AS123345

Revit and Dynamo for Interior Deign

William Carney
BSA LifeStructures

Learning Objectives

- Learn Revit techniques for Interior design and documentation when linking an architectural model into an interior design model
- Learn how to utilize Dynamo to simplify room management when linking an architectural model into an interior design model
- Understand how Dynamo can help connect construction documentation with modeled content from presentation graphics
- Learn Revit and Dynamo workflows to help streamline the design and documentation process

Description

A common challenge for an interior design firm is working as a consultant for an architecture firm and keeping room names, numbers, and finish information updated to populate room finish schedules. This is easily managed if you use Dynamo software to read information from the linked model and synchronize information in the interior design model. Another common struggle is coordinating annotated information, room finish schedules, and material finish legends. A number of work-arounds exist, though all produce complicated workflows leaving a disconnect between what is actually modeled and what is documented. A similar method to syncing room information from a linked model can be used to read finish information in a Revit model to populate the room finish schedule. This presentation will be a combination of Microsoft PowerPoint and a live demonstration of techniques that BSA LifeStructures interior designers employ to improve the process of designing and documenting in Revit software with assistance from Dynamo. We'll use PowerPoint to diagram the flow of information from phase to phase, or modeled content to construction drawing, followed by live demonstrations using Revit and/or Dynamo. Topics will include synchronizing room finish schedules with modeled materials, connecting rooms from linked models, coordinating items from linked models such as signage for doors or light fixtures, coordinating data external to Revit such as room data sheets or FFE requirements, and working through the design phases. This class will also cover finish drawings, signage drawings, and furniture drawings, with a focus on synchronization of information for producing a visualization-ready model while reusing those efforts to produce construction documents.

Speaker

William Carney is the BIM director at BSA LifeStructures where he is responsible for overseeing the firm’s use and implementation of design technologies. As an efficiency enthusiast, William seeks to find ways to use technology to streamline BSA LifeStructures multi-disciplined workflows. He obtained his Bachelor of Science degree in architectural studies from Southern Illinois University Carbondale and his master’s degree in architecture from the NewSchool of Architecture and Design in San Diego. William has worked on a wide range of project types and

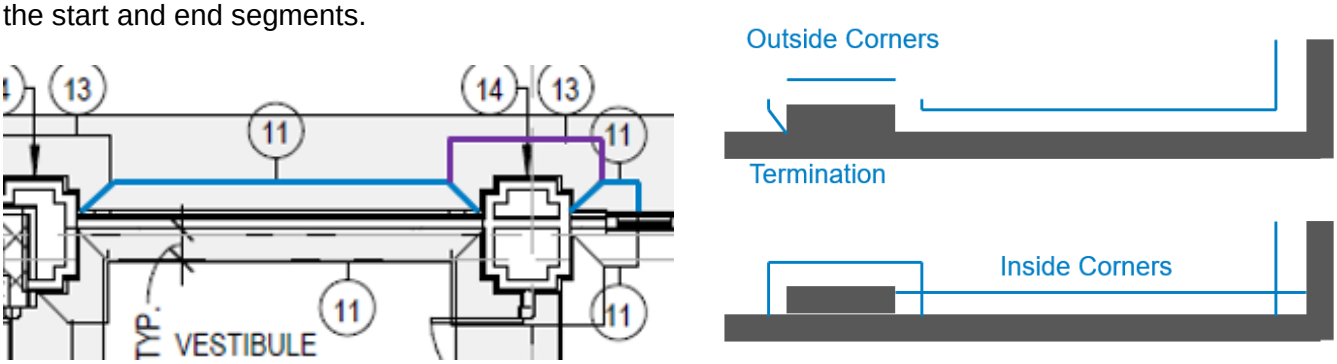
sizes throughout his architectural career. He is also an Author for Lynda.com and is actively involved with the St. Louis Revit User Group as one of its steering committee member.

Brief History

A Revit and Dynamo Interior Design workflow is a work in progress at BSA LifeStructures. Many of the processes shown in this handout/presentation will evolve with time. Months ago, our director of Interior design and I had a conversation regarding the Interior discipline's design and documentation process. We came to the conclusion that we had a typical Interior Design Process that we could improve upon.

As our director and I spoke about our process we honed in on how we indicate wall finishes. Initially this information is shown as a color fill scheme during preliminary design. As we proceed to design development phase, we present a plan indicating where wall finishes are located. Each wall finish is graphically represented by a thick colored line. The line is initially colored based on BSA standard colors until we know the actual color scheme of the space. As the project develops, we'll often select a color of paint and match our line to the material color. Line is currently an annotation and not a modeled element so when we create presentation graphics of the space, this information needs to be coordinated.

As we move into construction documents, the lines are displayed differently and get noted with a different annotation (Shown below). A line is drawn offset from the wall with an angled line at the start and end segments.

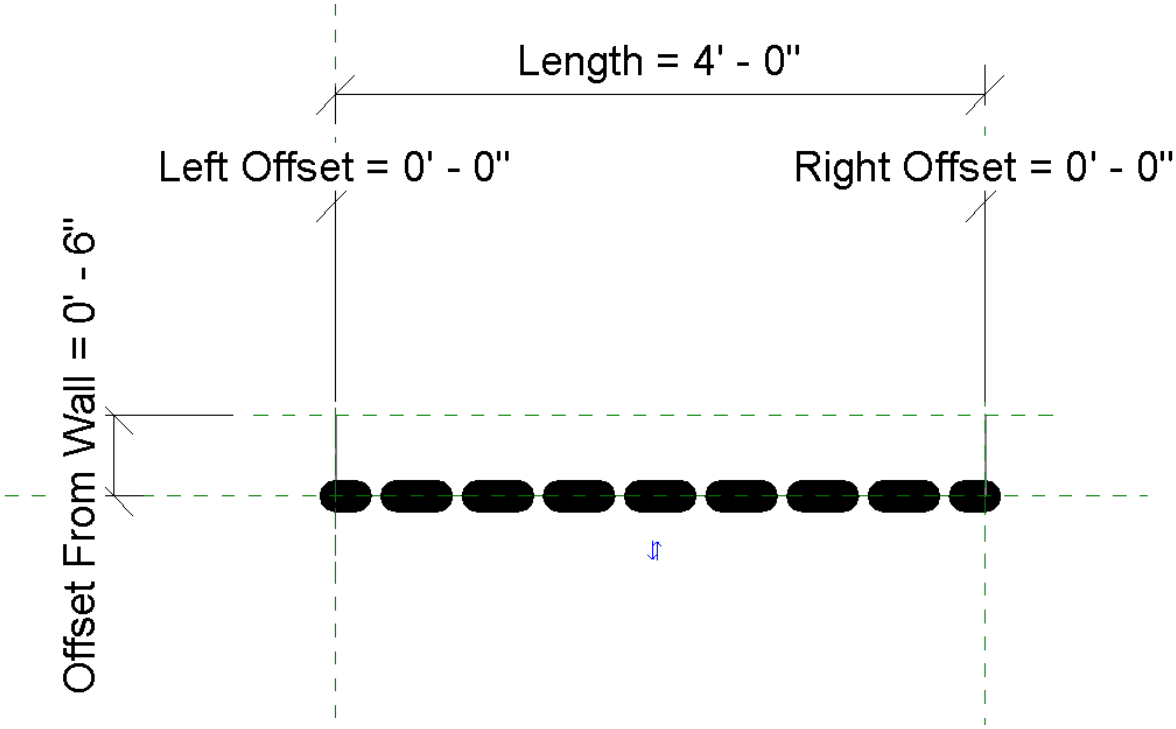


Typical annotation of Accent Wall (Colored for diagrammatic purpose)

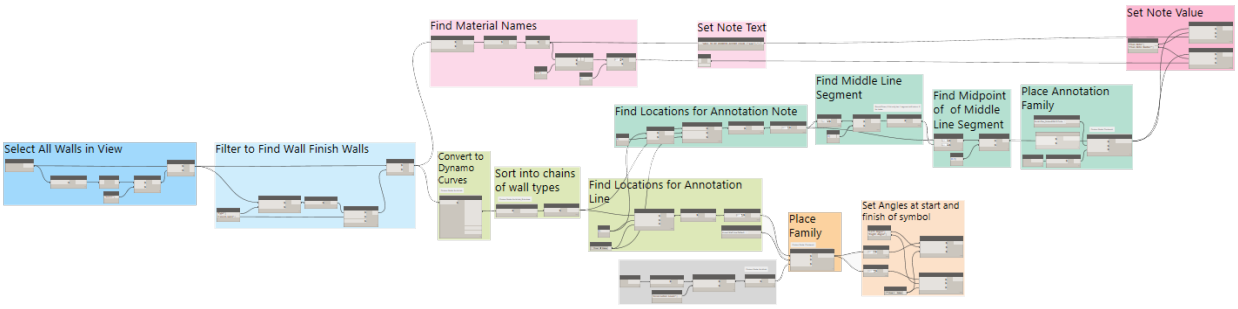
It seemed simple enough to make a family with the annotation and material color using either a yes/no parameter to toggle them or view detail level to show one or the other. I was wrong. Because of the offset from the wall, when the finish turns around an inside or outside corner, the line either needs to extend or shorten. Most solutions left this as either a cumbersome or worse a manual process. Based on the direction the wall turned, the family either needed to extend or shorten.

I knew we needed something better. Going back to the drawing board, I scrapped the idea of it being one family and considered automating the annotation from what was modeled. With Dynamo, I can read line based families, walls, or detail lines, and use the geometry to place other geometry. Rather than worrying about weather each end condition extend or shortened along its path, Dynamo could offset the path and draw the annotation. The angled line would

need to extend from the annotation line to the wall and this happened at the start and finish of the wall.



Accent Wall Annotation Family



.dyn to place wall finish annotation based on wall finishes in views

Our Process:

We have a pretty good foundation to build on as a process. Some of the steps to from schematic design to construction documents were cumbersome. Some of the steps to from schematic design to construction documents created a situation where we had to manually Re-enter information from a previous phase in a different form.

Preliminary Design

During the early design phase, we typically specify finishes for types of spaces, by describing the quality and general type based on the function of a space. This information is stored as a Room Group Parameter so we can quickly show the groups graphically on a colored floor plan with a Room Color Fill Scheme.

	PUBLIC	SEMI-PUBLIC	PATIENT PRIVATE	STAFF PRIVATE	BACK-OF-HOUSE
FLOORING	Terrazzo at main lobby and connector to existing Cancer Center transitioning to large format porcelain tile in main public concourse, dining, servery, and retail areas. Carpet (budget install allowance $ 38.00/SF) at Chapel and insets in main waiting areas with walk off carpet tile at all entrance areas. Open stair at main lobby to have terrazzo treads. Porcelain tile base at porcelain and carpet locations. Precast terrazzo or stainless base at terrazzo locations.	No-wax sheet vinyl flooring (heat-welded) with 4" resilient base at off-stage corridors and in patient units. Carpet with 4" resilient base at registration, consultation rooms, small or enclosed waiting areas, conference rooms and classrooms. Large format porcelain tile with porcelain tile cove base in public restrooms. Stainless steel accent trim as necessary.	No-wax sheet vinyl flooring (heat-welded) with 4" resilient base in most areas. No-wax sheet vinyl flooring (heat-welded) at procedure, treatment, pharmacy, labs, LDR, and Trauma with integral base. OR procedure rooms and decontam areas to receive epoxy flooring with integral base. Toilet rooms within patient rooms, dayrooms, exam and procedure areas which are adjacent to resilient flooring should also be resilient with 6" resilient base. Porcelain tile in public and staff restrooms with porcelain tile cove base. Spa/Toilet to receive upgraded tile with specialty tile or glass accents.	Porcelain tile in restrooms with porcelain tile cove base. Carpet in offices, retreats, conference/education rooms. VCT in locker rooms, break rooms, work/copy rooms, and storage with 4" resilient base. No-wax sheet vinyl flooring (heat-welded) with integral base in pharmacy and labs. No-wax sheet vinyl flooring (heat-welded) with 4" resilent base in med and clean rooms.	Epoxy flooring at central sterile with integral epoxy base. Rubber treads at stairways with rubber sheet on landings. Sealed concrete in materials management, utility rooms and utility corridors with 4 " resilient base. Penthouse floor and curbs shall have epoxy floor system similar to General Polymers EPO-FLEX MER II. Soiled and Clean linen to receive no wax sheet vinyl with resilient base except in Surgery areas which should have integral base. Areas adjacent to sheet vinyl without door separation to receive same finish. VCT with 4" resilient base elsewhere. Food Service kitchen to be quarry tile with coved quarry tile base. Grout to be epoxy.
WALLS	Painted drywall with high-durability paint. Reveals for seaming and architectural detailing. Exterior masonry (brick and or limestone) feature walls in public concourse. Specialty wood accents at feature locations. Porcelain tile wall-cladding at elevator lobbies Tile wet wall with decorative tile/glass accent at public toilets. Servery/dining to have porcelain tile wall with glass mosaic accents. Slat wall to be incorporated in gift shop. Stainless steel corner protection to be integrated into architectural detailing.	Painted drywall with high-durability paint, eggshell finish. Epoxy paint in wet areas. Type II vinyl wallcovering or paint at accent walls. Patient/Clinic corridors to have double bump rails (floor and handrail). Partial height PVC- free wall protection under handrail with paint above in surgery, radiology, procedure and dayroom corridors. PVC-free corner guards to be at patient corridors.	Painted drywall with high-durability paint, eggshell finish. Epoxy paint in wet areas. Patient headwalls/footwalls to be partial plastic laminate panels, metal trim/reveals, and accent material. Glass and ceramic tile used as accent at patient toilets.	Painted drywall with high-durability paint, eggshell finish. Epoxy paint in wet areas.	Painted drywall with high-durability paint, eggshell finish. Epoxy paint in wet areas. Partial height PVC-free wall protection in corridors, equipment rooms with PVC-free full height corner guards. FRP in Food Service kitchen .

