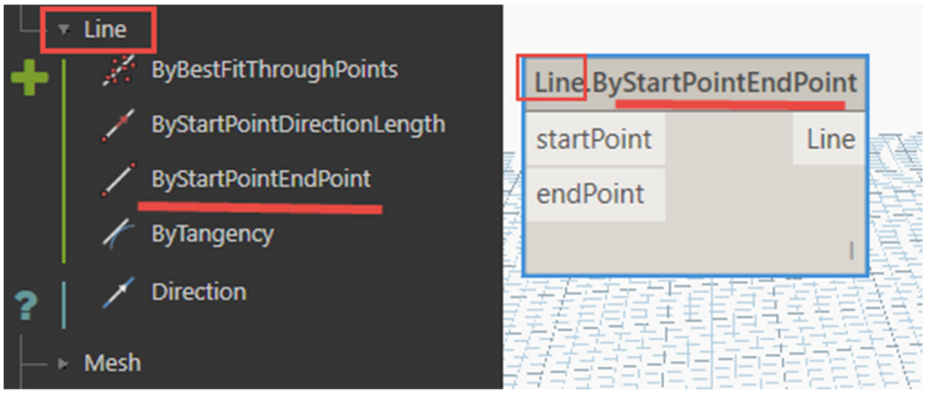
Definition

Definitions are what your finished Dynamo programs or graphs are called. They are the ".dyn" files you open to run a Dynamo program.

What's in a name

90% of the time the Nodes name will begin with the Sub-category that it belongs to.

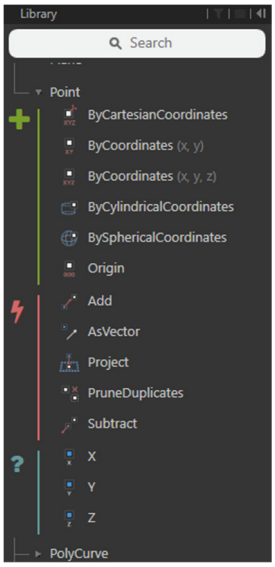
- Then it will have a short description of what it does
  - Example: Line.ByStartPointEndPoint



Library

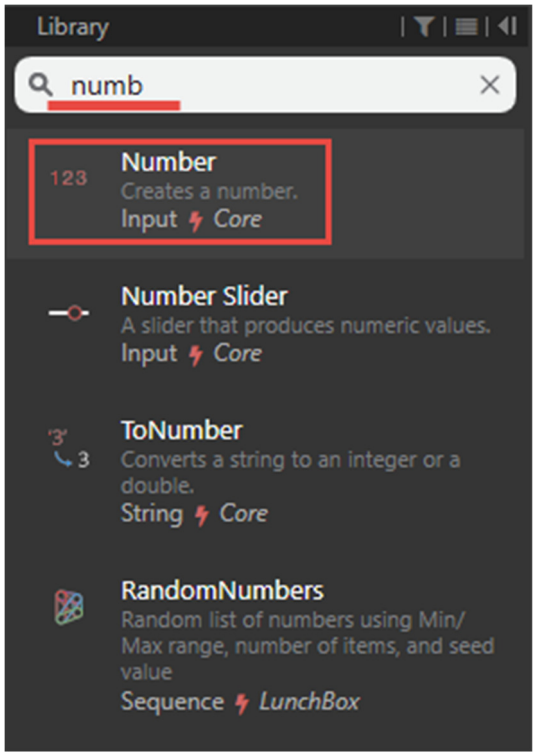
In the Library, nodes can be under 3 "areas" or "types".

- Create =  , Action =  , Query = 

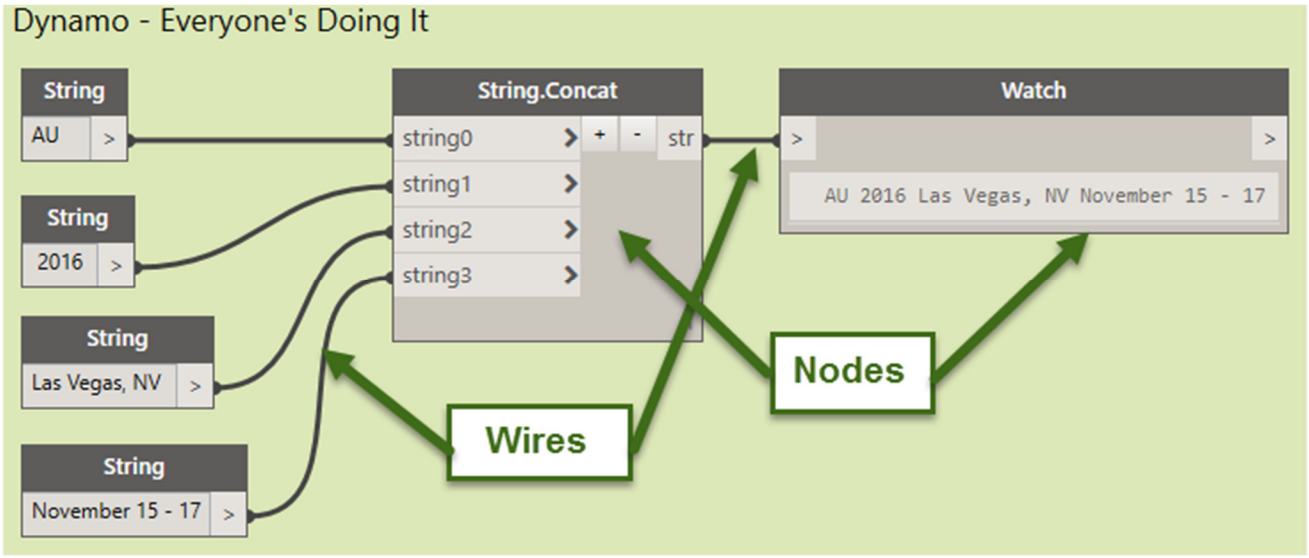


How do I find a node in the Library if I don't know where it is, or what heading it is under?

- You can type what you are looking for in the "Search" box at the top of the library.



Nodes & Wires

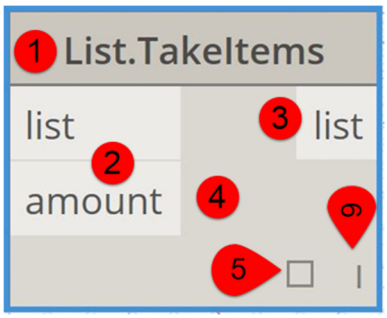


Dynamo is a visual programming language, in order to create a graph (or program) nodes are placed as the source of the information. Once these nodes have been placed on the canvas they need to be connected in order for the information to be processed. They are connected with connectors or "wires".

- Information in Dynamo flows the same as we read, from left to right

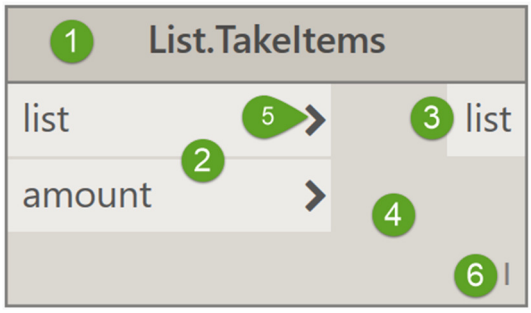
Nodes are generally made up of 6 parts:

Version 1.1 and earlier



1. **Name** – Name of the node
2. **Input** – Where data enters the node
3. **Output** – Where data (results)leaves the node
4. **Main** – Right click here for node options
5. **Data Preview** – Hover or click here for node info
6. **Lacing Icon** – Shows lacing choice (shortest, longest, cross product)

Version 1.2 and after

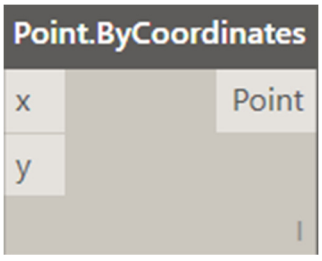


1. **Name** – Name of the node
2. **Input** – Where data enters the node
3. **Output** – Where data (results)leaves the node
4. **Main** – Right click here for node options
5. **List @ Level Menu** – Gives you two options *Use Level* and *Keep list structure.*
6. **Lacing Icon** – Shows lacing choice (shortest, longest, cross product)

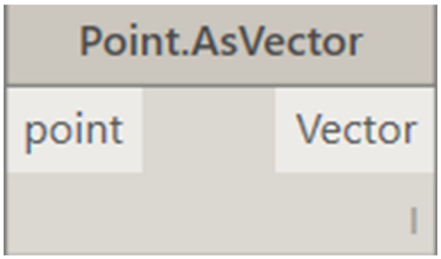
You can connect multiple wires to the output port of a node, but only one to the input(s) of a node.

- Output ports must be connected to input ports
  - Input ports are on the Left
  - Output ports are on the right
- Ports are connected with connectors or “wires”
  - Wires are dashed until they are connected.
    - The wires become solid once connected
  - The ports must be compatible

When a node has a dark header (Title) it means all of the nodes inputs have been satisfied.

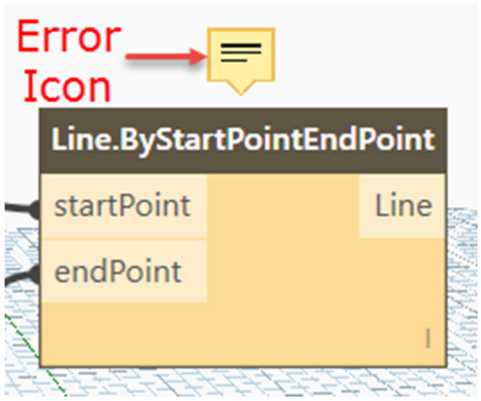


When a node has a light header (Title) it means all of the nodes inputs have not been satisfied.



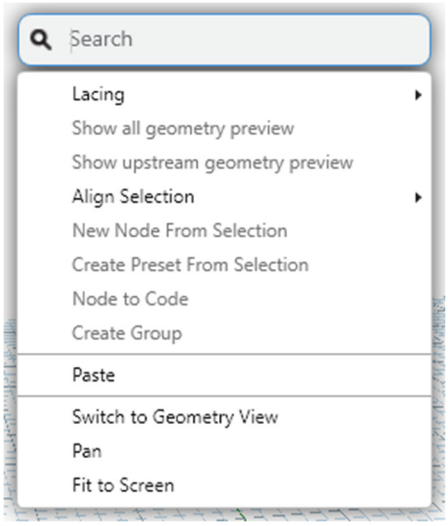
When a node is yellow it means that there is an error.

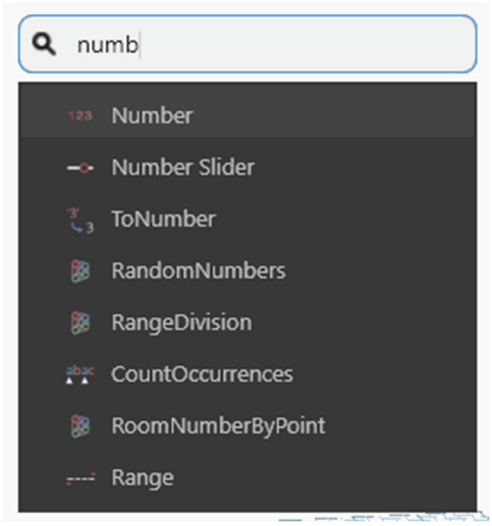
- There will also be a little error icon above to let you know what the error is



How do I find a node if I don't know where it is in the Library?

- You can right click on the canvas, and this will allow you to do a search in the pop-up dialog box.





Getting Custom Nodes

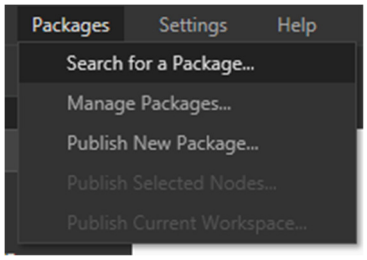
If you are not able to perform the task you wish with out of the box Dynamo nodes you have two options.

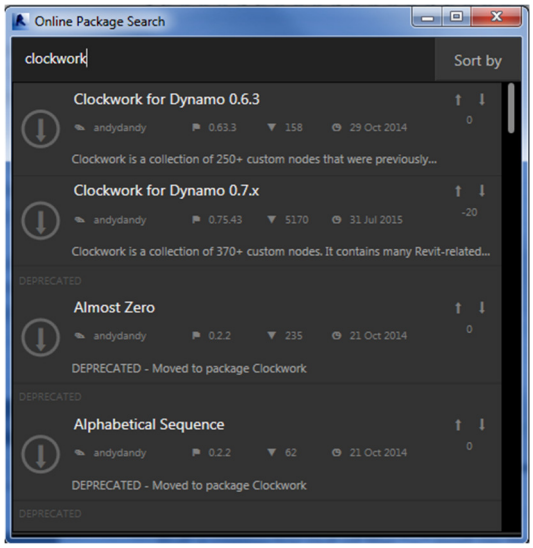
- Create Custom Nodes

- Search for Packages (custom nodes or groups of custom nodes created and shared by others)

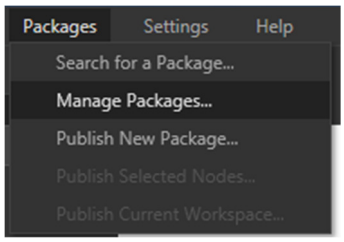
You can search for Packages under the “Packages” tab

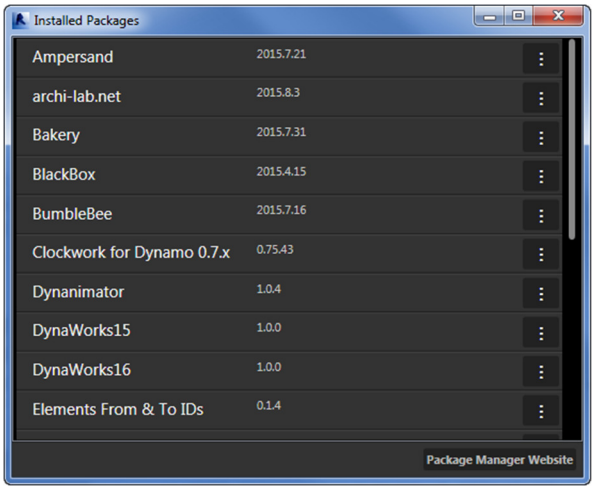
- If you click on a package it will show a drop down with more info





- You can use the “Manage Packages” to manage your packages.

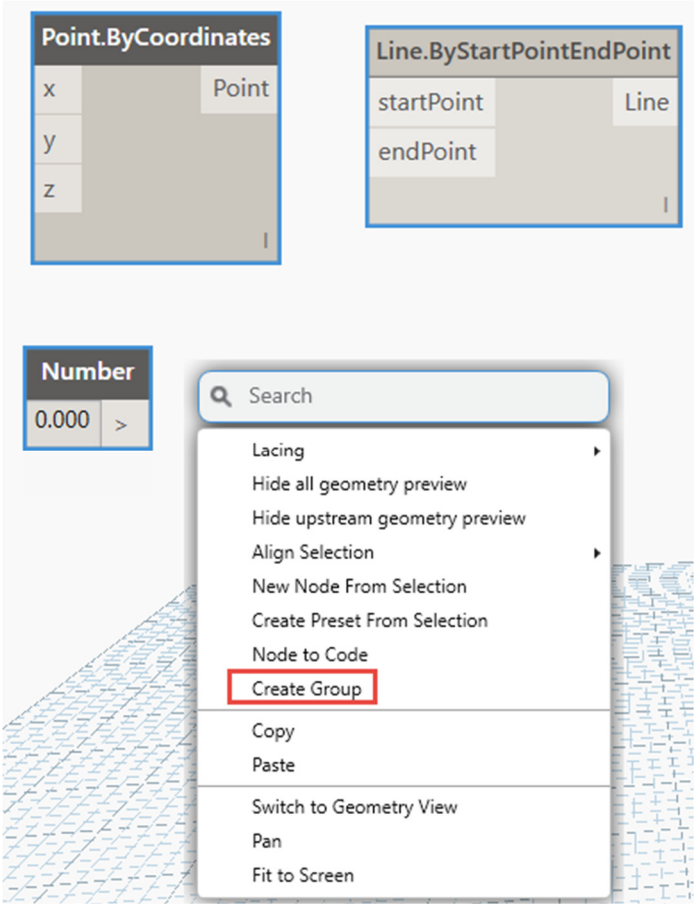




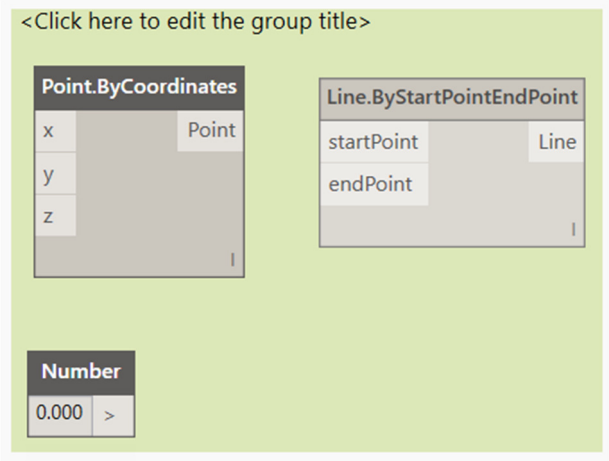
Groups

If you select more than one node you can right click and create a group.

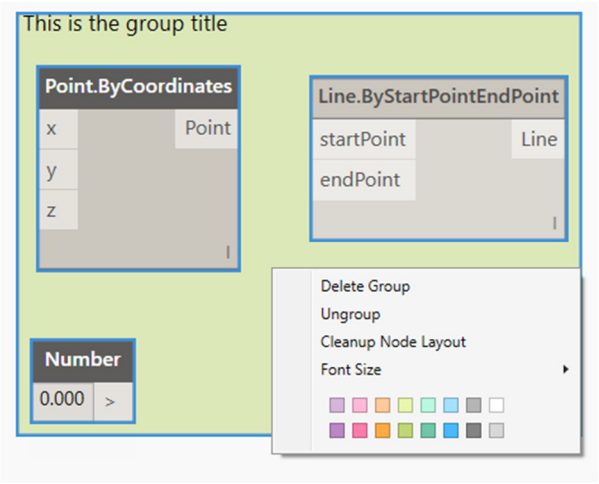
- You can also create a group under the Edit Tab > Create Group
- There is also a Keyboard shortcut “Ctrl + G”



- When the group is created a coloured background and a title are added around the nodes.



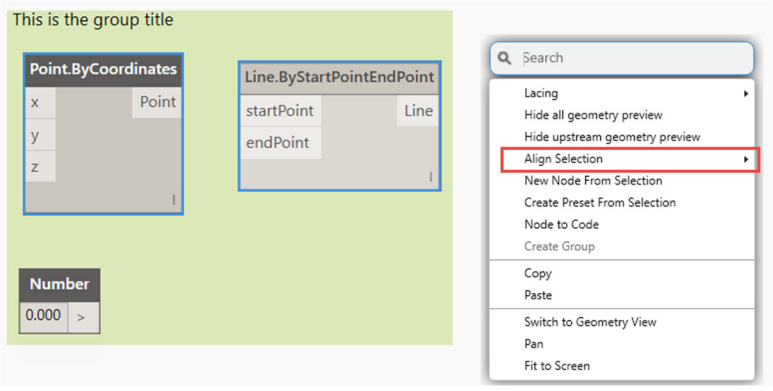
- If you right click on the coloured background of the group, you are given options for the group.

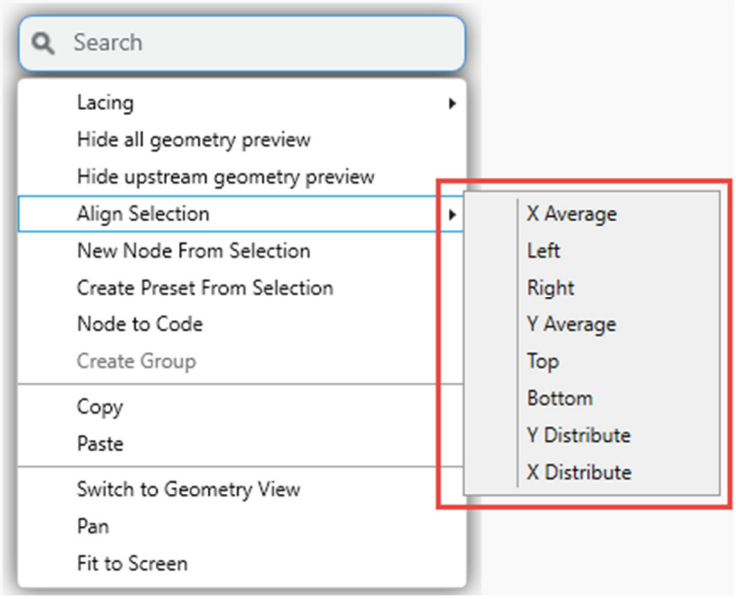


Align

You can clean up your graph or program by aligning the nodes.

- This can be done under the Edit tab > Align Selection > Choose from 8 options
- Or can be done by right-clicking on the canvas

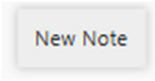




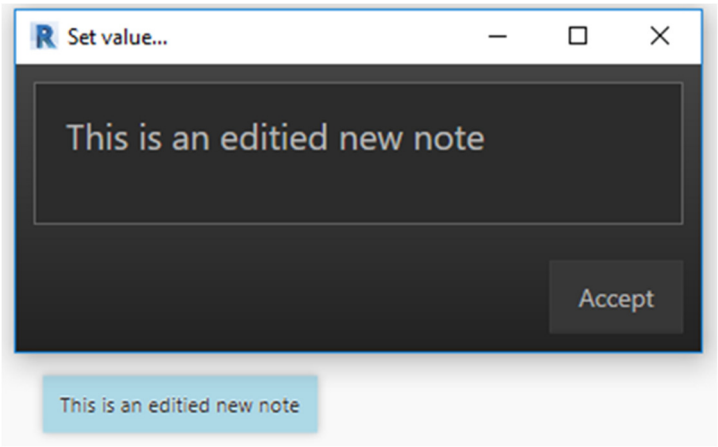
Notes

You can also add notes to your graph or canvas to help clarify it.

- This can be done under the Edit Tab > Create Note
- Or by using the Keyboard Shortcut “Ctrl + W”
    - Any of these methods will give you a “New Note”



    - The new note can be edited (double clicking) and moved to where required on the canvas.



