AUTODESK® REVIT

AUTODESK

Revit IFC Manual 2.0

Introduction

BIM (Building Information Modeling) is a model-based process that allows architects, engineers, clients, and contractors, to procure, design, build and manage buildings and infrastructure. The core of BIM is a digital representation of physical and functional characteristics of an asset. The most important Autodesk tools for the creation and modification of BIM data are Autodesk Revit for buildings and Autodesk Civil 3D for large scale infrastructure.

Provided that all of those involved in the design process are working with the same software tools, collaboration is hassle free and data exchange is straight forward. Collaboration tools for examining data quality like Revit's interoperability tools are integrated in the authoring software, reports may be generated at any time in user defined quality. This process is called **native BIM.**

In larger projects and complex team structures, native BIM can be a challenge, due to a variety of different authoring software tools from different vendors used for the individual design tasks. In order to work on an integrated collaboration across software platforms, Autodesk convened 12 industry leading companies to found the Industry Alliance for Interoperability (IAI) in 1996.[1] The main concept developed by this consortium was IFC (Industry Foundation Classes). The IAI was renamed to buildingSMART in 2005.

Today, Autodesk is member of the buildingSMART Strategic Advisory Council (SAC), "designed to appeal to those leading multinational enterprises who believe that full implementation and adoption of open BIM is strategically important to the built environment sector" and who forward IFC as a common data standard for interoperability." [2]

In addition, Autodesk joined the Open Design Alliance (ODA) in 2020 to fast track Improvements to Interoperability. [3]

IFC is the basis for exchanging data between different applications through **openBIM** workflows for building design, construction, procurement, maintenance, and operation, within project teams and across software applications. According to buildingSMART, IFC "is a standardized, digital description of the built environment, including buildings and civil infrastructure. It is an open, international standard, meant to be vendor-neutral, or agnostic, and usable across a wide range of hardware devices, software platforms, and interfaces for many different use cases." [4]

Since 2005 IFC - version 2x3 - was adopted as an ISO (International Organization for Standardization) standard (ISO 16739:2005). From ISO 16730: 2017 the standard was adopted by CEN (Comité Européen de Normalisation / European Committee for Standardization) and since then IFC has become a European standard, too. As collaboration is the core intent of IFC, buildingSMART developed a certification program for software-products.[5]

Due to the complexity of BIM projects, varying requirements for project design delivery, and differing capabilities between software platforms and vendors, it is essential for AEC practitioners and project teams to understand the basic principles of openBIM workflows, which this manual will discuss. Central to this manual are on the IFC capabilities of **Autodesk Revit**. We also include a chapter on IFC for AutoCAD products, and look to existing and emerging openBIM standards and capabilities for infrastructure projects.

For a current and updated list of useful links included in this document, please visit the Autodesk IFC resources.

1) https://en.wikipedia.org/wiki/BuildingSMART
2) https://www.buildingsmart.org/community/members/strategic/
3) https://adsknews.autodesk.com/news/open-design-alliance-membership
4) https://technical.buildingsmart.org/standards/ifc/
5) https://www.buildingsmart.org/compliance/software-certification/certified-software/

Understanding IFC

IFC (Industry Foundation Classes) is an object-oriented data model developed to describe the physical components of buildings, manufactured products, mechanical/electrical systems, as well as more abstract structural or energy analysis models, cost breakdowns, work and maintenance schedules, etc.

The official documentation by the buildingSMART covers all these aspects including implementation guidelines for software vendors, which is also the reason why it is often hard to understand for engineers and designers who just need to use IFC for data exchange.

When using IFC for data exchange, it is important to be consider which version, which Model View Definition (MVD) and which file format is to be used.

For a successful data exchange in a BIM project, it is essential to follow certain requirements which need to be defined by the client / BIM Manager. It is important to understand that it is not possible to create a universal IFC file for all use-cases, but that it needs to be created according to the specific requirements. These requirements are usually specified in the Employer's Information Requirements (EIR) .

The IFC definitions are regularly updated and developed by buildingSMART International. It is recommended that the design team members at the beginning of each collaboration identify which IFC version is the latest all parties can work with. However, it is always a good choice to use the latest versions wherever possible. Today, the IFC4 format, among other advantages, allows best rendering of complex geometries. The Articles by BIM professionals included in the Appendix of this manual provide an insight into quality management workflows of openBIM projects.

IFC file formats

The IFC data schema is represented in an alphanumeric format and can be stored in different file formats. Following file formats are commonly used and supported by Revit:

.IFC
Standard format, based on STEP (STEP: Standard for the Exchange of Product Model Data) [EN ISO 10303].

.IFCZIP
Compressed (zipped) IFC file with a smaller size; valid import format for most software applications supporting IFC. Can be unzipped to reveal the original .IFC file or also created manually by zipping.

.IFCXML
XML-based representation of IFC data, required by certain calculation software.

