

AB5115 - Fine-Tune Your Autodesk® Revit® Models for Autodesk 3ds Max®

Pierre-Felix Breton

Product Designer
Autodesk Media and Entertainment

Class Audience

Autodesk 3ds Max users and Revit users who want to bring Revit models into 3ds Max for rendering or animation

Class Description

There are several things you can do directly in Revit to minimize the work required in 3ds Max prior to rendering or animating. This class focuses on methods that can be adopted in Revit to prepare a model for 3ds Max.

Key Learning

- Combine entities effectively in 3ds Max for easier manipulation and faster rendering
- Leverage views and filters for preparing data for 3ds Max
- Prepare materials and render appearances for better results in 3ds Max
- Customize families for better results in 3ds Max

About the Speaker:

Pierre-Felix Breton is a software designer who specializes in the field of physically-based lighting simulation and rendering. Currently employed by Autodesk Media & Entertainment, he participates in the creation of products such as Autodesk® 3ds Max® where he focuses on the integration of the mental ray® rendering engine as well as daylight simulation technologies.

He also contributes to the design and development of materials and shader libraries included in Autodesk products where color consistency and physical accuracy is critical.

As his professional background includes electrical engineering, computer programming, and theatrical lighting, Pierre-Felix consults regularly on various architectural lighting design projects as a designer, technical coordinator, and simulation specialist. His list of recent projects includes the realization of the Scottish Parliament Building, the New York Times Building, and the Washington Air Force Memorial.

Get the latest Revit Updates

The following updates contains improvements and fixes to lights and materials that will be useful to you for exporting data from Revit to 3ds Max.

Autodesk Revit 2012 – Update 2

For Revit Architecture: <http://bit.ly/hWxcsg>

For Revit MEP: <http://bit.ly/kzQsiM>

Hotfix – Autodesk Revit 2012 – Apply material family parameters using the Paint tool

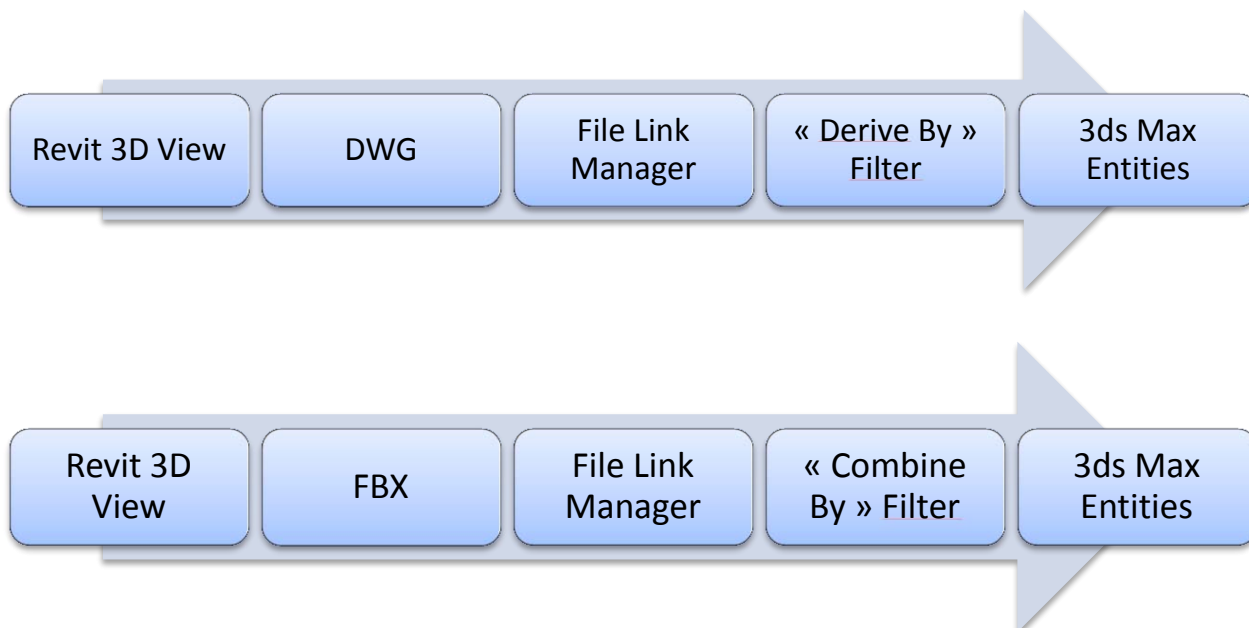
For Revit Architecture: <http://bit.ly/uvnjl>

For Revit MEP: <http://bit.ly/tEEzTi>

DWG vs FBX

The general concept

Whether you use FBX or DWG, the same principles applies. In essence, Revit exports the content of a view into a file format which then get imported into 3ds Max.



Whether you bring data in 3ds Max via FBX or DWG, both offer methods to organize your data inside 3ds Max to avoid getting what we typically call a “polygon soup”.

Quick comparison between FBX and DWG file formats related to 3ds Max

Although FBX and DWG have a fair amount of distinct characteristics, let's have a look at what is relevant for bringing Revit data into 3ds Max

	FBX	DWG	Comments
Curved Geometry	Fixed Tessellation	Controlled tessellation via Solids	<i>(More detailed information below)</i>
Organization of data inside 3ds Max	Object Meta Data in Scene Explorer	Layers	
Scene complexity management	Combine by Material Combine by Family Type Combine by Category Combine as One Object Do not Combine	Combine by Layer Combine by Color Combine as One Object Do not Combine	<i>(More detailed information below)</i>
Lines & Terrain Contours	No	Yes	
Materials	Yes	Yes <i>(but names are messed up)</i>	
Lights	Yes	No	
Daylight / Site Location	Yes	No	

Unless specified otherwise, the rest of the document will assume that data is brought to 3ds Max via the FBX file format.

“Combine by” filters overview

Typically, 3ds Max has hard time dealing with thousands of individual objects. In fact, 3ds Max deals better with fewer objects, with more polygons into them.

In 3ds Max, the File Link Manager offers a few data organization tools that should not be neglected, which are not all available with regular import options. Among other things, the File Link Manager offers options to combine entities together, to reduce the number of objects in 3ds Max.

Therefore, I recommend exploiting one of the “Combine By” option that is available for both FBX and DWG files, via the File Link Manager.

“Combine by” filters available with DWG

	Pros	Cons	Comments
By Layer, Keep Block Hierarchy	Preserve Materials (but their names are lost)	Creates many Max objects, to support complex block structures (one per family)	Leverage the Layer Mapping capabilities with Subcategories (see below)
By Layer	Reduce object count.	Materials are Lost Creates many Max objects, to support complex block structures (one per family) due to a bug in this filter.	Leverage the Layer Mapping capabilities with Subcategories (see below) Families exported as blocks not exploded due to a bug in Max DwG importer
By Color	Reduce object count.	Materials are Lost Creates many Max objects, to support complex block structures (one per family) due to a bug in this filter.	Leverage the Layer Mapping capabilities with Subcategories (see below) Families exported as blocks not exploded due to a bug in Max DwG importer
As One Object	---	Everything is in a “polygon soup”. Can be slow to import/link Creates a unique VIZBlock object which combines meshes and lines into a single object	Avoid this option on large Revit models

By Entity	Preserve each object individually selectable.	Increase object count in 3ds Max, reducing overall interactive performance.	Great for keeping furniture elements as individual elements.
------------------	---	---	--

“Combine by” filters available with FBX

	Pros	Cons	Comments
By Material	Reduce object count. Prevent Multi Sub Object Materials from being created.	You depend on the granularity of the material assignments in Revit. For example, if all curtain walls use the same glass material as interior doors, they will be combined in the same object in 3ds max.	<i>Great method if you do not plan to move objects around and don't want to deal with Multi Sub Object Materials</i>
By Family Type	Reduce object count. Prevent Multi Sub Object Materials from being created.	Multi Sub Object Materials are created.	<i>Good balance between scene complexity and flexibility Parts has problems</i>
By Category	---	Multi Sub Object Materials are created. All Walls are clustered together, all Roofs and so on...	<i>Hmm, what were we thinking when we have implemented this one....? Parts has problems</i>
As One Object	A single Multi Sub Object Materials is created, which is simple to manage	Everything is in a “polygon soup”. Can be slow to import/link	<i>Great for quick lighting studies where you only care about massing</i>
Do not Combine	Preserve each object individually selectable.	Increase object count in 3ds Max, reducing overall interactive performance.	<i>Great for keeping furniture elements as individual elements.</i>

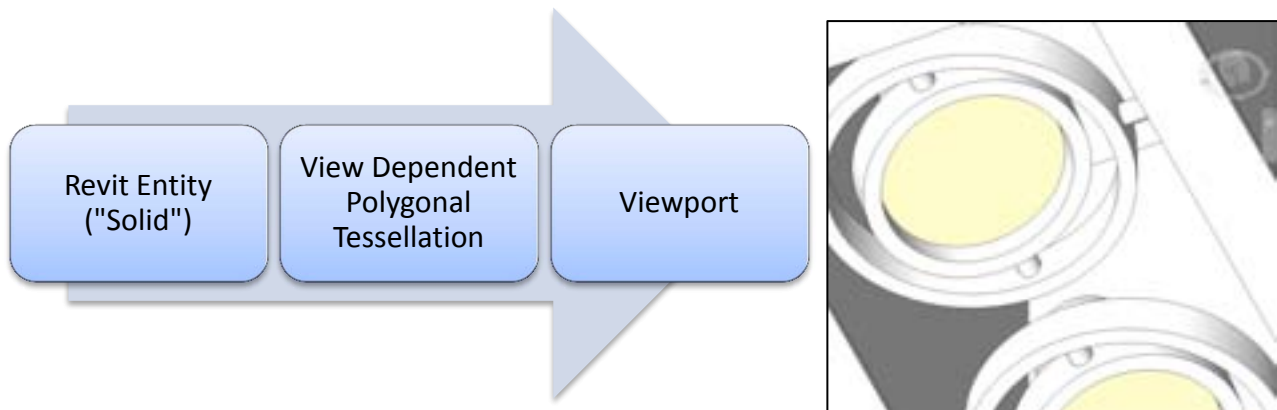
Curved objects from Revit to 3ds Max

The Revit database internally store entities in the form of “solids”. On the other hand, 3ds Max store entities in the form of faces (polygon meshes). Therefore, a tessellation (conversion) process has to occur somewhere in between.