Colored Floor Plan and Finish Matrix

Schematic Design

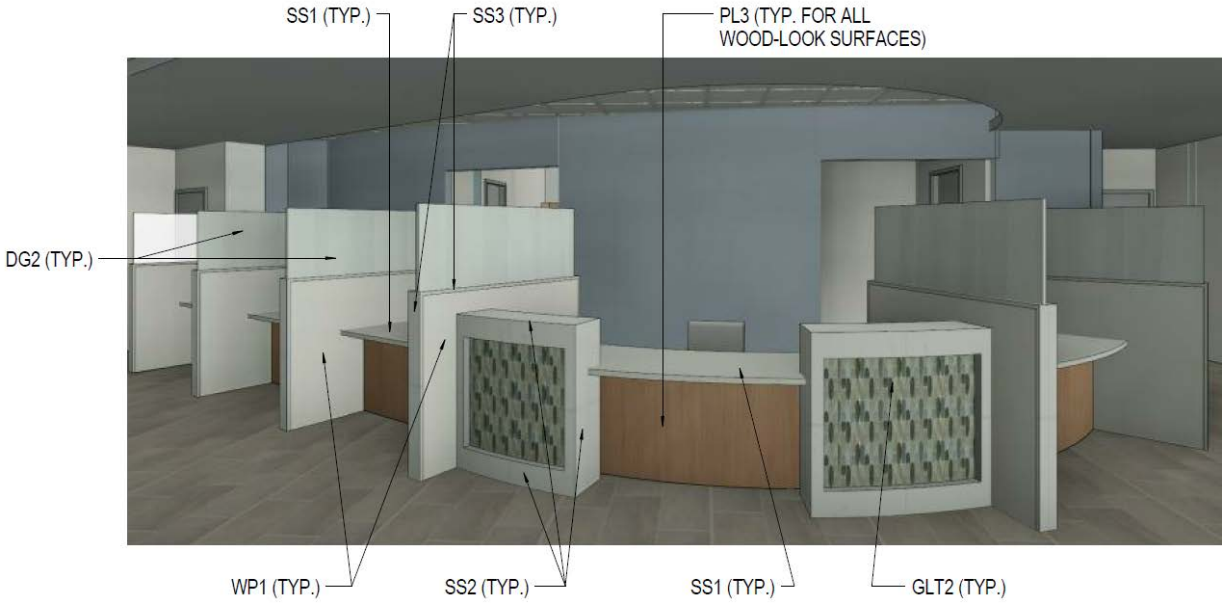
During schematic design we'll often use a blend of color fill schemes and modeled geometry. As an example our design team may have a concept of a corridor floor pattern that they'd like to discuss early as it can inform many of the design decisions along the way. With the frequency of design changes during this phase we'll avoid modeling finishes when it doesn't necessarily add a ton of value to the process. Our schematic perspective images show few finishes if any and are meant to convey space or a concept.

Colored Scheme and Modeled Floors

Design Development

In design development we usually use perspective graphics, a full colored floor plan and indicate where wall finishes will be applied. This is usually put together as part of a larger sign-off package. The goal here is to show what you need to in order to make sure the

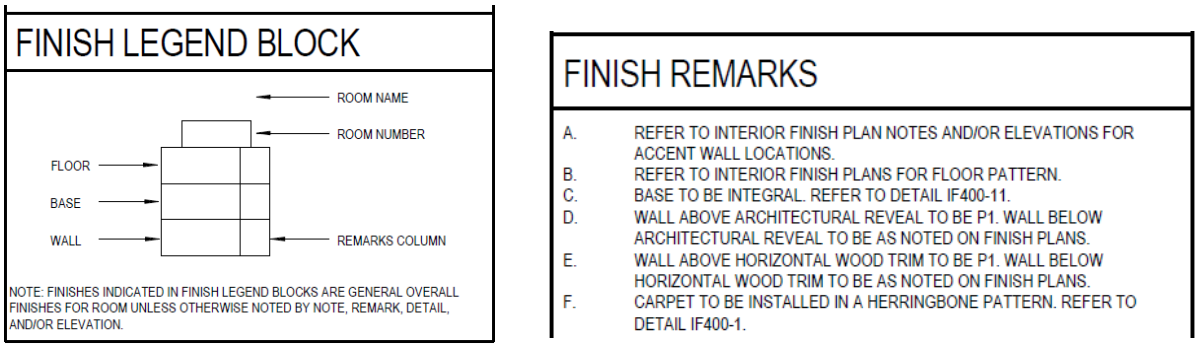
owner and user groups understand what the building will look like and how it will feel. All of these graphics coordinate information from the previous phases of work.



Revit View with Realistic Visual Style and Annotations

Construction Documents

The hope is that the design is final in design development and construction documents are used to create annotative directions for a contractor to construct the design. We take the information from our previous phases and display it in the best way that we can for the contractor's team to understand where and how each finish should be applied. Some of the things we do are decisions made as a group to best convey that information. Our interior designers indicate the room finishes in a tag associated with the room. There is not a traditional room finish schedule but we do tag rooms to display a room parameters material and a material take off to create the finish specifications



Finish Block Diagram

The Goal

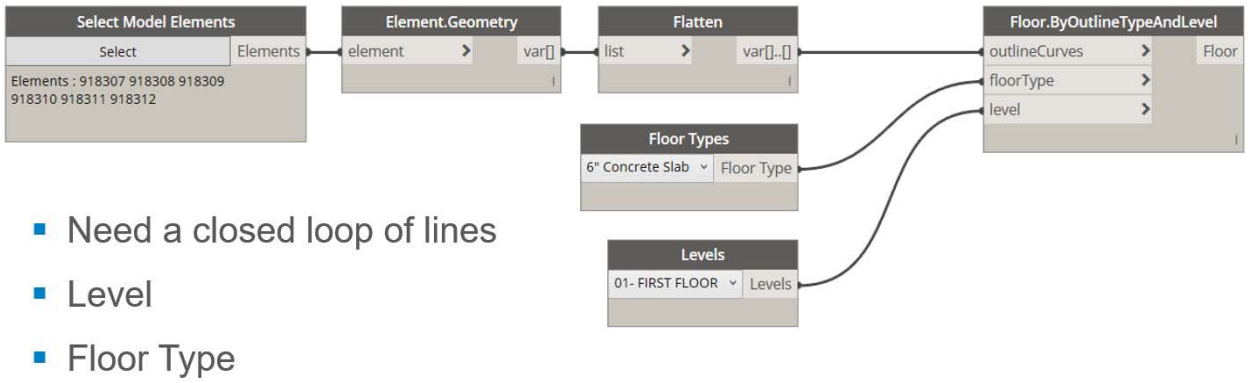
Our goal is to build upon our current process to eliminate duplicate work. We want to be able to create a better product that can more quickly be used to convey our design to the owner and the building's occupants while synchronize as much of the information in our model as we can. Most of the solutions for this overall goal involve automating modeling from rooms and reading what is modeled to keep the rooms and the model synchronized. The materials allow us to track what is in our model and quickly present a design in a number of methods that match our current look and feel.

Placing Finishes with Dynamo

With Dynamo, you can place line based families, point based, face based, hosted, and a slew of other methods of family placement. Often I find myself thinking this is so cool but really is it faster than the command in Revit? Often I find the answer to the question is that it is not. Where the benefit comes is the ability to place geometry based on other geometry or store information for updating the Dynamo placed geometry in the future.

Place Flooring

A floor is considered a sketch based system family. It requires the Revit user to sketch a boundary that will be filled with the specified structural makeup of a floor. In Dynamo it's pretty easy to place a floor. Below is a workspace where you can select lines and dynamo will use those lines to place a floor at the indicated level. This isn't any faster than drawing the floor in fact it's slower because you have to open Dynamo and pre-draw the lines you are selecting. However, if you build on this, it's actually pretty powerful.



- Need a closed loop of lines
- Level
- Floor Type

.dyn to place floor.

Change your thinking to "What can I use to get outlines?" or "Is there a way to get a floor type from something?" This is one of the many things that makes Dynamo really powerful. Rather than select the lines you need to make a floor, Select the room you need, read the floor finish parameter, and place a floor based on the boundary of the room. Or select a room to get the level and finish and the ceiling above to get the outline to create a floor that follows a soffit.

What if you store the room ID in a floor parameter to reference it later for boundary updates?

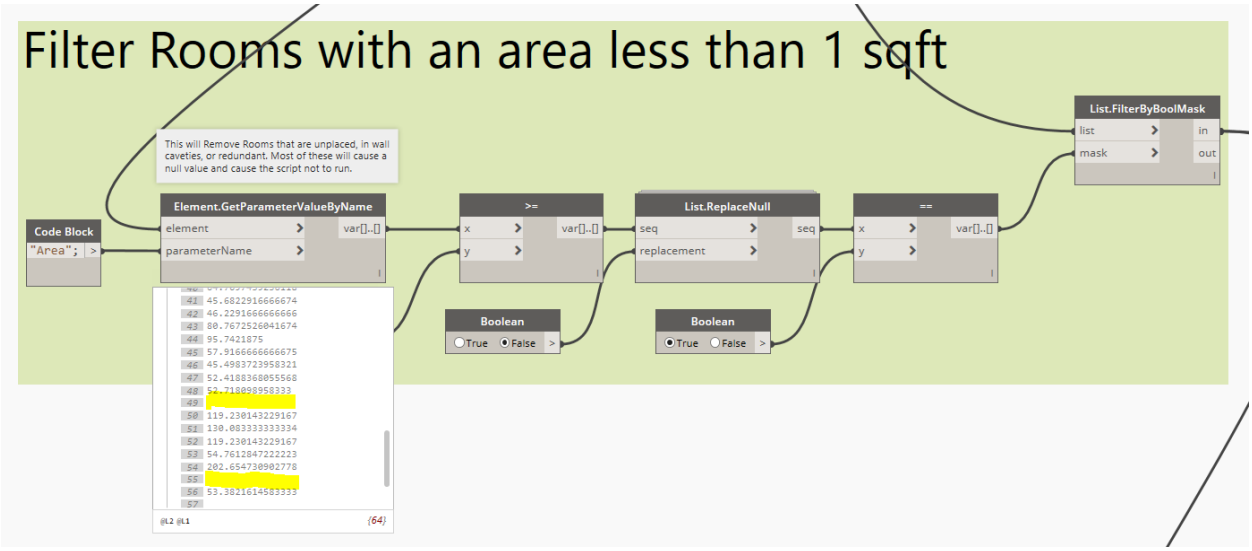
Selecting Rooms

This section is going to walk through a workspace that selects all the rooms in a model, finds their boundaries, reads the specified floor finish, and places a floor in their boundaries. In Dynamo it's fairly easy to select all the rooms in your model.

Select all rooms in the model

Filtering

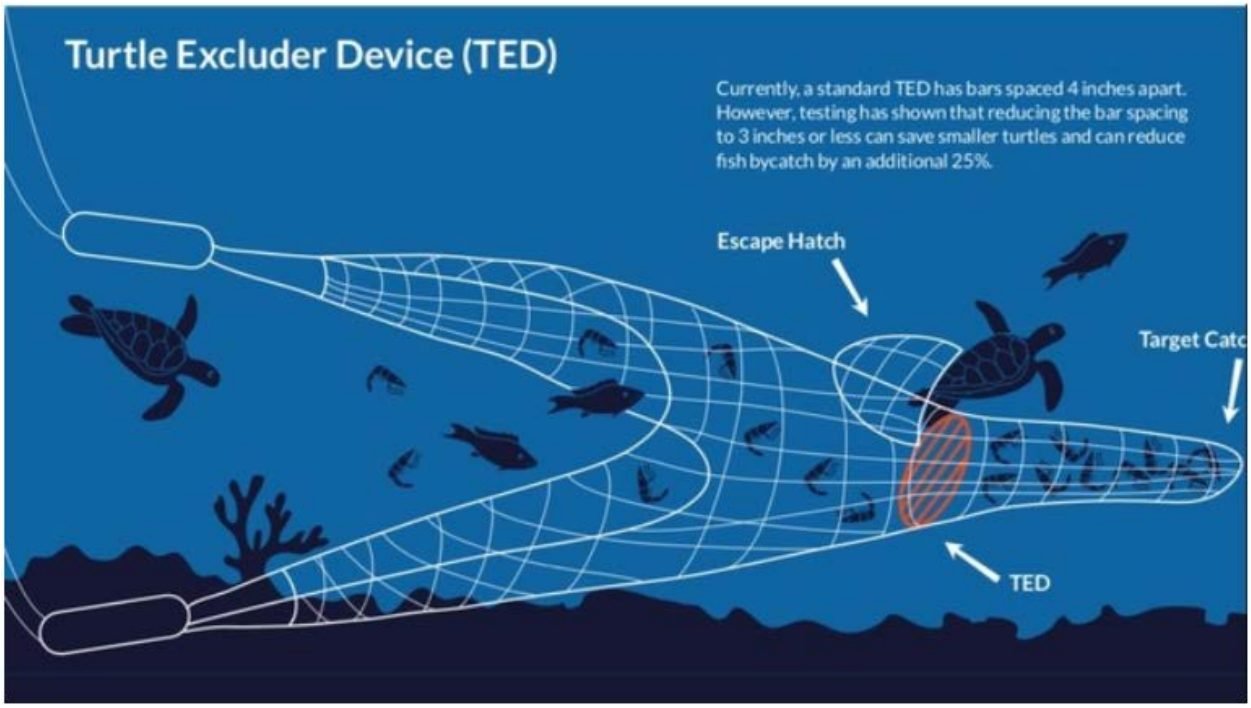
This reads **EVERY** room in the model. Why I highlight every, is that this includes the non-placed rooms, redundant rooms, rooms of each phase, and unbound rooms. All of these variables can cause you issues when you're trying to place the floors. You may also have rooms in your project that do not get a finish. An existing room where no work is expected or an unoccupied space may not receive a finish. There may also be rooms which you have already modeled a floor and you don't want to place a new finish in them. All of these are things that need to be filtered out.