.IFCXMLZIP
Compressed equivalent to .IFCZIP. [6]

IFC schema versions

Currently (April 2021) following IFC schema versions are in use:

IFC4: most current development, includes:

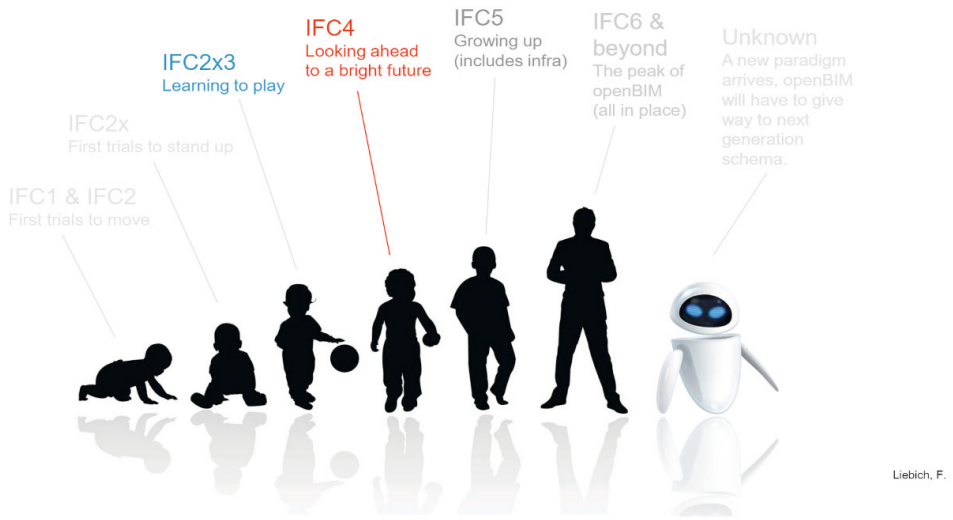
- Major efficiency improvements, better consistency of the schema and significantly smaller file sizes
- Extended definitions for building service elements, structural and analysis models
- GIS coordinate system transformation
- Support for property set templates, multi-language references and integration with the buildingSMART Data Dictionary
- General geometry enhancements (tapering in extrusions, arbitrary sweeps, non-planar surfaces, better tessellation, textures and lighting)
- Support for non-uniform rational b-spline representation (NURBS) in the Design Transfer View
- Point Releases (4.x) already in the pipeline including enhancements and new classes for infrastructure (bridges, railways, roads, ports, and waterways)

Note: Revit is certified for IFC4, however not all software tools fully support **IFC4. IFC2x3** is currently still the most supported and stable format.

6. EN ISO stands for European ISO standard and denotes an ISO standard that was adopted by CEN as European standard.

A full overview of all versions and direct links to the official documentation can be accessed at:

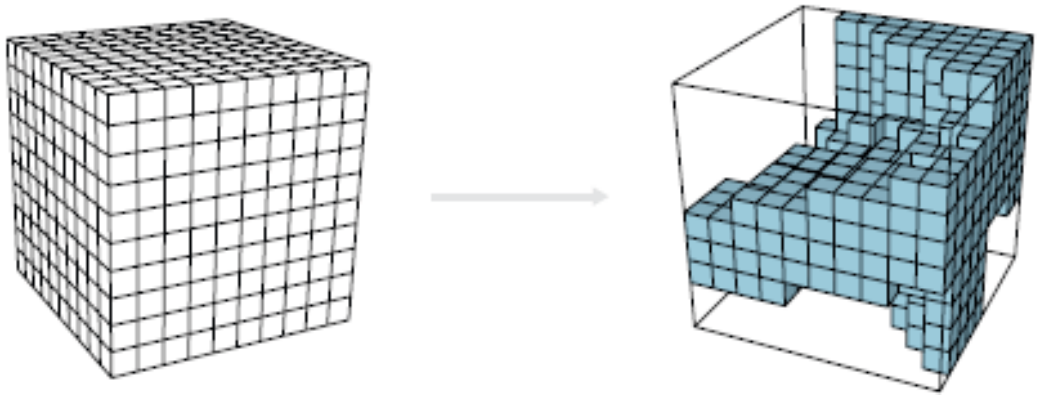
https://technical.buildingsmart.org/standards/ifc/ifc-schema-specifications/



The evolution of IFC (c) Keenliside / Liebich / Grobler

Model View Definitions (MVD)

One of the essential concepts for IFC data exchange are model view definitions (MVD). Model view definitions are data filters precisely defining the graphical and alphanumeric information that must be included in the data exchange. Hence an MVD is a subset of the overall IFC schema.
For instance, thermal simulations require information about openings in a wall and its materials, structural analysis relies on information about the analytical model, while FM systems require only the basic geometry and focus instead on spatial information and specific component features, such as MEP system information, fire protection features, and usable areas.



The IFC schema on the left compared to an MVD as a subset on the right (c) Mark Baldwin, The BIM Manager

The buildingSMART Association is developing the MVDs mentioned along with the IFC schema.[7]

MVDs are used to check whether incoming IFC files meet the data requirements as defined in EIR and BEP. The same applies to the specification of the quality of Revit files to be exported to IFC.