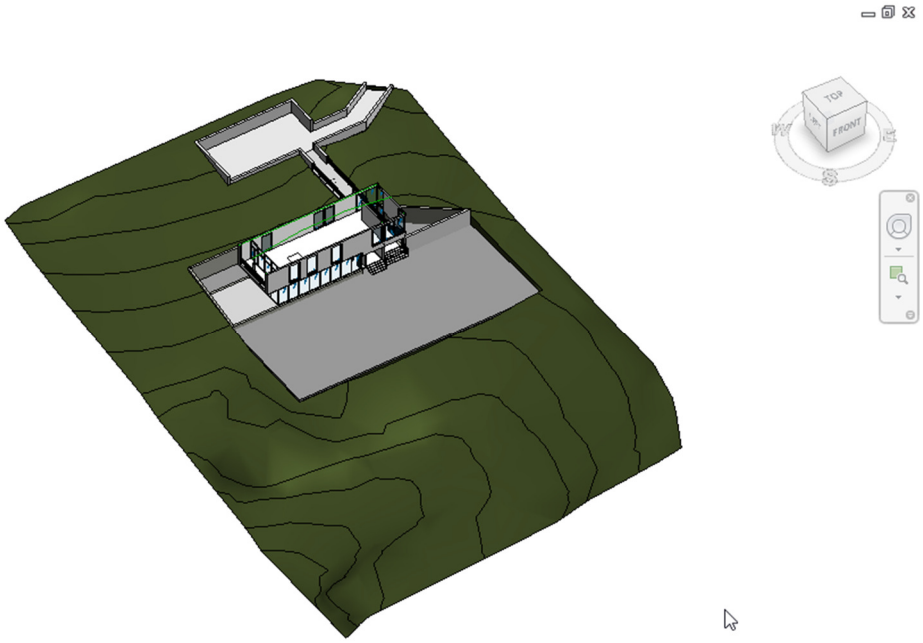
Exercises

Ex 1) OTB Samples

In this exercise, we will become familiar with using Dynamo inside Revit to open existing Dynamo graphs and execute them to see the results.

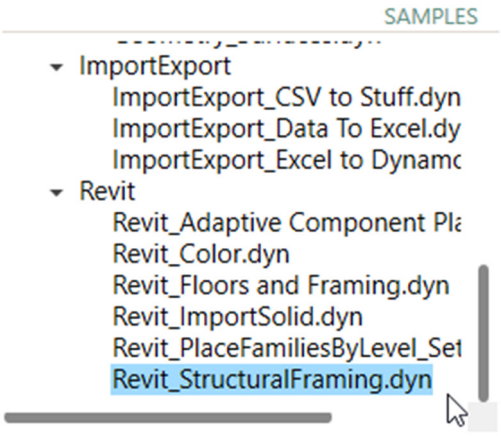
NOTE: All of the following exercises were created in versions of Dynamo before 1.2.

1. From the **Ex_1** folder in the dataset (AR16178_Storms_AU2016_Dynamo_Dataset) open the following file: **DynamoSample_2017.rvt** in Revit.



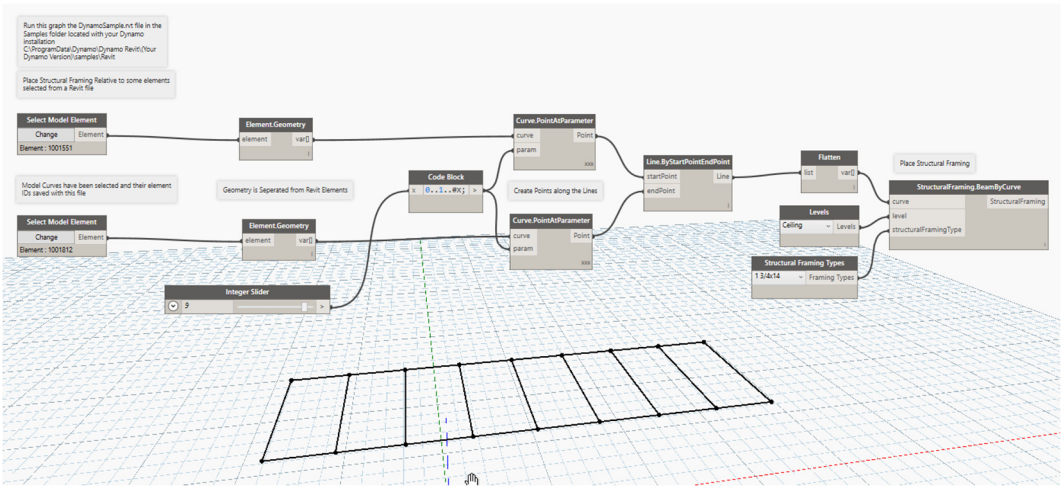
2. Start Dynamo from the Manage tab.

3. Select the **Revit_StructuralFraming.dyn** file from the SAMPLES area on the Start Page.

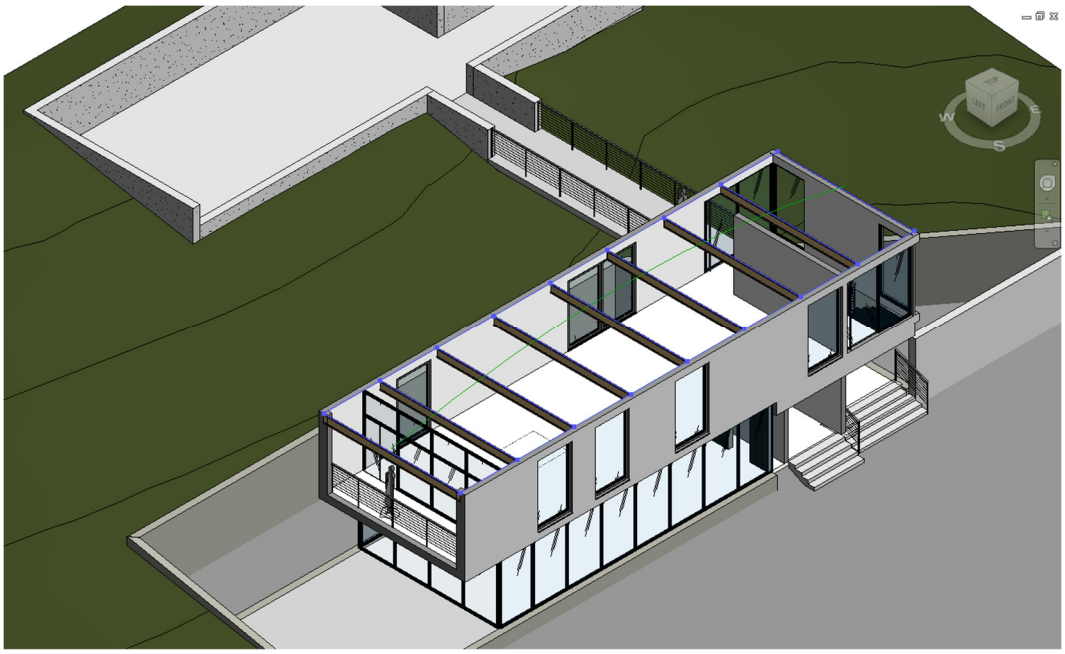


4. Ensure that the Execution Bar is set to **Automatic.**

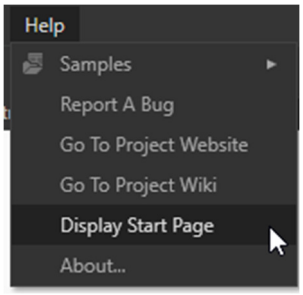
5. Once the run is completed you will see the completed version of the graph in Dynamo shown below.



6. Now if you switch back to Revit you should see the structural members in the project.

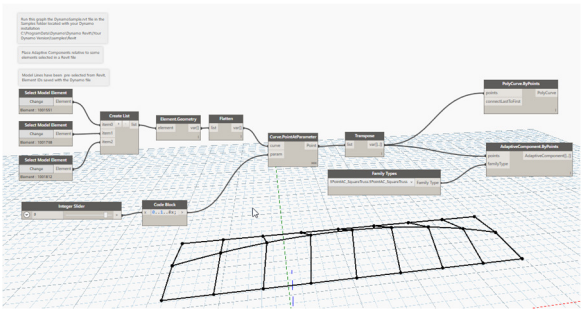


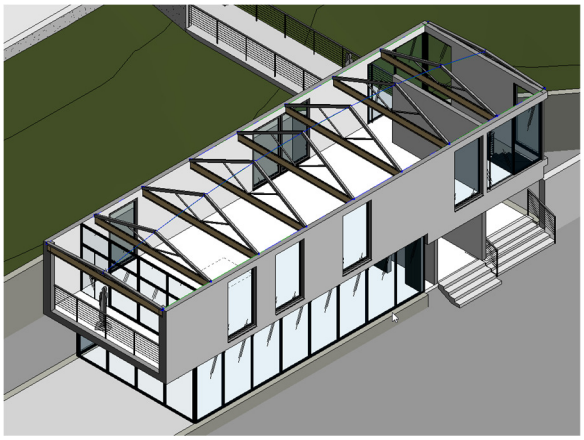
7. Go back into Dynamo and select the “Help” tab, then from the drop down menu select “Display Start Page”. This will allow us to return to the start page and select another graph (.dyn file) to use.



8. This time select the **Revit_Adaptive Component Placement.dyn** file from the SAMPLES area.

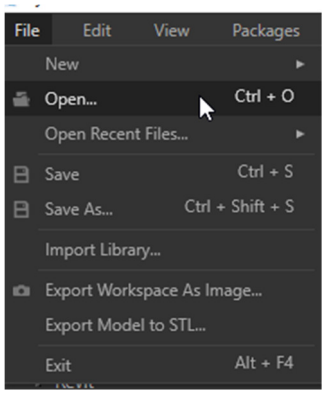
9. Review the outcome in Dynamo and then in Revit.





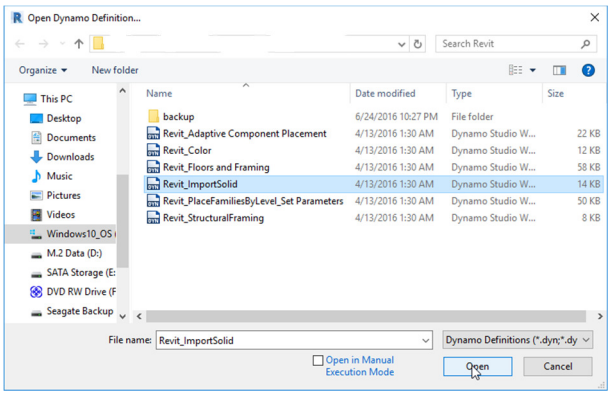
10. There is one more graph to be run to complete the roof, this time, we will load the .dyn file from the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset) in folder **Ex_1** we will open: **Revit_ImportSolid.dyn**

11. To do this we will go to the "File" tab in Dynamo and select "Open…"

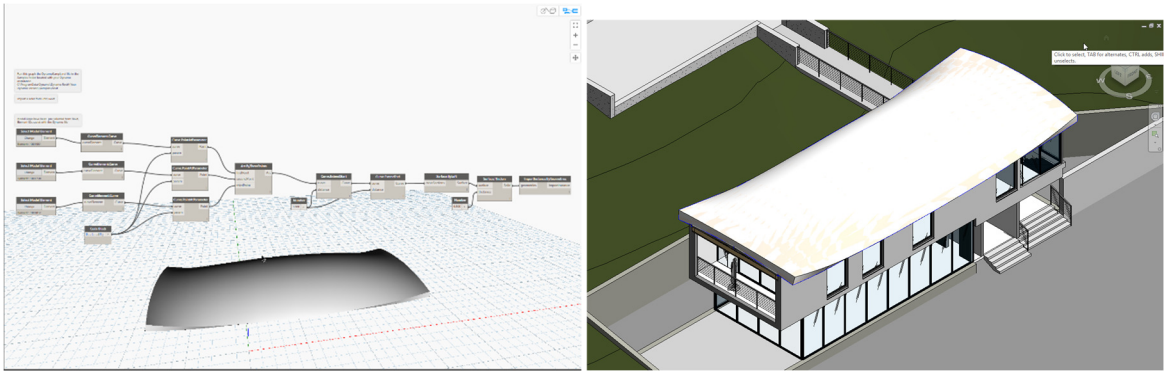


12. When the dialog box appears asking If you want to save changes you can select "No".

13. Then navigate to the Dataset folder location and open the **Ex_1** folder and open **Revit_ImportSolid.dyn**



14. For one last time Review the outcome in Dynamo and then in Revit.

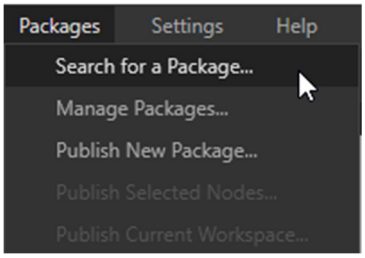


15. Close Dynamo without saving.

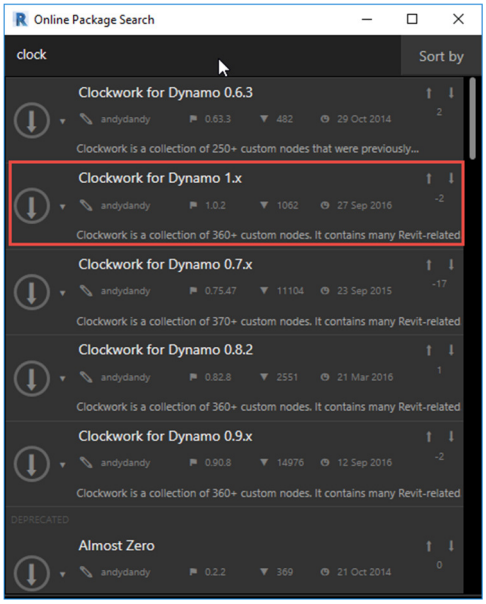
Ex 2) Installing a Package

In this exercise, we will install a couple of Dynamo Packages that will be used in later exercises.

1. Continue working in the same Revit file from Exercise 1 (**DynamoSample_2017.rvt**).
2. Start Dynamo from the Manage tab.
3. In the Menus Bar select the "Packages" tab, then from the drop down menu select "**Search for a Package…**"

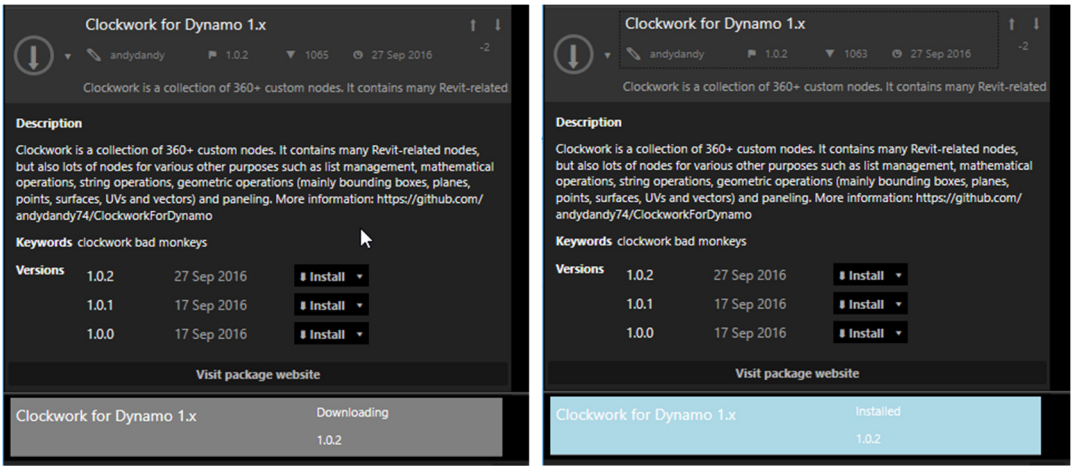


4. The "Online Package Search" dialog box will open, sometimes this can take a few moments before the dialog box is populated.
5. In the search area type in "Clock" as we want to install the **Clockwork for Dynamo 1.x** package. If you don't see something like what is shown below, then try changing the "Sort by" to **name** and try again.

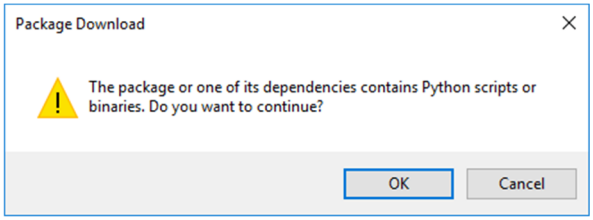


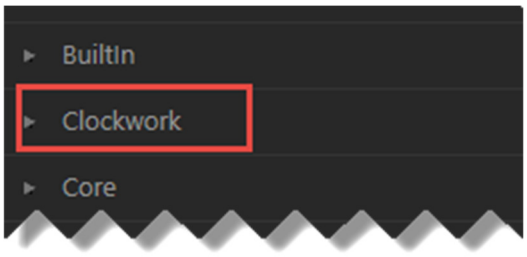
6. Select the **Clockwork for Dynamo 1.x** package and download it. This can be done by clicking on the down arrow in the circle (this downloads the latest version). The other option is to double click in the body of the package description which will open a dropdown menu showing all the versions of the package and you can choice the one you want, and select “install”.

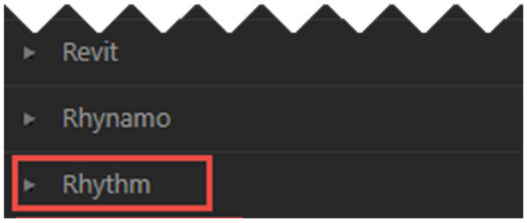
7. You will get a message asking if you are sure if you want to download the package, click yes.



8. Once the package is downloaded, repeat the process for the following package:
    a. Rhythm (accept Python warning)





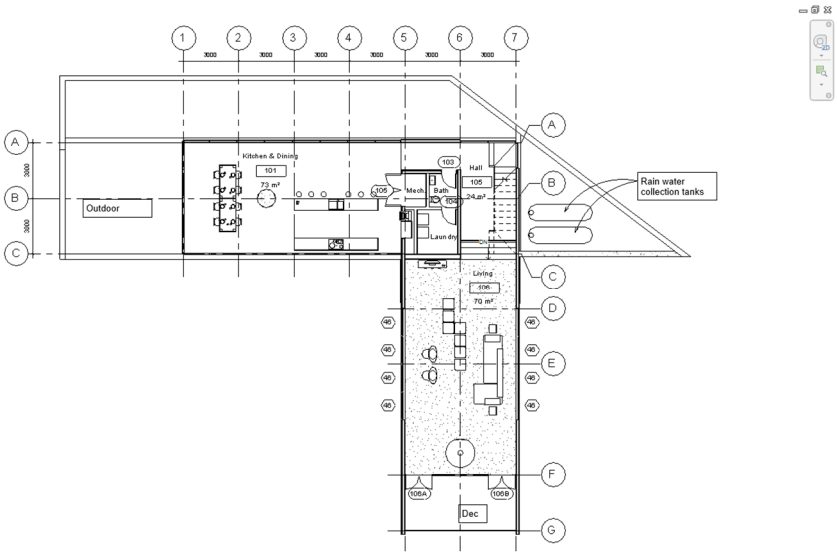


9. Close Dynamo without saving

Ex 3) ALL CAPS

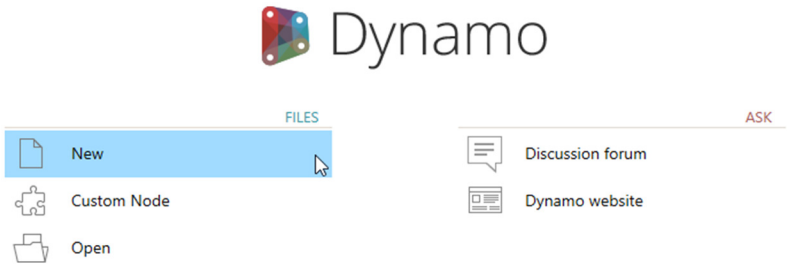
In this exercise, we change the room names from sentence case to ALL CAPS. This same workflow can be used with other Revit text parameters like sheet or view names.

1. From the **Ex_3** folder in the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset) open the following file: **rac_basic_sample_project_2017.rvt** in Revit. Open the Level 1 floor plan if it's not already open.

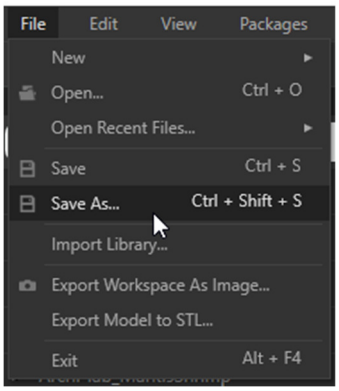


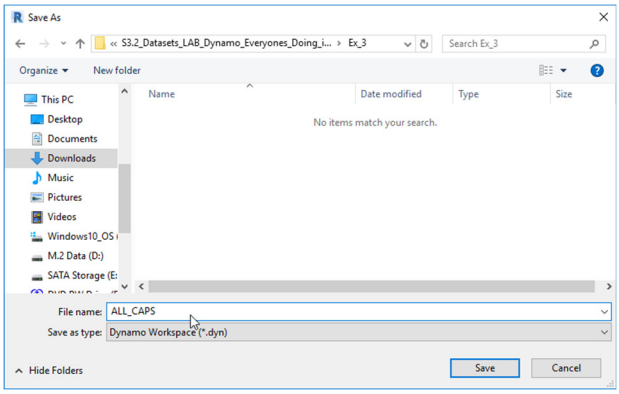
2. Start Dynamo from the Manage tab.