You will notice below that the results of this tessellation are not quite the same depending on the file format you choose. Some work better than others. Hopefully, this will improve over time but for now, let’s state how things are:

The Revit viewport

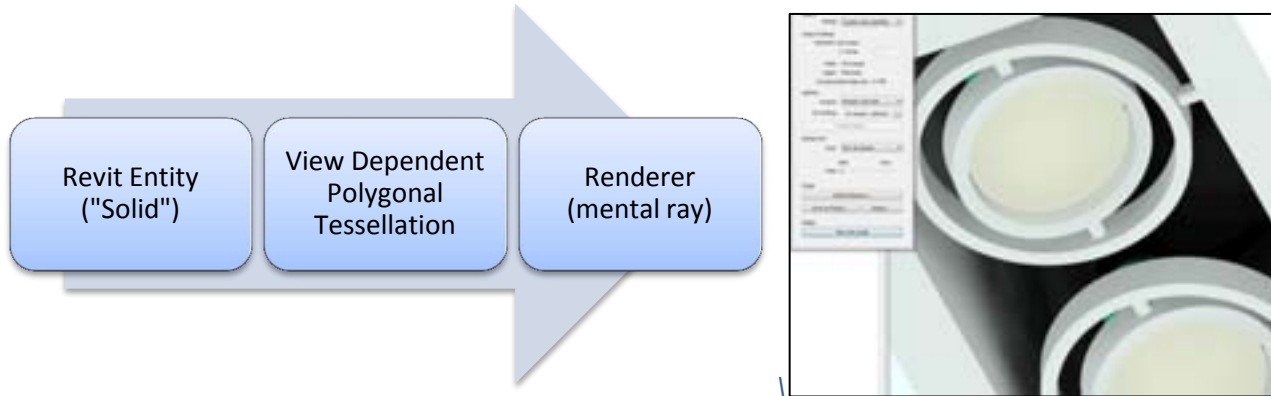
The Revit camera triggers the level of detail for each object on the fly, based on the viewing distance. This dynamic tessellation system has 16 different levels, or steps. The following images illustrate what you can get in Revit when zooming closely on small, curved objects:



Small curved objects (a light fixture in this case) remain smooth in the Revit viewport.

The Revit renderer (mental ray)

When rendering using mental ray inside Revit, the same view dependent tessellation occurs. As a result, the rendered image receives the same polygonal information than the viewport and the results look consistent:



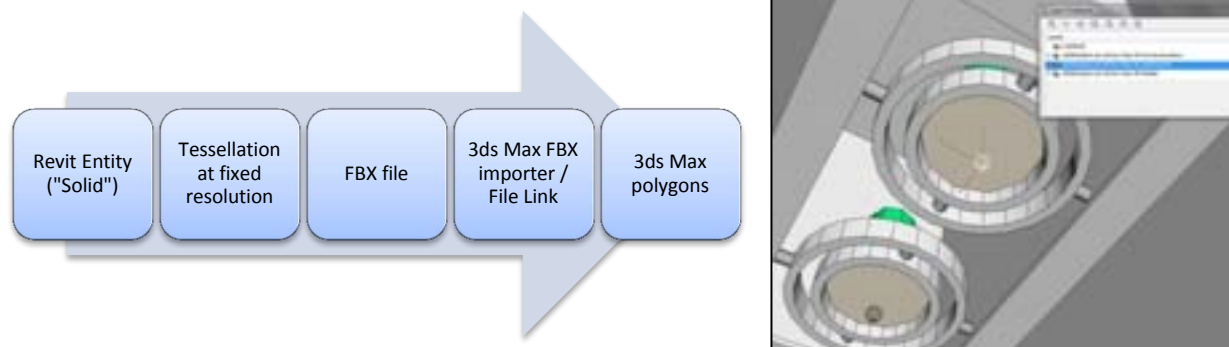
Rendered in Revit, curved objects remain smooth

Revit → FBX → 3ds Max

Fixed tessellation

Although Revit has the capabilities to subdivide curved geometry with a high level of accuracy, the internals of Revit associated with geometry tessellation are fixed and unexposed to the users and/or the API in the moment.

Therefore, when exporting to 3ds Max via the FBX file format, the level of detail used by the Revit FBX exporter is using a fixed value. Unfortunately, you get what you get and some objects will appear too coarse, especially small tubular elements:



Exported in FBX and brought in 3ds Max, objects are tessellated with a fixed level of detail / resolution.

At this stage, you cannot increase the resolution of the objects anymore as everything is baked into polygons at the export from Revit stage.

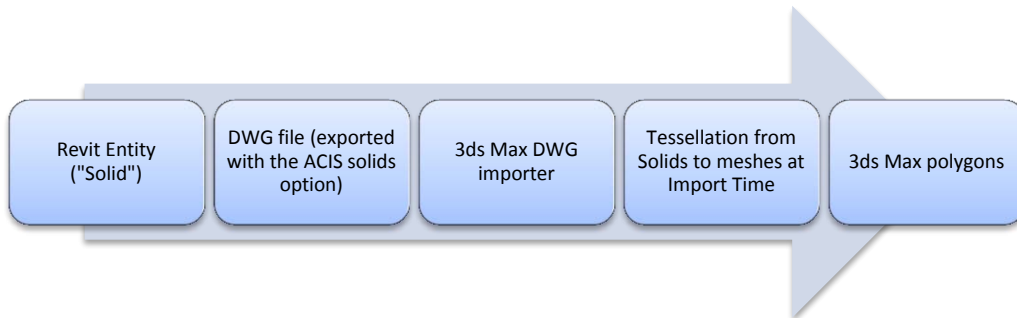
Revit → DWG → 3ds Max

A potential solution to the tessellation problem

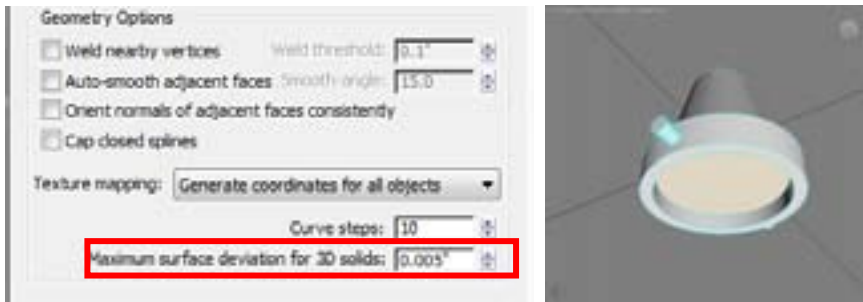
The DWG format offers an interesting option to preserve the quality of curved elements. By exporting to DWG as **ACIS solids**, we maintain the parametric curvature information in the DWG model. That is, we bypass the Revit tessellator.



The ACIS solids option from the DWG exporter will prevent entities from being tessellated as polygons by Revit. Solid information will be maintained.



Doing so gives us the option to use the 3ds Max tessellator available in the 3ds Max DWG import dialog, which offers control over mesh resolution for incoming solid entities:

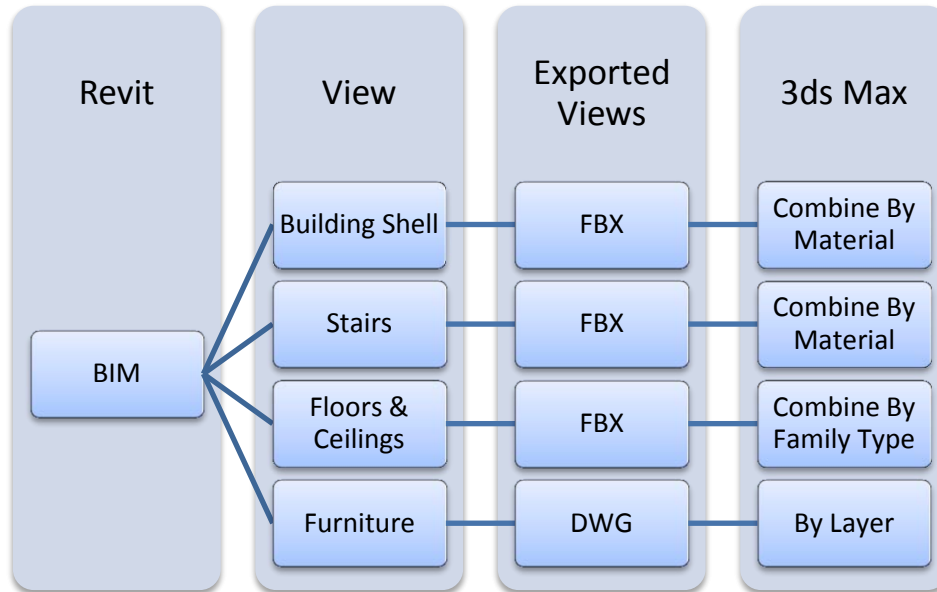


We then rely on the 3ds Max DWG importer to do the tessellation, which is not View Dependent and not locked to a fixed value.