"Due to the large scope, IFC is not implemented in software. IFC is the large set of agreements; an MVD uses entities from IFC to define an exchange standard for a specific use-case or workflow. This exchange standard (MVD) is being implemented by Software Vendors. Because an MVD is being implemented by Software Vendors, MVDs are the base against which the Software Certification takes place. Software implementations are checked against the requirements of an MVD." [8]

Following MVDs are certified by buildingSMART and widely used in all coordination workflows:

Schema	MVD	Description	Revit certifications
IFC4	Reference View	Simplified geometric and relational representation of spatial and physical components to reference model information for design coordination between architectural, structural, and building services (MEP) domains	Architectural Reference Exchange - Export Structural Reference Exchange - Export *In Progress:* *MEP Reference Exchange – Export* *Architectural Reference Exchange - Import*
IFC 2x3	Coordination View 2.0	Spatial and physical components for design coordination between architectural, structural, and building services (MEP) domains	Architecture, Structure, MEP – Export Architecture, Structure, MEP – Import

7. Complete List and status of MVDs developed by the buildingSMART: https://technical.buildingsmart.org/standards/ifc/mvd/mvd-database/

8. https://technical.buildingsmart.org/standards/ifc/mvd/

It is important to note that current IFC Model View Definitions primarily support 3D geometry and property data. For the exchange of 2D information, such as plan views and annotations, it is necessary to use traditional formats such as DWG or PDF.

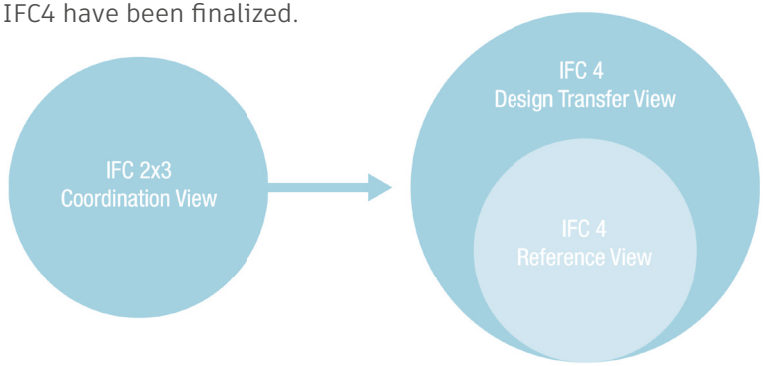
In addition, the intended use case covers only coordination in BIM coordination software, viewers or as reference in BIM modeling software such as Revit. The import of an IFC file for editing purposes is not recommending, not only due to responsibility issues but also to a certain data loss. The IFC schema is based on the STEP format and can not (yet) effectively cover the complexity and the internal dependencies of BIM modeling software.

With IFC 4, the buildingSMART has started the first developments into this direction and is working on a dedicated Design Transfer View, which will allow a better one-way transfer for these purposes:

IFC4	Design Transfer View	Advanced geometric and relational representation of spatial and physical components to enable the transfer of model information from one tool to another. Not a „round-trip“ transfer, but a higher fidelity one-way transfer of data and responsibility.	**Still under development** - not part of the certification process

The content and features of these MVDs are shown in the following graphic – while IFC4 has many new features if compared to IFC2x3, the IFC4 Reference View has a smaller scope than the IFC2x3 Coordination View and is designed for being used for referencing in BIM Software, besides being used in IFC Viewers and coordination software of course. Opening (importing) an IFC4 Reference View in a BIM editor like Revit or using it for other use cases like simulation or analysis will usually lead to less good results.

For these usecases we recommend the IFC2x3 Coordination View, until the IFC4 Design Transfer View as well as the other specialized MVDs for IFC4 have been finalized.



The scope of the IFC2x3 Coordination View compared to the IFC4 Reference View (c) Mark Baldwin, The BIM Manager (based on a visualization by AEC3)

When using the official buildingSMART documentation, it is recommended not to use the main schema documentation, but the dedicated MVD documentation which can be accessed through the following link: https://technical.buildingsmart.org/standards/ifc/mvd/mvd-database/

By doing so you can make sure that you are accessing only the features available in the MVD you are using, while the full documentation may include classes and properties which are not included in the MVD you are using.

Geometric representation in IFC

While BIM and IFC are a lot about data and information, the geometry also often plays an important role. Therefore it is helpful to understand how the geometry is described, as this can influence the file size and the overall performance of the IFC file significantly. The IFC format is based on STEP and solid geometry, which is generated using following methods:

Extrusions
Are the most common and simple graphical method and are used for most of the cases when the shape can be described by a simple profile.

Swept Solids
As the name implies, an element is created with the swept solid method using a sweep. In this case, a profile is swept along a path (direction vector) to generate the solid. This profile may change due to rotation or distortion along the path. Revit uses this method for describing various shapes which cannot be described with extrusions (rebar).