3. From the Start Page, under the FILES section begin a new Dynamo graph.

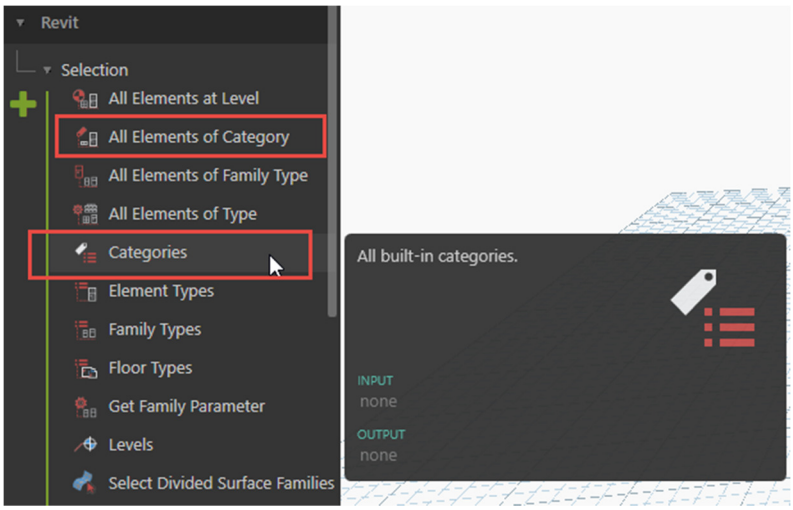


4. Under the File tab in the Menu Bar select "Save As…" and name the file **ALL_CAPS.**

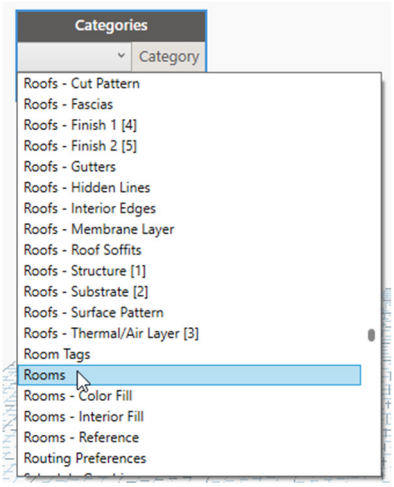




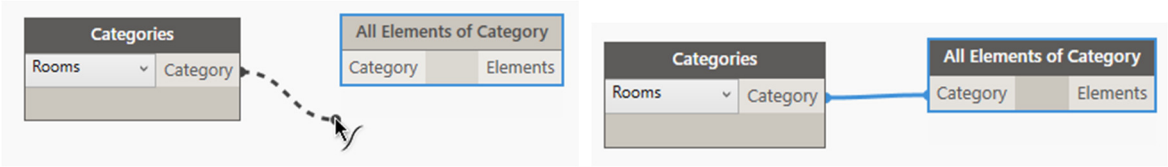
5. Under the **Revit** heading in the Library, open “Selection”, and then click on the **Categories** node, and then the **All Elements of Category** to place them on the canvas.



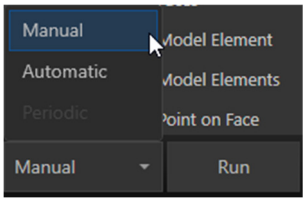
6. From the drop-down menu in the Categories node choose “Rooms”.



7. Click on the output (Category) of the **Categories** node, this will give you a dashed wire. Take this dashed wire and connect it to the input (Category) of the **All Elements of Category** node. Once this is done the wire will turn from dashed to solid.
   7.1. Note: Outputs are always on the Right of a node
   7.2. Note: Inputs are always on the Left of a node

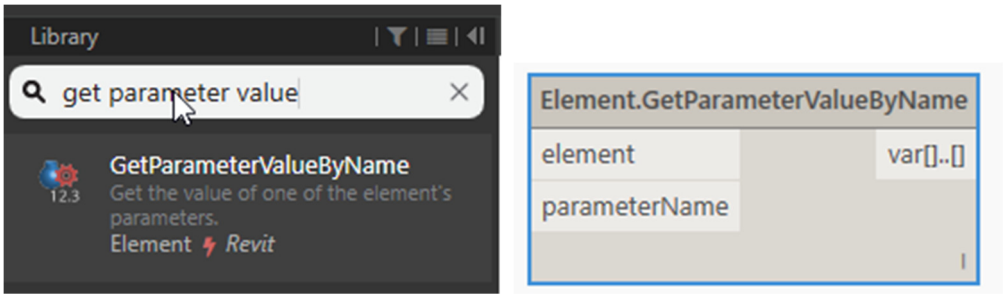


8. At this point, we are going to change the Execution Bar setting to Manual if it is not already set to Manual. This will ensure that the Dynamo program (graph) won't run until we click the "Run" button.



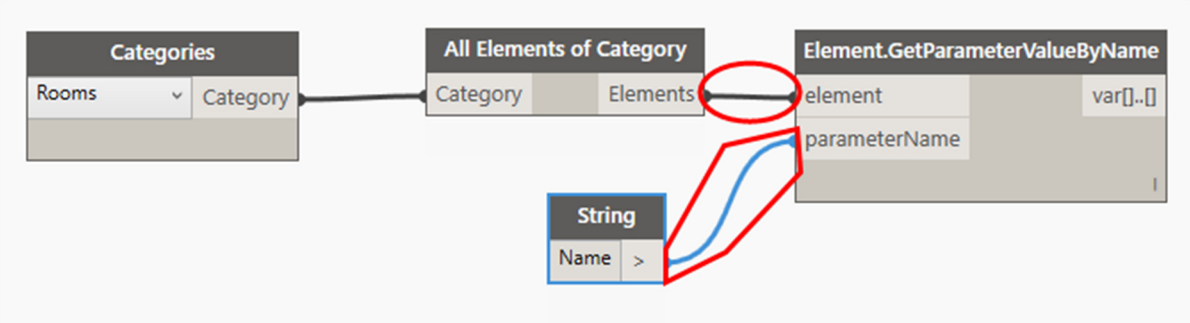
9. Start typing "get parameter value by name" in the search box at the top of the library, then select (double click) the **Element.GetParameterVauleByName** node when it appears to place it on the canvas.

   NOTE: if you hit the "X" in the search box it will clear the previous search.

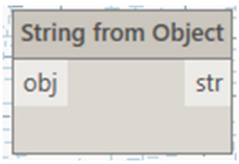


10. Next, we need a **String** node, this can be added to the canvas using the same search method by simply typing "string" in the library search box. We use the string node to add the parameterName to the **Element.GetParameterVauleByName** node. In this case, that name is "Name". This can be added to the string node by clicking in the white space, and typing in **Name**.

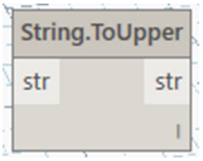
11. Next, we will connect our two newest nodes to our graph using wires as shown in the image below.

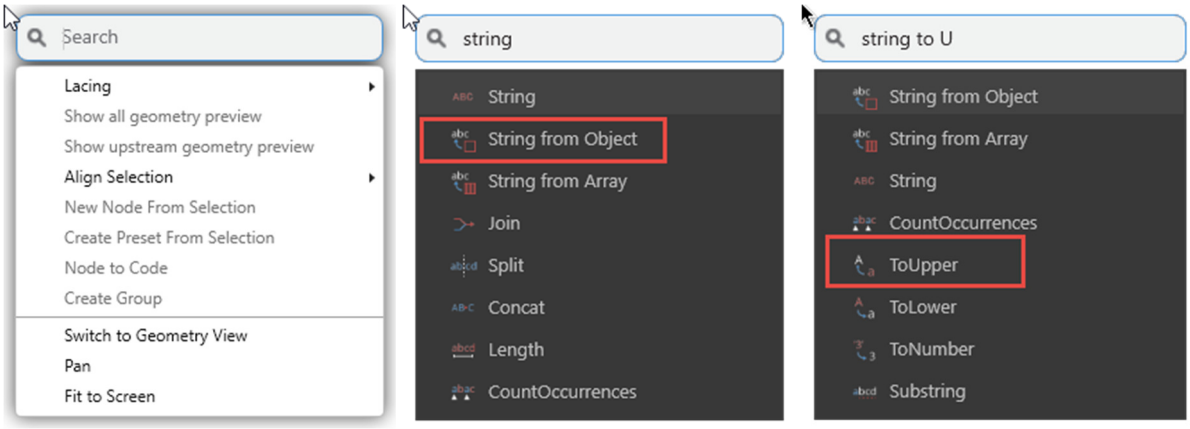


12. For the next two nodes, we will use a different way to search. We can get a search dialog box to pop up if we right-click on an empty space on the canvas. We need the following two nodes:
    12.1. String from Object

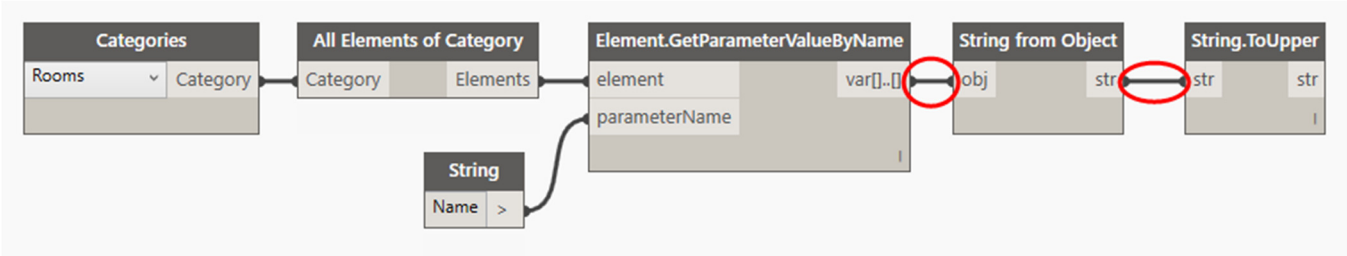


    12.2. String.ToUpper

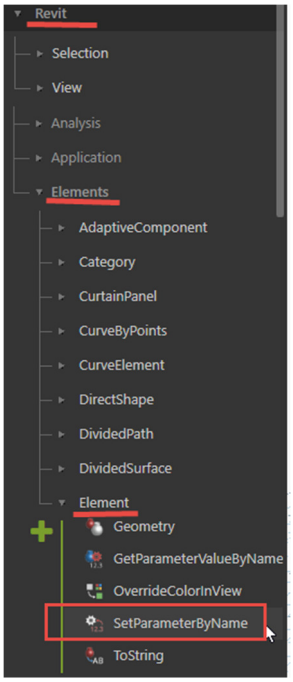


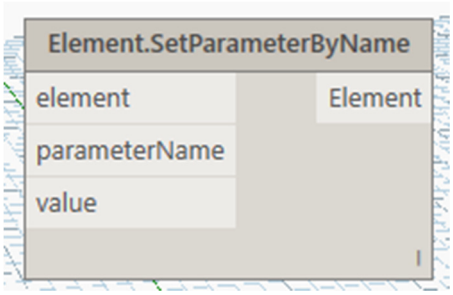


13. Again we will connect our two newest nodes to our graph using wires as shown in the image below.

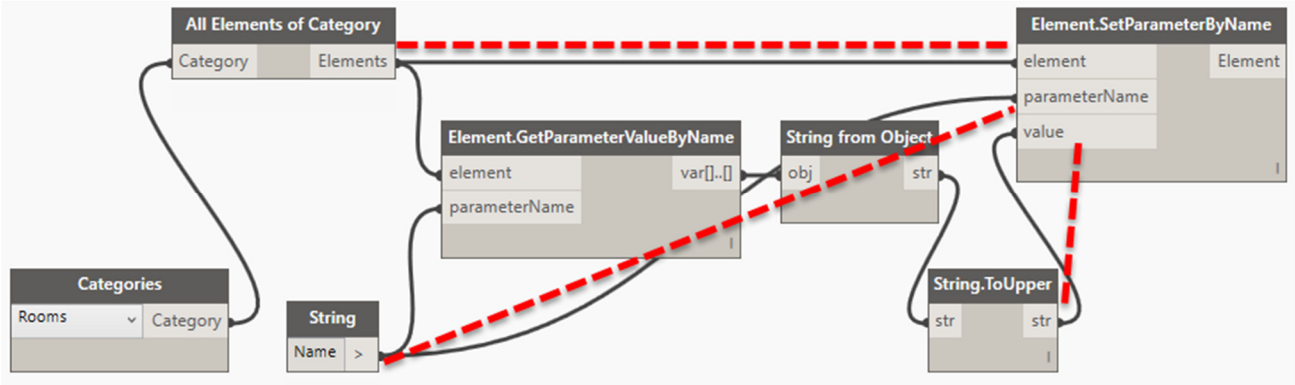


14. The final node needed can be found in the library under Revit > Elements > Element > **SetParameterByName.** Click on this node to add it to the canvas.



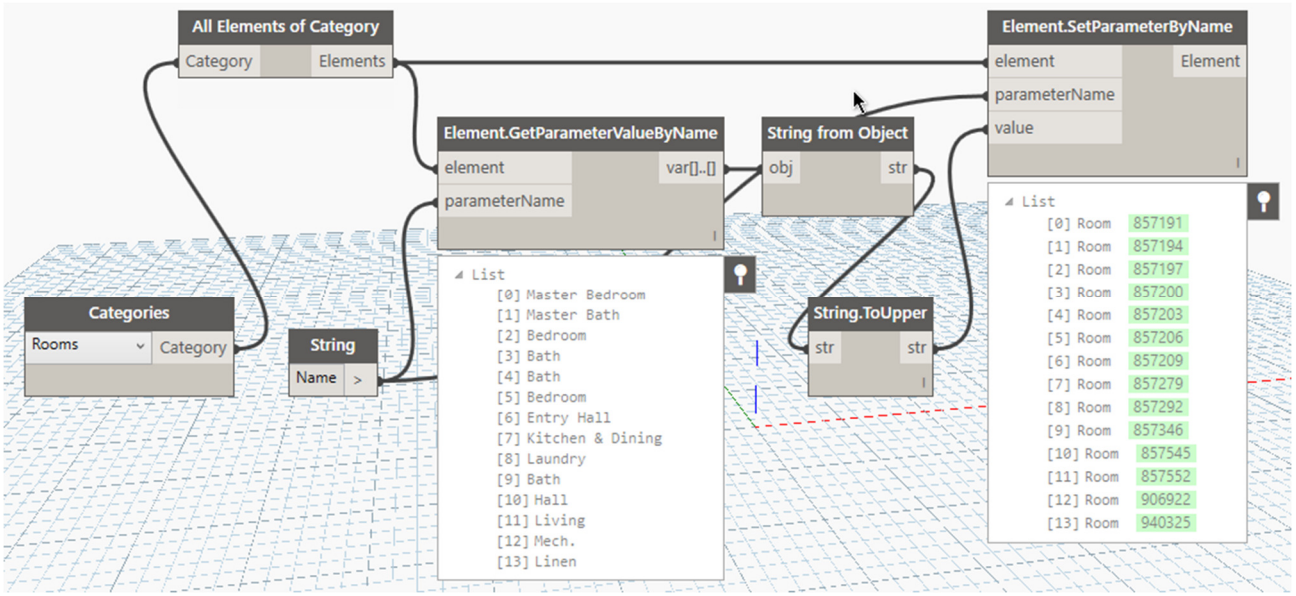


15. This final node requires three inputs from three of the nodes we already have on our canvas. Adjust the nodes as needed on the canvas to connect the final 3 wires as shown in the image below.

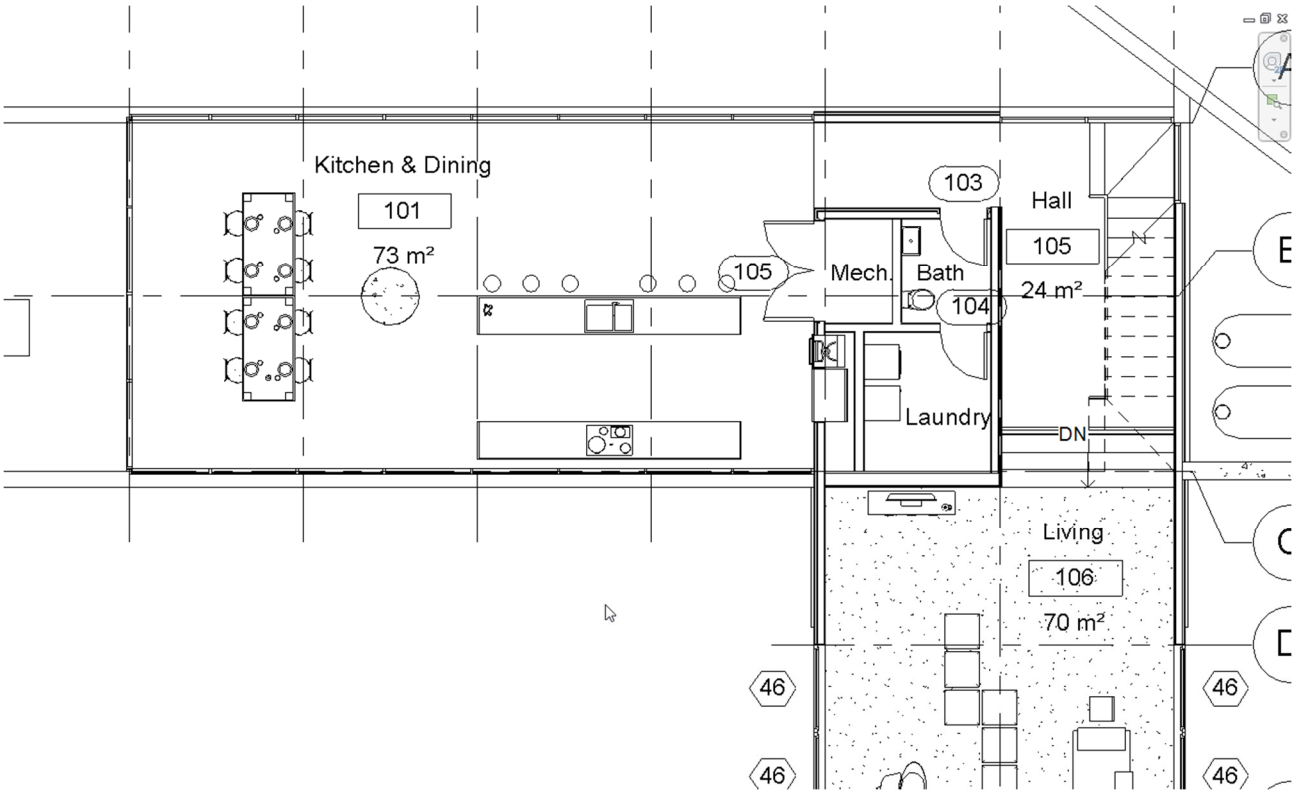


16. Now that our graph is complete we can click the “Run” button to see the results in Dynamo and Revit.

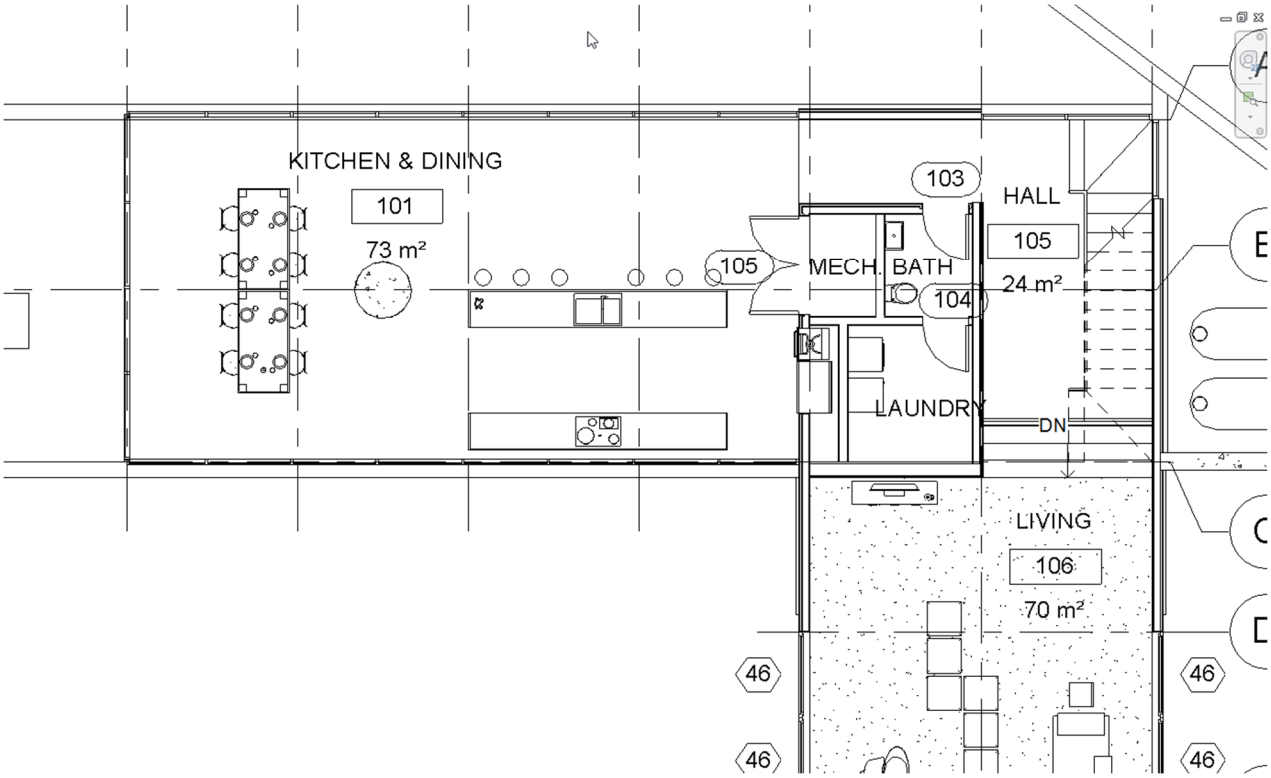
17. Final Dynamo Result.



18. Revit Before.



19. Revit Final Result.

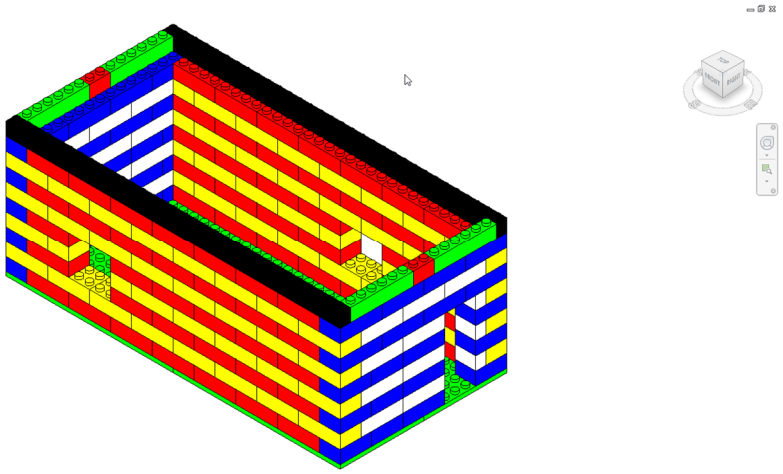


20. Save the Dynamo graph if you wish, and then close Dynamo.

Ex 4) Lego Part 1 – Extracting Info

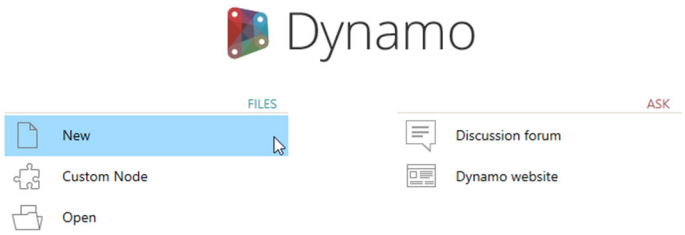
In this exercise, we look at some different ways to extract simple information out of our Lego model.