Highlighted rows show rooms without an area. These can be unplaced, redundant, or unbound

I love to use the analogy of TED to describe filtering. Commercial Fishing nets have methods of capturing the correct size and type of fish they are scooping/fishing for. They use different sized weaves and a **Turtle Escape Device** to let unwanted items like sea turtles escape to minimize their impact on the ocean. It's admirable but more important

to me a perfect way to describe filtering. You collect everything, in an easy scoop, and let some things pass through to get what you want.

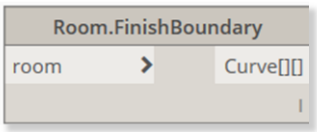


https://www.news4jax.com/weather/new-rules-aim-to-require-more-shrimp-boats-in-the-southeast-to-use-turtle-excluder-devices

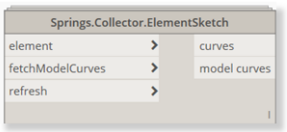
TED Concept Diagram

Making the Floor

We're using rooms as information containers to store information about the room that Dynamo can read and know what to place. There are two great nodes for getting a closed loop. You can also use the **select edges** node, but I find more often than not, you end up with an unconnected loop. Room boundaries are guaranteed to be enclosed or non-existent. Finding the sketch of a sketch based family, you can select any sketch based family and find the curves used to draw it. This allows you to use ceilings or filled regions to make a floor.



OOTB Node: Finds Room Boundary



Custom Node (Springs Package): Finds Sketch of Sketch created families
EX: Ceilings, Filled Regions, Roofs

Because we're indicating floors with a color fill legend early, more often than not, the floor finish is specified in our room properties. Also the level is stored with the room. Using the **get parameter value by name** node, I can find the level and floor finish I want to apply in the room and with the **room finish boundary** node, I can find a closed loop of curves. By selecting the room, I can make my floor, by selecting all rooms, I can make most of my floors.

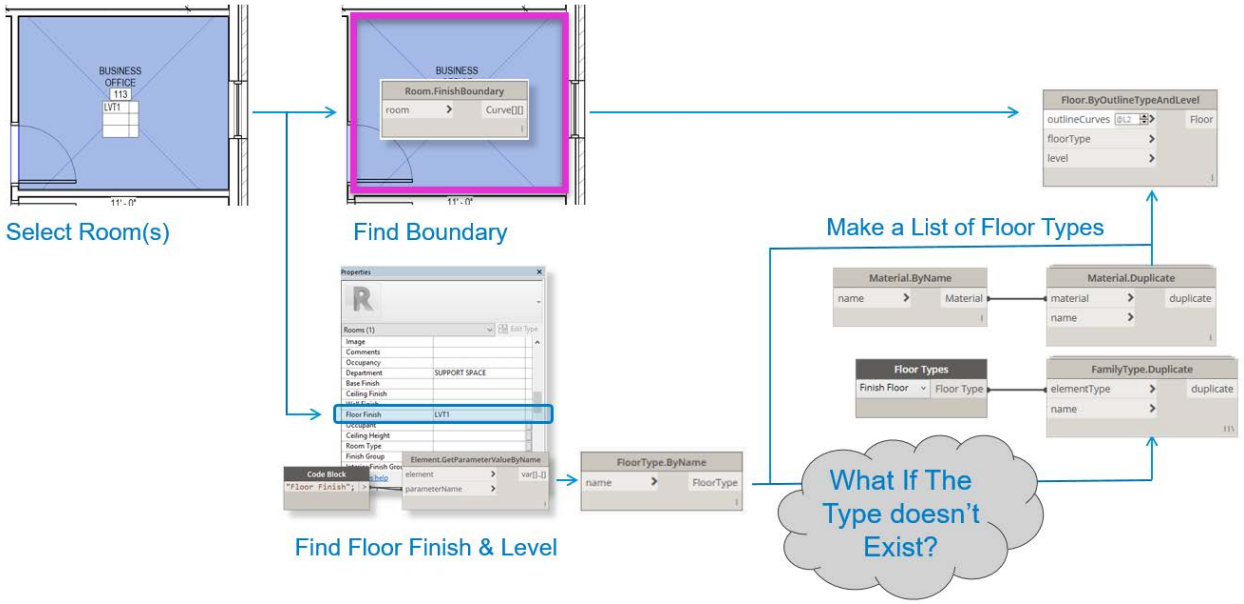
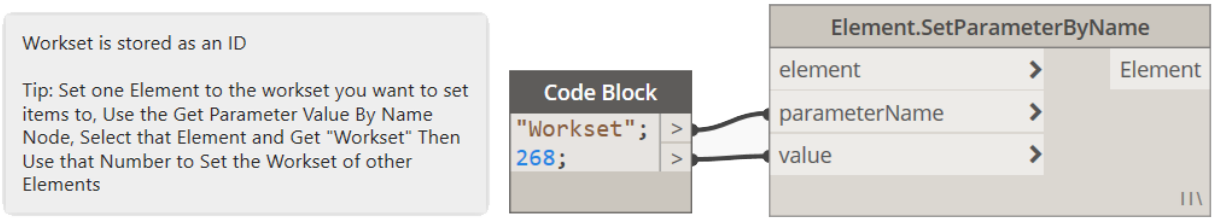


Diagram for making floor from room

Taking it further

A Floors geometry extends from the level down. When you're modeling carpet, it sits on top of the level. The floor needs to be offset. The beauty of using Dynamo is that that you can read the thickness of the floor material and set the offset of the floor by that thickness. You can also set the Floor's Workset for consistent documenting. Or store the Room Element ID to compare it later for updating.

Nodes to Get Thickness of Floor and Set Offset



Set Workset

Model Properties	
Dynamo Status	Created By Dynamo
LinkElementID	442505
Dynamo Notes	
Dynamo Ignore	☑
Data	

Use for Filtering to only update floors made by Dynamo

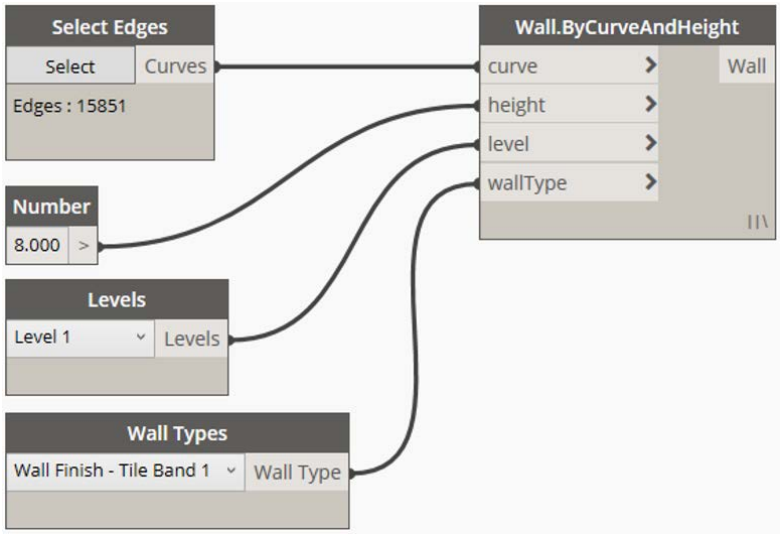
Use to only update floors to match

Use to ignore floors/rooms after they are set or too complicated to automate

Stored Parameters

Placing Wall Finishes

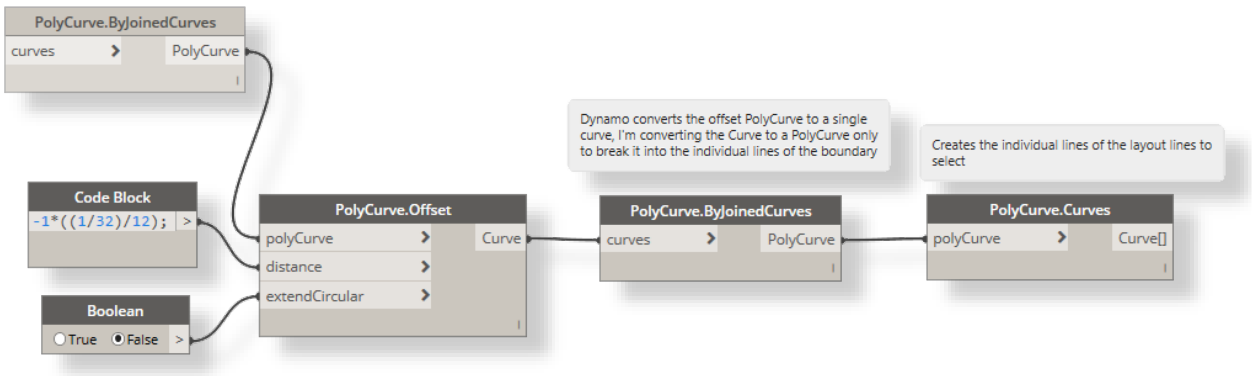
BSA LifeStructures interior design team have looked at a number of different methods for placing wall finishes. We have the luxury of being in the same firm as our Architects so we can if we choose use the split wall tool. We want to develop a workflow that is consistent. Using a thin wall and joining it to the adjacent wall allows us to place a finish in either the Architectural model or a linked model. With Dynamo, we can read that finish and associate it to the room's parameters later. I have typically used this method for materials of dimension like tile but had not for applied materials like paint.



.dyn to place wall

Placing a wall from a room

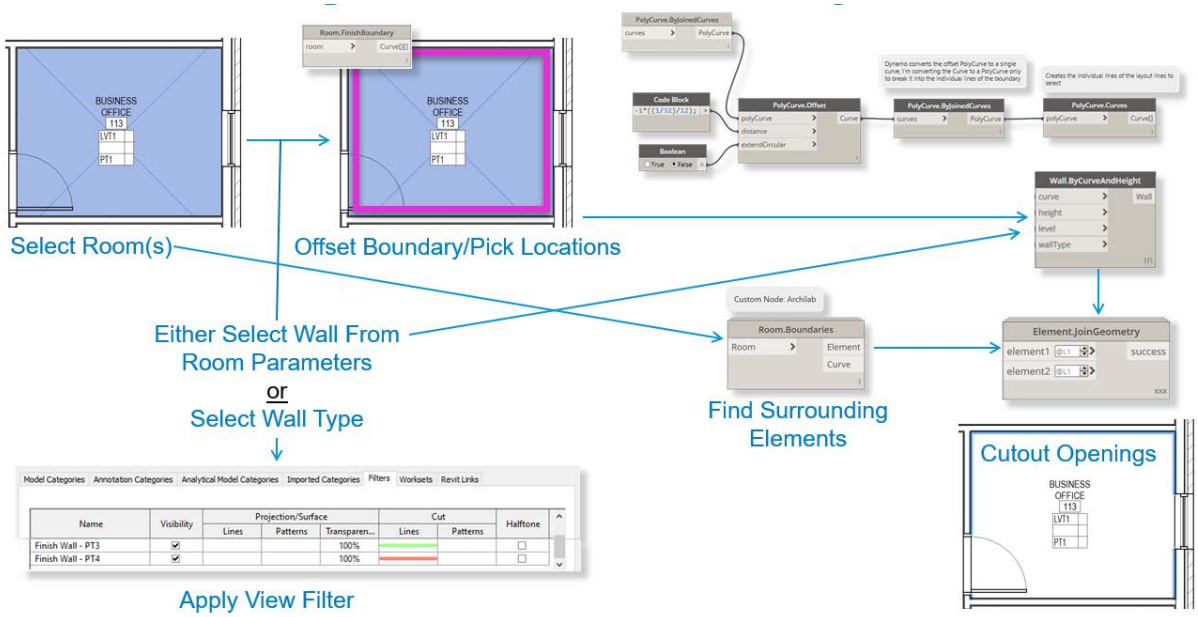
The line based location that is used when Dynamo places a wall is the centerline of the wall. If I use the boundary of the room to place the wall, the wall and the boundary overlap. To get around this, I can tell Dynamo to offset the room boundary by half the thickness of the wall. To do that, the screen grab below shows the nodes I used to do this. Basically, convert your room boundary to a polycuve, offset the line then explode it to get the list of lines to place walls.



