



AUTODESK UNIVERSITY 2015

AS9926

Revit++: More with your Revit, more for your workflow!

Homer P.S. Anave
TAISEI CORPORATION

Learning Objectives

- Create your own intuitive yet powerful Revit templates for preliminary design
- Put details on your plan and section models
- Use more of your HTML files from interference checking with the provided tools
- Explore other available workflows such as visibility tools, viewing tools using Dynamo, and Revit API

Description

Gain knowledge and acquire ideas on how to improve your architecture and construction design workflows in your company or office, and enhance your design drawing standards and cross-platform collaborative approaches using Revit. Get ideas on creating intuitive yet powerful Revit templates for basic design. Enhance your floor plan and section detailing through modification of basic models (walls, floors and ceilings). Make use more of your HTML files generated from Interference checking. Finally, get hold of suggestions on developing and implementing in-house tools using Revit API, citing some of our company's examples. We will also offer you some related tools for your use and evaluation. Likewise, we will orient you on how Japanese general construction companies such as Taisei Corporation introduce and make full use of BIM to comply and integrate their existing design and drawing workflows, and innovate new workflows using BIM.

Your AU Expert

Homer Anave has been working with Taisei Corporation for more than ten years mainly as a developer, mobilizing tools in AutoCAD and Revit for in-house and outsourcing use. He also conducts lessons on Revit and new tools, and advises solutions regarding their uses. Outside developing, he also does research on computational design and participates in projects requiring it. Before joining Taisei, he obtained a Bachelor's degree in Architecture at the Polytechnic University of the Philippines, and worked as an operator, developer and trainer at Taisei's subsidiary company in the Philippines. His other interests are in mathematics, videography, cycling, and music composition.

Session Overview

When working on the design phase, typical workflows in Revit tend to just focus on building models, creating drawing outputs from models, and engaging in the usual Building Information Modeling (BIM) workflows such as transfer of data between Revit and other software such as Navisworks and AutoCAD. As Revit has much more to offer, we can delve deeper and explore its functionalities. Customizing templates and families, and on-the-fly tool development and utilization are some of those that can make it happen.

The range of this workflow covers those made for small office-home office (SoHo) and even large design companies with sectional design departments.

Revit workflows in Customization Hierarchy

The Revit workflows to be discussed here each have premises in the different customization methods in Revit, which is hierarchically presented in Figure 0.1.

Creating the intuitive template, by which will call here Designer's Template, belongs to the lowest level in the hierarchy of customization in Revit. This is specially accompanied by intuitive families that are collectively the heart of the template.

Workflows such as visualization and viewing tools, grid and level creation and others, can be done with visual programming like Dynamo, which comes at second in the hierarchy.

Plan detailing and Interference Checking will be discussed here using add-ins developed with Revit API, which is on the highest level of the hierarchy.

We will discuss these workflows in hierarchical order.

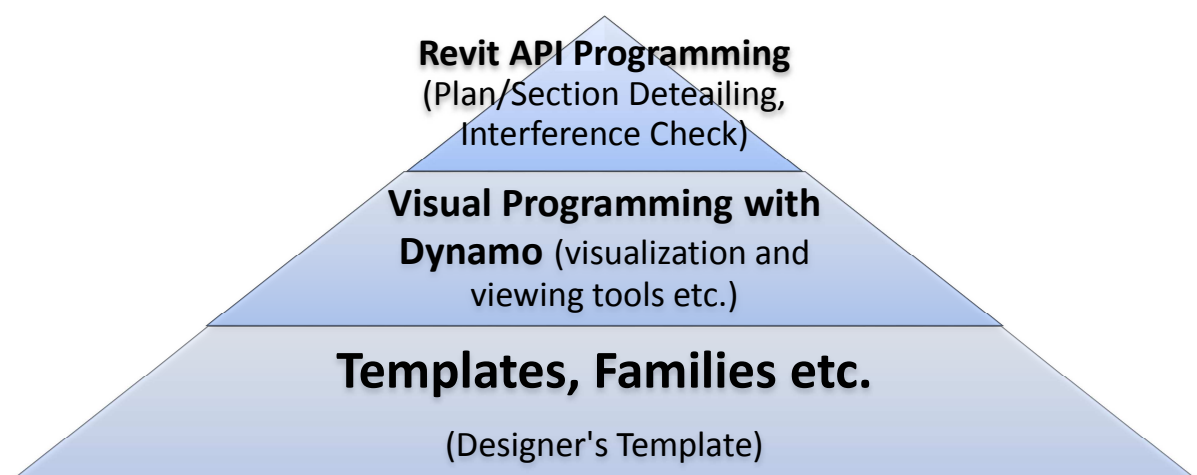


FIGURE 0.1 THE TOPIC REVIT WORKFLOWS AND WHERE THEY BELONG IN REVIT CUSTOMIZATION HIERARCHY



I. Creating intuitive but powerful Revit templates for Preliminary Design

Overview

We introduced Revit to our design and construction offices over ten years ago, and officially integrated it in our design workflows after two years. Until now, however, there are offices that are still unable to replace their two-dimensional workflows because they are so used to it and cannot compromise the time given them to finish such projects' phase. In the end, some of those projects will be done using Revit, including all the working drawings and contract documents.

The transition from 2D to 3D during this time conveys to us that we waste more time and effort. Designers who are used to the software for this kind of workflow would only tell us that they want the plans and elevations in the preliminary phase to be presentable. We thought we can do those workflows that can be started even in 3D by introducing BIM at an early stage.

This has prompted us to come up with a solution that will not compromise the designer's timetable (and even conserve time) and the product of the design as most of them will be done in BIM anyway.