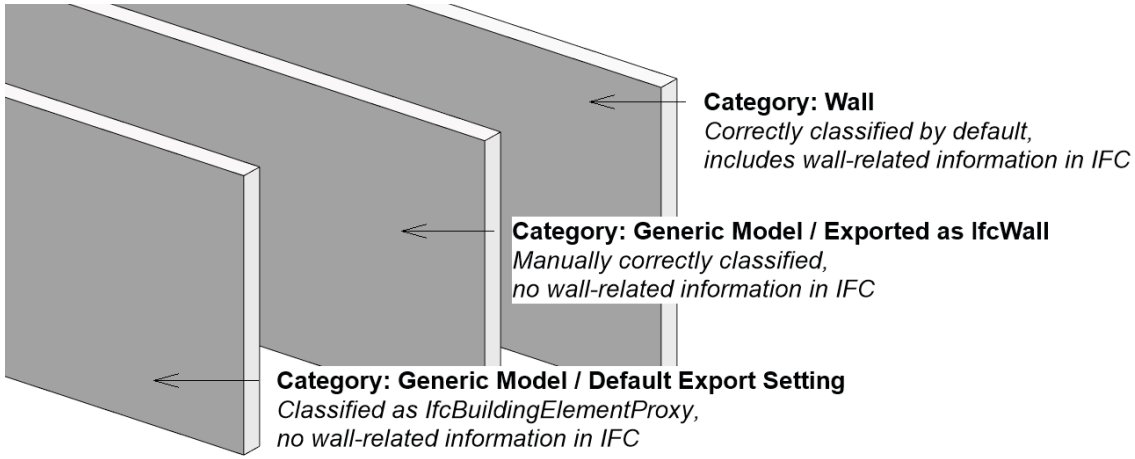
Brep
The method known as boundary representation (B-rep) can also be described as a boundary surface model. A component's individual surfaces are formed by coordinates and together represent the actual solid. Thus, even the most complex shapes can be reproduced geometrically correct. Since B-rep objects require complex calculations to display the individual surfaces, more memory is required.

NURBS (new in IFC4)
IFC4 can describe complex surfaces using NURBS (non-uniform rational B-splines) surfaces. This considerably reduces the requirements for available memory, while the quality of irregular surfaces is increased significantly.

Note: NURBS are not supported by the IFC4 Reference View and will be part of the IFC4 Design Transfer View

IFC Classes

Every object-oriented data schema is based on classes (entities). The IFC schema contains definitions for most physical objects in building (and increasingly also infrastructure) projects, but also for more abstract concepts in the entire lifecycle, like Tasks or Resources.



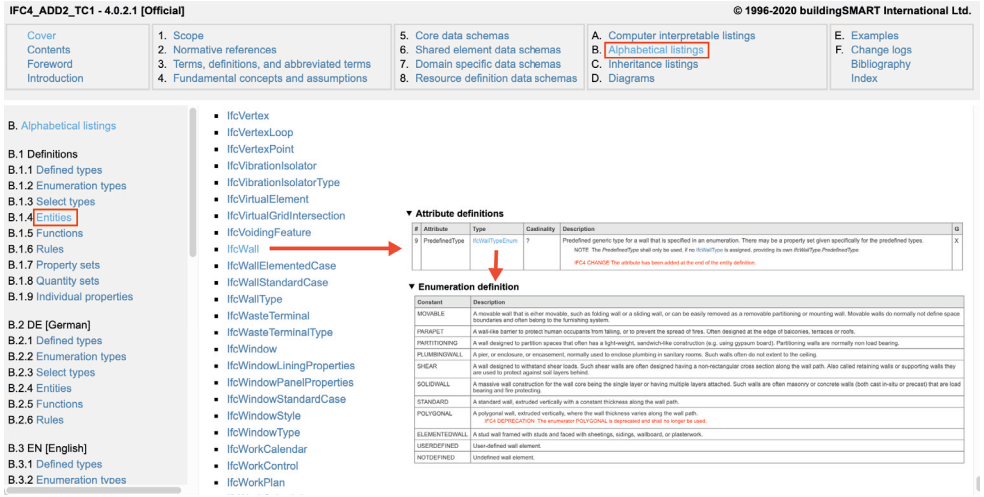
This manual is focused on the parts of the IFC schema most relevant for a Revit user, which are the physical objects.

When it comes to physical objects, the IFC classes are very similar to the Revit categories, as they define the relationships and the properties of every element. If a building element is created using the wrong Revit category and / or exported using the wrong IFC class, it will lack important information. Depending on the classification, each element has defined relationships to other elements and predefined property sets according to the model view definition used.

Revit supports all main IFC classes which are represented in the software itself. A current list can be accessed through the Revit help on AKN. [9]

9. Supported IFC Classes | Revit Products 2022 | Autodesk Knowledge Network8.
https://knowledge.autodesk.com/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2022/ENU/Revit-DocumentPresent/files/GUID-EE6C0CF8-7671-4DCC-B0C7-EEA7513C90A9-htm.html

In addition to classes, the IFC schema allows the distinction of types, which are similar to the subcategories in Revit and provide a further level of classification. The types are documented in the buildingSMART documentation under Type Enumeration and are written in block letters. An IfcWall in IFC4 RV for example can have following types: MOVABLE, PARAPET, PARTITIONING, PLUMBINGWALL, SHEAR, STANDARD, ELEMENTEDWALL, USERDEFINED, NOTDEFINED.



IFC Viewers

Before sharing your IFC file it is inherently important to verify it has been exported correctly. This is typically done in an IFC viewer – linking or opening the IFC file in the software it has been exported from is not recommended for this purpose. There are many free IFC Viewers to choose from:

Autodesk Solutions:
viewer.autodesk.com (free Autodesk viewer) supports 50+ file formats and allows sharing + commenting

Autodesk Docs (included in the AEC Collection) is based on the same technology as the Autodesk Viewer, but offers some extended features for document and project management.

Autodesk Navisworks (included in the AEC Collection) is the desktop coordination solution from Autodesk with extended features like 4D/5D Simulation and clash management. Navisworks uses the Revit IFC engine, which is updated together with the Revit IFC plugin.