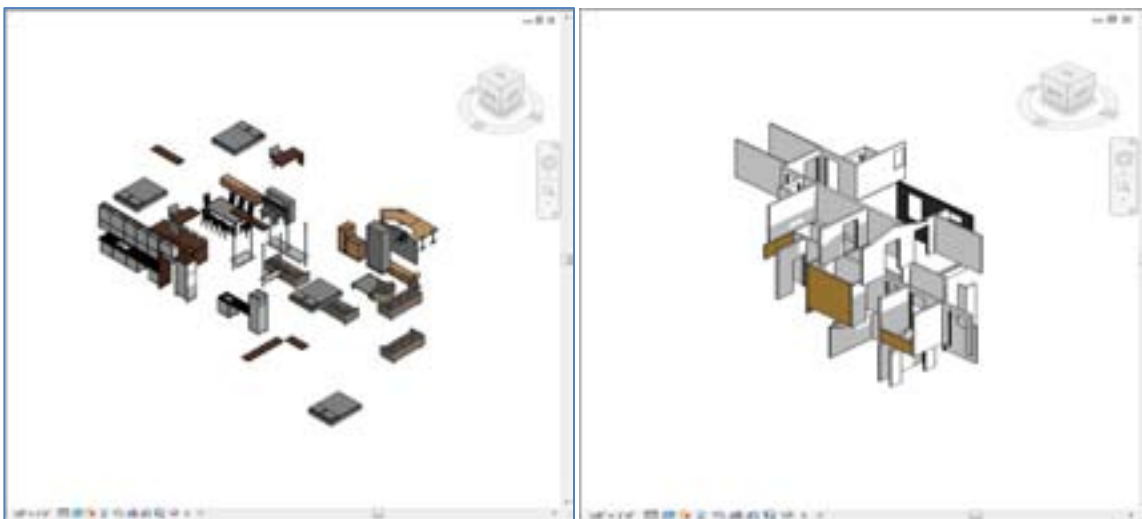
Strategies to adopt with Revit Views

Split your model across several views

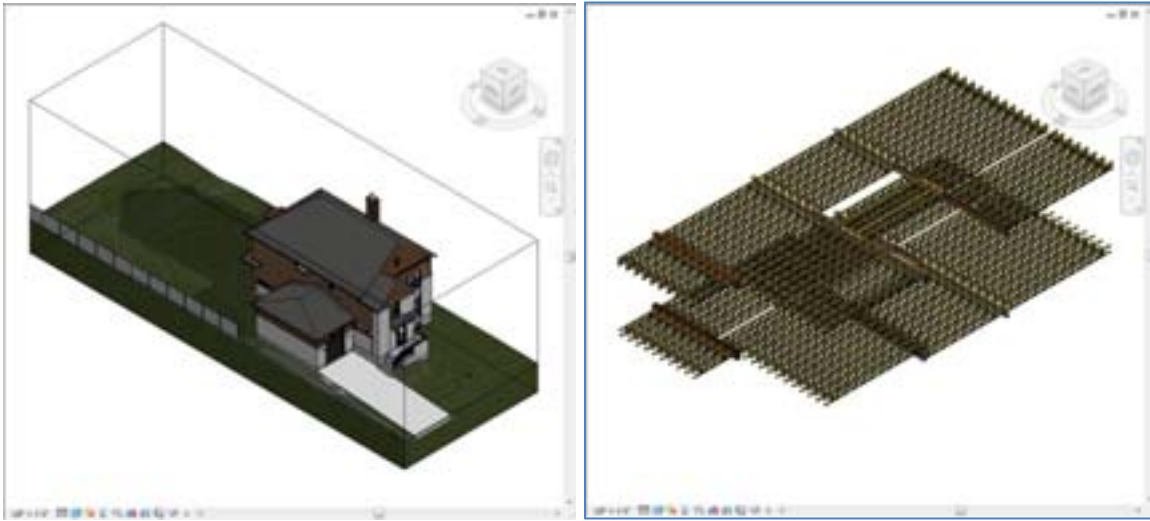
Instead of making a general 3D view of the Revit model and exporting it to a giant DWG or FBX file, I recommend to split the model across several distinct views with the appropriate filters defined. The gain here is that you will reduce the overall complexity of the model (manage it into smaller chunks) and be able to use different “Combine By” options.



This strategy will let you, for example, combine all walls and ceilings “By Material” and leave Furniture or Cabinetry uncombined. To achieve this, you can leverage the Visibility Graphics Override and Filters.

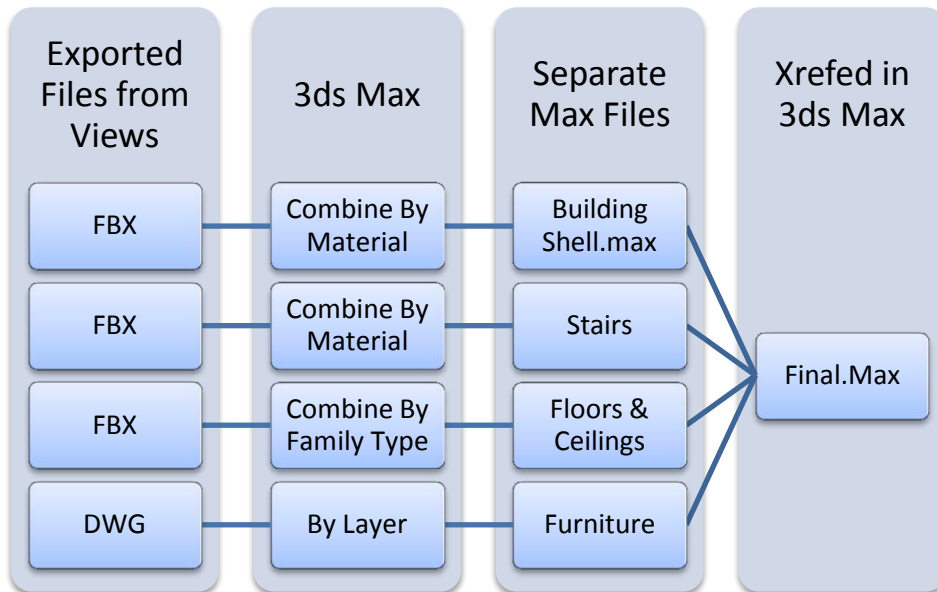


Two views where only furniture and interior walls are visible.



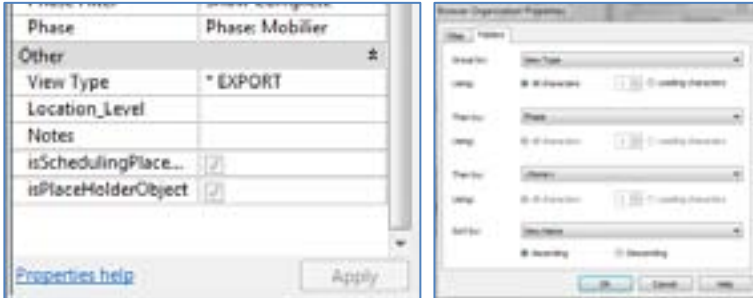
Two views where only the exterior shell and internal structure are visible.

With this strategy, you gain control over how the data is organized into 3ds Max. Furthermore, you can envision using this approach to allow multiple 3ds Max users to collaborate on the same project:



View Type & Project Structure Tip

To help with the above process, I like to identify views with special keywords. For example, assign a custom parameter to Views and mark them as “*Export to 3ds Max” as illustrated below:



A parameter named “View Type” applied to the View, with a custom property “*Export”...

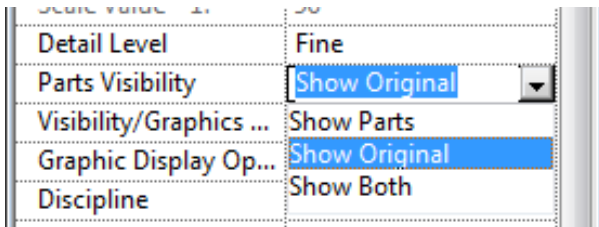


Views with the same parameter “*Export” grouped together the Revit project browser

Parts & Views gotcha!