1. From the **Ex_4** folder in the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset) open the following file: **Lego_Project_2017.rvt** in Revit.

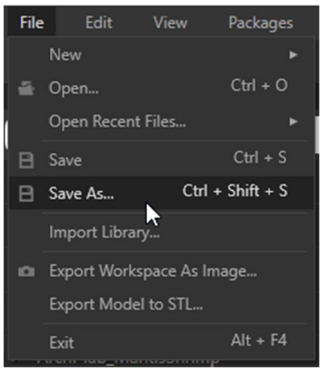


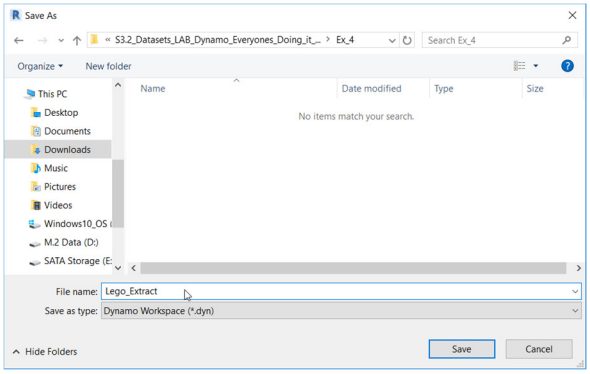
2. Start Dynamo from the Manage tab.

3. From the Start Page, under the FILES section begin a new Dynamo graph.



4. Under the File tab in the Menu Bar select “Save As…” and name the file **Lego_Extract.**

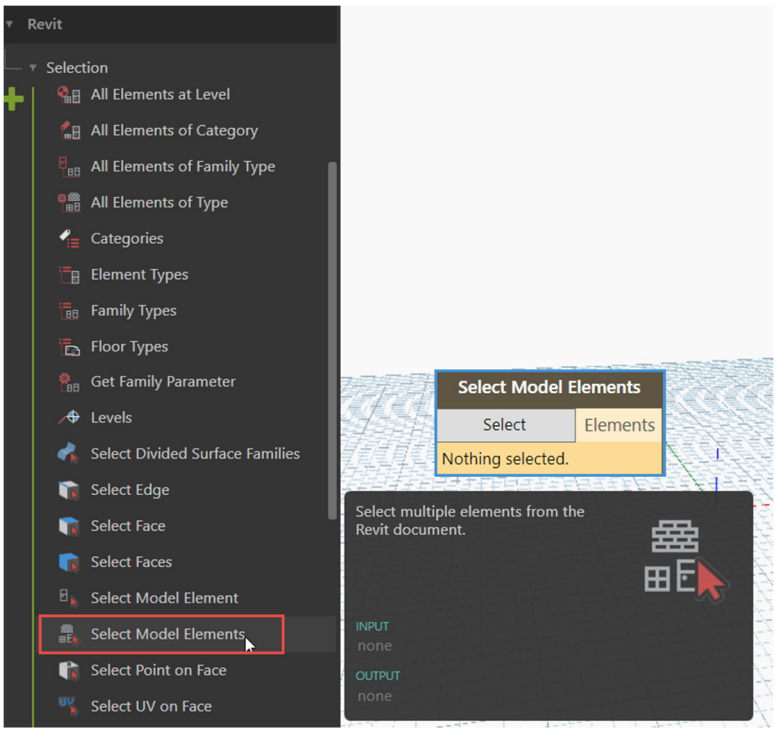




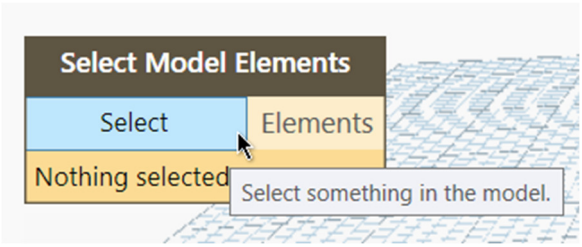
Task 1

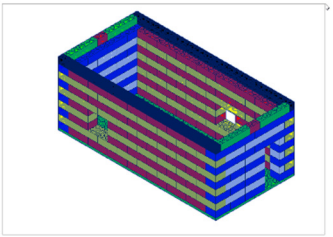
Finding out how many Lego pieces are in our model using different nodes and methods.

1. Under the **Revit** heading in the Library, open “Selection”, and then click on the **Select Model Elements** node (make sure it is Elements with the s on the end) to place it on the canvas.

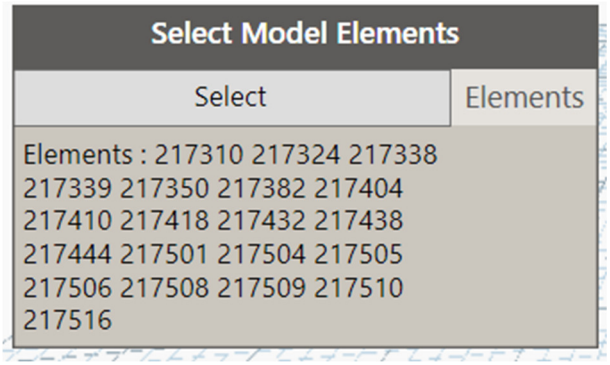


2. You will notice that the node is yellow, this is because we have not selected any elements from the Revit model yet. Before you select the content in Revit you need to click the “Select” button on the node, then return to the Revit model and select everything in the view.



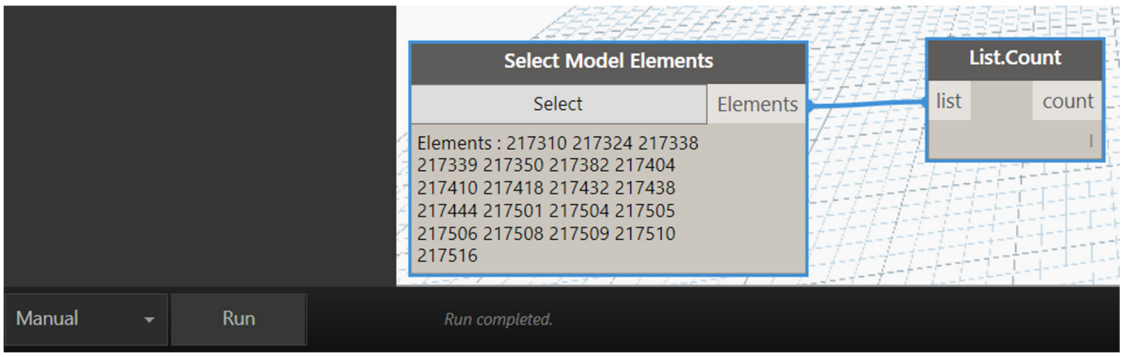


3. Once the Revit elements are selected you will see the Revit element IDs displayed in the **Select Model Elements** node.

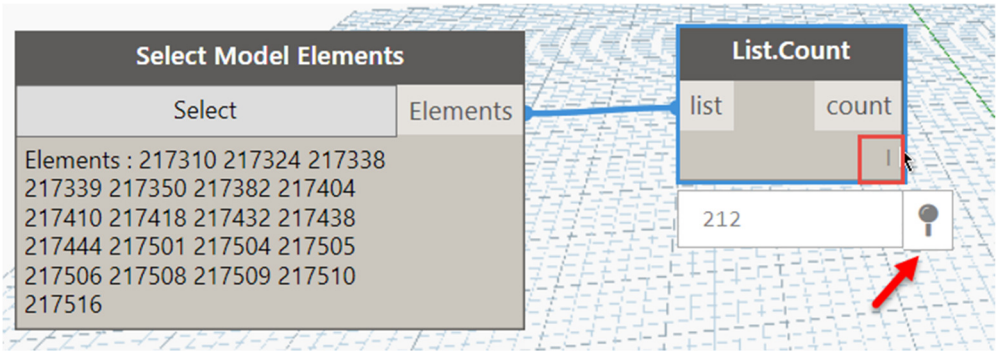


4. Now search for a **List.Count** node to add to the canvas, using the search method of your choice.

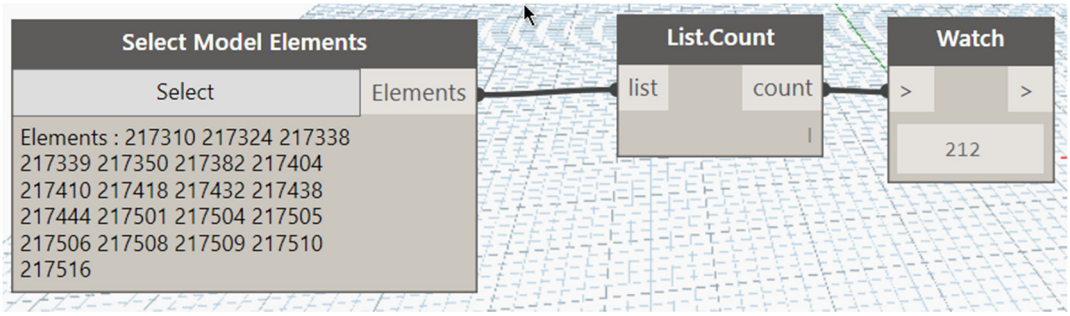
5. Next, we connect the two nodes with a wire to see the results after hitting the "Run" button if your execution bar is not on Automatic.



6. The results can be viewed by either hovering over or selecting the data preview area of the **List.Count** node. You can also keep the results visible by selecting the pushpin icon.



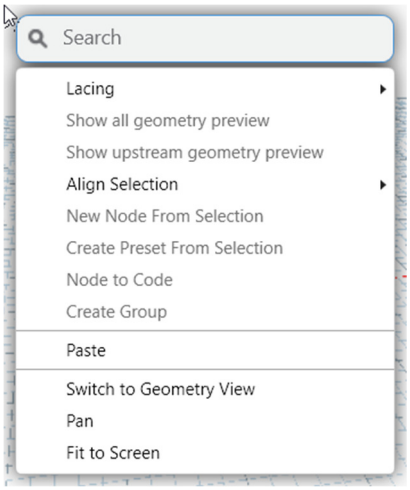
7. You can also see the results by adding a 3rd node, the **Watch** node and connecting it to the "count" output from the **List.Count** node.

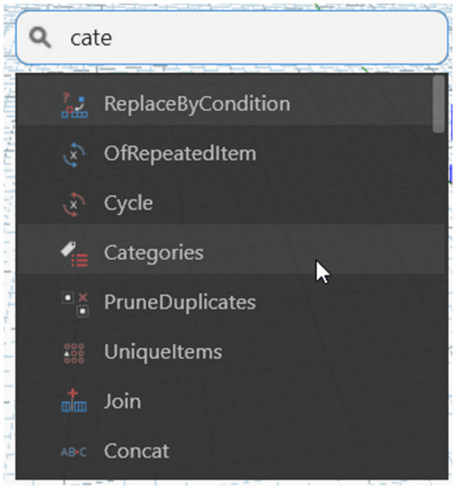


8. Both ways return the same results, **212** Lego pieces.

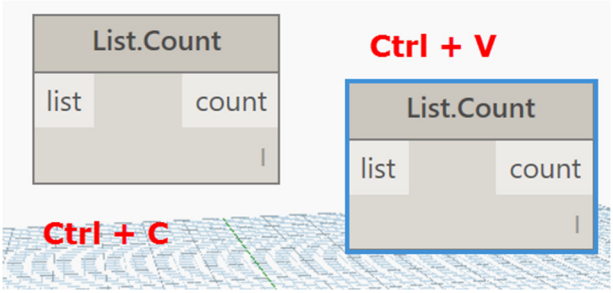
We can also find out how many Lego pieces are in the model by using the **Categories** node & the **All Elements of Categories** node, along with the List.Count node we already have on the canvas.

9. Right, click the canvas to access the search dialog box to find the **Categories** node. Then repeat the process for the **All Elements of Categories** node.

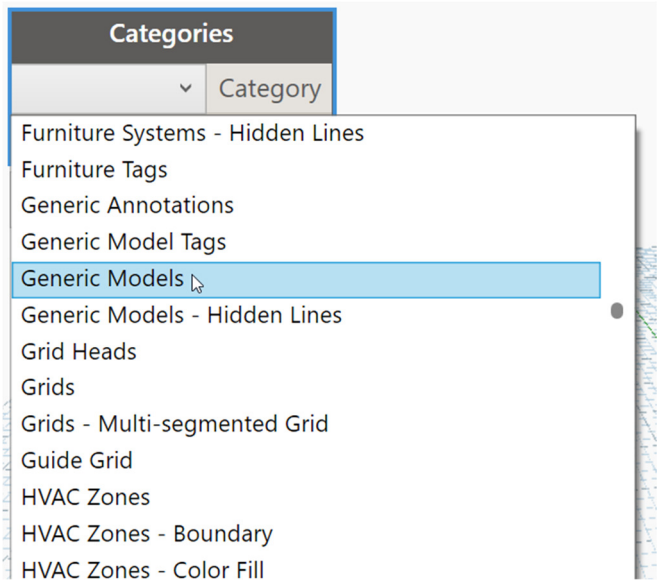




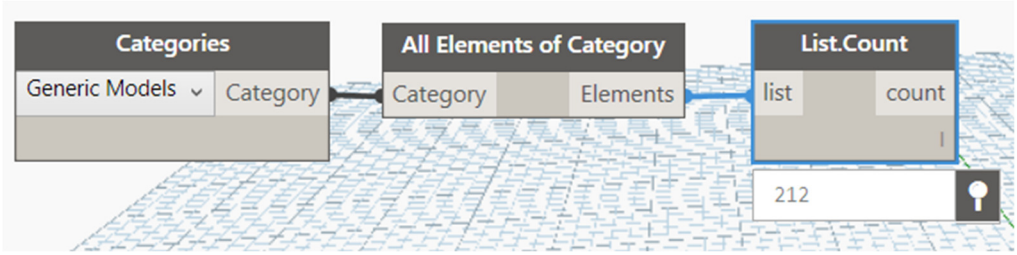
10. The List.Count node can be copied by selecting it and using the windows copy and paste commands (Ctrl + C, Ctrl + V).



11. Before we connect the 3 nodes to get our results, we need to select a Category from the drop-down list in the **Categories** node. The Lego bricks and plates are created as Generic models, so we will select **Generic Models** from the drop-down menu.



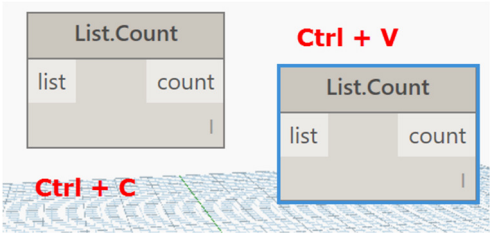
12. Now we can connect the nodes and get our results, 212, after hitting the “Run” button.



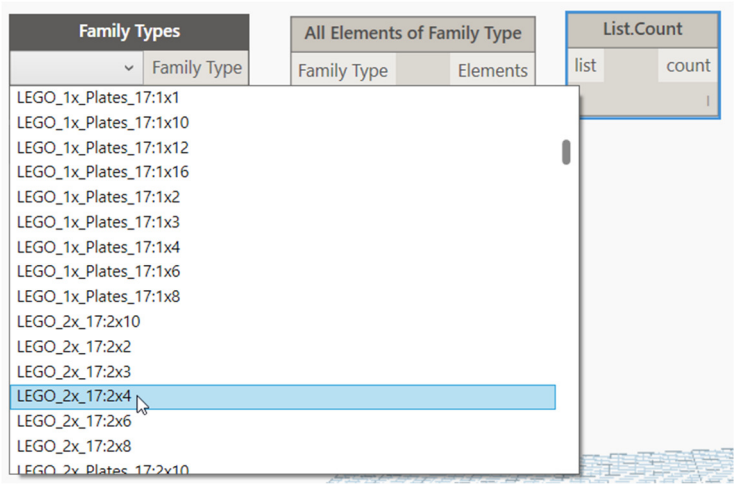
Now we are going to find out how many Lego pieces of a specific type (2x4) are in the model by using the **Family Types** node & the **All Elements of Family Type** node, along with the List.Count node we already have on the canvas.

13. Using the search dialog box at the top of the Library find the **Family Types** node. Then repeat the process for the **All Elements of Family Type** node. Add them both to the canvas.

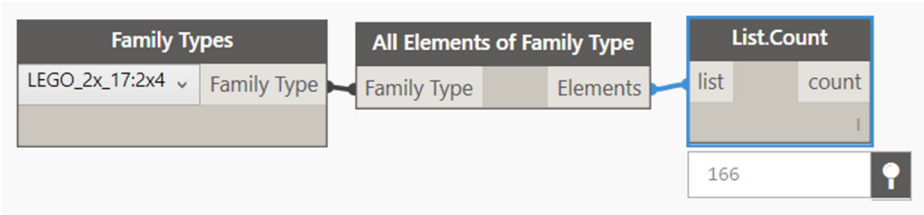
14. The List.Count node can be copied by selecting it and using the windows copy and paste commands (Ctrl + C, Ctrl + V). Be sure to disconnect any wires that come with the copied node.



15. Before the nodes are connected, we need to select the family type we want from the drop-down list in the **Family Types** node. The Lego brick we are going to count is the 2x4 brick. The official family type name is **LEGO_2x_17:2x4**, so we will select this one from the drop-down menu.



16. After we have selected the family type we can connect the 3 nodes together and find out the results after we hit the “Run” button. The results are 166 of the 2x4 Lego Bricks.



17. Save your Dynamo graph (file).

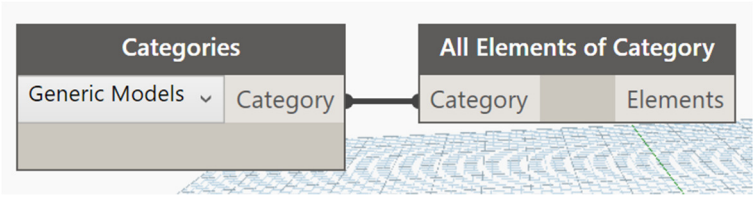
Task 2

In this task, we will find out how many Lego pieces there are of each colour and organize them in a list.

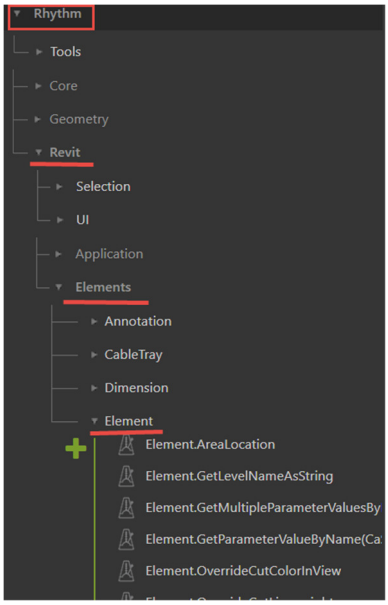
1. For this task, you can continue to work with the Dynamo graph from task 1, or you can open the **Lego_Extract_T2_Start.dyn** file from the **Ex_4** folder in the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset).

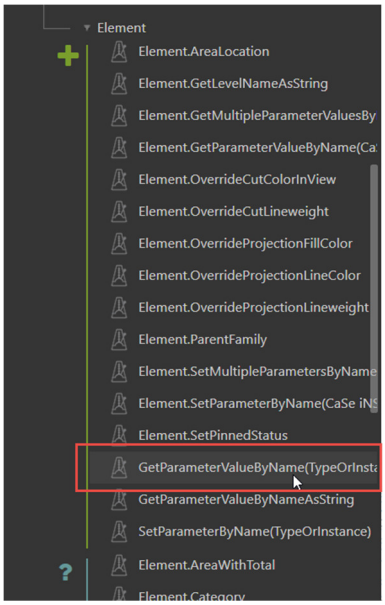
2. We will begin this graph by reusing the **Categories** node, and the **All Elements of Categories** node. We won't be needing any of the other leftover nodes so they can be deleted. You can delete the nodes by selecting them, and then hit the "Delete" key on your keyboard.

3. We also need to make sure that "**Generic Models**" is selected from the pull down in the **Categories** node.

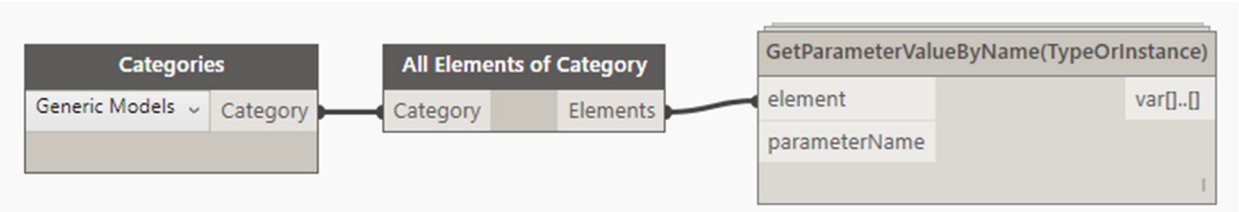


4. The next node we will use comes from the **Rhythm** package that we installed in Exercise 2, and it is called **GetParameterValueByName(TypeOrInstance)**. This node can be found in the Library under the Rhythm > Revit > Elements > Element > **GetParameterValueByName(TypeOrInstance).** Once you find the node add it to your canvas.