Who shall benefit?

Many things had been considered in showcasing BIM for these designers who are creating drawing for presentation in the early stages of design. The most important to them are

- 1.) They can produce models and drawing the easiest possible way;
- 2.) They can produce presentable 2D and 3D views;
- 3.) They can modify the model anytime, anywhere without the hassles; and
- 4.) They can plan, design, and, visualize non-stop while modeling

An adequate template has to be built that meets these considerations.

What is this adequate template anyway?

- 1.) It is one that must present as-if-to-draw, on-the-fly modeling, while keeping the BIM's essence, so that it will make you brainstorm design over Revit as if you are drawing on it.
- 2.) It must also be lightweight, as we cannot compromise intuitive designing with slow performance.
- 3.) Browsing views and families must be also organized.

Thus, this template must benefit designers, especially pure architectural designers. We will call this template **the Designer's Template**.



Starting the Designer's Template

Creation of templates is normally followed by standardization of its elements, including system family types (walls, floor, ceilings, roofs, etc.) annotations and materials. The mainstream template creation is for use of project all over the project period--from preliminary design to design development, and even up to post-construction management.

When we try to think of this template creation, it will include lots of system and external families, line and fill patterns, dimension and text styles, view templates, etc. This will be rational for design development.

But for creating a designer's template, we do not need all of them.

In fact, this template must be very compact and easy to carry; no overloads. We just use what we need particularly in making models in an intuitive approach.

As we all know, there are two basic methods of creating a template: 1) creating from scratch; 2) transferring standards from an existing project or template; but we can also combine both 1 and 2.

In creating a template from scratch, we recommend creating it without using another template as its reference that is, not creating from another template and instead starting from the basic template settings. In addition, some pointers in the template create are as follows:

View Templates and View Filters

- 1.) One to two view templates per view.
- 2.) Name the view templates to distinguish them with type of views.
- 3.) For view filters, create view filters for presentation and basic working state (optional).

Object Styles

- 1.) Line weights on cut must be the thicker than or the same with the projection line weight.
- 2.) Avoid using line patterns with dots.

Wall, Floor, Ceiling and Roof Types

- 1.) For wall types, at most two each for exterior and interior walls, with varying thicknesses.
- 2.) For the rest of the element types, at most two types with different thicknesses.

On the other hand, **when creating a template by transferring from project standards**,

- 1.) You may use any base template, and transfer necessary settings from an existing project,
- 2.) Clean the new template by removing other unnecessary setting created with the base template, and
- 3.) Enhance your template settings while keeping it simple.



The Key to the Designer's Template: FAMILIES

Revit templates are nothing without families, especially in this case. However, the families that we need here have characteristics that differ with the usual families. We call these families **Intuitive Families**.

Characteristics of Intuitive Families

An intuitive family is pretty straightforward. It lets Revit users easily change its properties to what they want it to be.

To characterize an intuitive family;

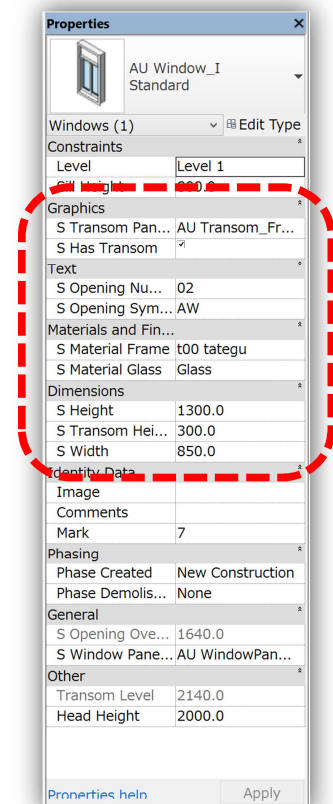
- 1.) It must be flexible or elastic;
- 2.) It also must be easy to edit;
- 3.) The family's nested families must be easily interchangeable; and
- 4.) Must contain only one or two types.

Putting it in the Revit context, all important properties of this kind of family must be editable in its instance level.

When we say an editable family in its instance level, it means there is no need to create several types per family (that is, one or two types for each door, window, structural members, fixtures, etc.) to create sets of patterns or fixed values.

The mechanisms that will drive inside the families are contained in instance parameters. For example, when changing a type of panel of a door or window, you can instantly change them from the family instance's instance parameters. Since parameters can be created either as shared parameters or as family parameters and parameters can now be created as a categorized family type, operation by just using parameters should be straightforward.

In order to achieve this using this example, however, a door family must contain some panel families loaded and nested in it. In addition, each panel must be created from a base panel (it may be a flush panel for doors for instance) of a particular category. This will prevent any inconsistency within all panels and avoid failure upon loading them into their parent family.



FIGURES I.1 A WINDOW INSTANCE SHOWING INSTANCE PARAMETERS



Pointers in creating Families and Nested Families for this purpose

Nested Families

- 1.) Start from a generic model template.
- 2.) Create an abstract or base family, and save with a distinctive name each time you create a derived family if you deem that you need another family, and be sure to derive them from its immediate variant or from the base model.
- 3.) Categorize a family based on which family it will be used primarily. For example, door panels must be set to Door category, window panels must be Window category, plumbing fixtures must be Plumbing Fixtures, etc. If nested families need to appear in two or more families (e.g. transom panels), the category can be Generic Model.
- 4.) Make sure that everything required in the creation (i.e., geometry, materials, dimensions, and parameters assignments) are done as well. Assign instance parameters to them.