Risk of getting overlapping objects

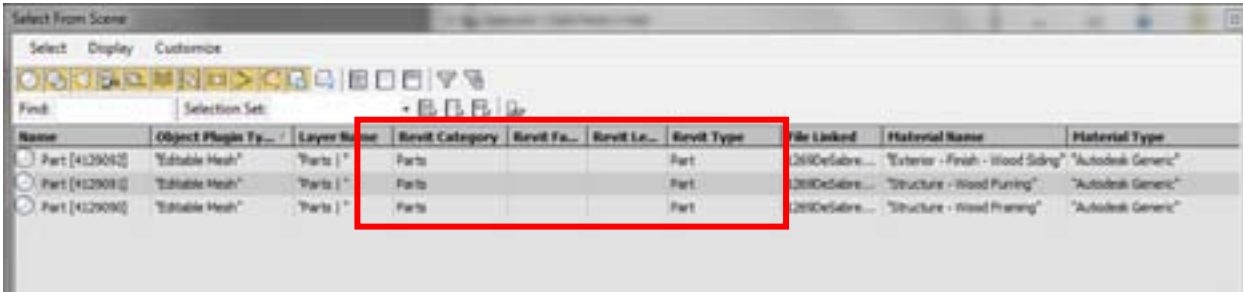
Keep in mind that Parts are separate objects than their originating Revit entities. Therefore, be careful about the Parts Visibility option in the View, as you can end up with duplicated / overlapping objects in 3ds Max which will automatically lead to render artifacts.



Parts Visibility set to “Show Both” will create overlapping geometry in the DWG/FBX file and you will get rendering artifacts in 3ds Max.

Problem with FBX Meta Data for Parts

Parts are a special “derived” object in Revit. As a result, their Type/Category is stored into a “Original Type/Category” field. This was not taken into account by the FBX importer of 3ds Max 2012. As a result, when you import Parts you will get something like this in the Scene Explorer of 3ds Max:



Parts in 3ds Max Scene Explorer: note how Category, Family and Types are all the same.

One major consequence with this problem is the fact that the *Combine by Category* or *Combine by Family Type* from the FBX File Link dialog will treat all Parts as belonging to the same *Type/Category*. As a result, they will all get combined together, *regardless of their original Type/Category*.

This is not a problem with *Combine By Material* or *Do Not Combine* options.

Partial scripted fix for Parts Meta data

If you used the “Do Not Combine” options from the File Link manager, you can still “fix” the Meta data issue with a scripted tool. This script is provided in the section entitled *Fixing Parts Meta Data (below)*.

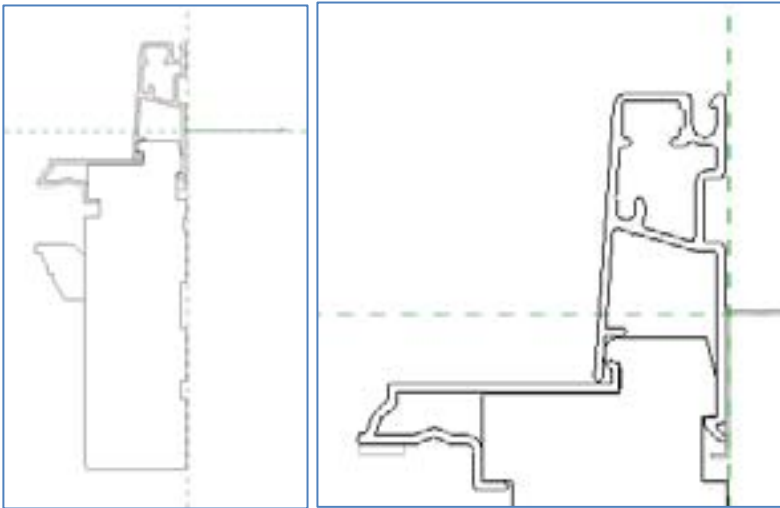
Since this script is fixing the Parts Meta data after the import/link process, it cannot be used to solve the *Combine by* problem discussed above.

Modeling with 3ds Max in mind

Level of Detail Considerations

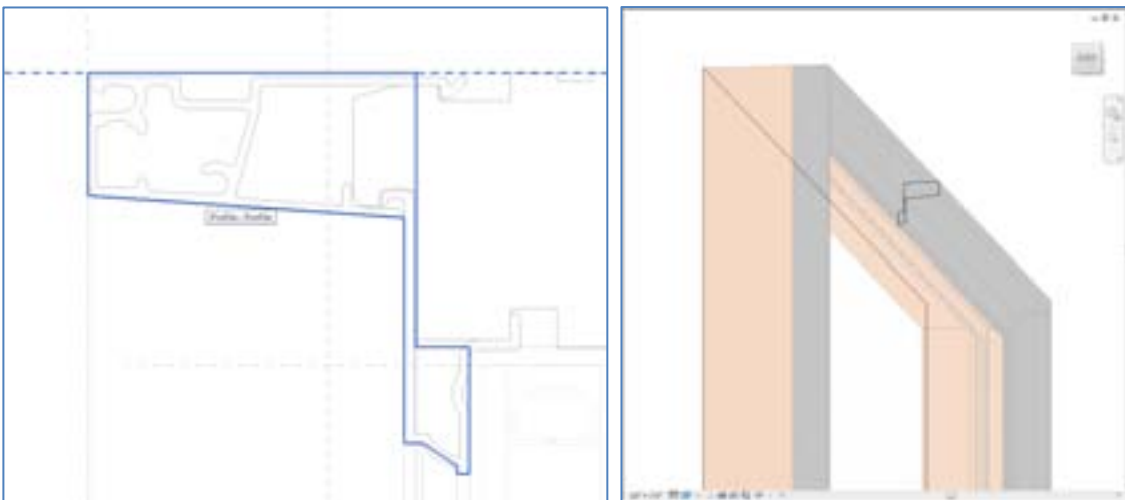
When it is “good enough”

Typically, manufacturers will provide detailed 2D profiles and drawings for their products. Here is an example I got from a window manufacturer:



Detailed profile for an aluminum window: too much detail to use in 3D.

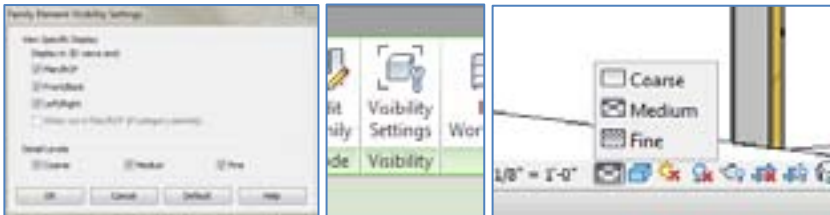
In no way I would want to use this “as-is” to create 3D extrusions: the Revit display would quickly get on its knees. Instead, the idea is to model a “rough” version of the original shape, where for example, rounded corners become sharp and dents and details become straight.



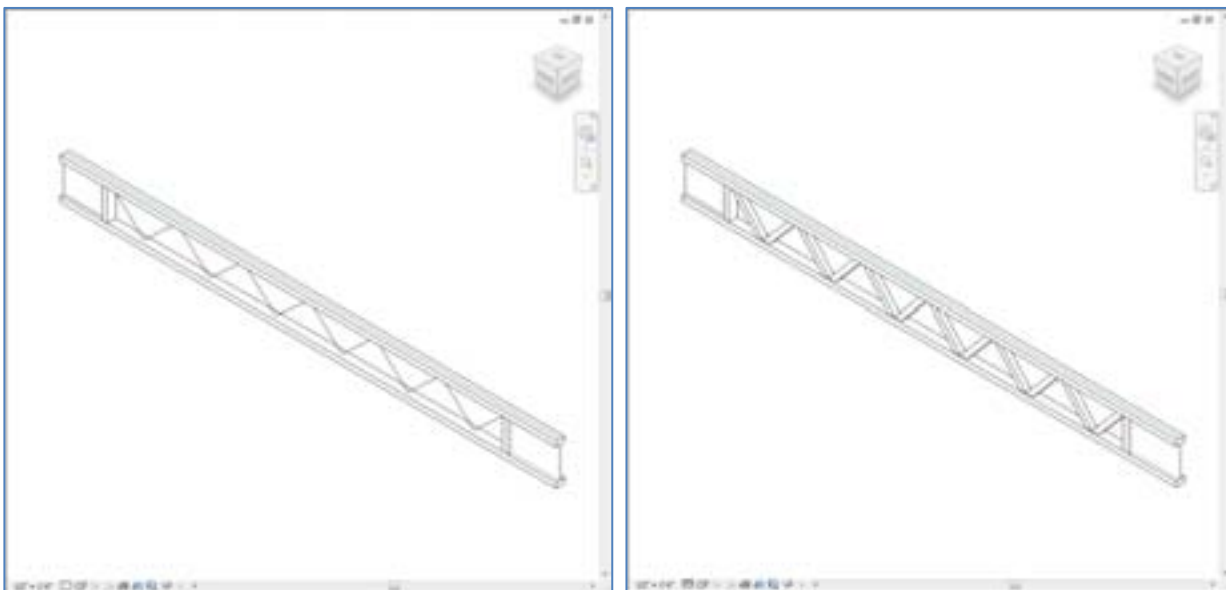
Simplified sweep profile matching closely the original shape, still making an acceptable 3D window.