Snippet of .dyn to offset curve

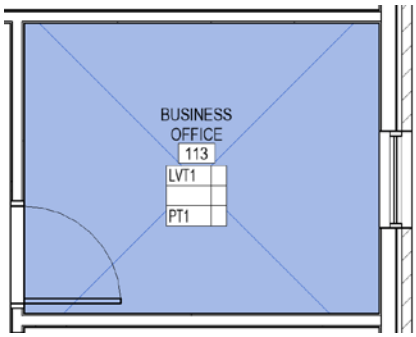
Taking it Further

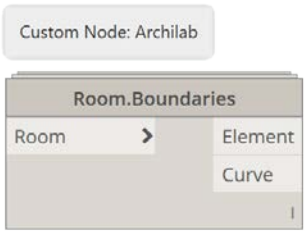
Where using Dynamo can be a benefit is like with the floors reading the room parameters to pick the finish. We can make the wall and material type if it is not already created. This is similar to making a floor so I'm not going to dig into it. There are two new things we can do here. We can join our thin wall to the adjacent walls to cut out openings and also create the view filters and apply them to our view.

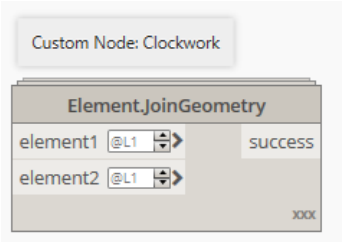


Auto Joins

Room.Boundaries is a custom node from the Archilab package that will extract the surrounding elements of a room. This does not work through a linked model. To do this you need to look at all elements of a category through the linked model. There is also an **Element.JoinGeometry** node from the Clockwork Package. Pass your room selection through the **Room.Boundaries node** Take the list of Elements and Join them with the **Element.JoinGeometry** Node. Element 1 and Element two equate to the selection process. When you use the join command in Revit the selection order matters.







A Word Of Caution: What is shown will cause errors. You are joining elements that are not adjacent. You'll want to use the geometry intersect node to find which of the room boundary elements are actually touching each wall segement and join those.

Creating Filters

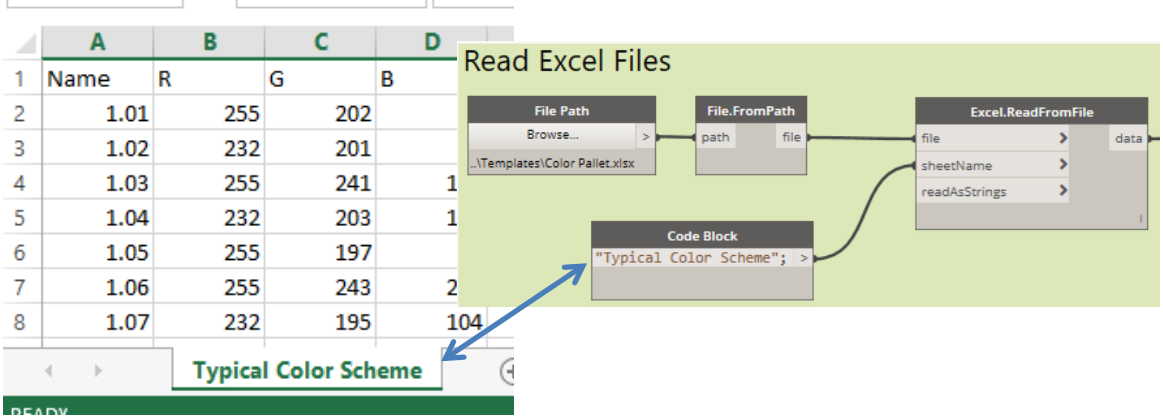
I was inspired on this one by one of the things that makes Dynamo great. The people that contribute their own code and workflows has mad Dynamo to be as powerful as it is. There is a great tutorial from Konrad Sobon on Thinkparametric.com. I suggest checking it out. Basically, Konrad made a bunch of nodes in the Archilab package for making and applying view filters. These have since been removed from his package and added to Dynamo's core functionality. I didn't create it so I won't go into it much but here's a link to the post that gave me the idea to add filters to my script and a tutorial.

Read the great blog post:
http://archi-lab.net/think-parametric-accessing-the-power-of-visibility-overrides/

Try the Tutorial:
https://thinkparametric.com/courses/create-custom-view-filters-using-dynamo

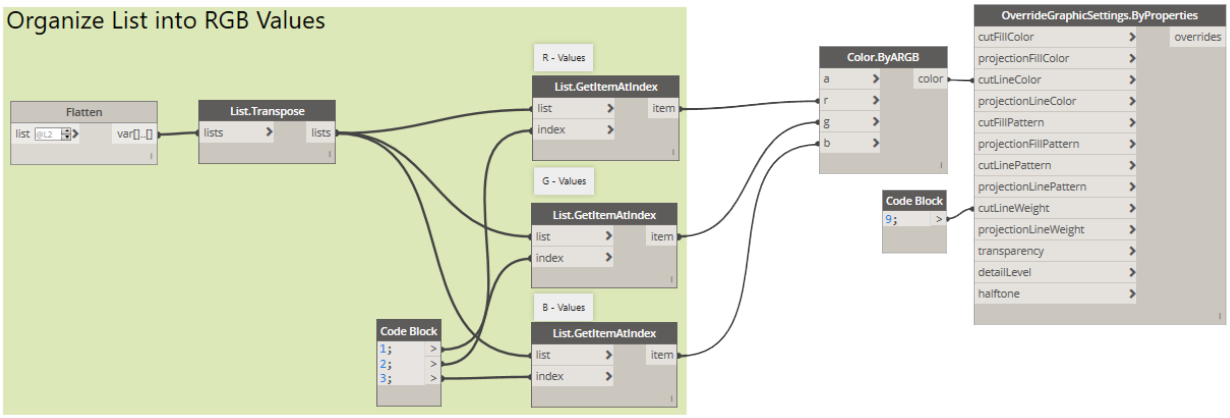
Coloring Walls

Often during early design, we may not know what color of paint we're using or we're between options but we do know where those colors are applied. We have typical BSA color value for PT1, PT2, PT3… when we know the actual color value, we'll apply that to the filter to display our wall value. Because of this, I have two methods for creating a filter legend. The first method, is to read a list of RGB values from Excel and apply those based on the material name. This can help create a consistency in look across our office. PT4 is always orange or pink or whatever we had decided. The second method is to read the material color value and use that as the filter color.



Reading Excel with Dynamo

Once you've read the excel file, you'll need to sort the information. Dynamo reads each row, as a list you want to break this up into columns to be workable. List.Transpose will flip the list order into columns. Then Get item at index can be used to extract each column from the overall list. As Seen in the screen grab below, this can be used to make an RGB value to apply to your visibility graphic overrides.

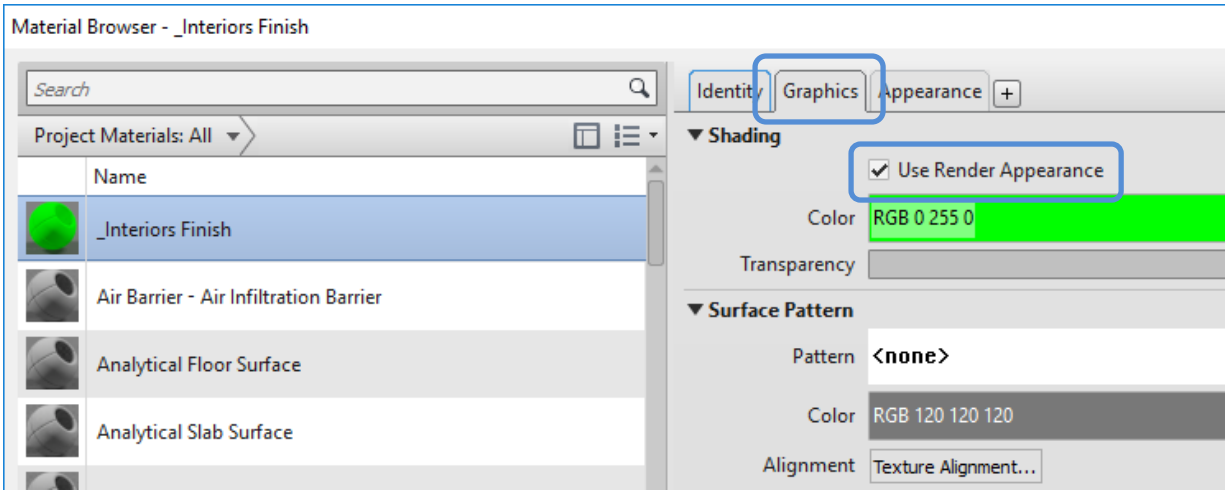


Create Color Overrides with Dynamo

When your materials have been applied with the actual color, you can find the material of the wall and use the Material.Color Node to find the color of the material.

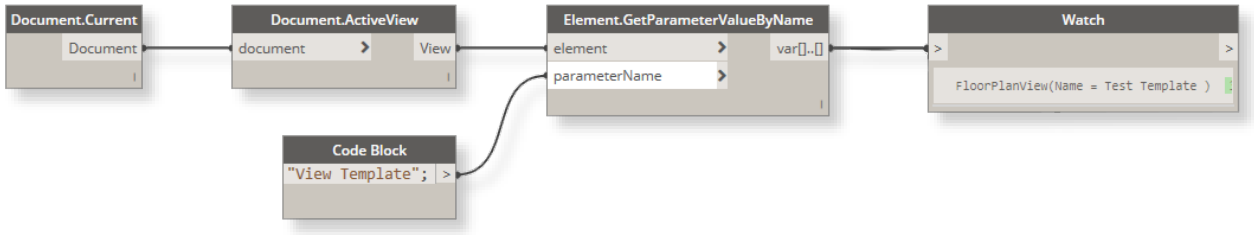
Get material color with Dynamo

This grabs the color of the material shown on the appearance tab and not the color of the render appearance. To keep these in sync, Check box color from Material on the graphics tab of the material properties. This will apply the overall color of your material. The nice part is now, if you have a shaded view or realistic view, the material will look like what you have intended.



View Templates

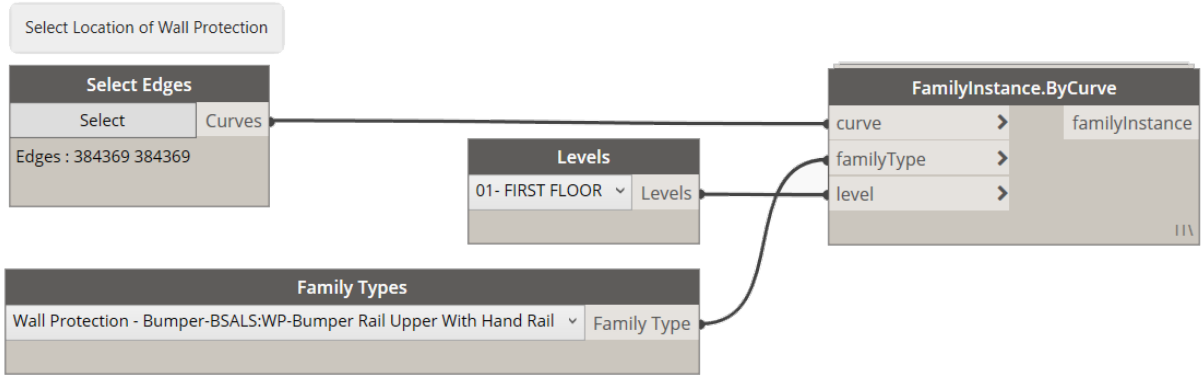
A view template is stored in a Revit model Database as a view. By using the **get parameter value by name** node, you can find the view template applied to your view and assign your view filter to the view template view.



Get view template of current view

Placing Wall Protection

Wall protection is tricky. It runs along a wall, has rules for where it starts and stops, how it starts and stops and has varying mounting heights. Sometimes the Architect is responsible for it, sometimes the interior designer is. It's tedious to place and even more so to maintain. This one is a work in progress for me but I'm showing it because I think it shows the potential of what Dynamo can do for your typical work process. The initial placement is simple. It's the variables that are challenging but if you spend the time looking at your rules, you can auto place wall protection. My script has manual entry but I hope to keep working on it to get it to a mostly automated process. Below is a screen grab of a simple way to place wall protection. This isn't better than using the pick line tool. I used this to see if I could do it and as a starting point to build from.



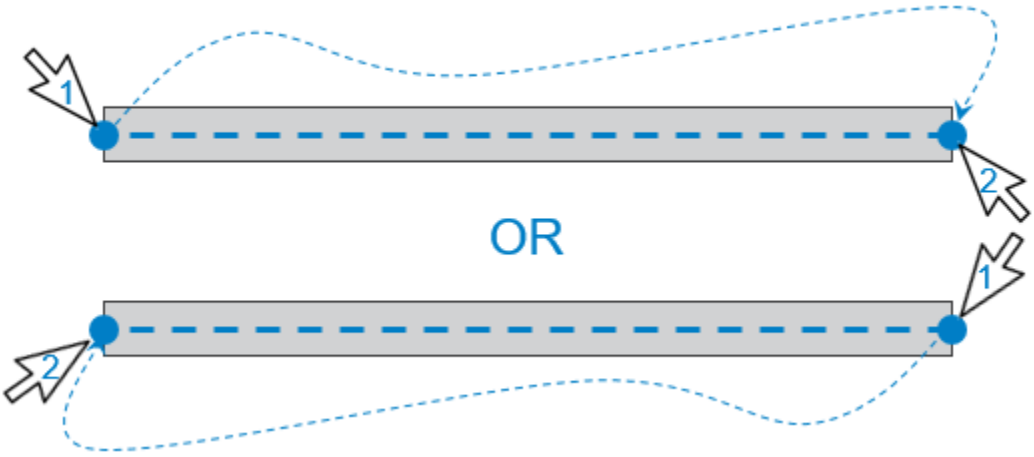
Place Wall Protection Family with Dynamo

Placing Wall Protection from Walls

In the image above, I am selecting the individual locations of wall protection. I prefer to protect the entire wall and have the wall protection adjust as new openings are added, removed, or adjusted. What I have found is there are a lot of variables to using this sort of method. The draw order of the wall determines the line location which changes which side of the wall your family is placed. The join conditions determine the length of the draw line. A butted wall is either seen as the full length of what I view as the wall surface to protect or short by the width of the wall on the outside edge. If a wall is joined by a miter, it is always short by half the thickness of the wall. I need to evaluate these things to ensure the wall protection will be placed consistently.