FLEXIBLE FAMILIES

A. Doors and Windows

Assign the parameters to appropriate dimensions and materials.

- 1.) Create a categorized parameter that will hold all nested families belonging to its category.
- 2.) Add a visibility parameter to the families that do not need to be visible anytime (for example, transom panels).
- 3.) Name the family type with one word like "Standard" or with one character like "*".

B. Furniture and Fixtures (ready-made)

- 1.) Use arrays to increase or decrease the quantity of the nested families.
- 2.) Create the furniture or fixture sets which are common on almost all of the building types. When families are limited to few building types, better not to load them inside the template and save them into a common folder.

C. Structural (and Architectural) Members

- 1.) Create a family that its geometry can be controlled with instance parameters.
- 2.) If you want to create a family with fixed sizes, create another family different from the above, and use type parameters instead.

Pointers in creating parameters

- 1.) For families that depend on nested families, the parameter must be set to a family type of a particular category. Likewise, the families that are to be nested in the parent family must be in the category compatible with that of the shared parameter. Following this, only the necessary families will be shown from the parameter. However, for nested families used across some parent families, assign a Generic Model parameter to it.



- 2.) Decide whether the parameter is a family parameter or a shared parameter. Normally, when using the same family for design development and scheduling, it is likely that all the parameters must be shared type parameters. With this in mind, to create a set of type parameters that you think must appear in your schedules. On the contrary, instance parameters may not be a shared parameter; it may only be bound as family parameters for the subject family.
- 3.) Create instance (family) parameters that you only need. Ideally, instance parameters to be created are fewer than type parameters. For doors and windows on preliminary design, you would only need Width, Height, Opening Symbol, Opening Number, Remarks and others that affect the geometry or indicate initial settings. Others such as parameters for provisions and details shall go as type parameters and they shall have values later in the design development phase and scheduling.
- 4.) When applying yes/no parameter combinations to control the visibility of nested families, assign a yes/no parameter to the nested families' visibility parameter, create several integer parameters and apply power-of-two logic formulas that triggers visibility. Refer to Figure I.2.

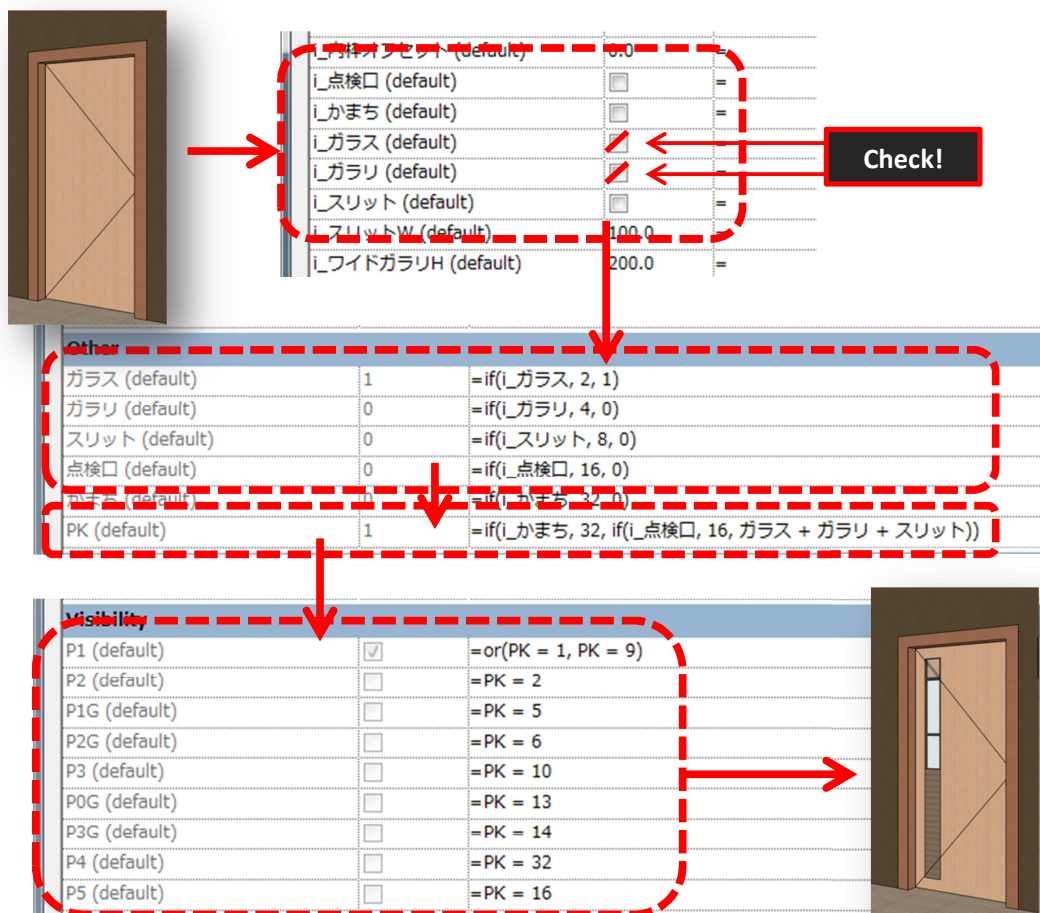


FIGURE I.2 THE YES/NO PARAMETERS AND THEIR ROLE IN NESTED FAMILY VISIBILITY LOGIC.

Problems to be anticipated (and how to address them)

While the extensive use of instance parameters to such families entails intuitive approach, there are also disadvantages in its use. They are:

- 1.) Hard to manage with schedules;
- 2.) Unavoidable inconsistencies and discrepancies;
- 3.) Prone to accidental editing and modification;
- 4.) Slow response when many nested families are loaded; and
- 5.) Troublesome when transitioning to design development/construction document phase.