Built-in Detail Levels

You are certainly well aware of the built-in Level of Detail functionality that you hook up in the Family Editor to determine when an object is visible or not:



Connecting the Visibility of an object to the Detail Level of a View



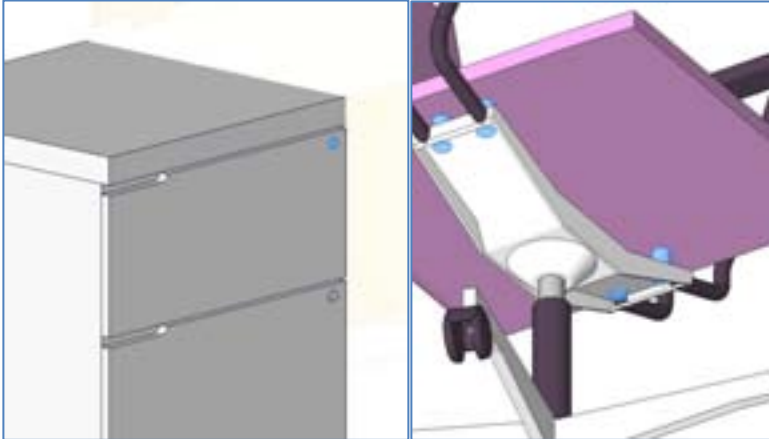
A family displaying less/more detail based on the Detail Level of a View

Along the same lines, it is a good idea to create families that will be repeated many times in a model (like a curtain wall mullion) taking advantage of this:

More flexibility with “custom” Detail Levels

Here is an idea: what if you were able to create as many detail levels for your families? By leveraging the principles behind subcategories, you have total flexibility with the amount of details you can put in (or hide from) a view.

For example, I came across a file that was designed with a fair amount of details for office workstations:



Small details on chairs and cabinets created as extruded circles could be placed on their own subcategory

Unfortunately, those objects were all classified as “3D Elements” as subcategories. As a result, it’s an all or nothing situation for display and export to 3ds Max.

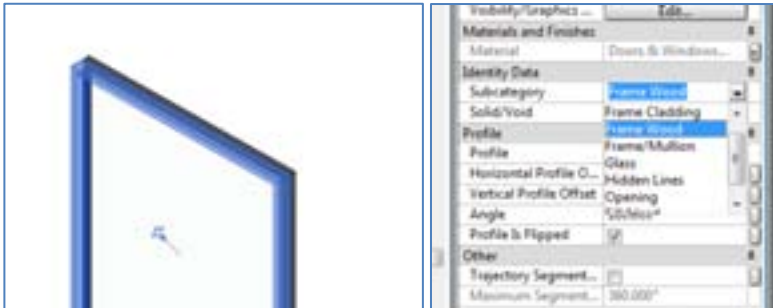
If those elements were assigned a different subcategory such as “3D Elements – Details”, we would be able to turn them off in our View and accelerate the File IO process with 3ds Max.

Take advantage of subcategories with DWG export

Revit DWG Export supports Subcategories

The Revit DWG exporter allows customizing on which layer exported entities will land. Furthermore, you can also define this at the subcategory level. Therefore, you can then leverage this flexibility if you import data in 3ds Max via DWG.

In the family editor, try to classify objects by subcategories:



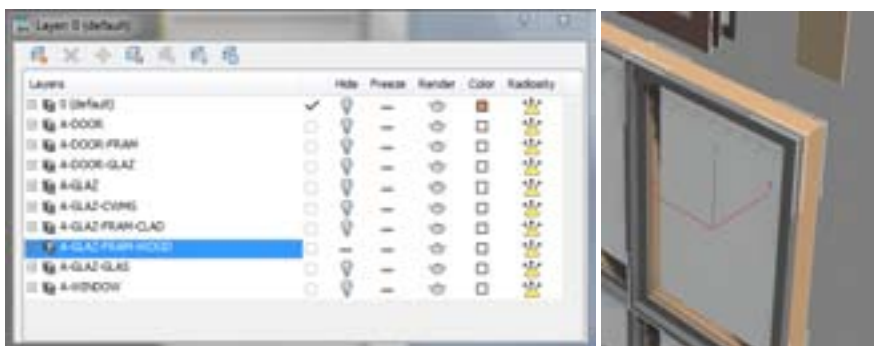
A window frame split into a Wood Frame Subcategory and a Cladding Subcategory

In the Revit DWG exporter, give those subcategories their own layer name:

Walls/Retaining	SITE-WALL	2
Windows	A-WINDOW	5
Elevation Swing	(A-WINDOW)	5
Frame Cladding	AGLAZ-FRAM-CLAD	5
Frame Wood	A-GLAZ-FRAM-WOOD	5
Frame/Mullion	AGLAZ-FRAM-WOOD	5
Glass	AGLAZ-GLAS	5
Hidden Lines	(A-WINDOW)	5

The DWG exporter has been set to export those two Subcategories on dedicated AutoCAD layers.

In 3ds Max, the entities will land on the same layers:



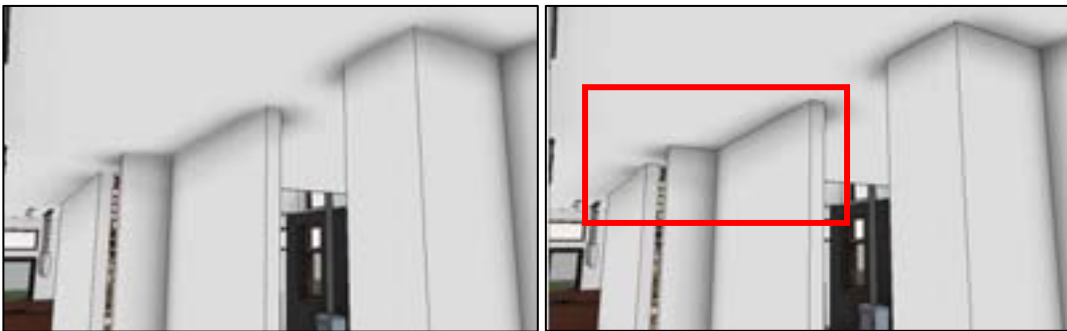
Families end up being split across several layers, as specified in the DWG export dialog of Revit.

Join geometry when possible

Joined geometry provides clean edges between intersecting objects. This also gives hints to the rendering engines of max, shaders and global illumination algorithms about where object “ends”.



Join geometry is a great preparation tool for making render friendly models



In Revit, the ceiling on the left is not joined with the walls. On the right they are joined together.

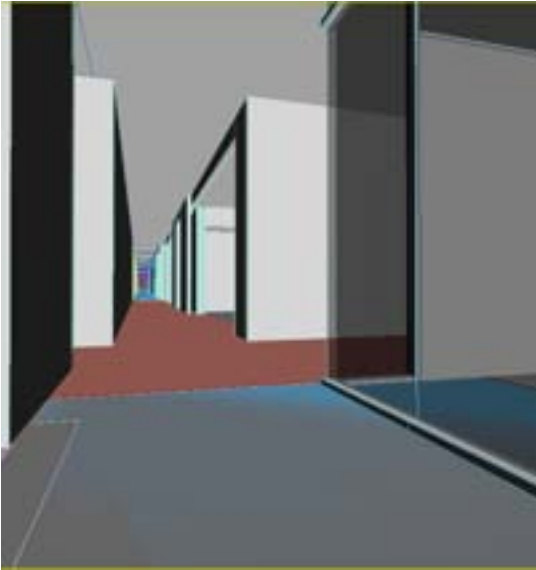
As a result, some rendering algorithms misses the edges that you are looking for in 3ds Max

If you intend to render in 3ds Max with contour shading you may want to consider getting clean, joined, geometry. The same goes for mental ray global illumination as it tend to leak light from outside. On the other hand, render engines like *NVIDIA iray* remains unaffected by geometry that is not joined.

Dealing with overlapping geometry in 3ds Max

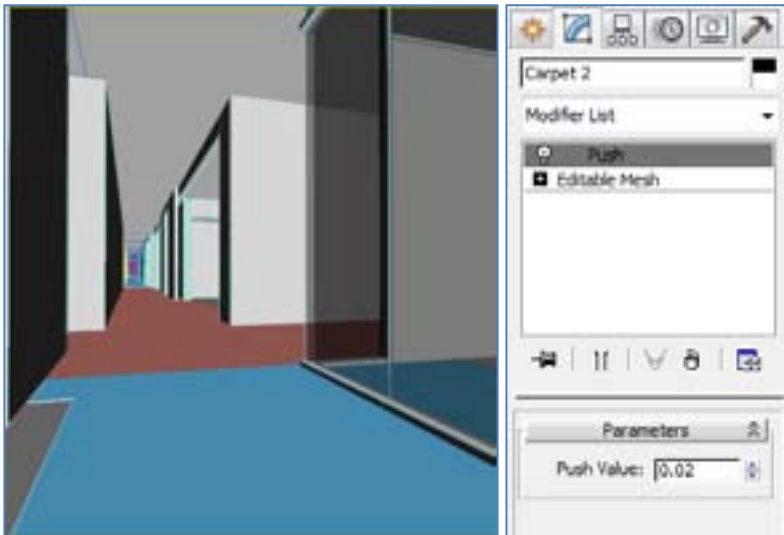
There are cases where objects overlap each other and you can't fix it because they are tied into a workset that you can't touch for any given reason. This is typical for Revit models that contain floor objects for finishes such as carpet on the *same* level as the concrete slab. Normally, an offset would need to be defined but this is often missed.