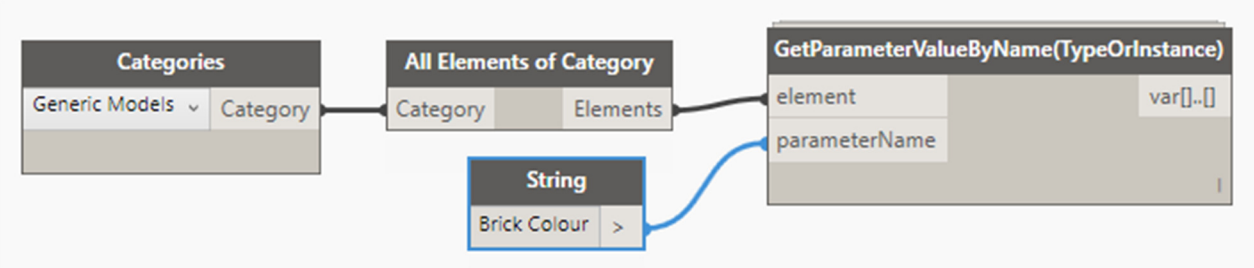




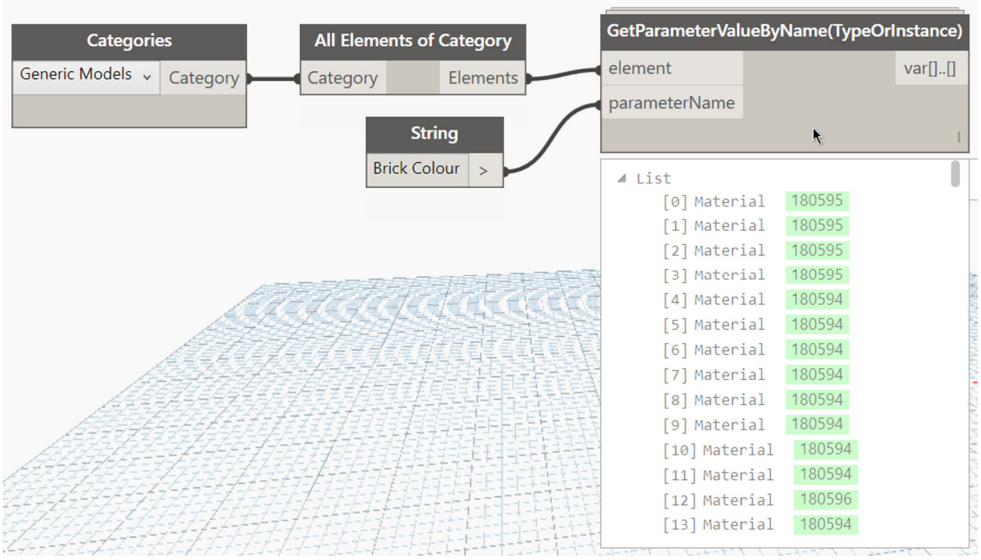
5. The **GetParameterValueByName(TypeOrInstance)** node has two inputs, the first input is “element” and the input will come from the output of the **All Elements of Categories node.**



6. The other input needed is “parameterName”, for this we need a **String** node so we can add the name of the parameter we are looking for. We are looking for the colours of the Lego pieces, and this parameter is called **Brick Colour.** Use any of the section methods to find the **String** node and add it to the canvas. Once the **String** node is on the canvas click inside it and add the words “Brick Colour”.

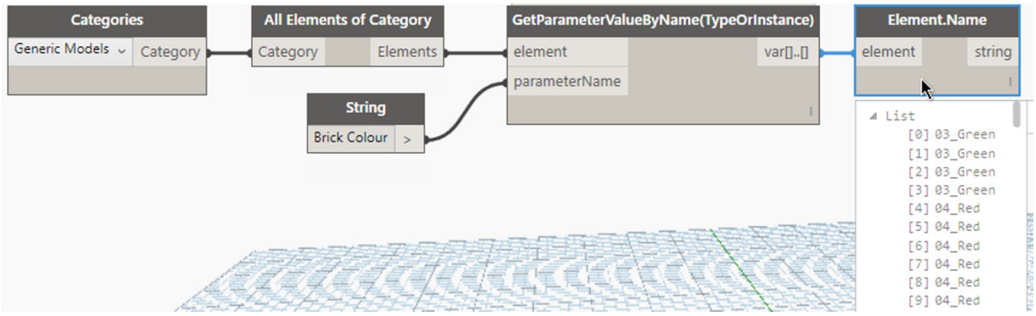


7. The information that comes out of the **GetParameterValueByName(TypeOrInstance)** node is not in the right format for us. We need a node to convert the name of the parameter to the name of the parameter value (from Material to the Brick Colour name, such as 03_Green).

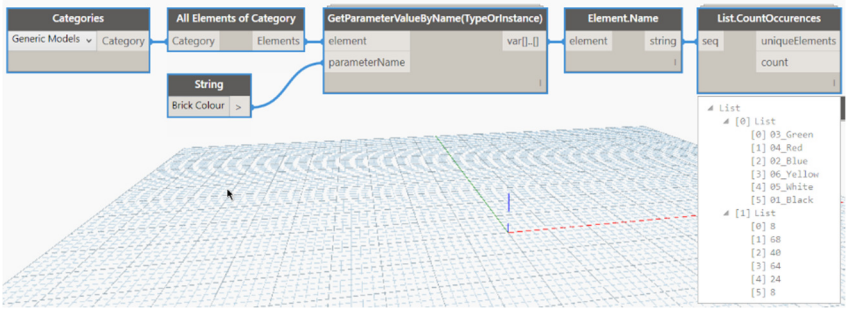


8. The node we need for this task is the **Element.Name** node, use your method of choice to find the node and add it to the canvas.

9. After connecting the latest node to our graph, and selecting “Run” we now see the Brick Colour name as the list output instead of Material and the Revit Element ID.

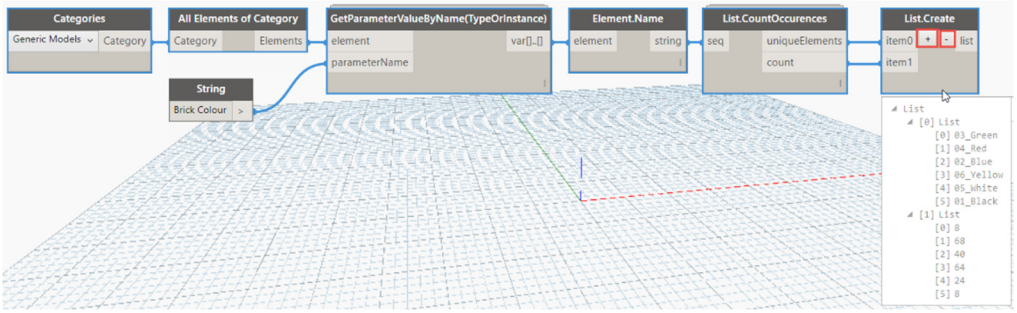


10. The next node will show the unique items in our list, which will allow us to see how many different Brick Colours there are. This node will also show us how many Lego Pieces there are in each of those colours. This node is called **List.CountOccurences** and comes from the Clockwork package we installed in exercise 2. Find the **List.CountOccurences** node, add it to the canvas, and then connect it to our graph.

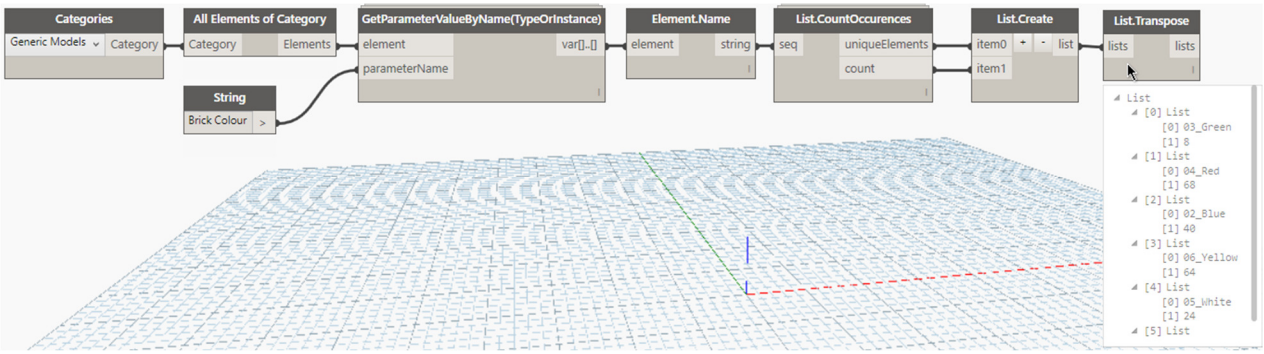


11. We now have a list of 2 outputs. The first output [0] is a list of the unique Lego pieces colours. The second output [1] is the amount of pieces in each colour. With 2 more nodes, we can turn these inputs into a list in a more useful format.

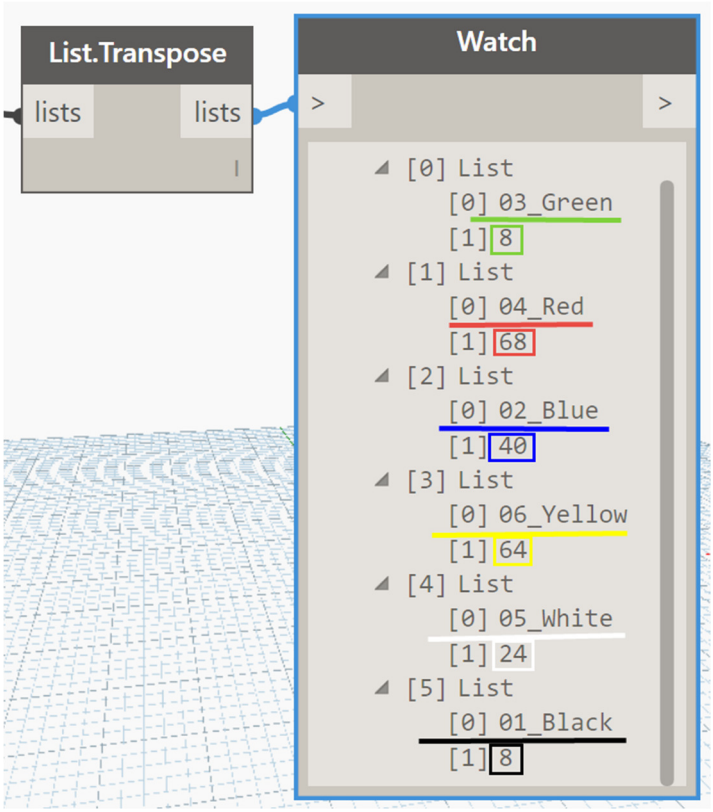
12. We will use a **List.Create** node to turn the two outputs into a single list. This node is different because it allows the user to add or remove inputs as required by hitting the “+” or “-“ buttons on the node.



13. Now that we have a single list that contains 2 lists, each with 6 items in them. We have one last node to add to the graph to get the list sorted, or transposed into our final version. Search for and add the **List.Transpose** node to the canvas, and connect it to the end of our graph.



14. The final output is a list, of 6 lists, each with 2 items in it. We can now see each colour and the number of Lego pieces that are in that colour. By adding a **Watch** node we can see the 6 lists better.

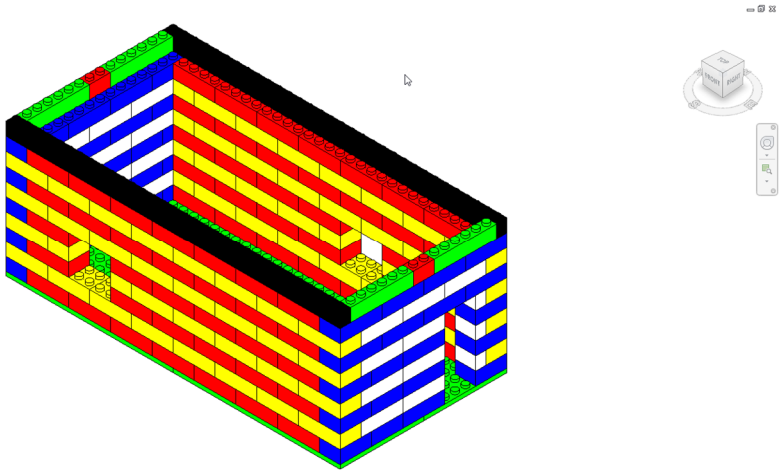


15. Save your Dynamo graph (file), and then close Dynamo.

Ex 5) Lego Part 2 – Writing to Excel

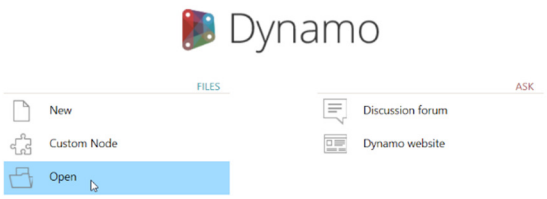
In this exercise, we will write some information from Dynamo & Revit out to Excel.

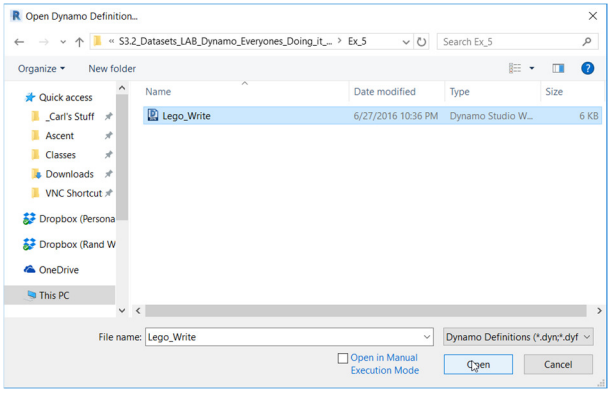
1. From the **Ex_5** folder in the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset) open the following file: **Lego_Project_2017.rvt** in Revit if it is not already open.



2. Start Dynamo from the Manage tab.

3. From the Start Page, under the FILES section open, then open the Dynamo graph **Lego_Write.dyn** file from the **Ex_5** folder in the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset).

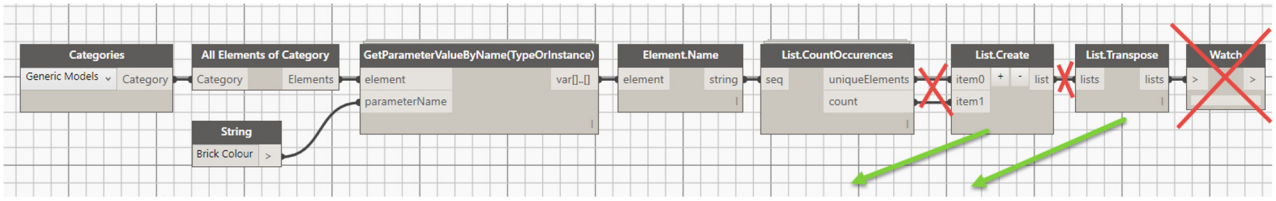


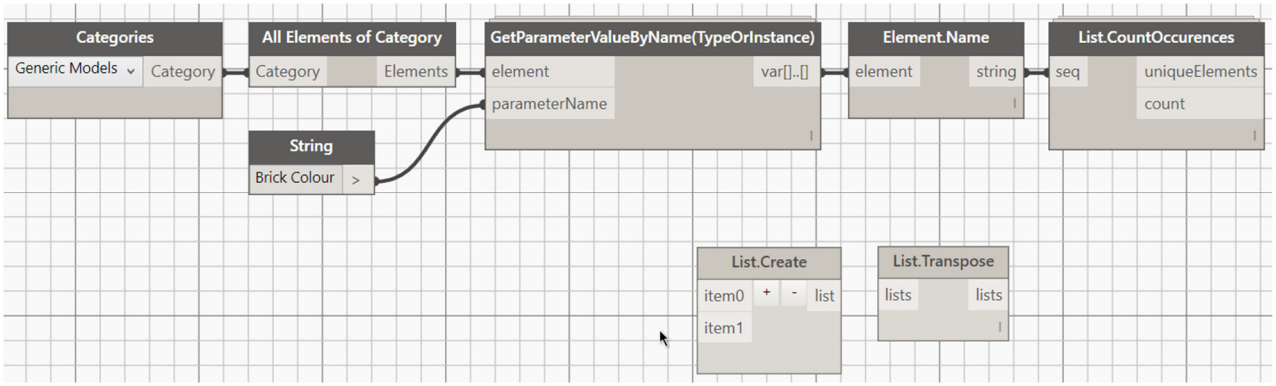


Task 1

For this first task of exercise 5, we will write the information about the number of Lego pieces in each colour that we found out in Task 2 of exercise 4 to an Excel file.

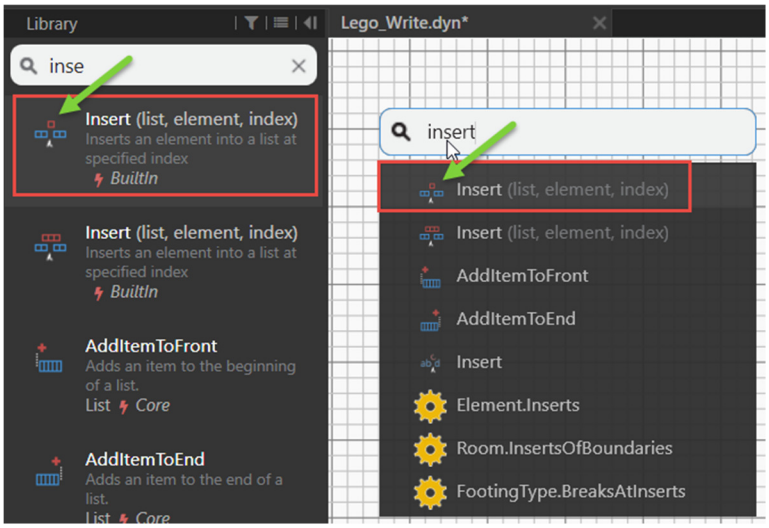
1. We will start this task by deleting the **Watch** node as we won't need it. Then disconnect the **List.Create** and **List.Transpose** nodes and move the aside on the canvas to be used later in the task.

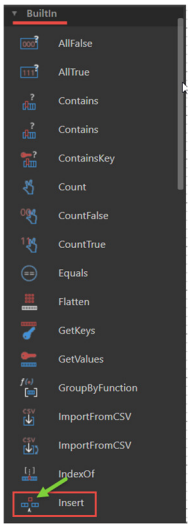


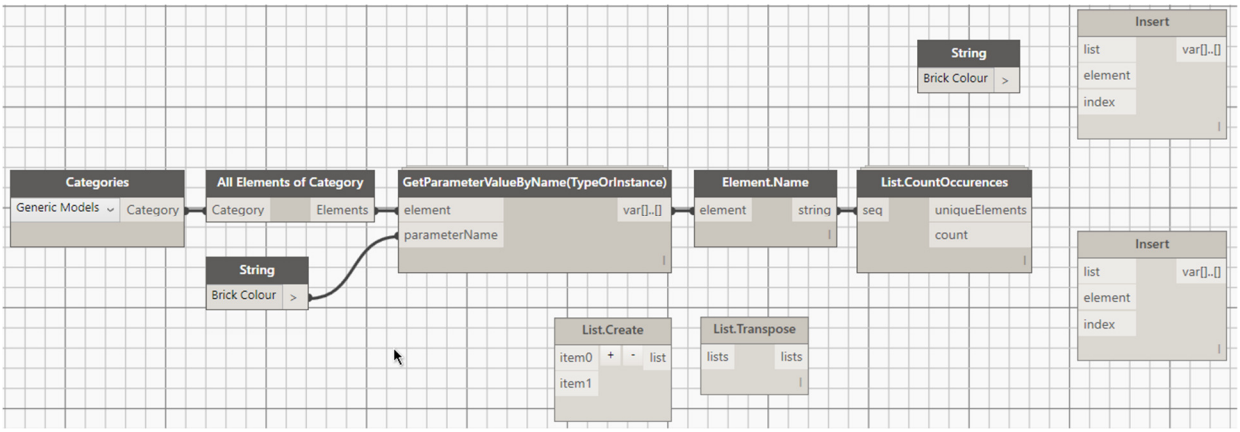


2. We will need a copy of the String node that is already on the canvas, we can do this by using "Ctrl + C" and Ctrl + V". Keep the same text inside the node "Brick Colour".

3. Next, we will need a node that will allow us to add an item to our two lists. The reason for this is so we can have headings when we write the information out to Excel. The node we need for this is called **Insert** and we will need 2 copies. Search and find the **Insert** node and add 2 to the canvas.
   3.1. **Note:** Select the **Insert** node with the single red square in the icon, not the one with three red squares in the icon. This node can be found under the "Builtin" heading in the Library.

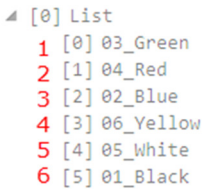




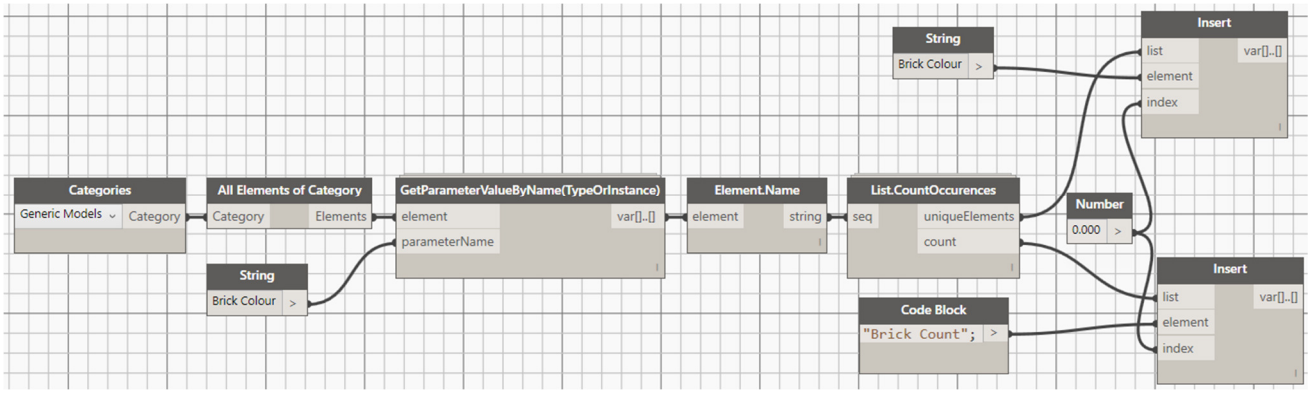


4. Each **Insert** node requires 3 inputs, these inputs allow you to insert an element into a list at a specific index.
   4.1. *list* = list of values
      4.1.1. This will be the output from the **List.CountOccurences**, one output for each **Insert** node.
   4.2. element = element to be inserted
      4.2.1. This will be the names we add as the heading for our 2 excel columns, "*Brick Colour*" and "*Brick Count*". We can do this by using **String** nodes, or a **Code Blocks**.
   4.3. index = where to be inserted (within the list)
      4.3.1. This is where in the list we want to add the element. We want the element (column header) to be on the top of the list, or at index [0]. We can do this with a **Number** node.