The first three are straightforward when you operate the families. The fourth problem shows when you try to change the nested family that you want to assign with the family type parameter, thus nullifies the purpose of the family being lightweight if not property managed. The fifth problem is going to be a big hurdle when you have to use the project in the later design phases, and on top of that, it also presents high risk of information loss especially when transition to a more manageable family is done manually.

To be able to perform a fast transition, we provide you a program that does the work (as shown in Figure I.4).

Prerequisite and instructions are in my blog <https://mathcadbimthingy.wordpress.com/>.

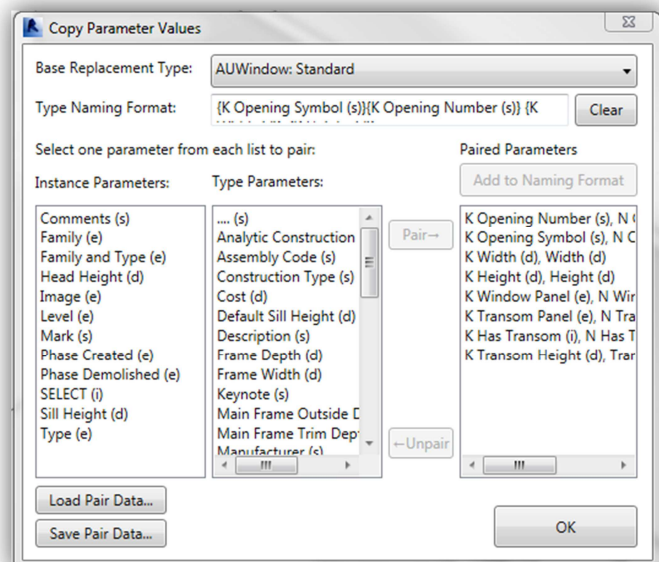


FIGURE I.3 AN EXAMPLE OF PARAMETER-COPYING PROGRAM.

II. Programmatic Workflows

Workflows using Dynamo

Most of us were accustomed with Dynamo as a computational design tool that takes advantage of the power of Revit with visual programming. To attain such an objective, Dynamo has to go deep inside Revit's application programming interface, or API.

Due to this necessity, Dynamo has also opened its doors to users who want to create and modify elements in Revit, extending beyond Dynamo's true objective of computational design. This possibility further extends Dynamo's capabilities and further lets users have control of Revit with it.

Some examples you can refer to when integrating Dynamo in your workflow are given below:

Grid and Level Layout

The method of creating grids and levels in Revit comes with an input string of delimited numbers defining intervals (e.g. "5000*3,6000").

For grids, the creation orientation will be left to right for vertical grids, and bottom to top for horizontal grids. If grids must be rotated, the pivot will basically be at the intersection of the first horizontal and vertical grids.

Apple

This is one basic example of a project using Dynamo's computational design tools. It was originally a sculpture outside a hospital requested by the client. Given a half profile of an apple in a conceptual mass project, the apple is generated with equal number of rotated profiles and divided segments, and end points from the segments are extracted and brought to multiple spline creation. The splines are then used to create a solid mass.

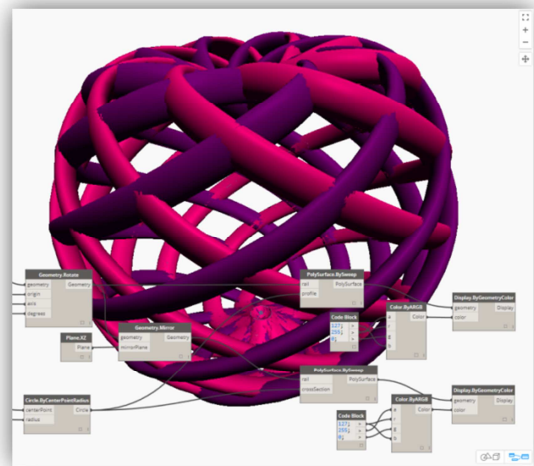


FIGURE II.1 THE APPLE IN DYNAMO

Conditional Exterior Panel Layout

Lay outting of exterior panels with colors at random might seem easy, but this is the toughest job I had while participating in a project. This project was not done originally with Dynamo, but it is another proof of the extensibility of Dynamo.



The most challenging part in generating the result for this project is the application of several conditions that the clients want. However, these conditions cannot be done with just the existing Dynamo nodes. There is a need to input codes for more convincing results.