As a result, you get coplanar objects that produce display and render artifacts in 3ds Max:



Coplanar objects often show flickering effects like this in the display.

One quick method to “fix” the problem is to apply a Push modifier on the object, to push the polygons out slightly, just enough to resolve the problem about coplanar geometry. Perfect for carpets and other thin elements:



A Push modifier on the offending object will push the polygons out a bit, resolving the coplanar issues

Materials & Finishes

Paint materials on edges

Holes cut in walls will leave a default material assigned to the inner edges. This will be a problem inside 3ds Max, depending on the *Combine by* option you use with FBX.



Edges/Borders around windows are assigned to a default material.

I recommend using the *Paint Material* tool to apply a finish that matches what you want upfront, rather than waiting to fit it in 3ds Max:

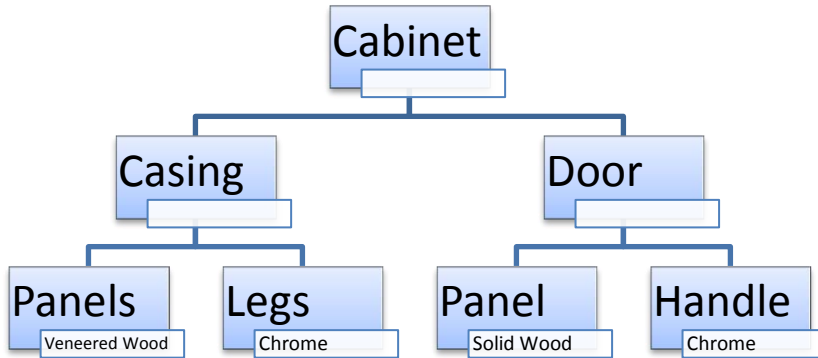


Edges of walls painted with the appropriate Material in Revit

Materials in Families

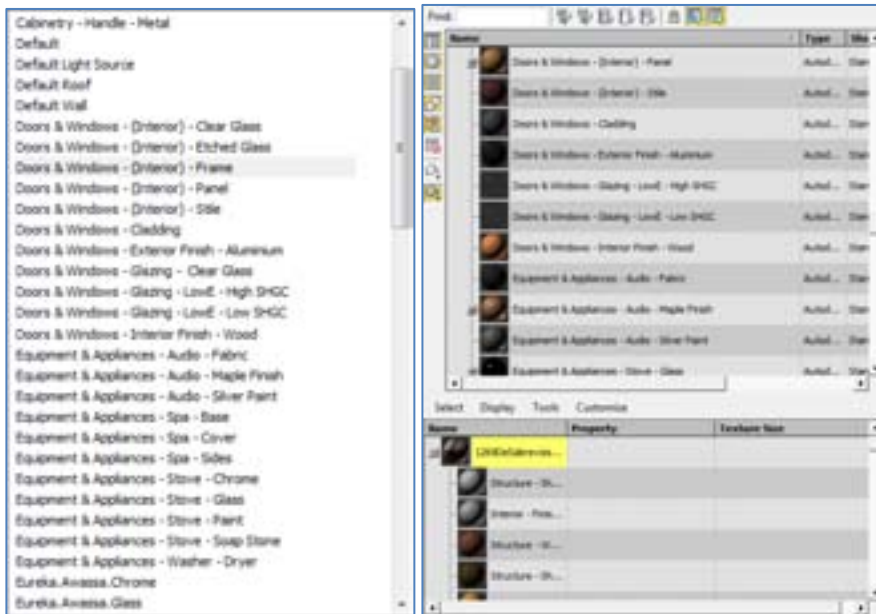
As we previously seen, a lot can be done in 3ds Max when family elements have been assigned to subcategories or materials. This is made possible by the different *Combine By* options that FBX and DWG importers of 3ds Max offer.

As a general strategy, I recommend to assign family components a dedicated material that will be retrieved later in 3ds Max. I suggest avoiding leaving them as “By Category” unless you adopted a workflow where each subcategory is properly defined.



Assign a dedicated material to all the components of a family, whether it is explicitly assigned to the object, or defined By Category/Subcategory

This is where adopting a naming convention becomes useful: it avoids confusion. It does not matter what convention you adopt, as long as you get “something”.

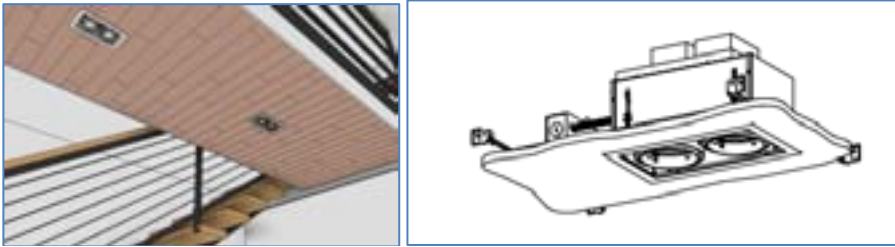


Revit 2012 material names will carry over 3ds Max via FBX

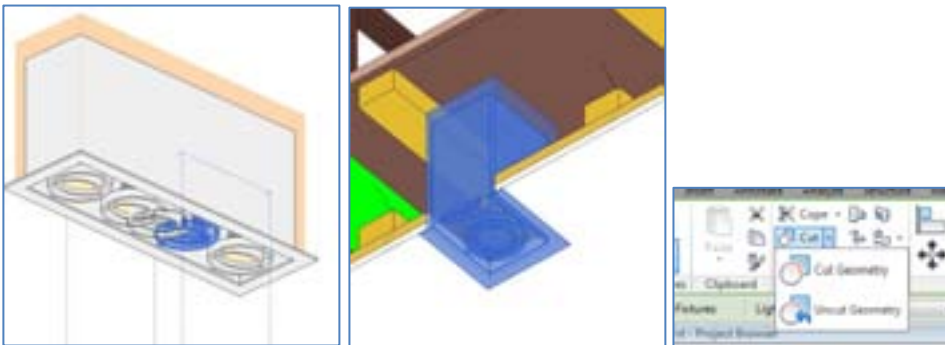
Light fixtures

Recessed lights cuts the ceiling

If you insert recessed fixtures in your model, I recommend using a void to appropriately cut the ceilings when they are inserted.



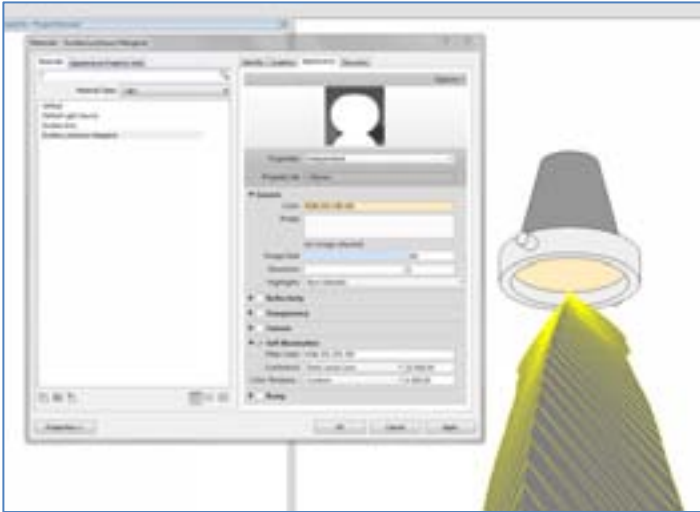
Recessed fixture need to cut holes in ceilings to look right in renderings.



A void object in the family solves the issue. The Cut tool will let you cut the ceiling.

Lights glow in 3ds Max

To prepare lights as glowing objects in 3ds Max, make a Glowing Material and apply it to a fake lamp surface with the Paint tool. This material will carry over FBX and appear as glowing in the final rendered image.



A glowing material applied in the family with the paint tool to a single surface.



The rendered image in 3ds Max, using NVIDIA Iray renderer.

More Tips and Tricks

Site location / Coordinates

It is possible to transfer to 3ds Max, via FBX, information about the project location as well as time & date for Daylight studies. Let's see how:

Sun settings

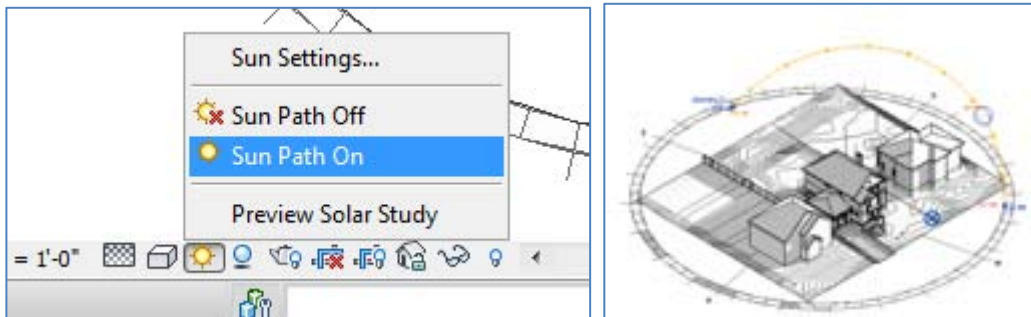
In Revit, Sun Settings are view based. That is, each view can have its own Sun Settings. Therefore, the view that will be exported to FBX will

The Sun Path tool of Revit has useful (but limited) abilities to perform shadow studies. In Revit, you can perform the following tasks:

- Inspect shadows from the Sun in real time in the 3D environment, for outdoor and indoor situations.
- Perform an animation of the Sun over the course of a day or a year.
- Save presets for the date and time (winter, summer etc.)