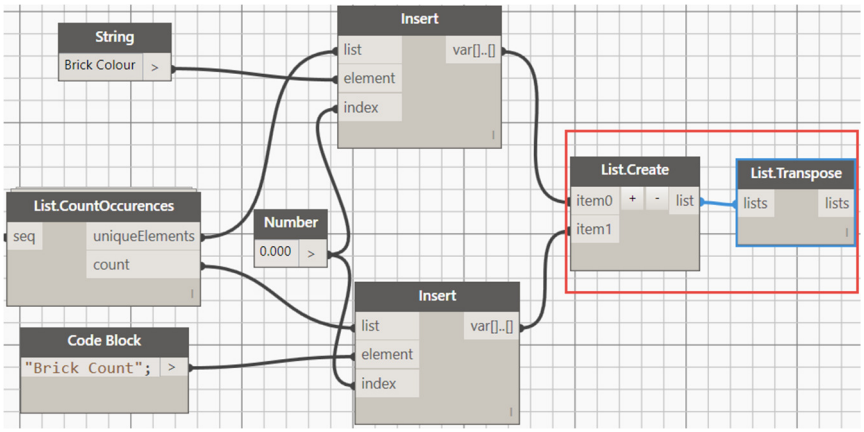
Note: If you double click on the canvas you create a **Code Block**. These can be used to add code to a program, but they can also be used to make quick **String** nodes by simply adding " " around whatever you are typing, this makes it a string.



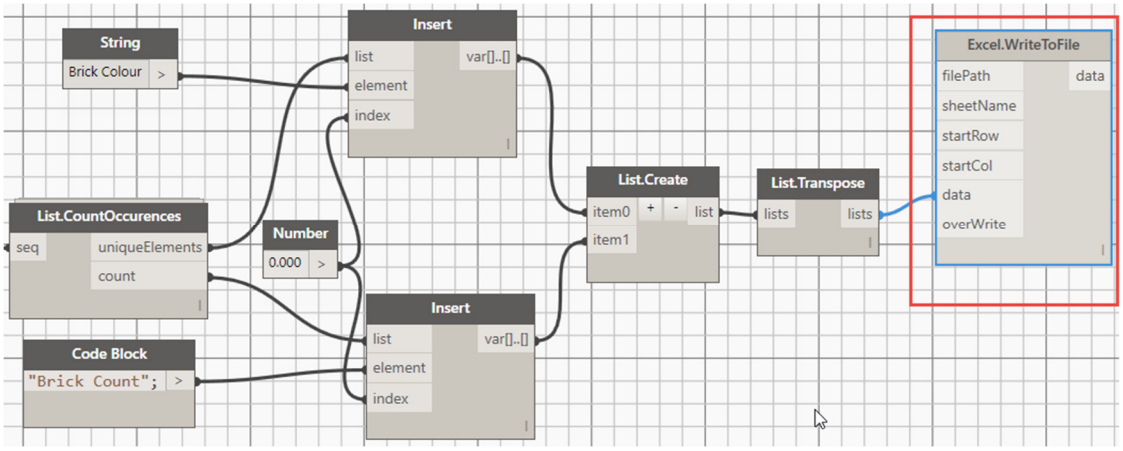
Note: All lists in Dynamo start at "0", so a list that goes to 5, is a list of 6 items.



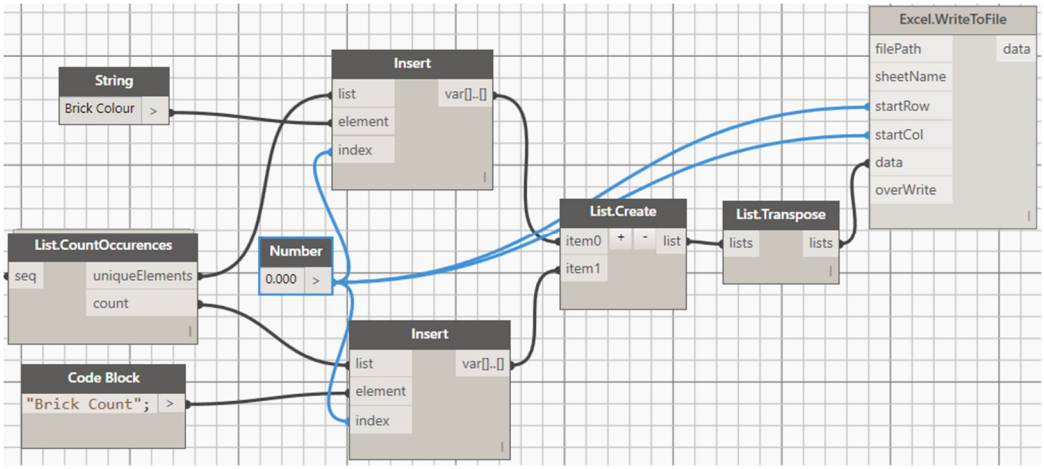
5. Now we can reconnect the **List.Create** and **List.Transpose** nodes we disconnected in the first step.



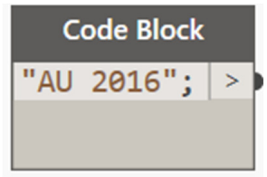
6. To write this information to Excel we need to add the **Excel.WriteToFile** node to our graph.



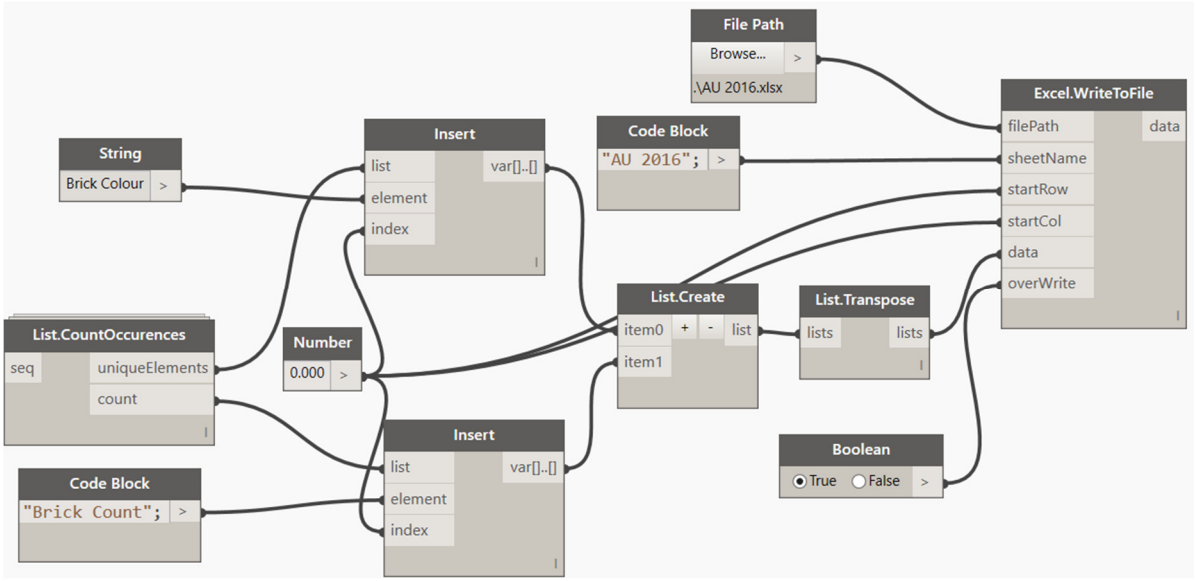
7. To complete the graph, we need to connect all of the inputs in the **Excel.WriteToFile node.** First, we can use the existing **Number** node and connect it to the "*startRow*" and "*startCol*". These inputs tell Excel where to start each Row and Column.



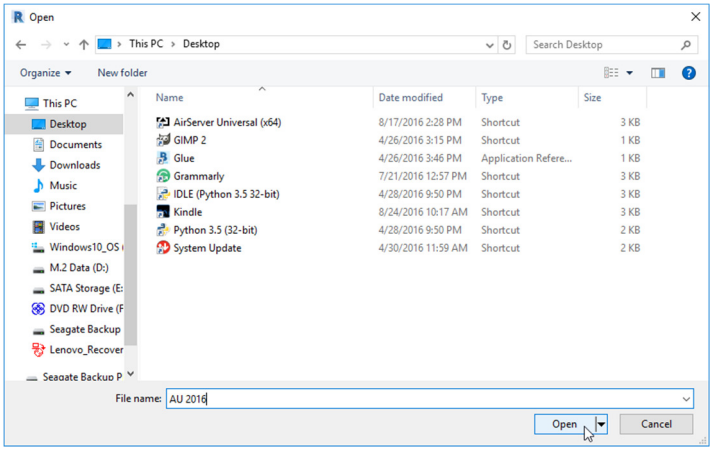
8. The "*sheetName*" input can be connected to a Code Block. This input is to put the name on the tab that is on the bottom of the Excel sheet that is created. We will use "AU 2016".

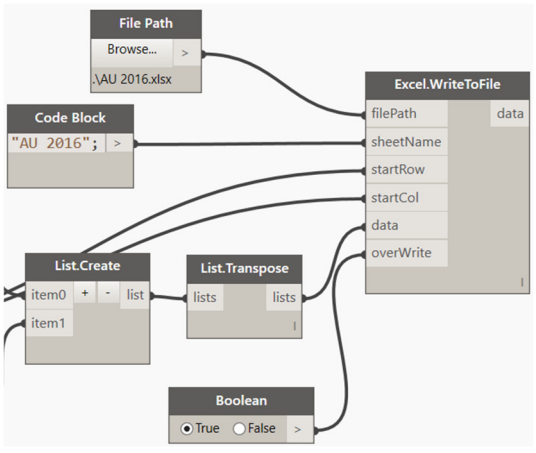


9. The "*overWrite*" input needs a **Boolean** node that will tell Dynamo if you want to overwrite the Excel file or not. If the **Boolean** node is set to True the file will be overwritten, if it is set to False it will not be overwritten.

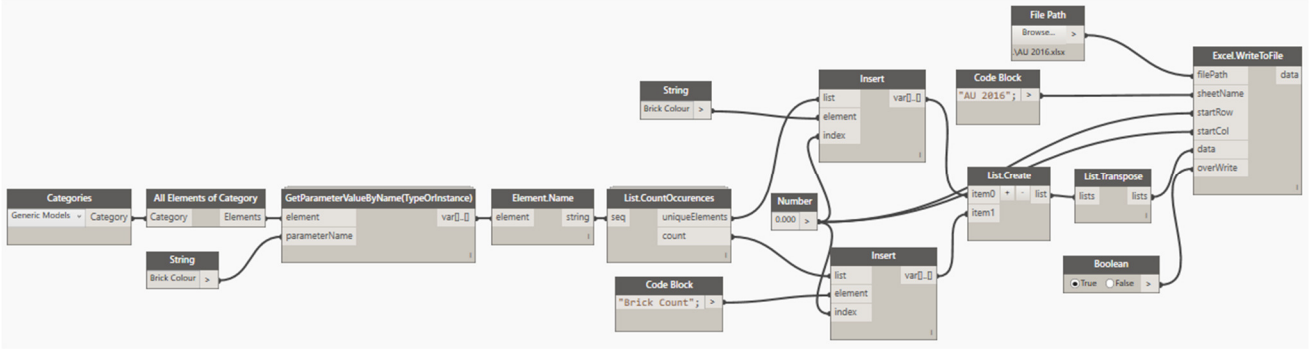


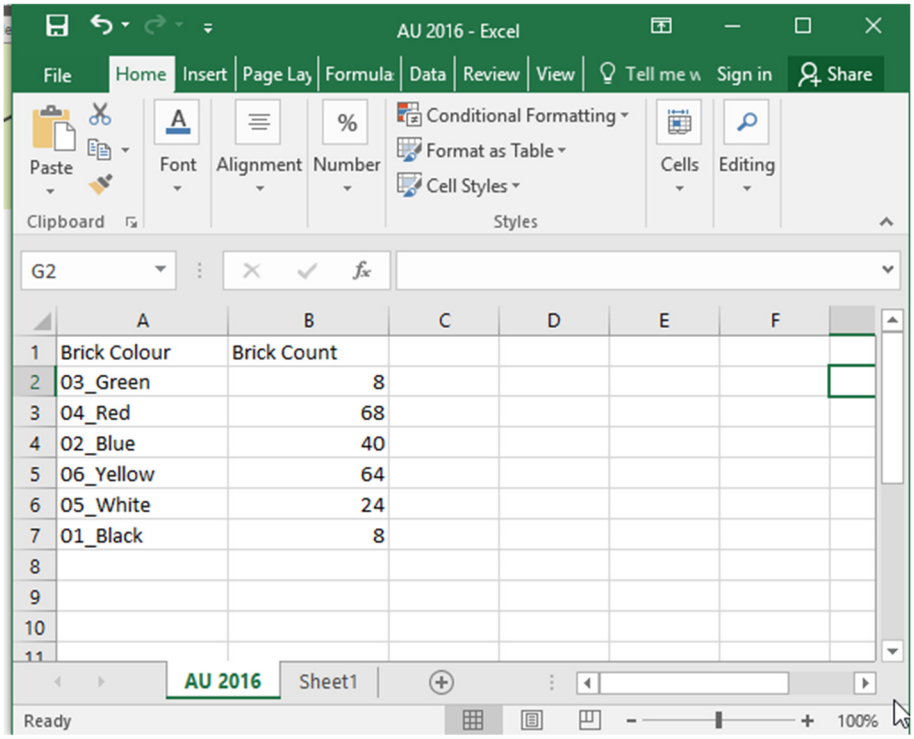
10. The final input "filePath" will require the **File Path** node. This node allows you to input a location of where you want the Excel file to be written. For this exercise, we want to save the file to the computer desktop. You can also name the Excel file once you select the save location, we will call this file "AU 2016".





11. Now that all the information is in place in our graph the last thing to do it hit the “Run” button and see what shows up in in the Excel file.





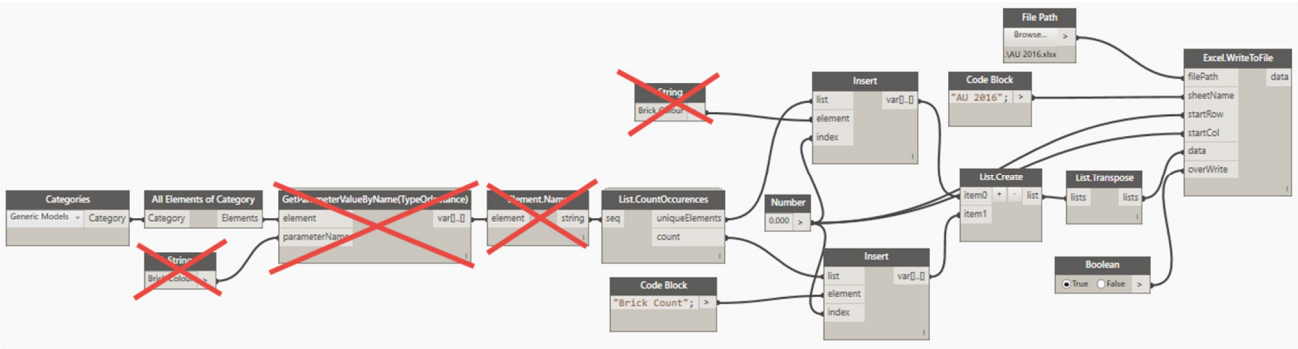
12. Save your Dynamo graph (file).

Task 2

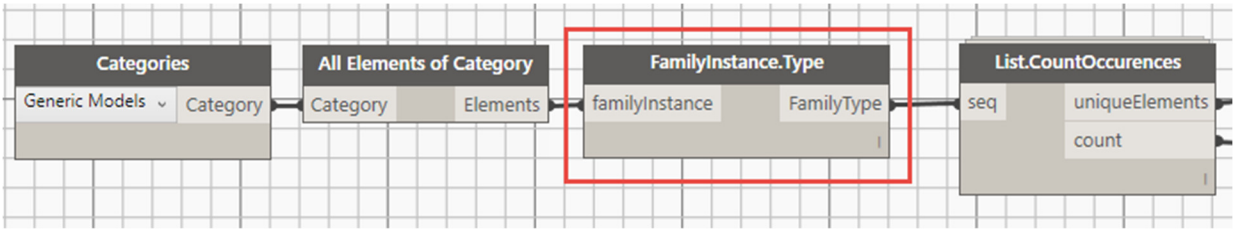
In this task, we will write another Excel file, this time with the Revit Family name of the Lego pieces in the model, the Revit Family Type of the Lego pieces, and the count of each.

1. For this task, you can continue to work with the Dynamo graph from task 1, or you can open the **Lego_Write_T2_Start.dyn** file from the **Ex_5** folder in the lab dataset (AR16178_Storms_AU2016_Dynamo_Dataset).

2. We will begin this graph by reusing most of the graph from task 1. You can delete the nodes that aren't needed by selecting them, and then hit the "Delete" key on your keyboard. The 4 nodes to be deleted can be seen in the image below with the red "X" on them.

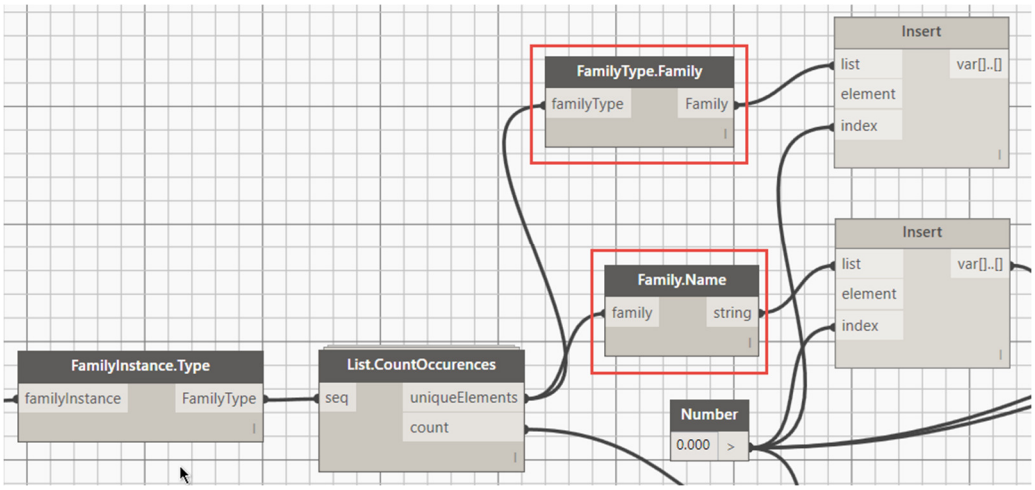


3. The first node we will search for to complete this graph is the **FamilyInstance.Type** node. Add it into the graph between the **All Elements of Category** node and the **List.CountOccurences** node.

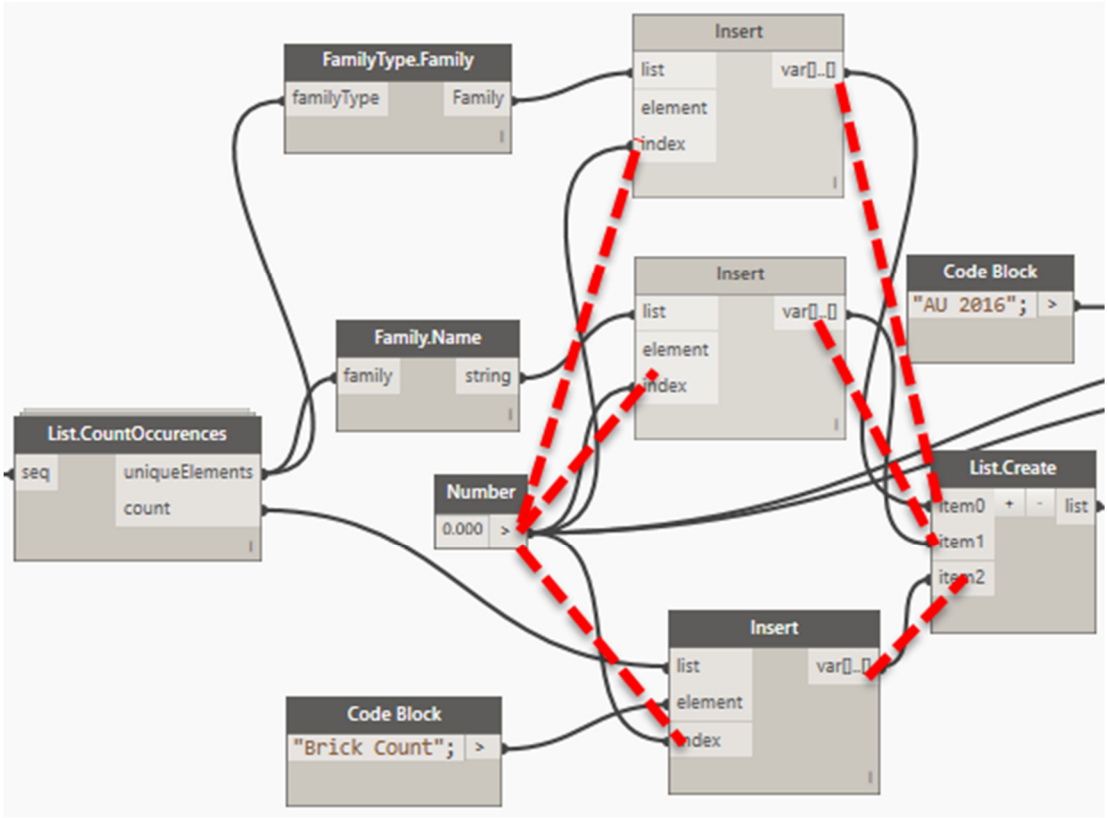


4. This Excel file will have 3 columns in it, so we will need to copy one of the **Insert** nodes. We can do this by using "Ctrl + C" and Ctrl + V". You will need to remove the wires that come pre-connected to the copied version of the node.