Selected third-party IFC Viewers:

Open IFC Viewer developed by Open Design Alliance (ODA), a very fast and advanced IFC viewer, supporting latest IFC versions including IFC 4.3

FZK Viewer developed by Karlsruhe Institute of Technology (KIT), supporting IFC versions including IFC 4.3, mvdXML, GML, LandXML, gbXML, e57, ...

BIMvision developed by Datacomp, supporting IFC versions including IFC 4, is extendable with commercial plugins.

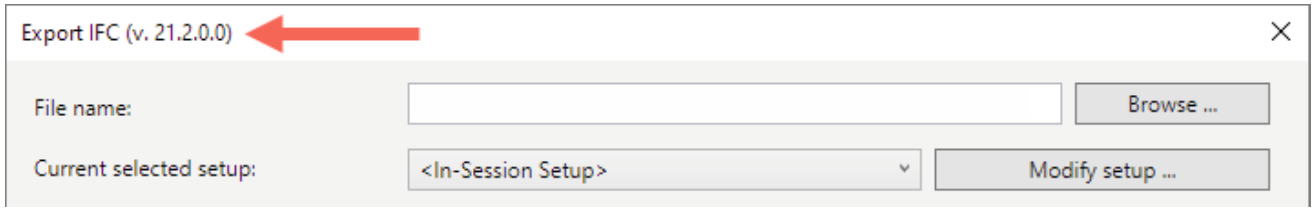
BIMcollab Zoom developed by BIMcollab, supporting IFC versions including IFC 4, a commercial more powerful software is available.

Revit IFC Open Source

Revit comes with an integrated IFC interpreter for reading and writing IFC files; this is a part of an open-source project and is therefore updated independently from Revit. New versions are published in two places:

- Github (installation file and source code): https://github.com/Autodesk/revit-IFC
- Autodesk AppStore (installation file, typically 1-2 weeks after Github): https://apps.autodesk.com/

The currently installed version is displayed in the Export Dialog (Revit > Export > IFC):



No version displayed indicates the original version shipped with Revit.

Important: There is a separate installer for every version of Revit and the installation also updates the interpreter in Navisworks.

The installation updates the current version of Revit IFC and also comes with the additional assets. The most relevant among these are the IFC Shared parameter files which are used to add IFC properties to Revit. These are stored in: C:\ProgramData\Autodesk\ApplicationPlugins\IFC 20xx.bundle

Using IFC Files in Revit

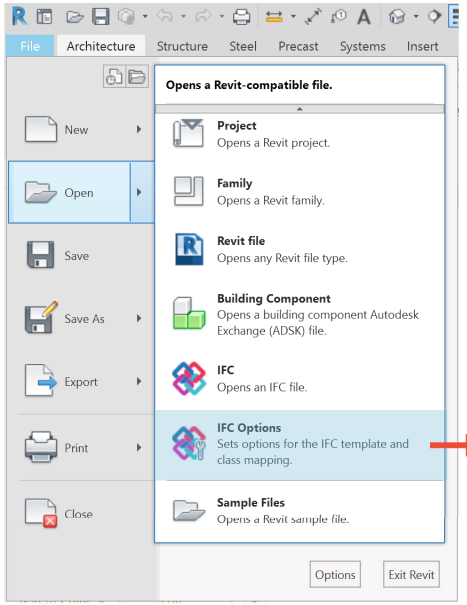
In order to use IFC files in Revit, these can be linked as a reference (recommended) or opened.

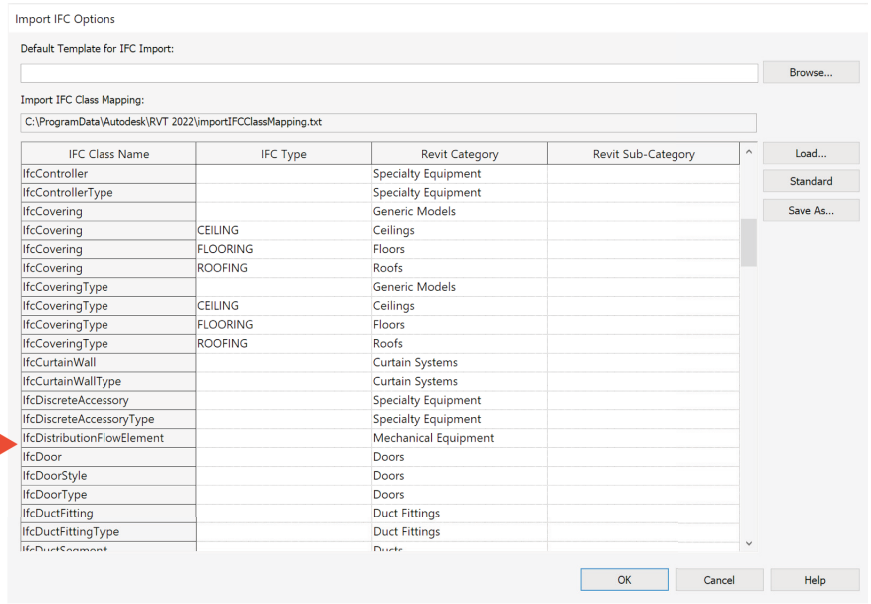
General Settings

The settings found in Revit under File > Open > IFC Options are valid for both opening and linking IFC files:

Default Template for IFC Import (and Link): will use the first template from your list defined in the general Revit options, which is also presented when creating a new project file. It is recommended to select a minimal template for IFC Import / Link to avoid blowing up your file with unnecessary information like views or families. A minimal template can be created from scratch by selecting New > Project > Template: <None> and then saving it as a new IFC template.