However, keep in mind that Revit also has the following limitations:

- Revit do not handle quantification of light (unless using an add-on pack, which is also limited to exterior lighting).
- Revit do not perform calculation from the sky dome (unless using the rendering functionality)



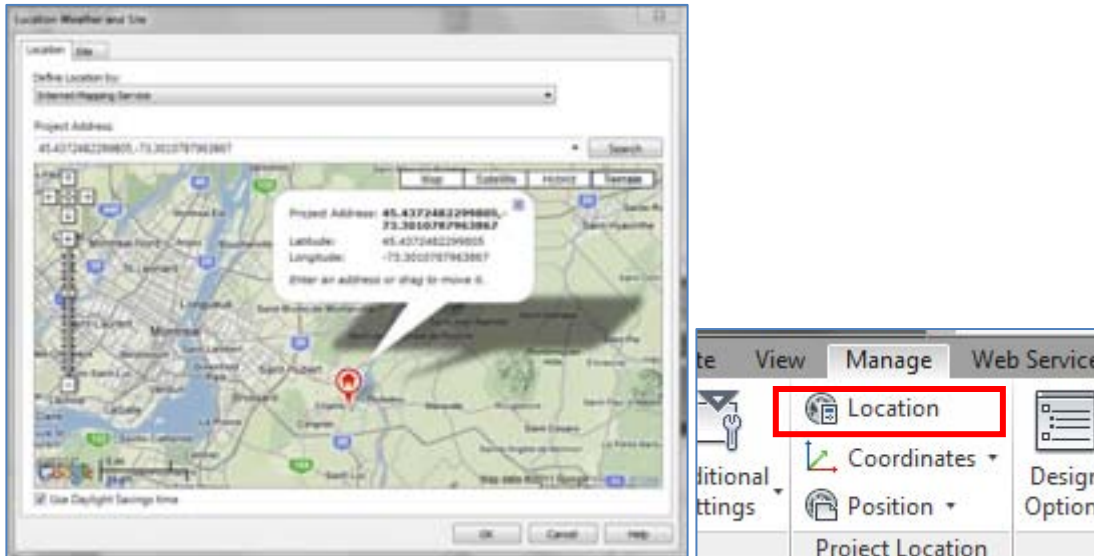
The Sun settings are available in the lower part of the Revit interface. Each view can have its own Sun Settings.

Prior to export to 3ds Max,

Project Location & North Direction

Displaying the Project Location

Revit uses Google Map services to let you find the exact project location. This also feeds in the algorithms used to calculate the Sun angle.



The Project Location will translate to 3ds Max via FBX.

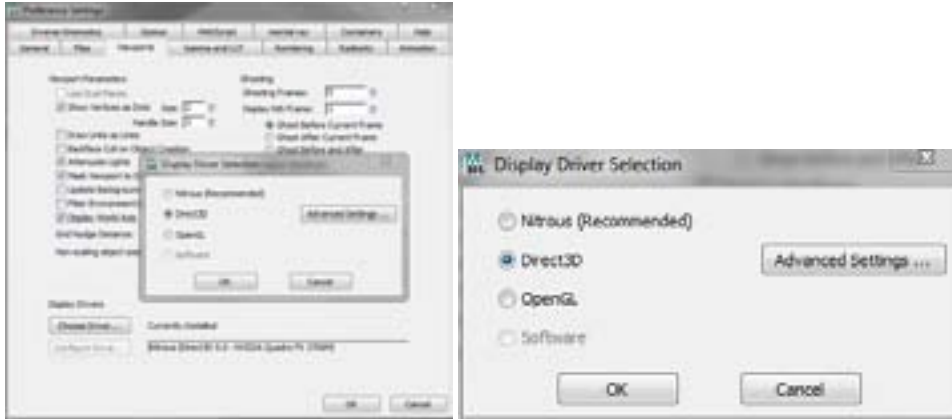
Note that the Use Daylight Savings time is in my opinion misplaced. You need to keep that in mind when doing shadow studies.

Nitrous display driver issue with DWG data imported in 3ds Max

3ds Max 2012 shipped with a new display driver called “Nitrous”. This is the default driver. Unfortunately, entities created by the DWG import plugin have a display bug with this driver and disappear.

There are two (2) solutions to this problem, which are explained below:

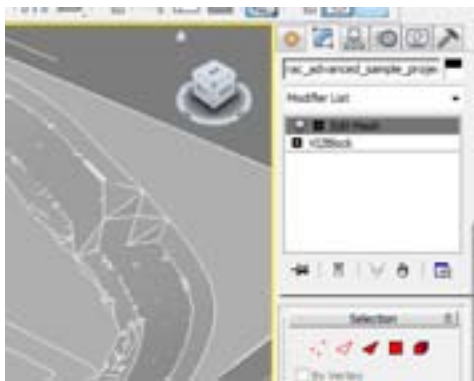
(1) Revert to the DirectX display driver



Reverting to the Direct X display driver is done from the Customize | Preferences... | Viewports dialog

(2) Apply an Edit Mesh modifier on VIZBlocks

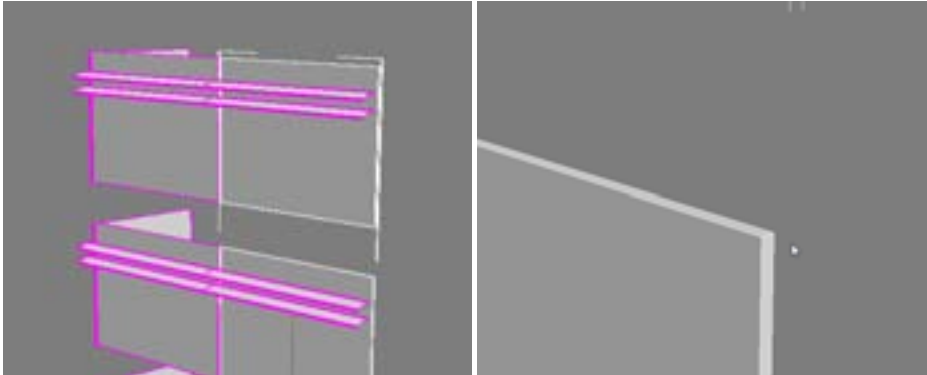
VIZBlocks are unsupported by Nitrous in the moment, due to simple bugs. To work around this problem, apply an Edit Mesh modifier on them. This will “fool” the Nitrous display driver making it think that VIZBlocks are now regular Editable Mesh objects, and will display correctly in Nitrous



Note: I provide a script to automate this process in the section entitled Helpful scripts to help working with Revit Data *below*.

Save render time by simplifying glazing

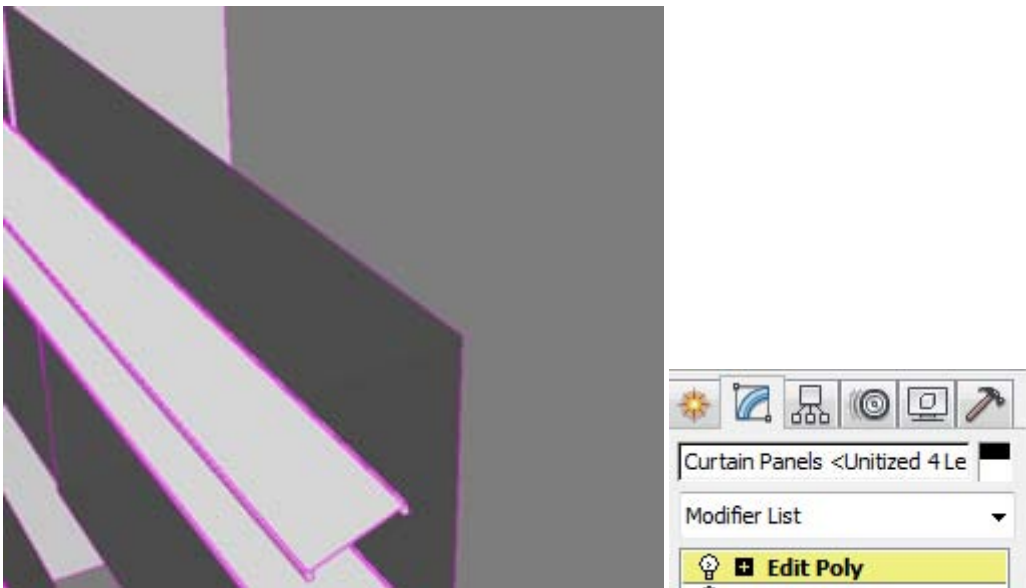
Glazing can be time consuming for ray tracers. One method to improve render times is to simplify glazing in a way that they have a single flat polygon instead of representing glass with thickness.