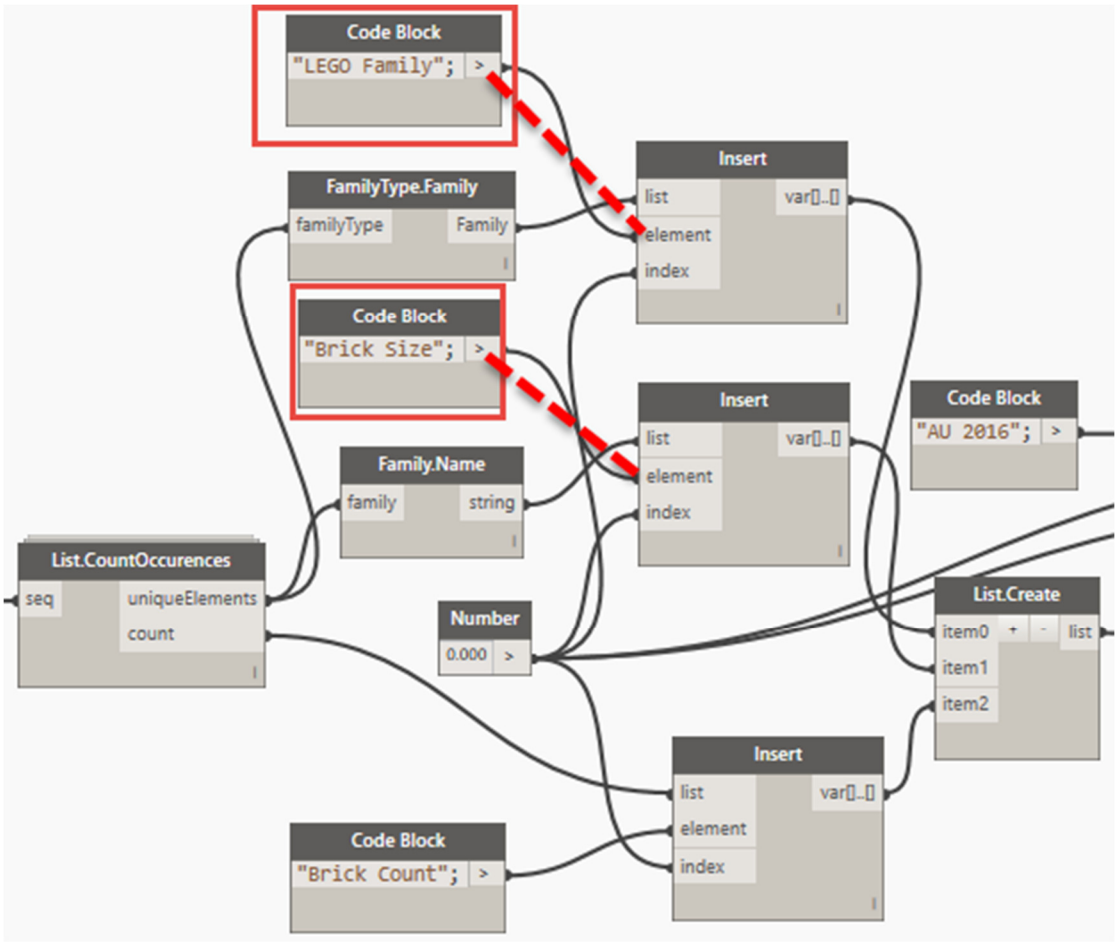
5. The next two nodes we need to search for are, **FamilyType.Family** & **Family.Name.** These nodes are connected to the output of the **List.CountOccurences.** Then their output will connect to the "*list*" input of two of the **Insert** nodes.



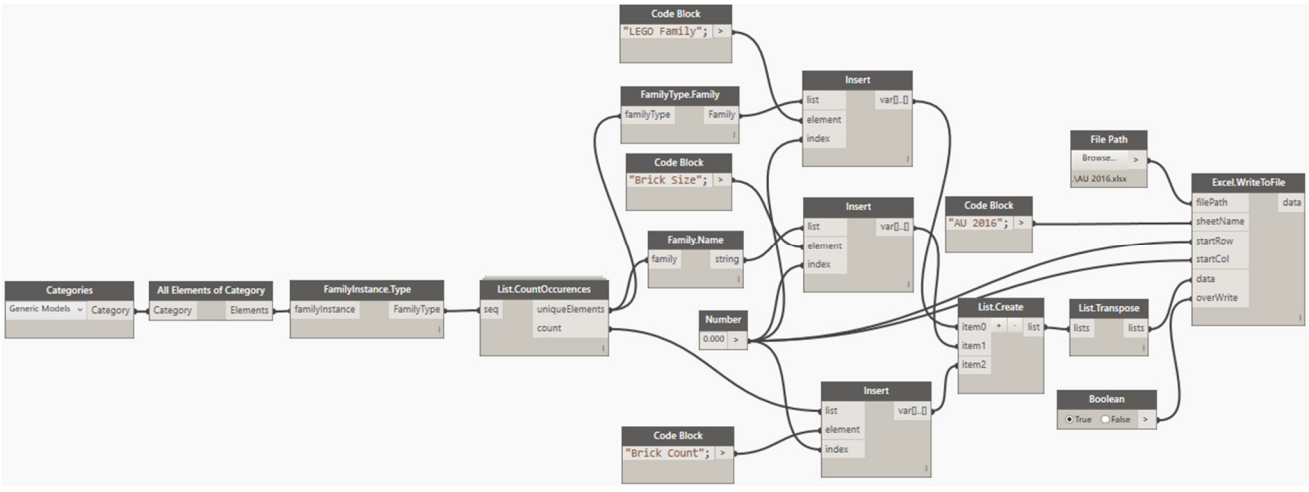
6. Because of the 3 columns, we will also need to add another input to the **List.Create** node. We can add the 3rd input by hitting the "+" button on the node. After creating the 3rd input connect the output from each of the 3 **Insert** nodes into the 3 inputs of the **List.Create** node. We also need to make sure that the Number node (with the number 0 in it) is connected to the "*index*" input of all 3 **Insert** nodes.



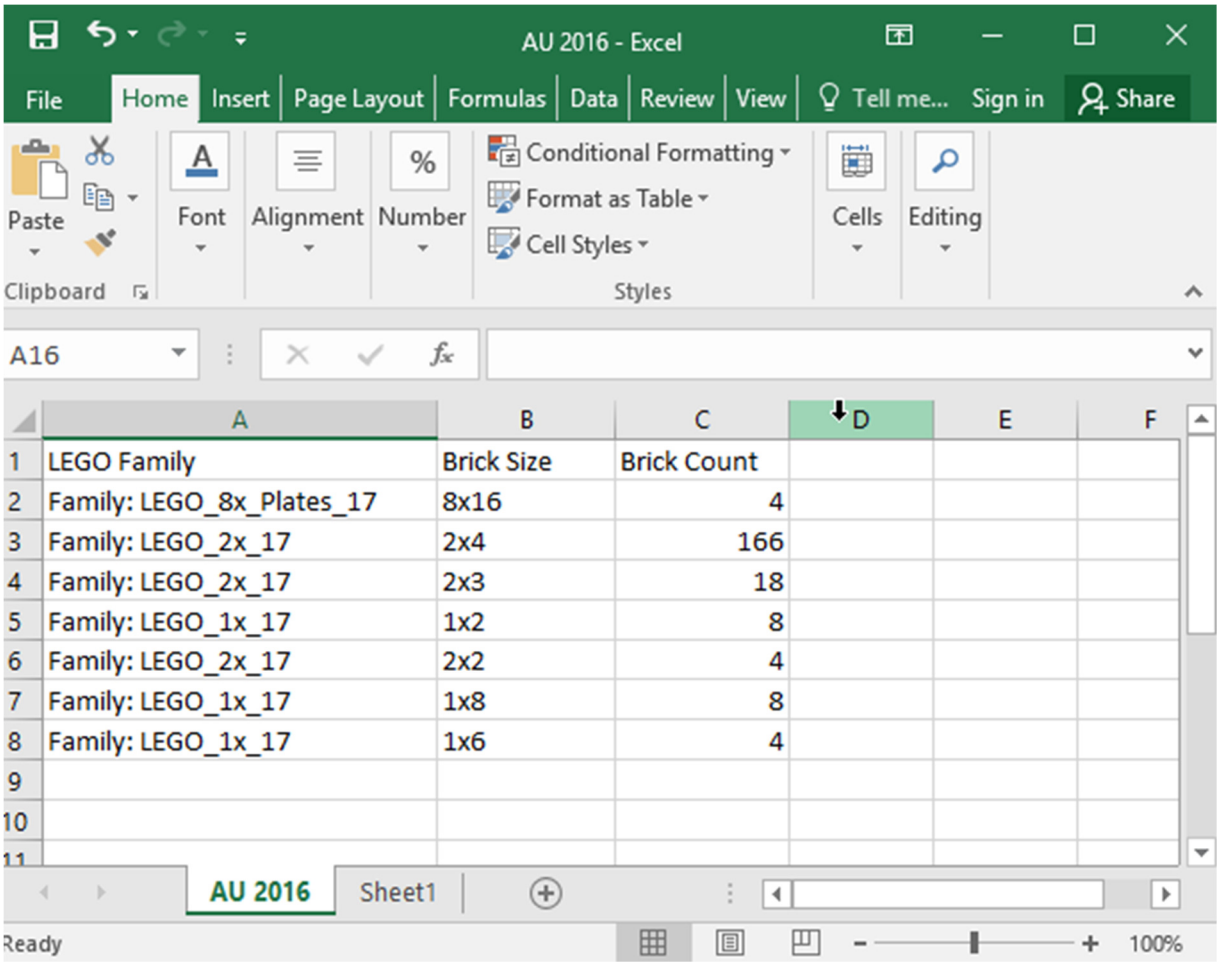
7. Next, we need to add 2 **Code Blocks**, so we can add in the name for each column in the Excel file. The 2 remaining names to go along with "Brick Count" are: "LEGO Family", "Brick Size". These Code Blocks will connect into the "*element*" inputs of the **Insert** nodes.



8. This graph should now be complete, using the remainder of the graph as it came from the last task. Now we can hit the "Run" to see if all the columns and rows ended up in the right locations.



9. If after hitting "Run" the Excel results don't look like the image below you will need to adjust the order in which the **Insert** nodes outputs are connecting into the **List.Create** node until you find the correct order.



	A	B	C	D	E	F
1	LEGO Family	Brick Size	Brick Count			
2	Family: LEGO_8x_Plates_17	8x16	4			
3	Family: LEGO_2x_17	2x4	166			
4	Family: LEGO_2x_17	2x3	18			
5	Family: LEGO_1x_17	1x2	8			
6	Family: LEGO_2x_17	2x2	4			
7	Family: LEGO_1x_17	1x8	8			
8	Family: LEGO_1x_17	1x6	4			
9						
10						
11						

10. Save your Dynamo graph (file), and then close Dynamo, then close Revit.

Resources

Blogs & Websites

- ArchSmarter – Michael Kilkelly's blog about Revit, Dynamo, and Architecture
- archi+lab – Konrad Sobon's blog with lots of great Dynamo content, examples and packages
- Autodesk Revit Structure – A great blog for structural Dynamo users
- Bad Monkeys – A website for a great group of Dynamo cool cats
- BIM 42 – Simon Moreau's blog, lots of good Dynamo and Revit content
- Dynamo Blog – The official DynamoBIM.org blog
- Dynamo Dictionary an open source, searchable database for Dynamo functionality.
- Dynamo Learning Resources – An ongoing post in the Revit Forum about Dynamo
- DynamoNodes – A blog about Dynamo nodes, great place to learn about standard and custom nodes
- Enjoy Revit – A blog with some good dynamo content, it's a little older but still helpful
- Github – Where the open source Dynamo fun is
- Havard Vasshaug – Havard's blog with good (and pretty cool) Dynamo content
- KM – Kyle Martin's blog with some Dynamo content and uses
- landarchBIM – Lauren Schmidt's blog, Dynamo and Revit content for landscape architecture
- Parametric Monkey – A website about computational design with Dynamo (and other) tutorials
- Revit Dynamite and Ammo – blog about Dynamo and Python
- Simply Complex – Marcello Sgambelluir's blog about Dynamo, Revit & making complex things simple
- sixtysecondrevit – John Pierson's blog about Dynamo and Revit in 60-second bites
- Stuff and BIMs – Adam Sheather's blog (the creator of DynaWorks) with some good Dynamo content
- The Dynamo Primer the unofficial online user's manual
- The Revit Saver – Brian Nickel's blog about Revit and Dynamo from an MEP perspective

Courses & Training

- ArchAmarter – Learning Dynamo – This is a paid course created and deliver by Michael Kilkelly
- CADLearning for Dynamo – DVD course for Dynamo intro by Jason Boehning & Marcello Sgambelluri
- Learn dynamo – This is a free resource with more tutorials always being added
- Lynda.com – Dynamo Essential Training – This is a subscription paid course by Ian Siegal
- Lynda.com – Dynamo for Revit Workflow – This is a subscription paid course by Ian Siegal
- Plurasight – An Introduction to Dynamo for Daily Use Within Revit - This is a subscription paid course by John Pierson
- Plurasight – Exploring Dynamo Geometry - This is a subscription paid course by Sol Amour
- ThinkParametric – Dynamo 101 Fundamentals - This is a subscription paid course by Konrad Sobon
- ThinkParametric – Create Custom View Filters using Dynamo - This is a subscription paid course by Konrad Sobon

Podcasts & Videos

- Dynamo – Autodesk's Dynamo YouTube Channel
- Dynamo Thoughts – A Vodcast about teaching and learning Dynamo by Bill Debevc and Ian Siegel
- Konrad Sobon's YouTube Channel – Creator of archi+lab and 4 helpful dynamo packages
- San Francisco Dynamo User Group YouTube Channel
- The Simply Complex Podcast – A Dynamo podcast by Marcello with help from Jason and John

30 Dynamo Packages You Should Check Out

Number	Dynamo Package Name	Latest Version*
1	Ampersand	2016.6.7
2	archi-lab.net	2016.12.1
3	Archi-lab_Mandrill	2017.12.1
4	Archi-lan_Mantis Shrimp	2017.12.4
5	Bakery	2016.9.12
6	BumbleBee	2017.11.2
7	Clockwork for Dynamo 1.x	1.0.2
8	DynamoMEP	0.4.0
9	DynamoUnfold	1.01
10	Dynanimator	1.04
11	DynaWorks 15 / 16 / 17	1.2.0 / 1.2.0 / 1.1.1
12	Flux	2.0.15567
13	Hollandaise	2015.12.23
14	If Equal Return Index	0.1.0
15	Illustrator	0.1.0
16	Juggernaught	1.0.8
17	Ladybug	0.1.3
18	Landform	2016.5.24
19	LunchBox for Dynamo	2016.9.30
20	MeshToolKit	1.1.0
21	Optimo	0.1.2
22	Prorubim Common Kit (Dynamo 4 Revit)	0.1.3
23	Rhynamo	2016.5.3
24	Rhythm	2016.10.6
25	Slingshot! For Dynamo	2014.11.30
26	SpaceLayout	1.0.5
27	Solar Analysis for Dynamo	0.9.2
28	spring nodes	100.0.1
29	SteamNodes	1.0.0
30	Zebra	2016.7.2

*As of October 6th, 2016

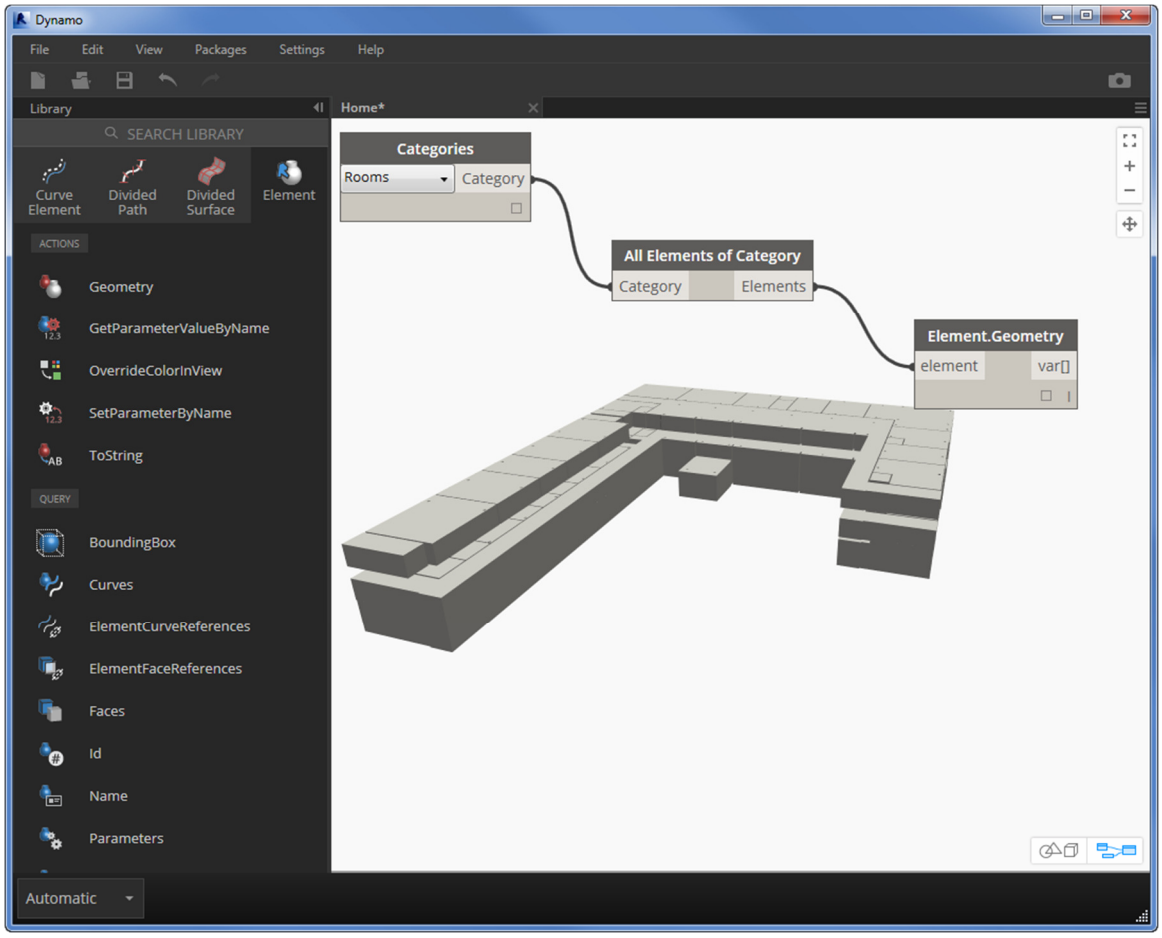
Real World Dynamo Examples

These real world workflow examples have been included with the permission of Marcello Sgambelluri. They have come from his blog "Simple Complex" as well as his past Dynamo sessions from RTC and AU.

NOTE: All of the following examples were created in versions of Dynamo before 1.0.

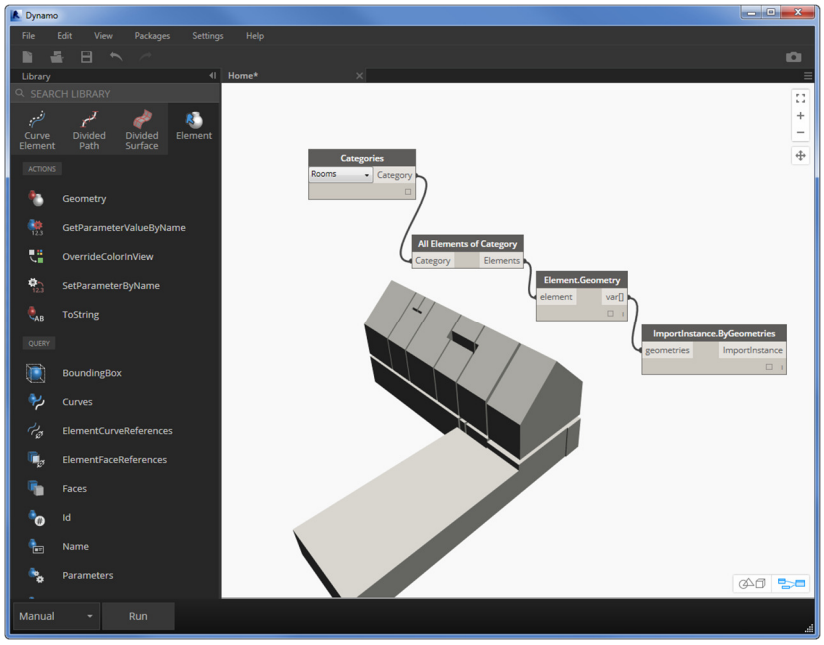
View Rooms in 3D in Dynamo

- This only requires 3 nodes
  - Categories
  - All Elements of Category
  - Element.Geometry
- Example:

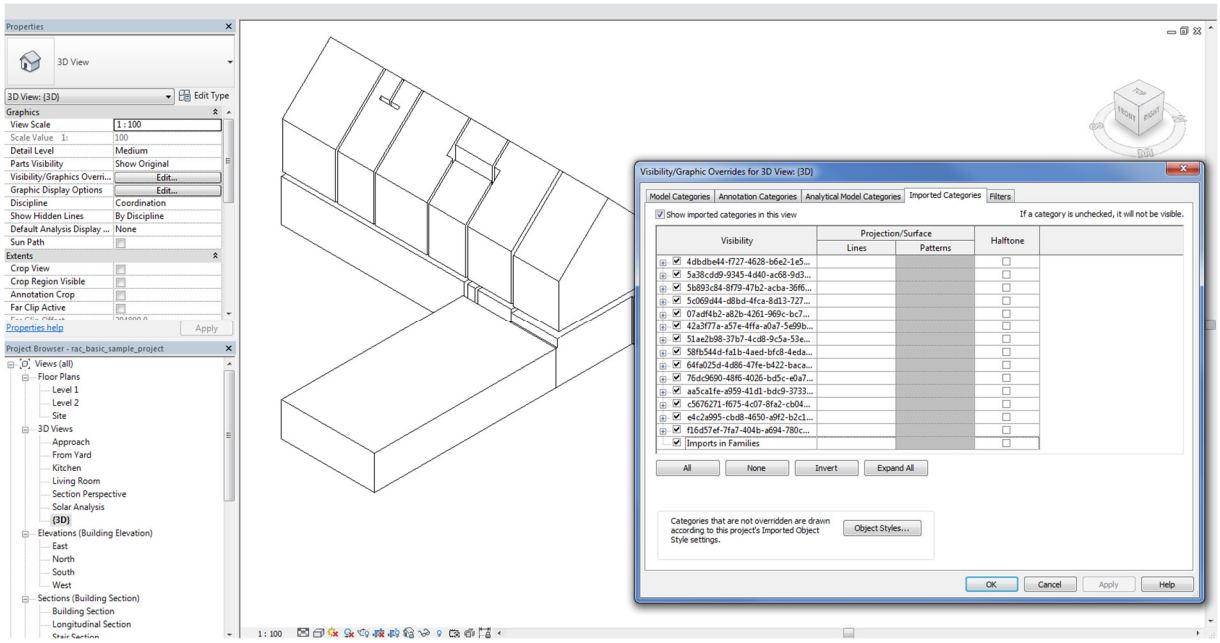


View Rooms in 3D in Revit

- This only requires 4 nodes
  - Categories
  - All Elements of Category
  - Element.Geometry
  - Importinstance.ByGeometries
- Example:

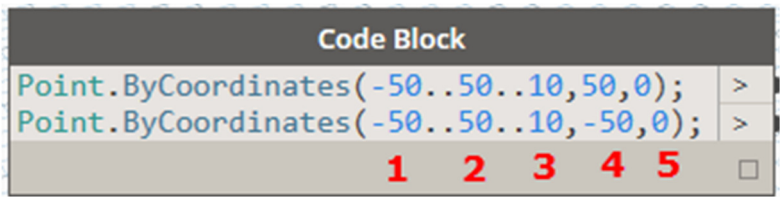


- The geometry is imported into Revit in the default 3d view (which needs to be already created)
- It comes in as imported symbols (check under “Imported Categories” in the VG dialog box.)
- Turn off all model elements in VG dialog box to see the 3d rooms