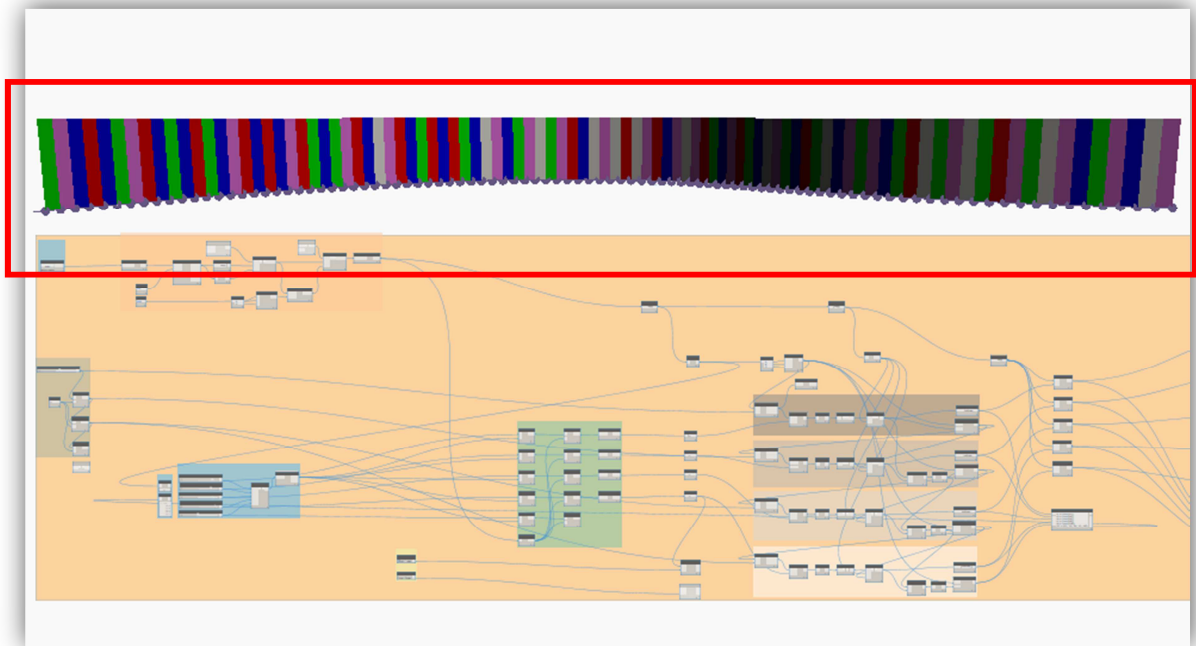


FIGURE II.2 A LAYER OF RANDOM COLORED VERTICAL PANELS IN A CONDITIONAL LAYOUT

Vicinity Visualization

An example of our computational analysis tools, vicinity visualization generates a result indicating graduated colors that define how much the parts of the area are able to view the subject building. The brightest color indicates full visual of the subject building, while the darkest color does not.

This vicinity visualization program is used to determine how much an object is visible in a particular area on the ground. With this result, we will be able to understand if a mass must or must not be seen, or is tolerable to be seen in a specific area, and at one point we will also be able to understand if such mass meets any visual prerequisites of a particular place that is sensitive to some visualization issues.

The drawback of doing this with Dynamo--at least from my design--is that the finer the result required a significant amount of calculation time to generate the color display. This is because the example maintains an algorithm that every piece of face from the topography fires a vector to the subject building's faces, and from the valid faces it also produces pieces of small faces that are required for further calculation.



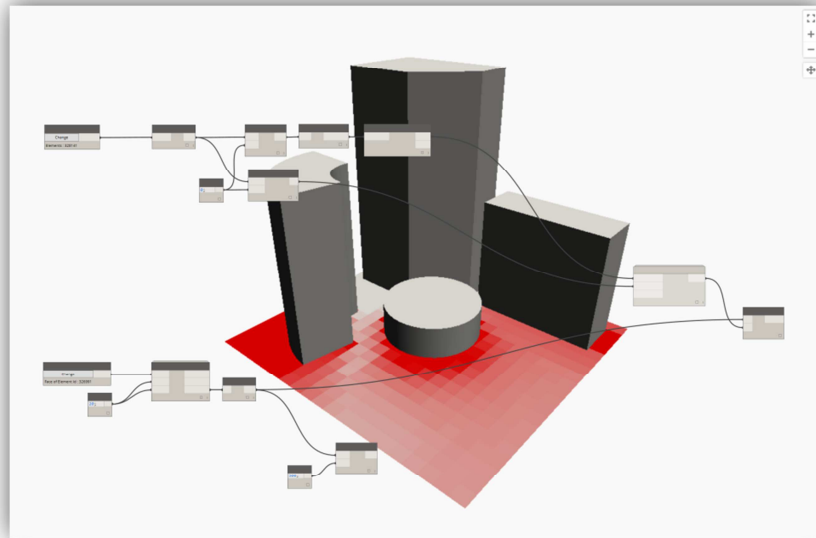


FIGURE II.3 THE VICINITY VISUALIZATION TOOL

Cabbage

Originally one of our visualization tools, Cabbage indicates how much you view an area from several positions. This visualization tool is ideal if you want to check the overall visual percentage of--say a soccer field in a stadium or a stage/screen in a theater--taking into account the other viewers' heads (thus, the name "cabbage") and other possible obstructions in front of the subject viewer(s). The subject viewer(s) is then rated according to the percentage of the field it can view.