Diagram of Wall Lines

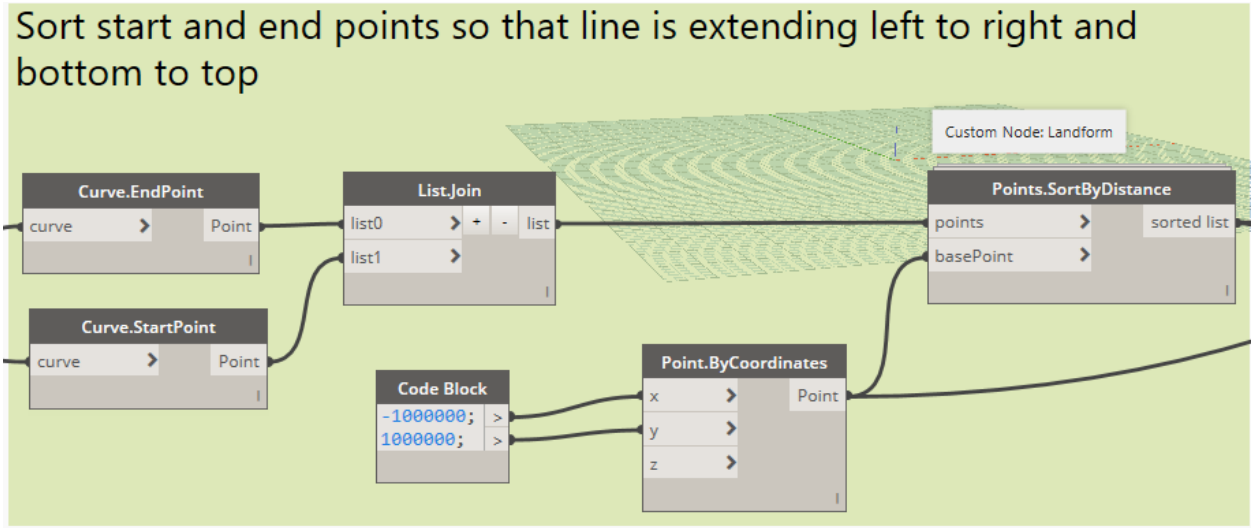
- Sort End and Start point by proximity to point



Wall Draw Order Diagram

Evaluate the variables

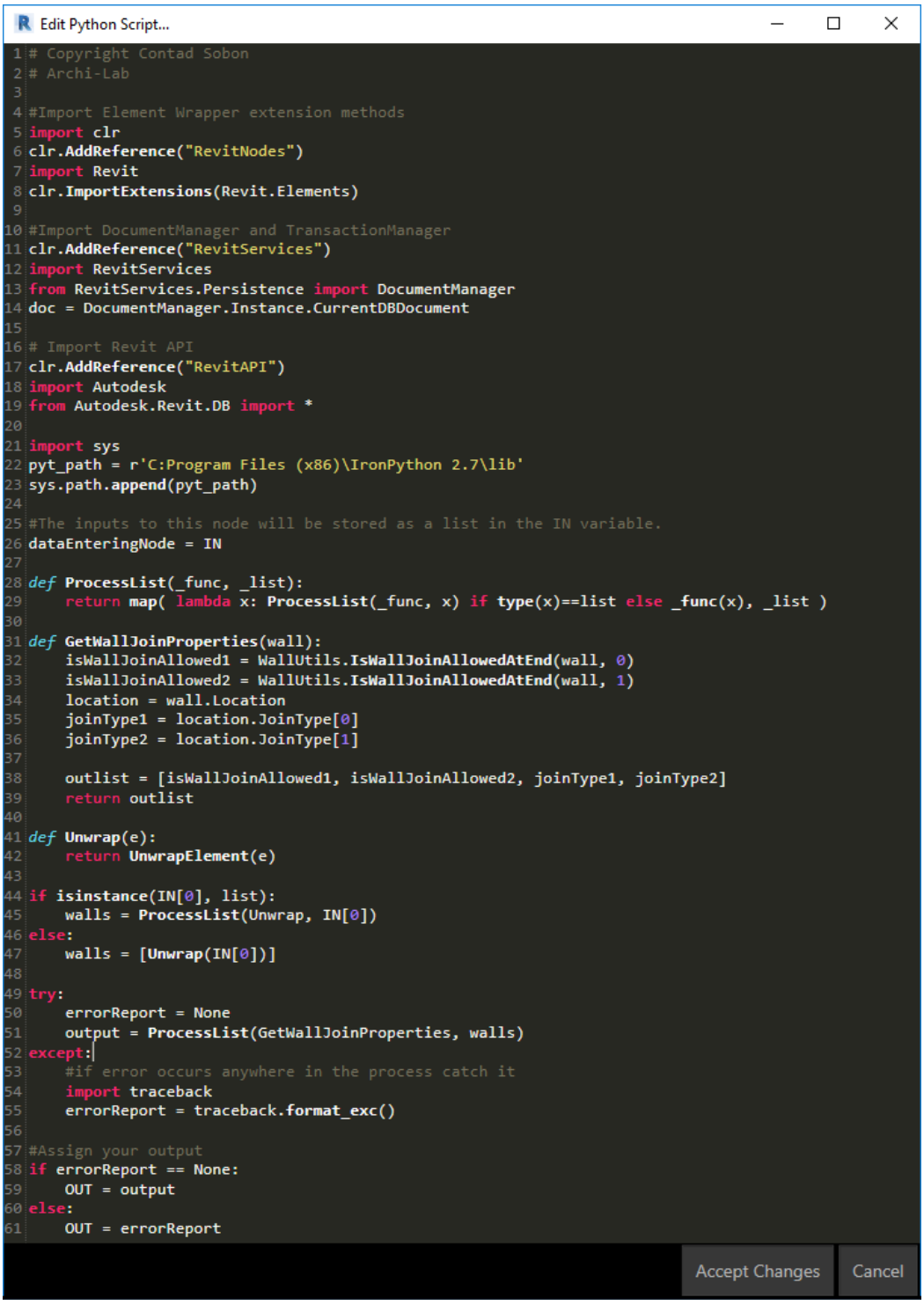
To evaluate the click order, I look at the start point and end point of the wall line and compare them to point far away from the project and sort by closest to. Then Redraw the line dynamo will read to place the wall protection family.

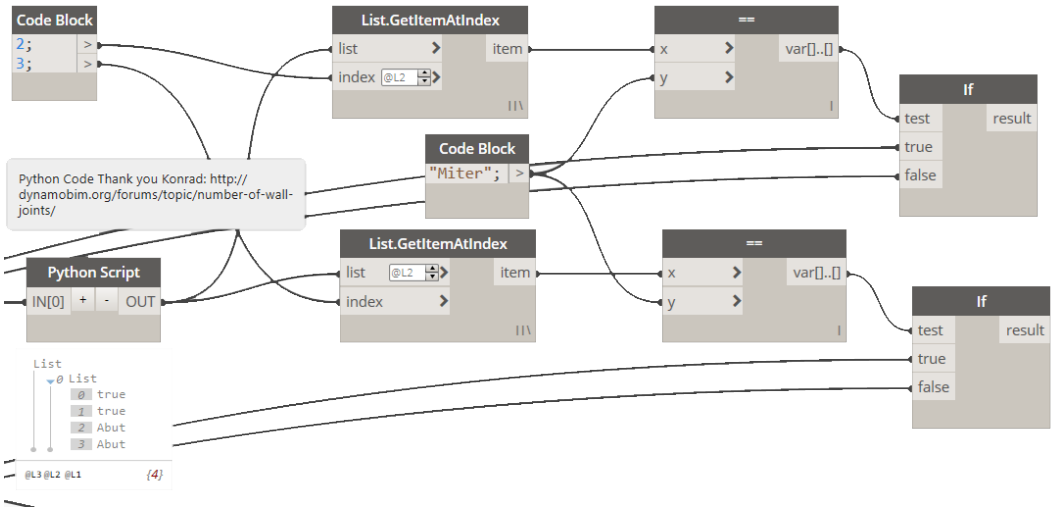


Reorder Line Start and End Points

Evaluate the variables

Reordering the lines can help for consistent placement location. For the end conditions, I've pasted a link below a post on the Dynamo Forum. Using Python, we can query walls to see what they're end conditions are.
Link to Python Code post on the DynamoBIM forum. Thank you Konrad:
http://dynamobim.org/forums/topic/number-of-wall-joints/

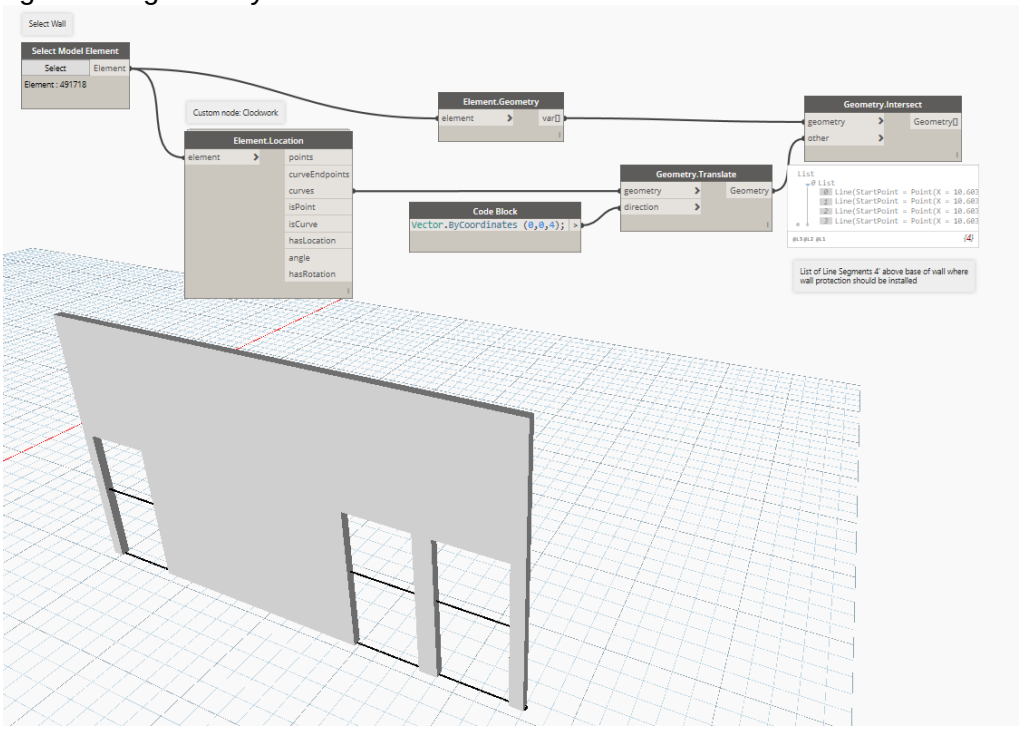




Look at each end condition of wall

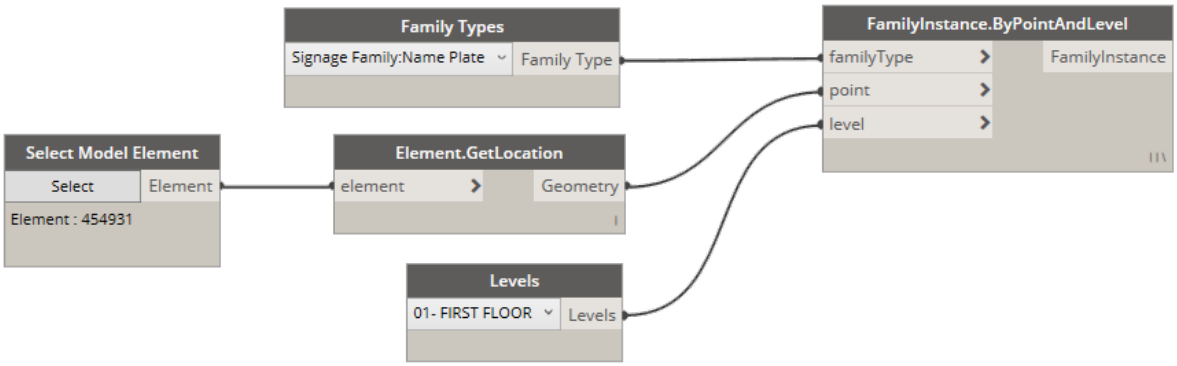
What to do the new line

Knowing the end condition and how the wall is built allows you to know the distance to extend or shorten a wall. With this information, you can pick a wall and create a line to place your wall protection family that extends the entire surface of the wall. Wall protection isn't placed in a single line along a wall, it is placed in segments along a wall. We can see where the line of the wall intersects with wall geometry though by offsetting the line at the mounting height of wall protection. By using the **Geometry.Intersect** node, you can find the line segments of wall protection by clashing the line of the wall against its geometry..