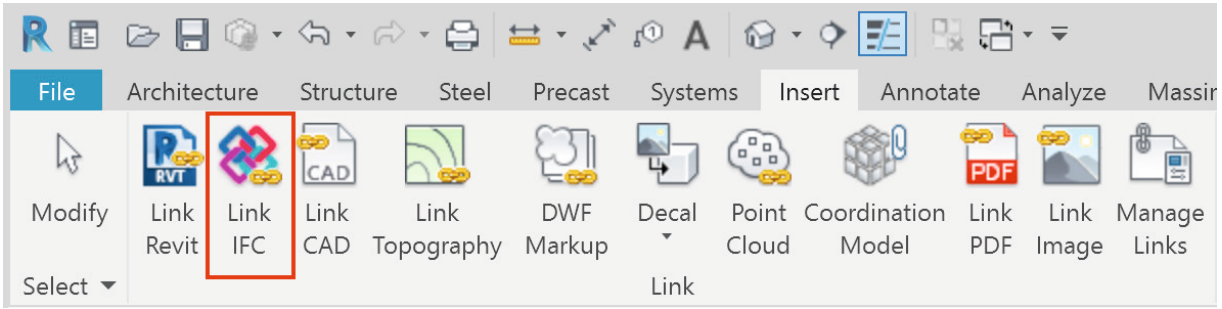
Import IFC Class Mapping is a mapping table very similar to the Export mapping table. It can be directly edited in the dialog or also by opening and editing the referenced text file. This is particularly useful if the default mapping table does not contain a specific IFC class and type already. Classes can also be excluded by entering *Don't import* instead of the Revit Category. It is recommended to exclude classes which are not relevant in Revit for best performance.





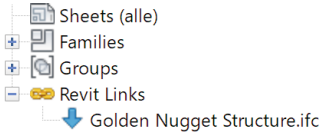
IFC Class Name	IFC Type	Revit Category	Revit Sub-Category
IfcController		Specialty Equipment	
IfcControllerType		Specialty Equipment	
IfcCovering		Generic Models	
IfcCovering	CEILING	Ceilings	
IfcCovering	FLOORING	Floors	
IfcCovering	ROOFING	Roofs	
IfcCoveringType		Generic Models	
IfcCoveringType	CEILING	Ceilings	
IfcCoveringType	FLOORING	Floors	
IfcCoveringType	ROOFING	Roofs	
IfcCurtainWall		Curtain Systems	
IfcCurtainWallType		Curtain Systems	
IfcDiscreteAccessory		Specialty Equipment	
IfcDiscreteAccessoryType		Specialty Equipment	
IfcDistributionFowElement		Mechanical Equipment	
IfcDoor		Doors	
IfcDoorStyle		Doors	
IfcDoorType		Doors	
IfcDuctFitting		Duct Fittings	
IfcDuctFittingType		Duct Fittings	

Link IFC

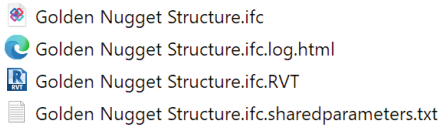


Linking or referencing IFC files in Revit is the preferred and most reliable method for using IFC data in Revit. This method will process the IFC file in the background and display it as a reference. If the linked IFC file is updated, it will be automatically reloaded and updated in Revit the next time the project is opened. Alternatively, it can be updated manually by selecting it in the project browser and clicking *Reload*.

Linking IFC files in Revit creates automatically three files in the same directory:



Revit

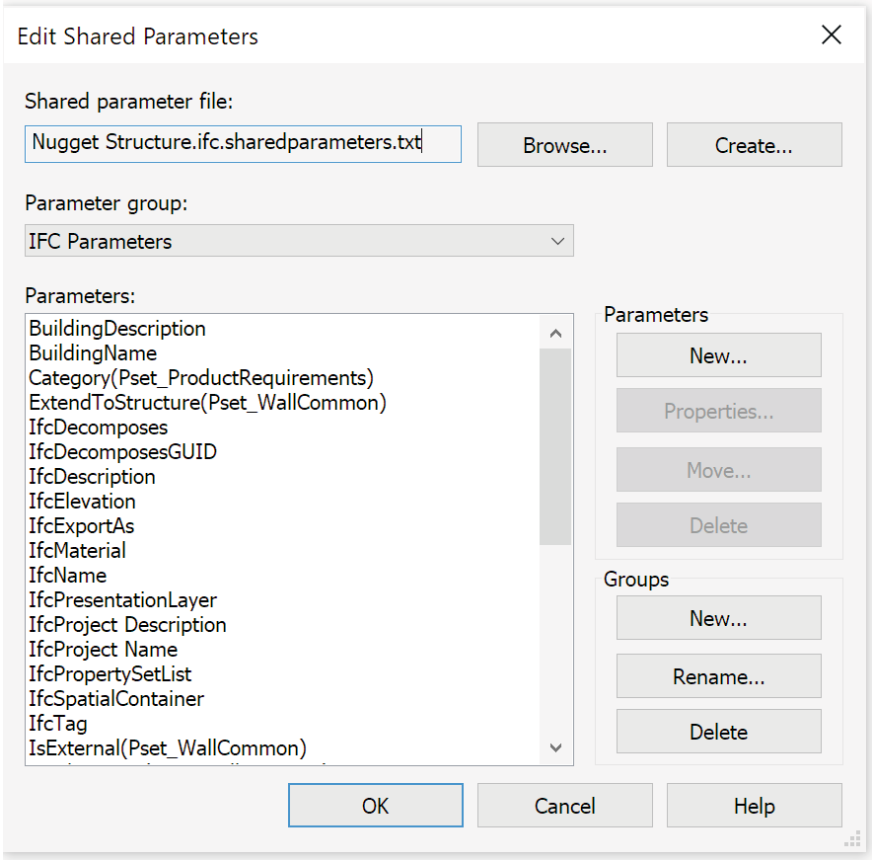


File Folder

* ifc.RVT is used by Revit internally and must not be moved or edited in order to maintain the relationship between the Revit project and the IFC file.

* ifc.log.html which is basically a logfile of the conversion process and contains a report about the linked elements, but also error message and hints which can help with troubleshooting

* ifc.sharedparameters.txt contains the shared IFC parameters found in the IFC. In order to be able to schedule certain parameters contained in the linked IFC file, these can be added to the project from this file.



Open IFC

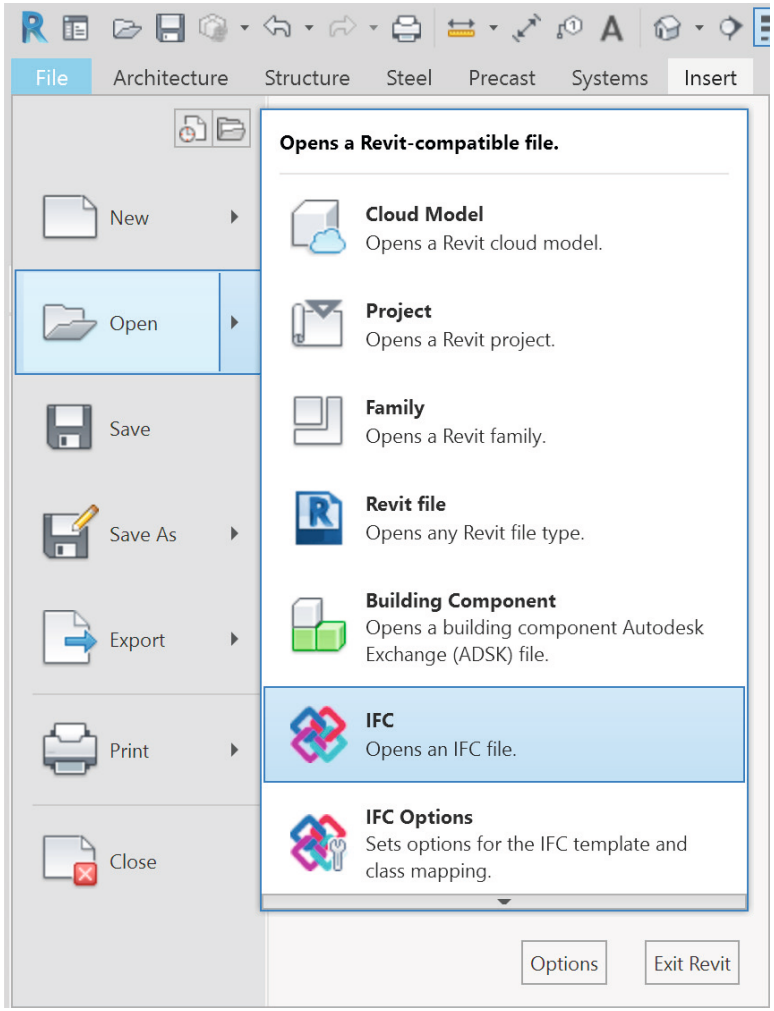
IFC files can also be opened in Revit, which will convert all IFC geometry to native Revit families and make them editable. As introduced at the beginning of this manual, IFC has been developed as a coordination format and it still has limited capabilities when it comes to converting and editing. This is being addressed with the newer concepts like the IFC4 Design Transfer View, however this is still under development by buildingSMART.

In addition, altering IFC data can lead to liability issues.

In certain cases it can happen that importing an IFC file is necessary due to a change of the authoring software. It is important to be aware of the fact that currently this process will lead to a data loss and therefore the imported model needs to be checked for errors or missing elements. The most important factor is however the actual content and quality of the IFC itself, which depends on the export settings.