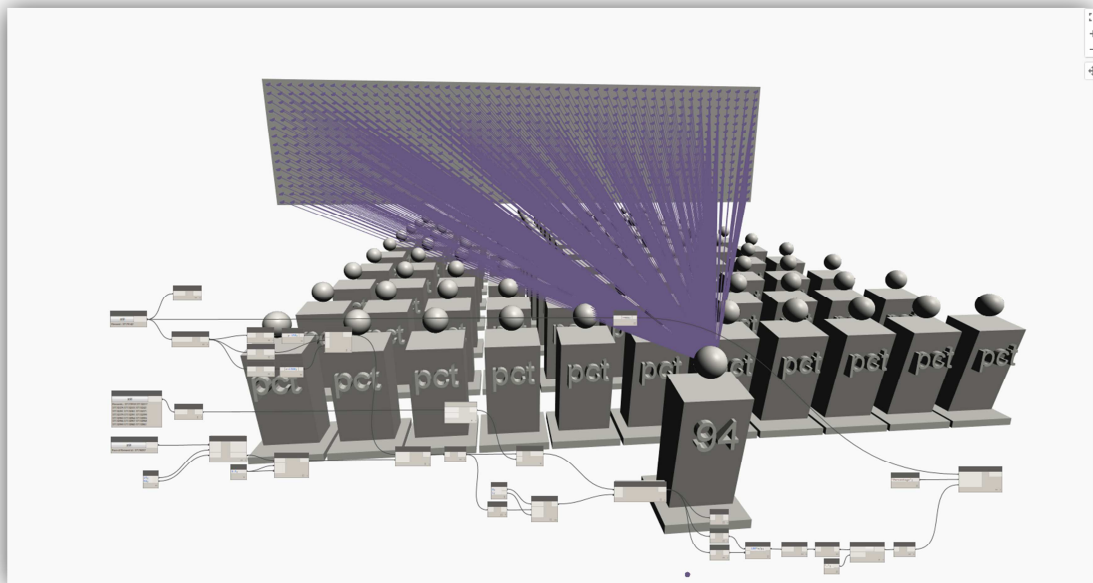


FIGURE II.4 THE CABBAGE ON DYNAMO AS A VISUALIZATION TOOL



Workflows using (programs developed with) Revit API

The best option for customization to inject in your workflow, however, is using programs developed for your needs. One option is to use tools that can be found at the Autodesk App Store (apps.autodesk.com) and look for the one which suits your workflows. When none is available, you can develop your own programs using Revit API (Application Programming Interface).

Luckily, Microsoft has already showcased Visual Studio Community, a program developing software which is now downloadable for free and for personal use. Then you can use Revit API for you to develop programs in Revit.

While developing with Revit API demands your programming skills and experience in either VB.NET or C#.NET, there are many tutorial sites where you can learn them for free. In addition, Autodesk also offers some sample programs from their software development kit (SDK) that comes with the installation of your Revit.

For those who are interested in developing programs with Revit API, I have some tips that are specified on a separate sheet for your reference.

In connection to this, the following workflows are those which tools that will be introduced are developed with Revit API.



III. Using more of HTML files for Interference Checking and its documentation

Revit has the functionality of making interference checks among elements in category, as well as among linked models. The best use of this is when checking with linked structural and/or MEP models. After interference checking, instances that clash in the model can be viewed when the results are shown. Moreover, this checking tool can produce an HTML file that contains the result of this checking.

But the role of the output HTML file in Revit basically ends there. The use of this file is mainly on Navisworks, which is the best tool for further checking the clashes. However, you may not need Navisworks unless you want to further your investigation on the interferences on your project.

With this in mind, we will make the HTML files usable in identifying more of the clashes inside Revit.

Introducing the tools

We developed tools solely for reading HTML contents, as well as on what to do with these contents.

HTML files contain enough data that represent elements in Revit that interfere with each other. We take advantage of this information to create indicators and views as well. The sphere indicator possesses information on both interfering elements and an index copied from the HTML file.

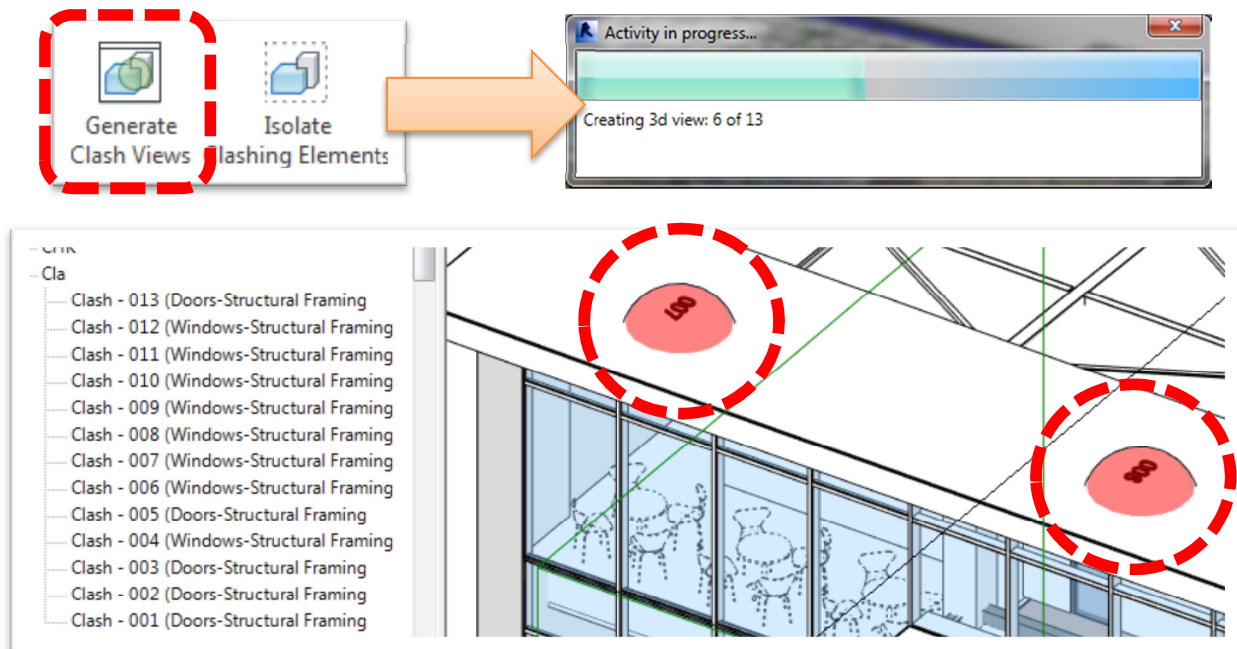


FIGURE III.1 (CLOCKWISE FROM TOP-LEFT) TOOLS FOR INTERFERENCE CHECK DOCUMENTATION, ACTIVITY IN PROGRESS, AND RESULT FROM A 3D VIEW



What is important in these results is the generation of the red spheres and their corresponding 3D views. These indicate the locations of interference, as well as the indices pertaining to the 3D view.