Placing Signage

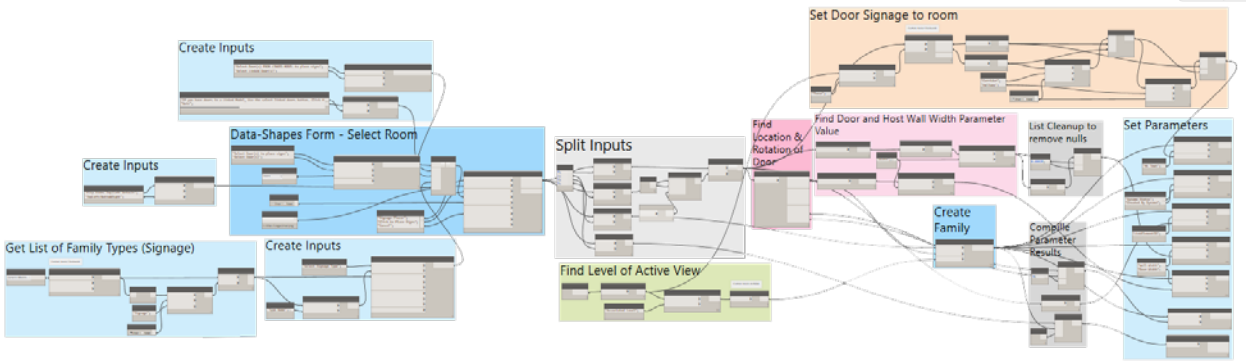
The overwhelming majority of signs are placed around doors and usually display the information of what is beyond the door. The door signage mounting locations are specified in details as distance from the floor, offset from the frame, which side of the door, or top of the door the sign is installed. We can select doors, find their location, place a family in that location, and store the door information for synchronizing.



.dyn to place a family at the location of a selected family

What are the variables?

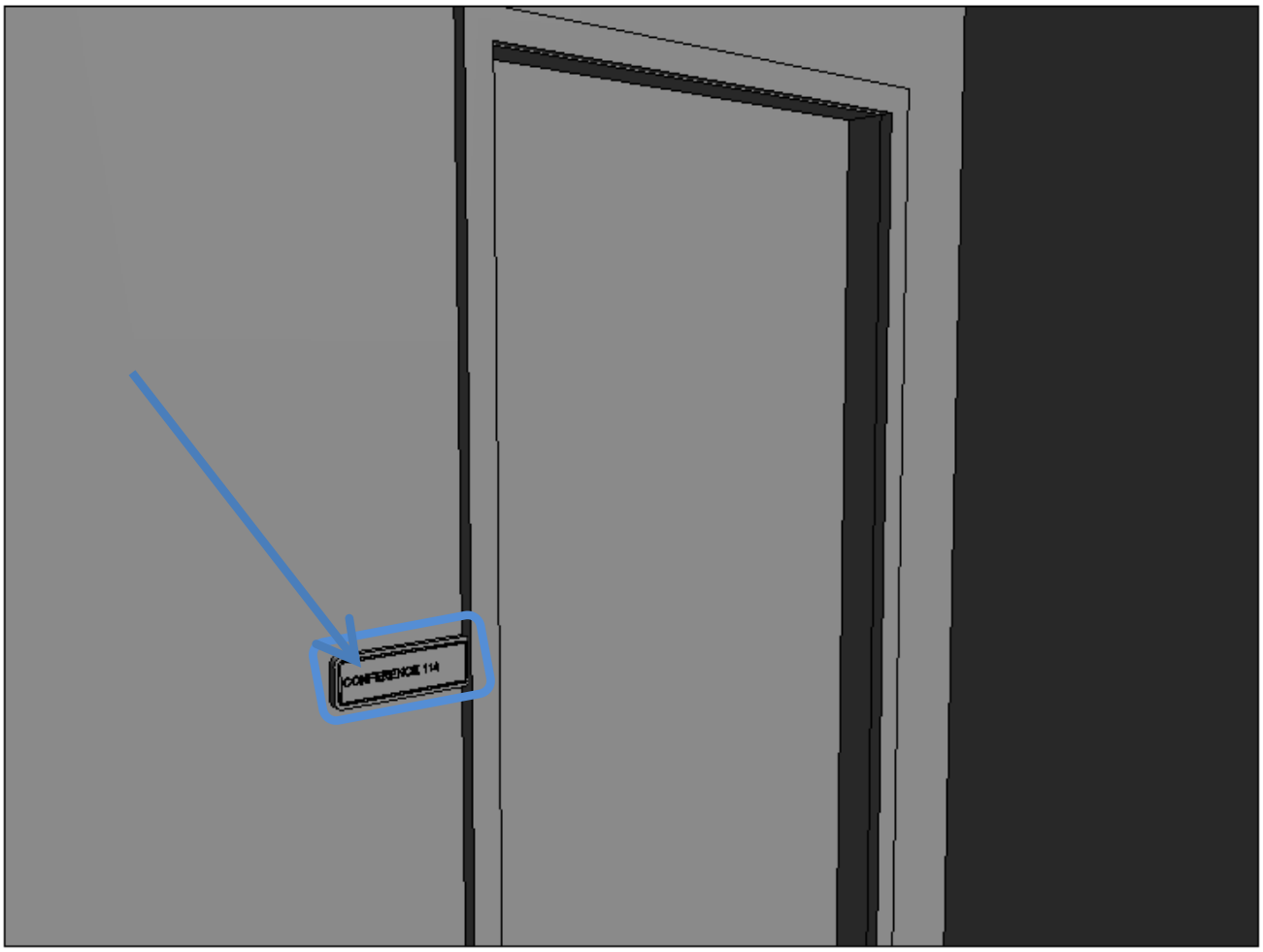
I have made non-hosted signage families that have parameters for a doors height and width, the wall width, side of door, and sign text. Using Dynamo, I can select a door, get its location and place a family at that location and set the parameters mentioned above.



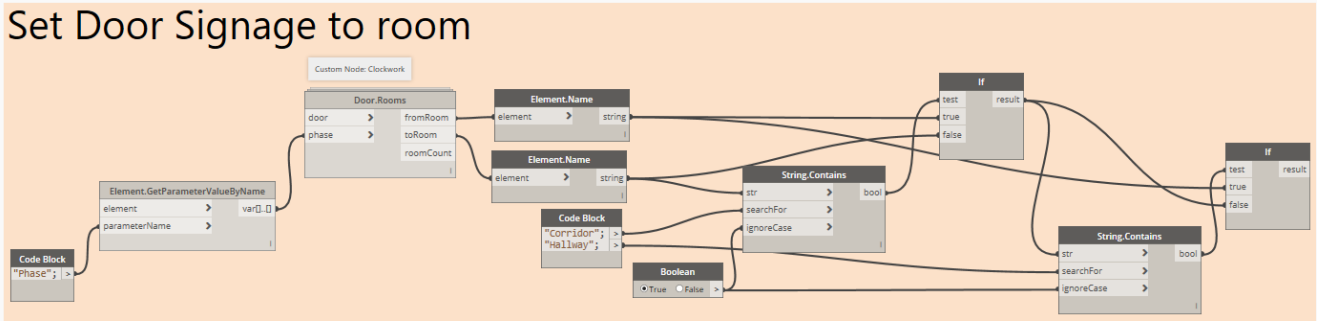
dyn to place a family at the location of a selected families

What is special?

The script above was one of the easier ones that I've written. What I am proud of is that it is setup to look at the door to Room and from Room parameters assuming they are placed incorrectly and ignore the value that has hallway or corridor and assign that value to the signage text.



.dyn to place a family at the location of a selected family

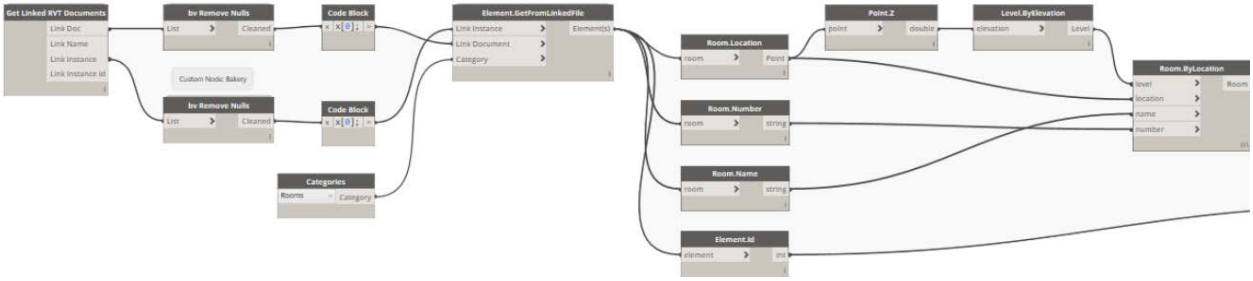


.dyn to place a family at the location of a selected family

Placing Rooms

A BIG challenge for interior designers is working with rooms across a linked model. I've traditionally seen three methods of how to coordinate the rooms. The first is to not use rooms. Either use spaces or don't provide a room schedule. Another method is to place a family in the room, associate finishes to the family, and schedule the family instead of rooms. This can work but it requires a lot of maintenance and just like signage associated to a door, I hate placing something and coordinating it when it should just be associated to that object. I'd rather see the door sign carried with the door family and I'd rather see the material finishes carried with or read

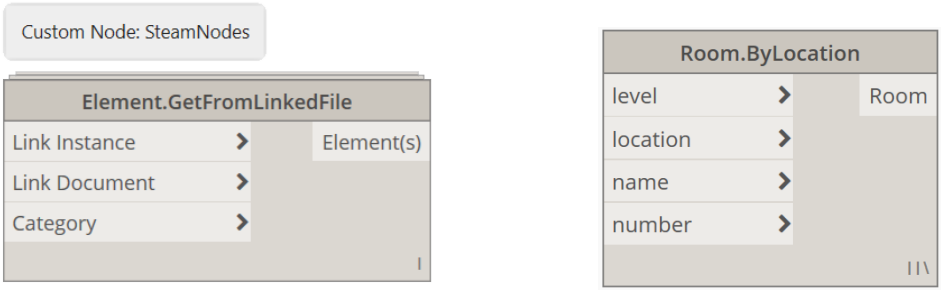
from a room. Another method is to use rooms but coordinate them. There is not a copy monitor feature for rooms but there are some tools that can help. For this section, I'm going to discuss how to place rooms from a linked model. This is the first step of a workflow using Dynamo to coordinate the rooms.



.dyn to place a family at the location of a selected family

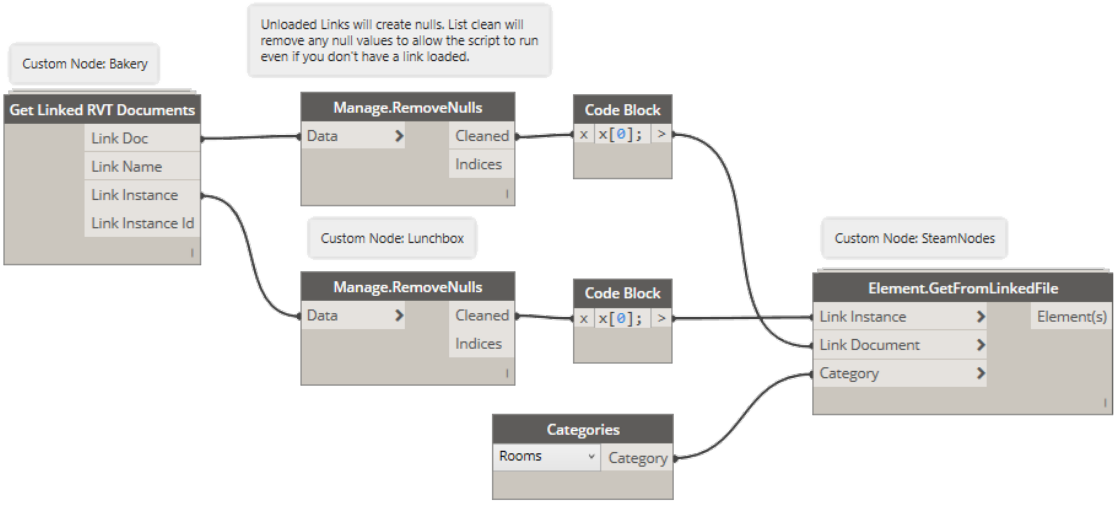
Place a room

Dynamo has a node to place rooms that is pretty easy to use. You need a level, Point, Name & Number. The already placed rooms in the linked model and there is a custom node in the SteamNodes package that will get elements from a linked model. This node can be used to find all the rooms from the linked model.

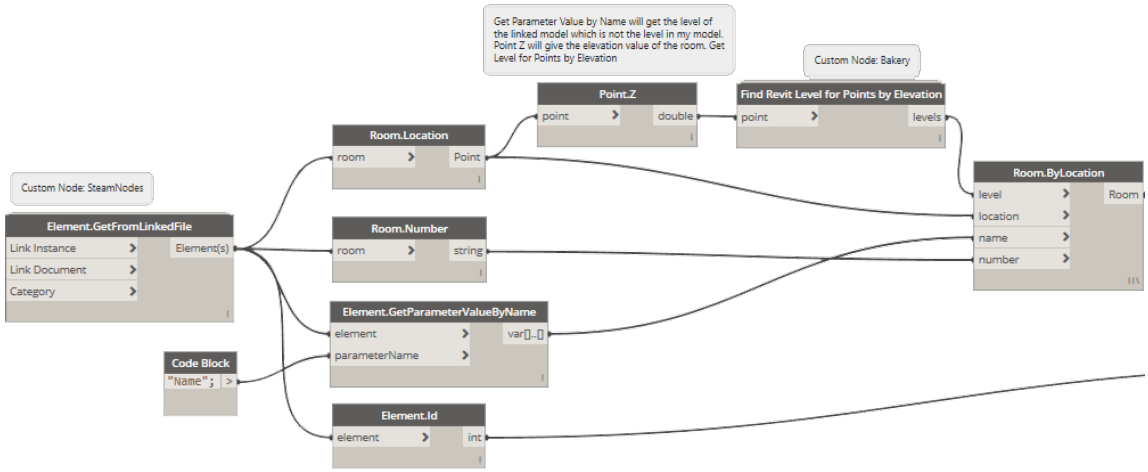


Nodes to Find Rooms in Linked Model & Place Rooms

There are other methods of doing this but this is how I get all the rooms from a linked model. There are custom nodes from the Bakery and SteamNodes packages. Something that frequently causes issues for me is when a link is unloaded. Unloaded links produce null values from the get links node. If I use the **List.Clean** node, I can avoid having my script error out on me.