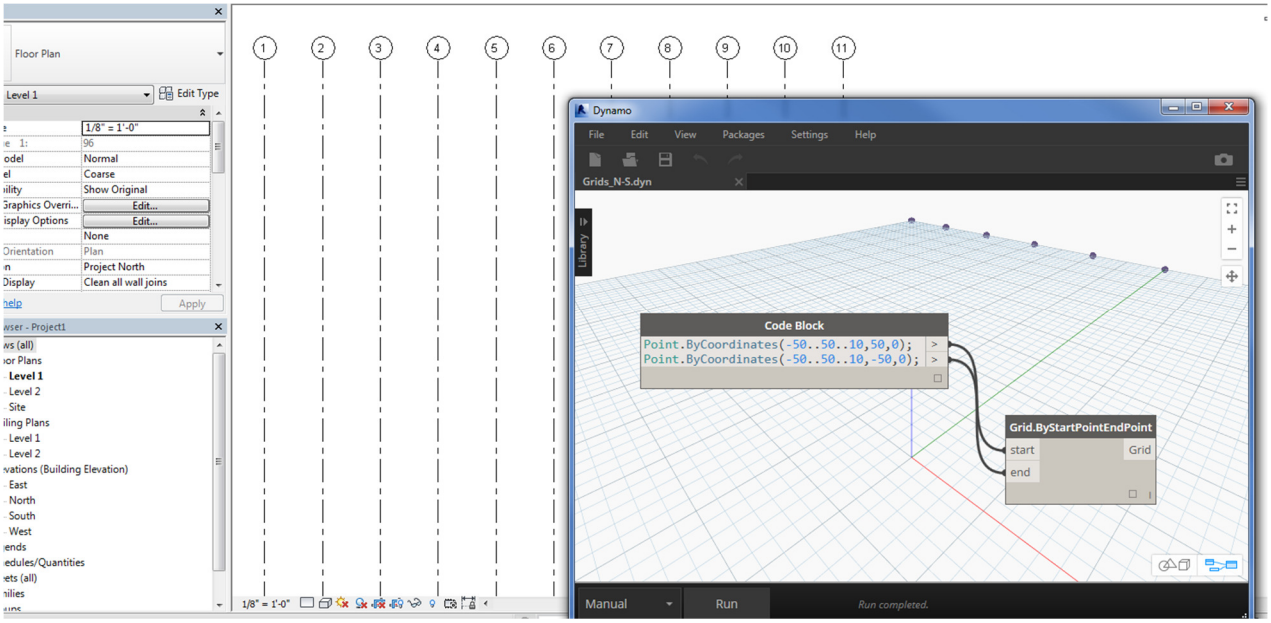


Create Revit Grids with Dynamo

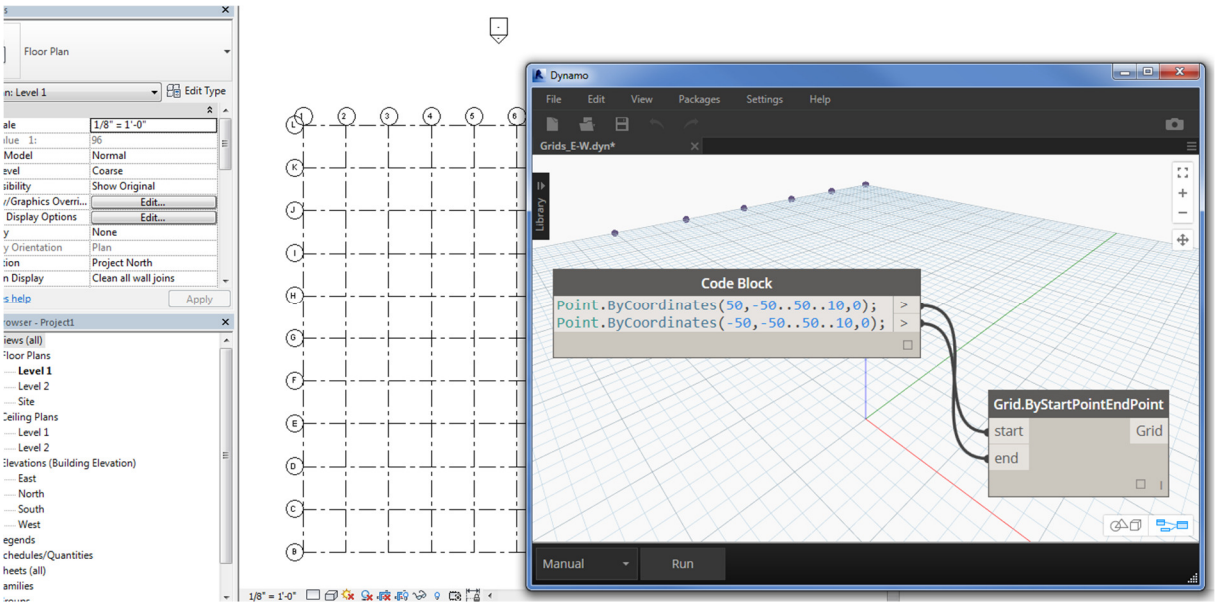
- This only requires 2 nodes
  - Code Block (create by double clicking anywhere on the Dynamo canvas)
    - In this example, the numbers in the code block represent the x,y,z coordinates for the start and end of each grid. The "x" is created by using a "range"
      - 1st = Start location (x) of the grid (the range start)
      - 2nd = End location (x) of the grid (the range end)
      - 3rd = The step between each grid, in this case, 10 units (the range step)
      - 4th = (y) coordinate
      - 5th = (z) coordinate
        - This means Dynamo will create grids ever 10 units starting at -50 and ending at 50 (10 grids)



  - Grid.ByStartPointEndPoint
- **Note:** if you change one of the grid numbers Dynamo will renumber them all
- **Note:** for east – west grids simply switch the "x" and "y" values in the first node (see example 2)
- **Note:** the number and letter always appear 1 higher than you enter (0 =1, A = B)
- Example 1: North- South Grids

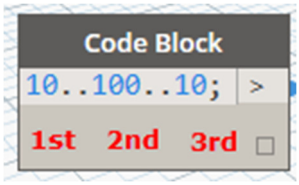


- Example 2: Addition of East-West grids (separate graph)

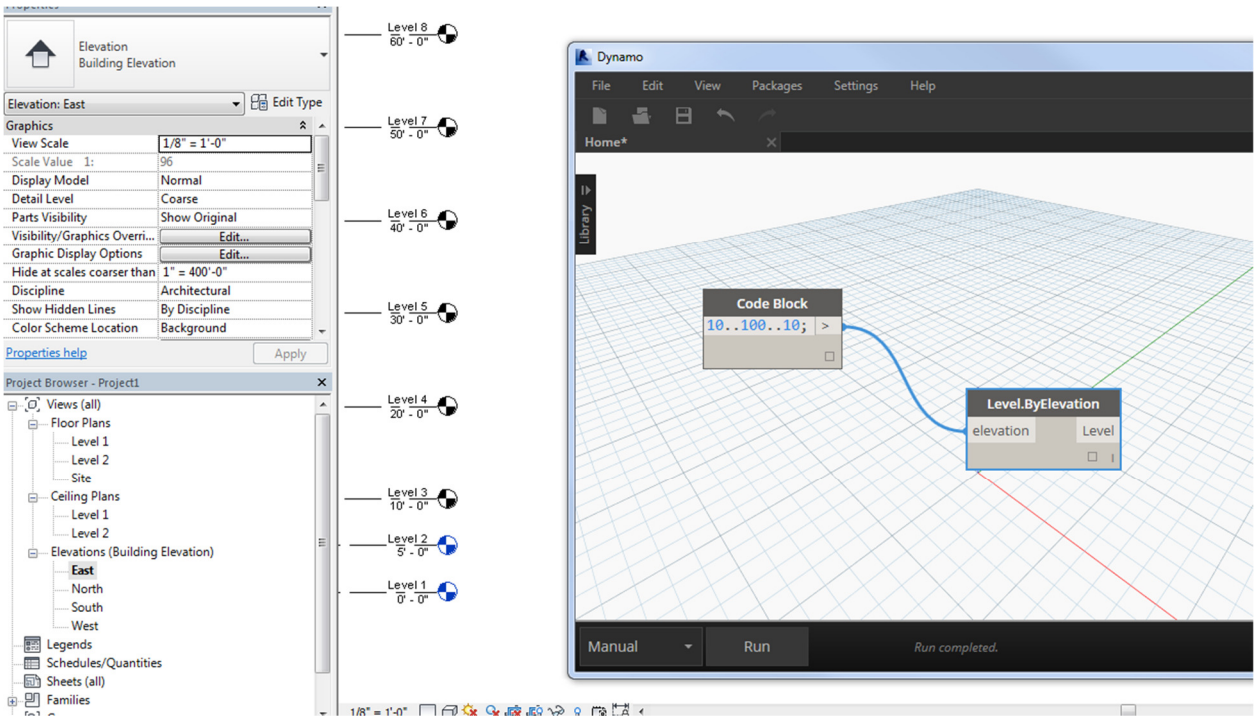


Create Revit Levels with Dynamo

- This only requires 2 nodes
  - Code Block (create by double clicking anywhere on the Dynamo canvas)
    - In this example, the numbers in the code block represent the following
      - 1st = Starting height of first level, 10' in this case
      - 2nd = The Range of the levels, 100' in this case
      - 3rd = The step of the level, in this 10'
        - This means Dynamo will create levels ever 10' starting at 10' of height until it has filled 100' (10 levels)

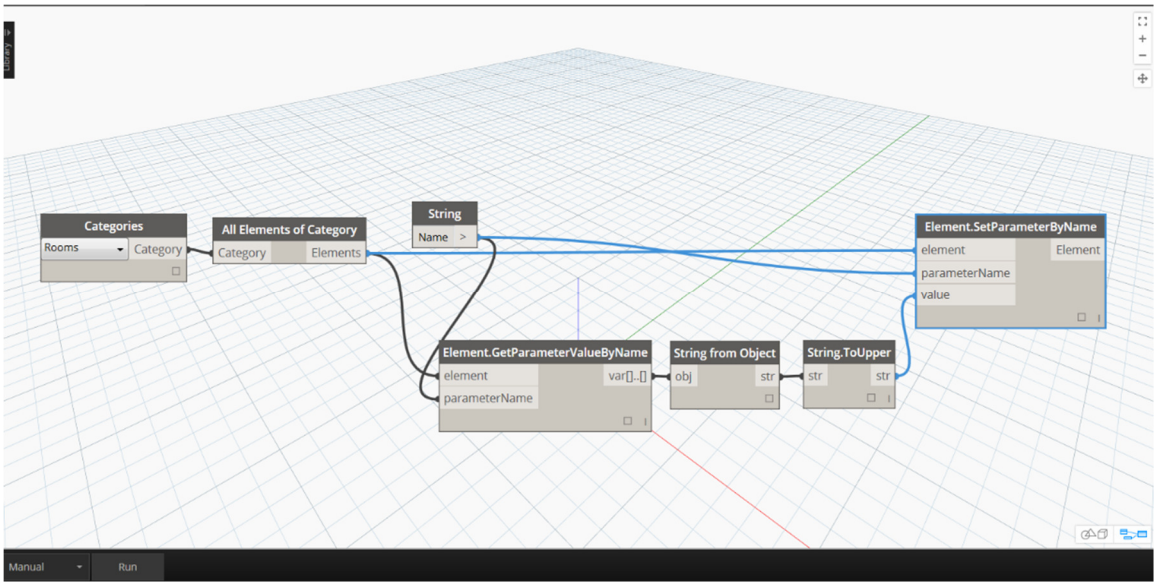


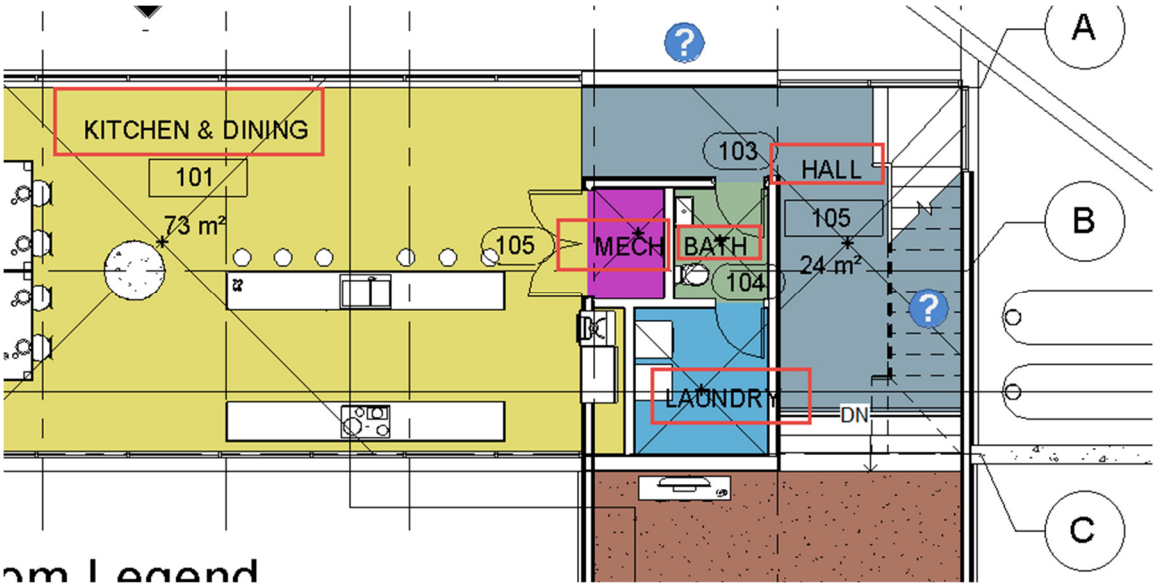
  - Level.ByElevation
- **Note:** if have created levels manually in Revit (the first one is helpful) Revit will continue with that naming convention sequentially.
- **Note:** make sure the start location works with any existing level within your project, so level aren't created on top of each other
- Example:



Change Text in Room Tags to all Caps with Dynamo

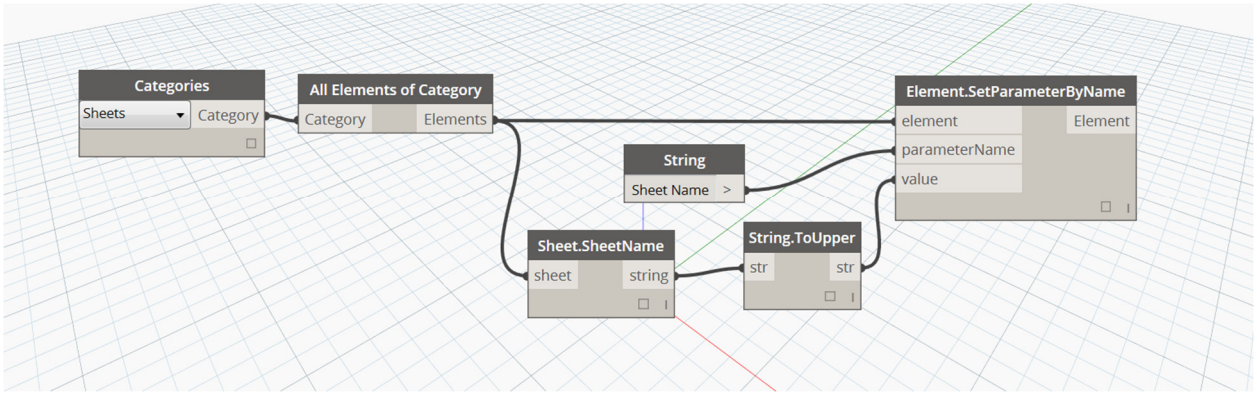
- This requires just 7 nodes
  - Categories
  - All Elements of Category
  - String (can also use Code Block)
  - Element.GetParameterValueByName
  - String from Object
  - String.ToUpper
  - Element.SetParameterByName
- Example:

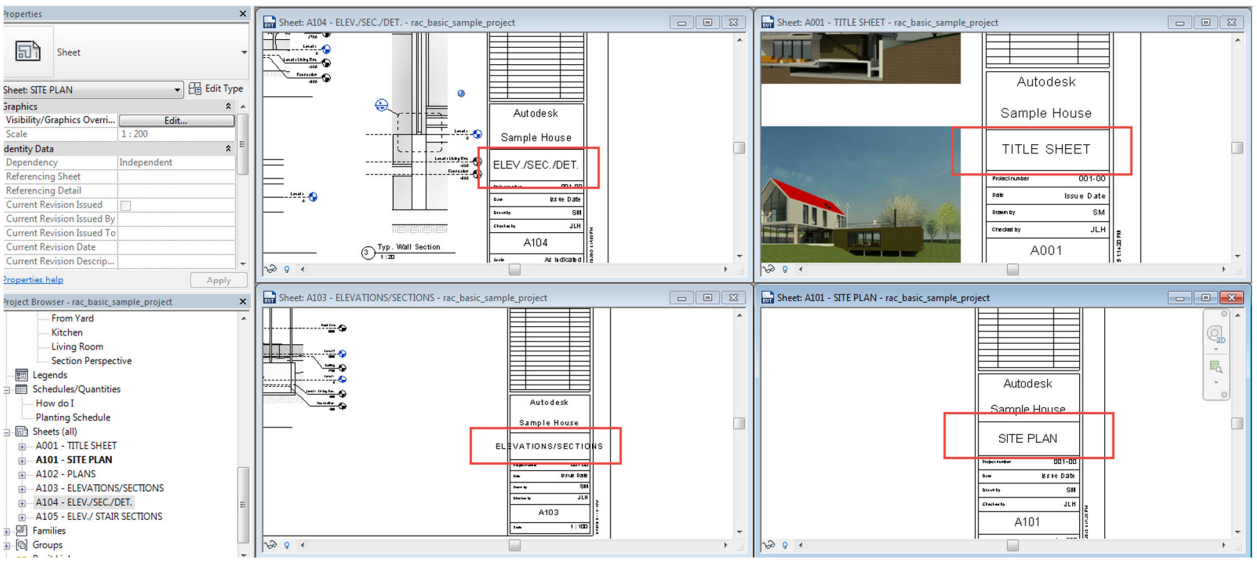




Change Text in Sheet Names to all Caps with Dynamo

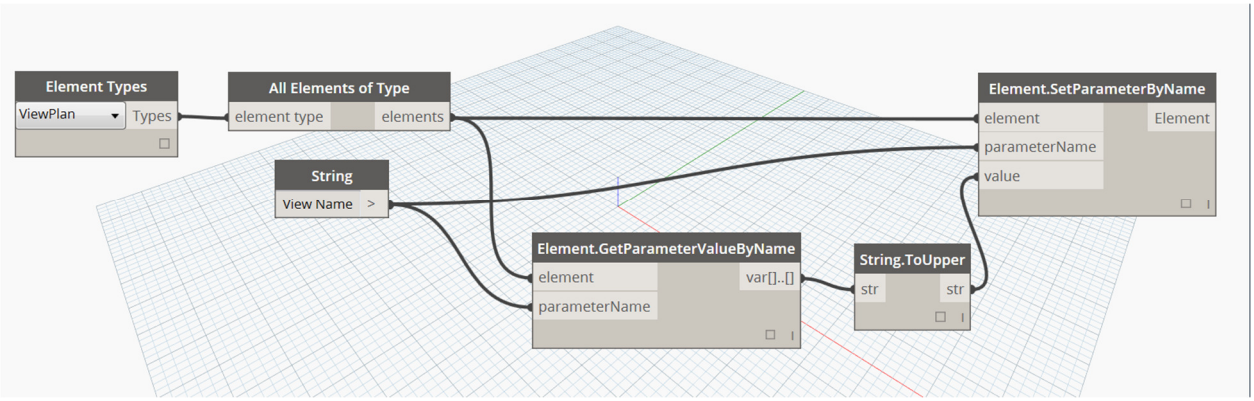
- This requires just 6 nodes
  - Categories
  - All Elements of Category
  - String (can also use Code Block, but I find string easier)
  - Element.GetParameterValueByName
  - String.ToUpper
  - Element.SetParameterByName
- Example:

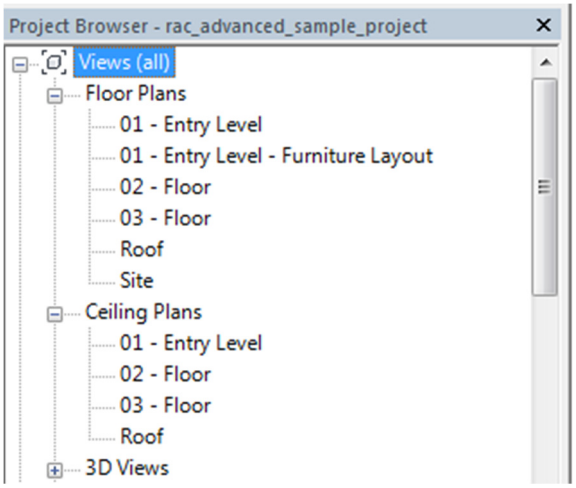




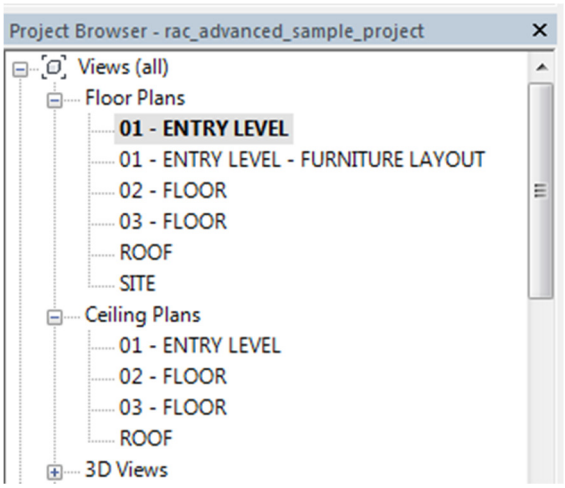
Change Plan View Names to all Caps with Dynamo

- This requires just 6 nodes
  - Element Types
  - All Elements of Type
  - String (can also use Code Block, but I find string easier)
  - Element.GetParameterValueByName
  - String.ToUpper
  - Element.SetParameterByName
- **Note:** this only works for “Plan” views
- Example:





Before



After