However, results such as those shown on the right do not really tell which are interfering. Unless the elements are clearly visible, they do not give us decent results.

To make things better, we need to use another tool to isolate the elements.

Result shows clearly the elements that interfere with each other.

(Note: At this point, elements in a linked Revit file are still shown after running the command. In this example, a structural model is linked to the architectural model. This will not occur when the linked model is bound to the architectural model, or all the elements subject to interference check are all in the main model.)

Running the tool for another time after we fix the interfering elements will not pose any problem. The tool will reset the 3D views generated earlier in this stage.

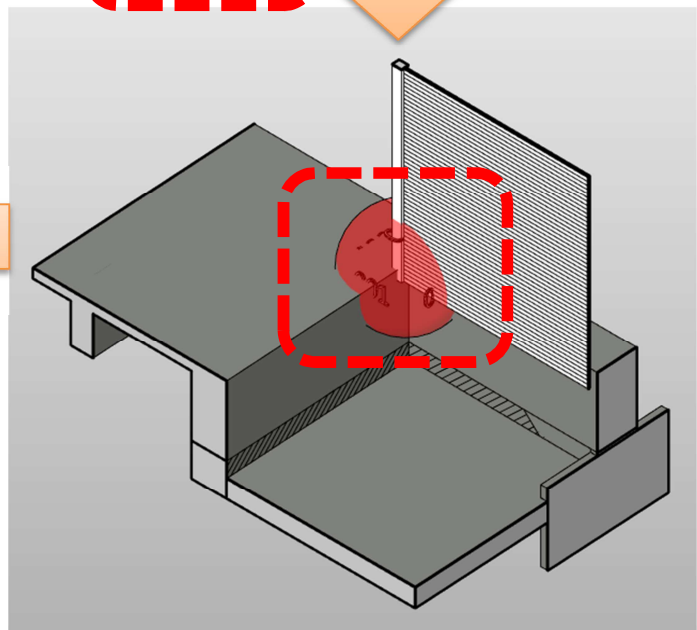
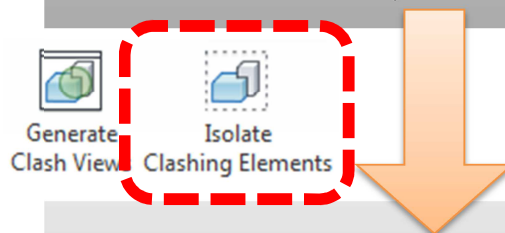
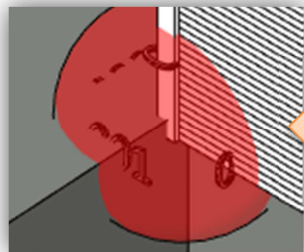


FIGURE III.2 (CLOCKWISE FROM TOP) THE VIEW3D RESULT GENERATED AFTER GENERATING CLASH VIEWS; RESULT AFTER ISOLATING CLASHING ELEMENTS; RESULT AT A CLOSER LOOK.



Recommendations when using the HTML files for this purpose

As stated earlier, HTML files contain the IDs of the elements that interfere with each other. These IDs, however, change when the project is being work-shared. Once these IDs are changed, you need to redo the interference check to output of HTML files according to its present state.

Also, linked models are sometimes not aligned with the main model (it happens when there is not enough coordination between models). Although the program supports linked models, positions of the results might be incorrect in some or all areas.

To avoid this effect and to secure the right position of the elements subject to checking, we recommend to:

- 1.) Disintegrate from the central file before interference checking;
- 2.) Save the project under a different file name after disintegration; and
- 3.) Bind all linked models after saving the project.

When all of these are done, you can execute the program and obtain better results.

Documenting your Interference Check 3D Views

When you are done with generation of 3D views, you would want to compile them by laying them out in sheets. You can also add annotations to further describe or identify the clashing objects in each particular view.

Furthermore, since the resulting 3D views contain red spheres with indices, you can also include a key plan or key view to indicate where in the model a particular interference occurred.

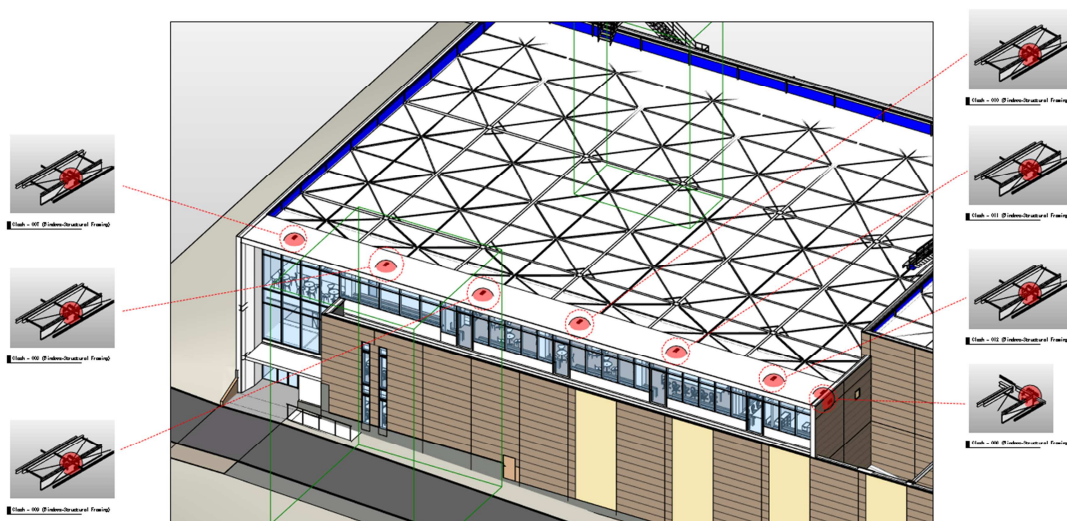


FIGURE III.3 A SAMPLE OF DOCUMENTING INTERFERENCE CHECK VIEWS.