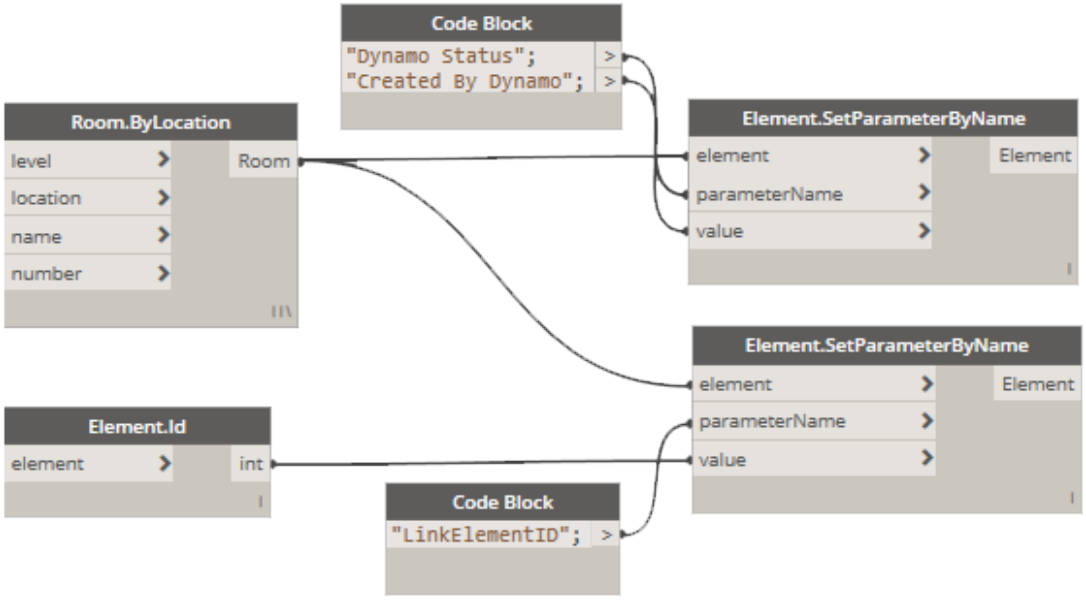


Once we have the rooms, we need to get the information from them. Dynamo has several nodes dedicated to extracting information from rooms making this an easy task



Portion of .dyn to get room parameters and place room

The final part of this workspace is not necessary to placing rooms but it is necessary if you want to sync them later. We need to store some information about the room in the link model that we can compare later. I have added a few project parameters to my Revit file to store this information. I'll cover what we can do with this later in this handout. I'm tracking the Element ID of the Room in the Linked Model and noting that the room was created by Dynamo.



Portion of .dyn to set room parameters

Synchronizing

When we show an image to a client or have them look at something in Virtual Reality we do this to get them to understand what the space will look like and how it will feel. Our hopes are that they like it and we get to document that for someone to build. It's easy to make yourself look bad if you show something the construction documents show something else. A great way to avoid this is to pick presentation methods that allow for synchronized information. It's why the concept of BIM is so popular. What you show in plan, section, and elevation is true thanks to working from a three dimensional database.

Many Interior Design teams struggle with this because there is the Revit model where you may or may not model everything, your presentation graphics which may or may not be produced from your Revit model, and the documentation you are creating. This next section is going to cover methods of updating finishes you've placed, or reading the finishes you've placed to populate parameters of objects that are not synchronized with each other.

Synchronizing Rooms

Earlier we looked at how to copy rooms from a linked model. It's rare for a project's lifecycle to avoid redesign. Redesign usually causes changes to rooms. Because we stored the Element ID of the room we used to make our initial room, we can compare that room's location to our current room's location.

The tools you need for this

The script for synchronizing rooms across links can look intimidating and complex but when you break it down, it's only performing a few routines. It's the order that matters. These next few sections will cover the tasks needed to compare rooms to a linked model and set the room parameters and locations to match those of their corresponding rooms in the linked model.

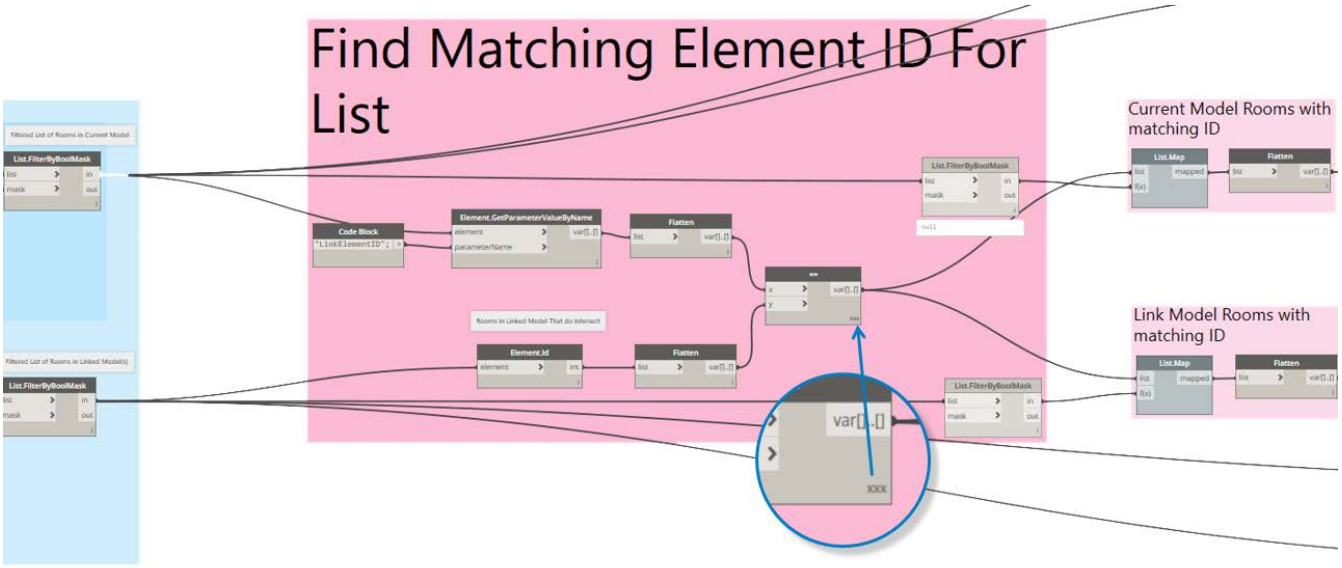
What you'll need to know:

- How to Compare a list of values to a list of values and pair the matches.
- How to Compare Locations of items in lists of lists and pair the matches
- How to perform the actions of those results.

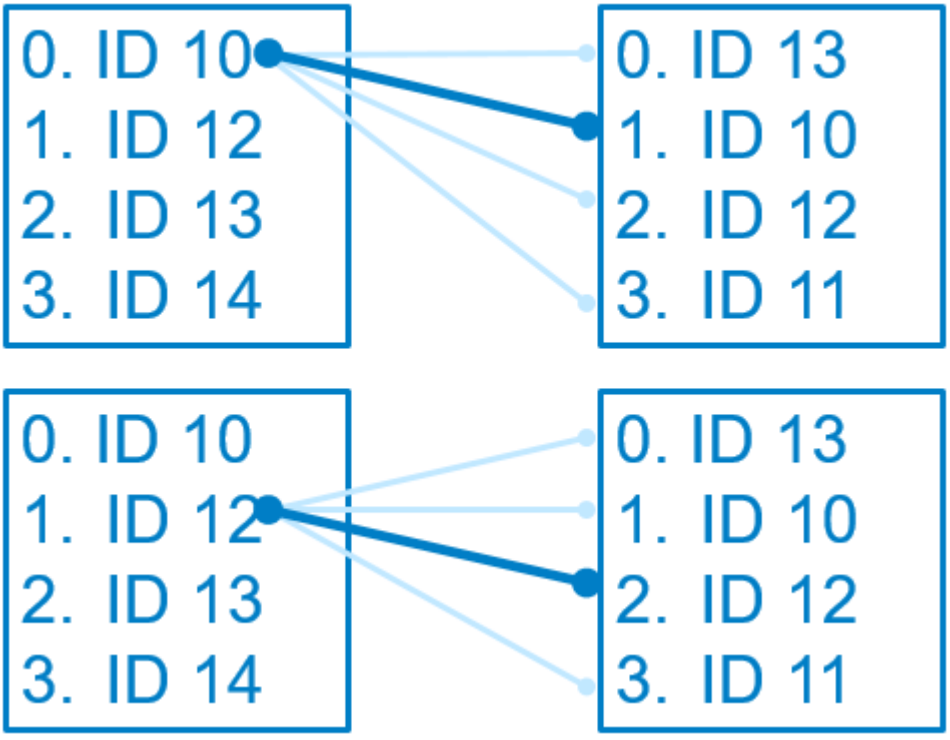
How to compare a list of values to a list of values and pair the matches

When you compare a list to a list. It's not as easy as just setting lacing to cross product. You are comparing one item from a list with the cross product of the other list for each item in your first list. This can be a bit confusing. The diagram below is the best I've been able to do to explain this.

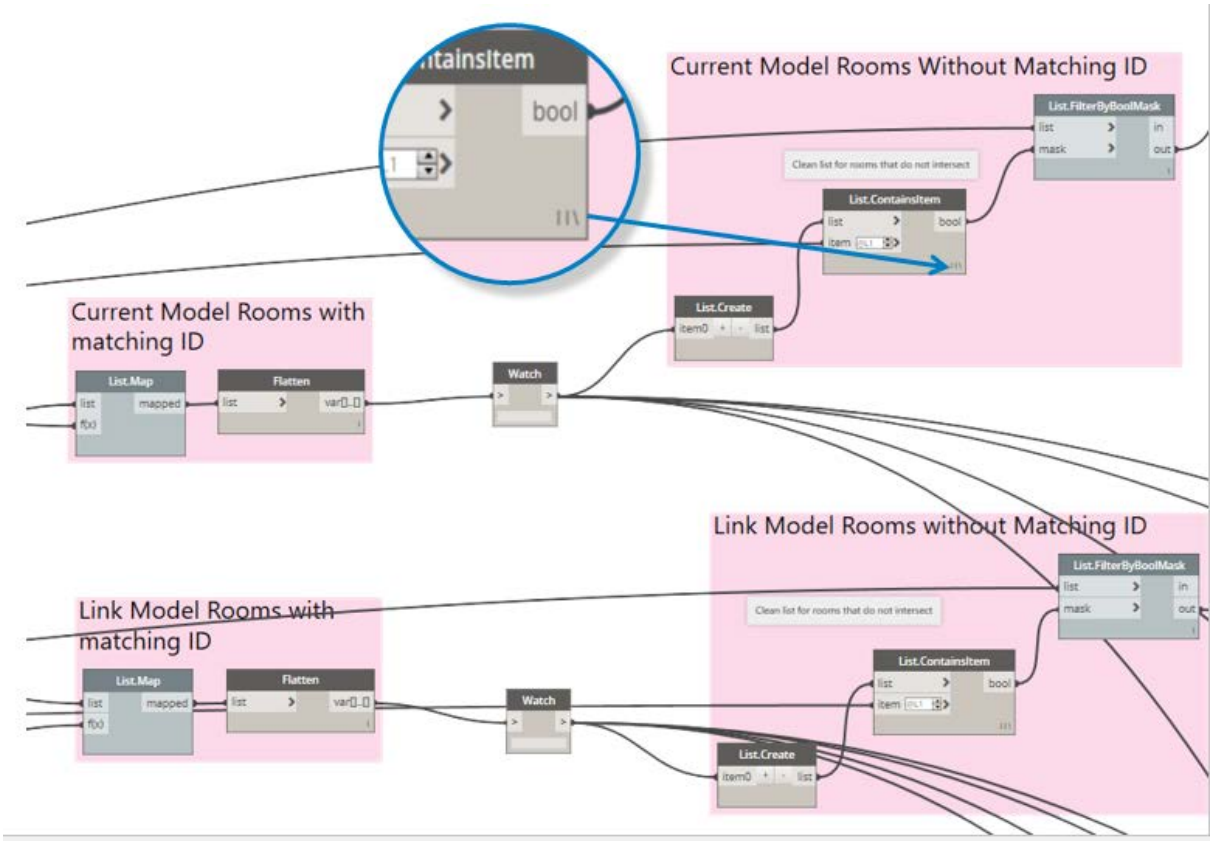
In Dynamo I am passing each list through the **==** node with the node set to cross reference and then using the **list.map** node to apply this same function to each item of the initial list. This gives me the true results for rooms that are matching. The idea does not apply for finding the false results. If I do this, I get a list of each result that did not match the item I was looking for. It basically returns a list of repeated items. I can however, Compare the list of matching results to the initial list I passed through the == node to see if each item is contained in the list. I can take the results that are false as the list of rooms that did not have a match.