A curtain wall glazing panel is solid geometry

Delete polygons

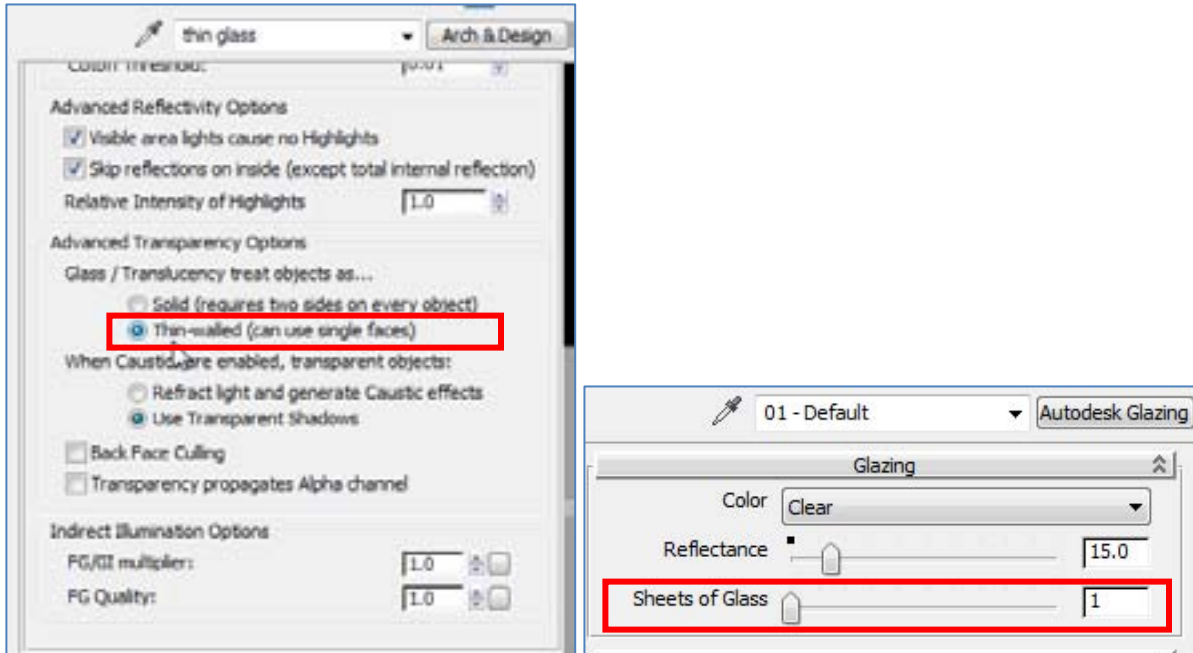
The main idea is to delete 5 polygons out of 6 and leave only a thin surface. This is achieved with an *Edit Poly* modifier.



The flattened glazing panel: use an Edit Poly modifier and delete the unwanted polygons.

Compensate the Glazing material

When a glazing pane has been turned into a flat polygon, you need to change its material to take it into consideration; otherwise ray tracing engines will get "confused" with refraction.

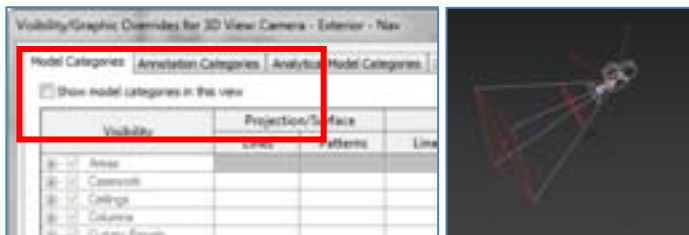


An A&D Material set to Thin-Walled and an Autodesk Glazing Material set to use a single "Sheet of Glass"

Exporting multiple cameras from Revit

Don't look for it; it's not possible to export all Camera Views into a single FBX file in the moment. However, there is a workaround:

1. Make a Camera in Revit
2. In the Visibility Graphics/Override, turn off all objects
3. Export to FBX
4. The FBX file will contain only that camera....



Nothing prevents you from exporting a FBX file with no geometry in it... This is a quick way to get only a Camera in 3ds Max!

Helpful scripts to help working with Revit Data

Note: The following scripts are provided as-is and are not supported or guaranteed by Autodesk. Be sure to back up you data prior to use them.

Fixing Nitrous display driver issues with DWG data imported in 3ds Max

Description

As previously discussed, the Nitrous display driver will not display VIZBlocks unless you apply a Mesh Select modifier on them.

Download

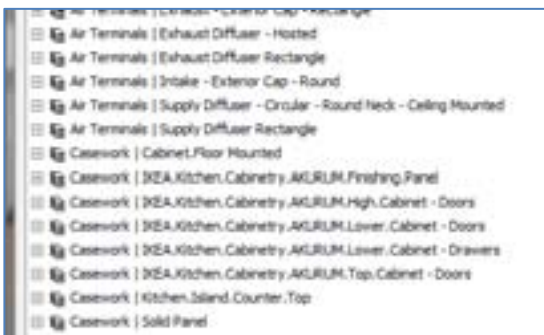
I wrote a script automating this process, download it here: <http://bit.ly/slp7yh>

Simply run the script on freshly imported data from AutoCAD. It will spot any VIZBlocks and apply a Mesh_Select modifier.

Create a Layer per Revit Category

Description

A script that loop through all objects and move them to a layer matching the Revit Category, by extracting meta information that FBX files exported from Revit carry over 3ds Max:



After running the script over a Revit model imported/linked with FBX data

It will leave untouched newly added objects and native 3ds Max objects (or anything missing this Meta data).

It will “fix” parts by assigning them to their Original Category Name, Family Name and Type Name so they also land on the appropriate layer.

Limitations:

- This script will not work when using Combine By Options that destroy meta data, such as Combine By Materials, As One Object or else.
- If you run this script on an existing model from which you had cleaned up layers already, you may lose your original scene organization.

Download location

Download it from here: <http://bit.ly/vp8miH>

Scene Explorer Customization for Revit Data

Description

It is generally useful to list objects in the Scene Explorer with a bit more information that what is available out of the box. I am providing a script that will allow you to add extra columns in the Scene Explorer:

- Material Name
- Material Type
- Object Plugin Type
- Layer Name

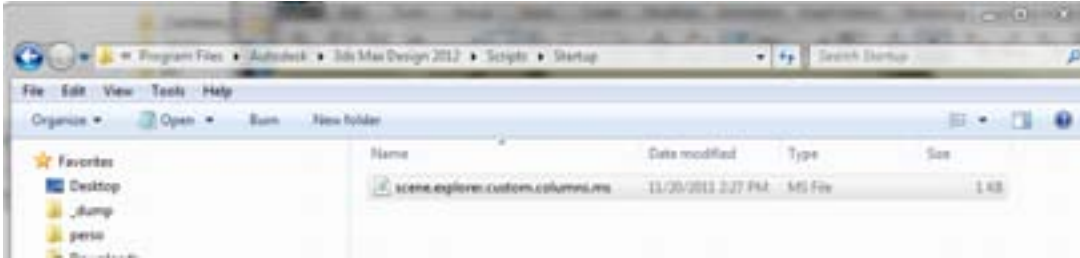


Also, note that it does not save anything in the scene so it will not break file compatibility with other 3ds Max users who have not installed those scripts.

Download the script & install instructions

Download it from here: <http://bit.ly/rA2GqM>

The script has to be placed in the /scripts/startup/ folder of your 3ds Max installation as it needs to be treated as if it was a 3ds Max plugin:



Extracting Revit parameters in 3ds Max

Revit parameters are stored in 3ds Max as “Custom Attributes” on objects and can be inspected with MAXScript. Here are a few examples:

Inspecting properties on the selected object:

```
-- script begin
(
  objCustAttribs = custAttributes.get $ 1 baseobject:false -returns custom
  attributes from the selected object
  showproperties objCustAttribs - lists all custom attributes returned
)
-script end
```

Will return something like:

```
.Category Name : string
.Family_Name : string
.Type_Name : string
.Keynote : string
.volume : string
.Area : string
.Manufacturer : string
.Comments : string
.Type_Comments : string
.description : string
.length : string
.Assembly_Code : string
.Cost : string
.Structural_Usage : string
.Top_Offset : string
.Base_Offset : string
.Base_Constraint : string
.Unconnected_Height : string
.Top_Constraint : string
.Room_Bounding : string
.width : string
.Location_Level : string
.Notes : string
.isUsedForMasonryCalculations : string
.isWindowOpening : string
```

Therefore, invoking:

```
$.Family_Name
```

Will return:

```
"Basic Wall"
```

Fixing Parts Meta Data

Description

The following example is used by the *Create a Layer per Revit Category* (see above) to restore Meta data into Parts which are imported without Categories, Family Names and Family Types.

```
--loop through all scene objects
for obj in Objects do
(
  --get custom attributes from object,
  --as Revit meta data is stored as such on objects in Max
  local objCustAttribs = custAttributes.get obj 1 baseobject:false

  --check if the object has custom attributes
  if (objCustAttribs != undefined) do
  (
    --check if the object has a Type_name property,
    --insuring that the object effectively comes from Revit via FBX
    if (hasProperty objCustAttribs #Type Name) do
    (
      -- Check if the #Type_Name is a Part
      if ((getProperty obj #Type_Name) == "Part") do
      (
        --recopy all original properties into the real properties
        --so the objects land on the right layers later in the same code / script
        obj.Type_Name = obj.Original_Type
        obj.Category_Name = obj.Original_Category
        obj.Family Name = obj.Original Family
      )--end if
    )--end if
  )--end if
) --end for loop
```

Download location

Download it from here: <http://bit.ly/vp8miH>