IV. Plan and Section Detailing with Walls, Floors, Roofs and Ceilings

When transitioning from preliminary/schematic design to design development and construction document phases in BIM, you will likely consider detailing your plan views and section views. In all BIM software, however, most detailing workflows tend to be done in two dimensions. Every category has every tag for detailing required. Lots of line styles and patterns, and fill patterns will come up to the stage abruptly. Oftentimes, putting annotation lines to models in plans and section cannot be denied.


In Revit, however, there are things that can be done using the model, especially with walls, floors, roofs, and ceilings.

These elements' compound structure property has it.

Let us review the compound structure of the above mentioned elements.

The contents of a compound structure are made of layers where each layer has combination of functions, materials, and thicknesses. These layers can be core layers or shell layers, depending on where they are positioned in the compound structure.

Creating composition of layers of different properties will mean creating different types. As you go further in detailing these models, you will find out later on that you are creating many types, and at some point you may not be able to manage them.



	Function	Material	Thickness	Wraps	Structural Material
1	Finish 1 [4]	Silica Board D	6.0	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	Substrate [2]	Gypsum Board	12.5	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	Core Boundary	Layers Above Wr	0.0		
4	Structure [1]	RC	200.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
5	Core Boundary	Layers Below Wr	0.0		
6	Substrate [2]	Gypsum Board	12.5	<input checked="" type="checkbox"/>	<input type="checkbox"/>
7	Substrate [2]	Gypsum Board	9.5	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Membrane Lay	Wallpaper	0.0	<input checked="" type="checkbox"/>	<input type="checkbox"/>

FIGURE IV.1 AN EXAMPLE OF COMPOSITION OF A COMPOUND STRUCTURE OF A WALL TYPE.

How to Deal with Creating Types

When you are about to detail these elements using Compound Structure, one problem you face when creating their types is the repetitive task of:

- 1.) Composing different configurations of compound structures for every type;
- 2.) Too many mouse clicks when configuring your compound structures; and
- 3.) Finding the type we want to use to other elements using the Type Selector.



Especially if you have drawing standards and rules in detailing walls, floors etc. for drawing documentation, you will always have to put in mind these standards and rules while building element types. This will lead to inconsistencies and discrepancies, and eventually take more of your modeling and documentation time when you found them along the way.

To eliminate these problems, in our company's case, we developed an editor program (*refer to Figure II.2, Edit Wall Type*) that handles everything; from keeping drawing and material standards and modeling rules, to configuring compound structure layers.

Aside from this program, we also made other programs in line, such as:

- 1.) Standards Setting initializer (*Initialize Project*);
- 2.) Real-world Material and Revit Material Updaters using Excel (*Export/Import Materials*);
- 3.) Compound Structure Re-configurator (*Reconfig/Rename SS Walls*); and
- 4.) Room Material Setting Synchronizer (*One-click register Fins*).



FIGURE IV.2 PROGRAMS WE DEVELOPED THAT HANDLE DETAILING WITH COMPOUND STRUCTURES

While these programs are not available for your evaluation, you might want to consider developing or ordering to create programs such as these, especially if you work on projects that deal with too many types (especially wall types). This is so that you do not have to worry about building your wall, floor, ceiling and roof types on your own while maintaining your standards and rules.

If ever you consider having programs that will do the detailing for you, things that you might need to prepare are:

- 1.) Drawing standards and rules in building your types in Revit;
- 2.) Widths or thicknesses of all the commercial building materials you use;
- 3.) Revit materials, line and fill patterns you always use;
- 4.) Specific shared parameters you want to use, if any; and
- 5.) Other matters and mechanisms specially applied when modeling with these types.

Likewise, you might need enhancing tools for your Revit detailing workflows, such as programs that generate wall core centerlines, place tags and transfer data to room parameters from its surrounding elements for scheduling.

When you already have such tools, not only your drawing standards and rules are kept;

- 1.) you may no longer use the Type Selector; and
- 2.) you may no longer press the "Edit Type..." button;



Summary and Outlook

Revit alone has really much to offer especially when it comes to its hidden capabilities. We only have to unleash them by further customization. The ease of designing with Designer's Template in Revit has been brought out, as well as extending your workflows in Revit using Dynamo and Revit API, eventually giving us the capability to connect end roads like Interference Check HTML file utilization, and detailing enhancements with compound structures.

The above mentioned enhancements to your Revit workflows do not cover everything yet. There are many things that can be discovered and spread all over the Revit world. But with these, you may already be able to gain more ideas and start your experiments with Revit, and thereafter discover and contribute to more enhancements and developments in the application itself.

As Revit is poised to release another version next year, we expect changes in its internality, particularly the availability of cloud services or software in the cloud (just like AutoCAD 360). These imminent changes bring challenges in your Revit workflows and it is crucial to always adapt to these software enhancements to get only the best of what Revit has to offer.

As for me, I very much look forward to these developments-- and I hope you do, too.



Get materials here!
Available after the session