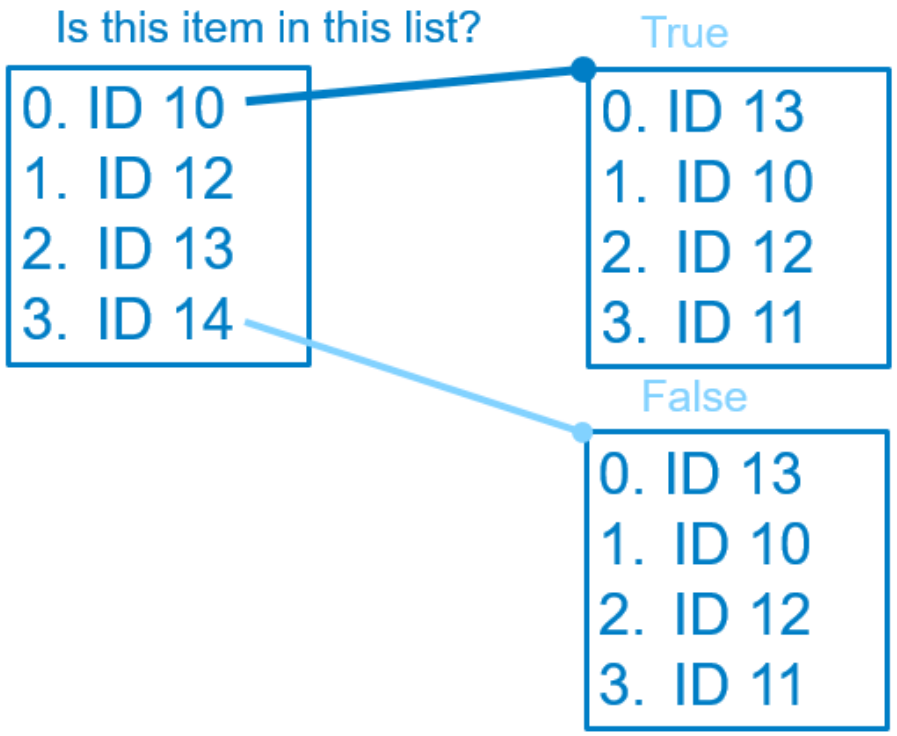
Portion of .dyn to compare Two List of Lists and pair matching results



Compare Two List of Lists and pair matching results diagram



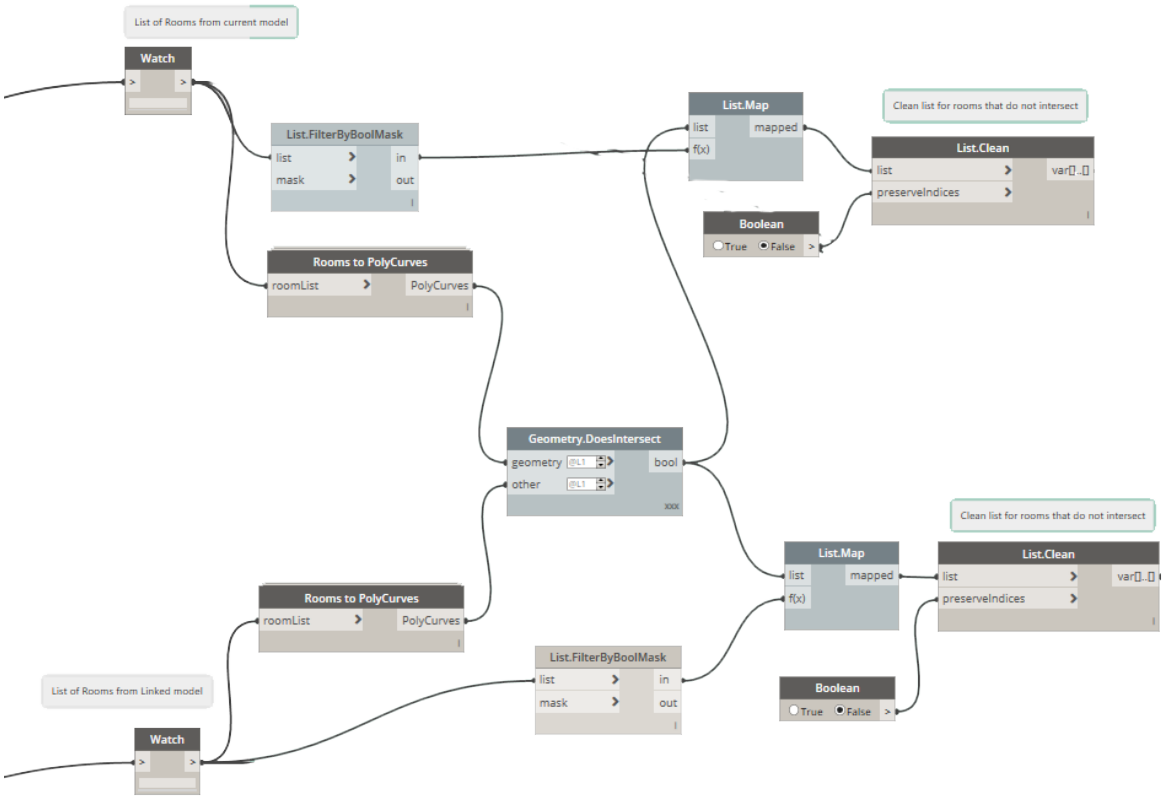
Portion of .dyn to compare list of matching items with full list of items to find missing items



Compare list of matching items with full list of items to find missing items diagram

Comparing a list of locations

This is basically comparing a list of lists to a list of lists but using the **Geometry.DoesIntersect** node instead of the **==** node. I did want to show this and point out a few other things. When I first wrote this script, I compared used the **element.geometry** node to convert the volume of the room into Dynamo geometry and compared those. What I found is that it often produced results of multiple intersections for cases where a room's edge met with another room. I then looked at the point of element location and thought I had success but eventually found that this location can vary when a new room is placed of the size of the rooms change. I have settled on comparing the polycurve of each room. This is most likely to be consistent. There are cases with room separation lines in the current model where this would not work but I can fix that by removing those. The other item worth noting is that when you find the true results, there will be lists without any items because there is not a true result. Use the **List.Clean** node to remove the empty lists. Empty lists and null values frequently cause scripts not to complete.



Portion of .dyn to compare Two List of Lists room's locations and pair matching results

How to perform actions on those results

It's important to identify what can change if you are scripting your workflow. This helps you avoid errors that prevent a script from completing. It also allows you to identify the things that you need to make decisions regarding. With rooms there a few scenarios that I've been able to think of.

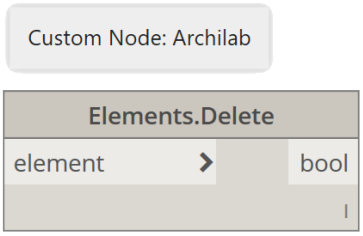
- A room can get removed from the project
- A room can get deleted from the model and a new room can be placed
- A new room can be added to the project
- A room name or number can change

What do you want to do?

Once you've determined what can change you know what you need to look for. You also need to determine what you want to do in these cases. Once you've decided what to do you can use dynamo to look for those cases and perform the action you determined.

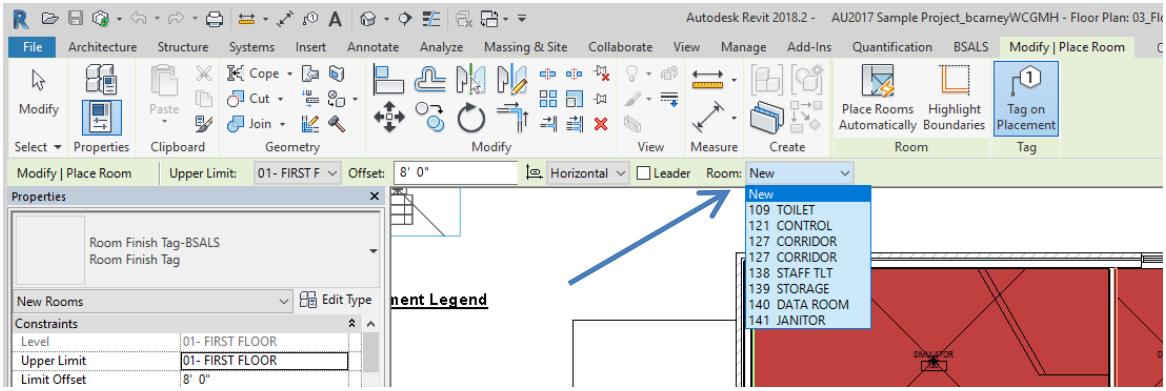
A room is removed from a project

This is a pretty easy decision. If a room is removed from a project entirely, then there aren't many reasons to keep it around. You may want to archive your project file just in case but it's safe to say you can delete the room. How do we know if a room is deleted from the project? Hopefully the lead designer tells everyone? But if they don't we can look at all the rooms with an Element ID matching your stored Element ID. If there is not a match, we can compare all the rooms to see if another room has a matching location. If there are no true results for either, we can delete the room. The Archilab package has an **Element.Delete** Node.



A room is deleted from the model

When a room is deleted from a Revit model, the room still exists in the model and can be placed again if it was deleted in error. There are probably a lot of people that would avoid automation for synchronizing rooms if a deleted room immediately went away after it was synchronized.



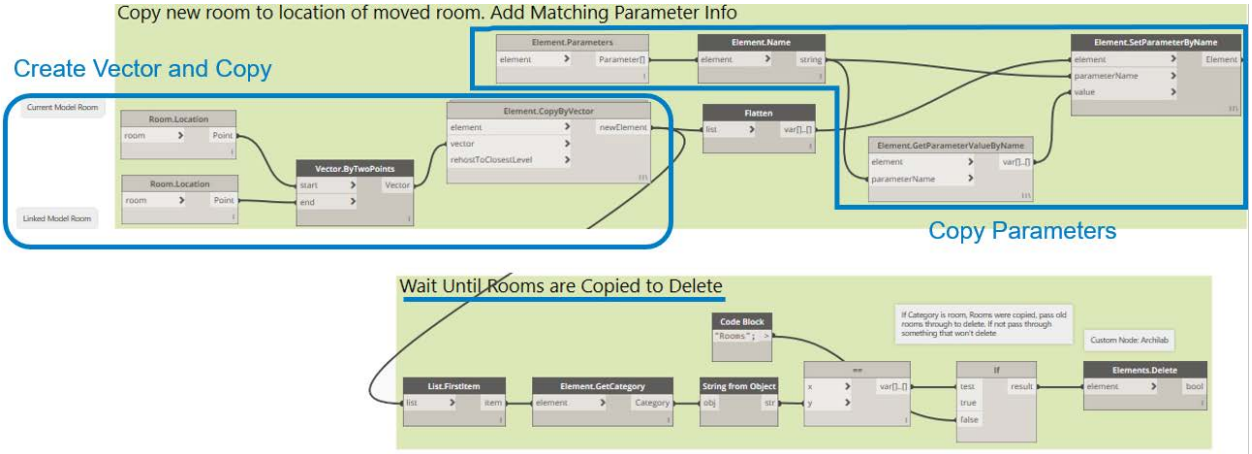
Deleted Rooms are available to re-place from the options bar

A new room is added to the project

When a new room is added to the project, we need to add a new room to our Revit model to match it. Using what you know from the place rooms from linked model workspace, you can get the list of new rooms, find their locations and place them.

A new room is added to the project

If a room is moved, this is something that is typically done on purpose. A room's location has moved and the Revit modeler dragged the current room to it's now location. Dynamo can move families but as far as I'm aware, it cannot move a room. What I do is copy the room, set it's parameters to match and delete the old instance.



Portion of .dyn to copy room to its new location

Room Parameters Change

Room names and numbers change frequently in projects. A room named "Steve's Office" My change to just office. Numbers can change to accommodate wayfinding, renumber because a new room was added in the same vicinity, or the owner has a specific naming convention that you need to adhere to. If the list of rooms in my current Revit model match location of a room in the linked model, I need to check to make sure the room name, number, and any other parameter we care about matches. Just because the room name changed doesn't mean it's a different purpose. I update the names and numbers to match and store the last sync so we can compare what changed. If I add project parameters to rooms for Las Sync Name and last Sync Number, I can use conditional formatting and color fill legends to graphically show me what rooms to review.

<ROOM SCHEDULE>

A	B	C	D	E	F	G
Name	Number	Area	Department	Last Sync Name	Last Sync Number	Dynamo Status
ACCELERATOR	125	532 SF	EXAM			
SIMULATOR	120	351 SF	EXAM			
DOSCIMETRY	118	153 SF	SUPPORT SPACE			
CONFERENCE	114	186 SF	SUPPORT SPACE			
NURSE WORK AR	113	103 SF	SUPPORT SPACE			
EXAM #1	112	99 SF	EXAM			
EXAM #2	110	99 SF	EXAM		111	Updated Numbe
PHYSCIAN'S OFFI	108	156 SF	SUPPORT SPACE			
TOILET	109	Not Placed	BUILDING SUPPORT	Toilet		Updated Name
RECEPTION	105	169 SF	SUPPORT SPACE			
WAITING	101	233 SF	WAITING/LOUNGE			

Conditionally Formatted Room Schedule