Following best practices can help when importing IFC files in Revit:

- Check the IFC file in a Viewer and make sure that all elements are classified correctly – if not, request a new IFC file with a correct classification
- Open the IFC file in a text editor an check the header for information around the IFC schema and MVD. IFC2x3 Coordination View 2.0 is currently recommended for best results when opened in Revit.
- Exclude all IFC classes not needed in Revit by inserting DontImport in the mapping table found in IFC Options
- Disable *AutoJoin Elements* and *Correct lines that are slightly off axis in the Open Dialog* to speed up the import process

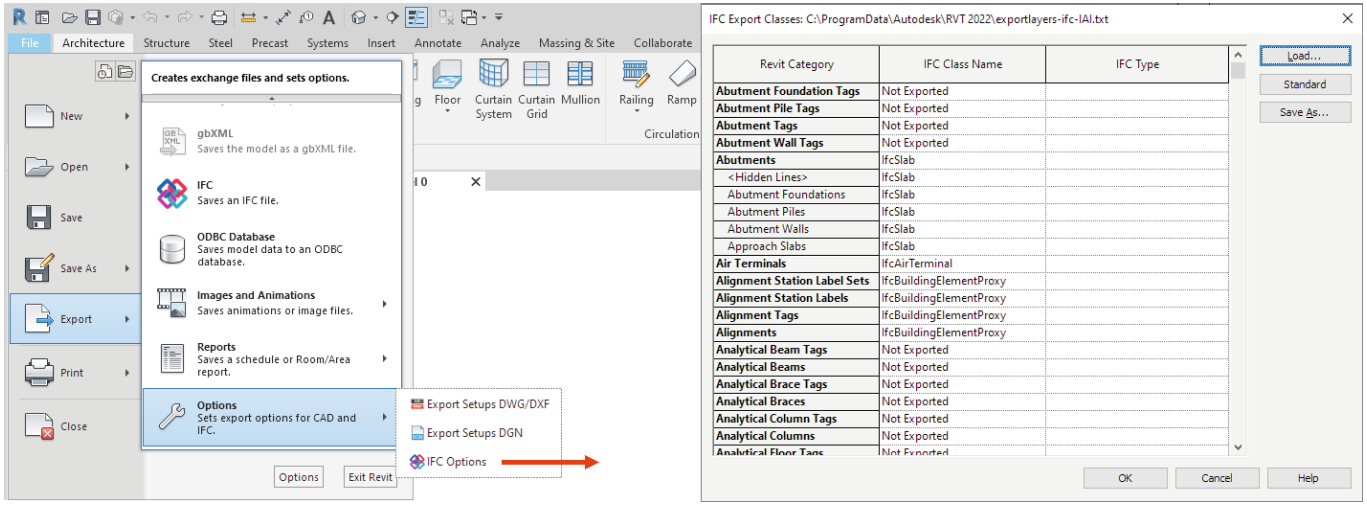


IFC Export from Revit

Default mapping

The most important export setting is a correct mapping of Revit categories to IFC classes.
This is done through a common mapping table, that is usually found as "exportlayers-ifc-IAI.txt" in the "C:\ProgramData\Autodesk\RVT20xx [10] " directory.
To edit / change this mapping table from the Revit UI select the menu item "File -> Export -> Options -> IFC Options":

Note: Overriding of Revit Subcategories as well as the IFC types is limited at this level – only the main Revit categories should be mapped to IFC classes. For more granular mapping, the elements can be mapped individually. Replacing the IFC Class Name with *Not Exported* will completely exclude the Revit category from export.



If using Revit in different languages, the "exportlayers-ifc-IAI.txt" will be generated according to the first language the dialog is launched in. To reset the mapping table to the default settings and/or current language, delete the text file (path indicated in the header) and then click "Standard" in the dialog above– this will recreate the mapping file with the hardcoded settings.

It is recommended to save your own settings in a separate file.

10. 20xx stands for the Revit version used.

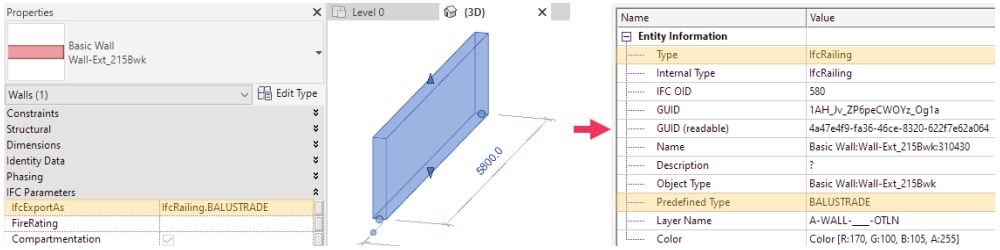
Individual mapping

There are many cases where the global mapping discussed previously needs to be overridden on element basis, as the IFC classes are often more granular than Revit categories and also have their own predefined types.

The element-based mapping is achieved by assigning values to the parameter IfcExportAs. It is strongly recommended to add this parameter as a shared parameter to your project by using the shared parameter files included with Revit IFC.

The value of this parameter needs to be IfcClass.TYPE – both of these are defined in the IFC schema. Similiar to the main mapping table, *Don't export* can be used to exclude a particular element from export.

It is also possible to map Revit categories to classes they were not originally intended for, however keep in mind that only the information available in Revit can be exported. In our example this means, the mapping of a wall to IfcRailing with the predefined type BALUSTRADE works well:



However, if compared to a regular railing not all the custom properties automatically mapped when exporting from Revit are available for the overridden railing and would need to be added manually:

Revit Railing exported as IfcRailing

Pset_RailingCommon	
Height	900 [mm]
IsExternal	FALSE
Reference	900mm

Revit Wall exported as IfcRailing

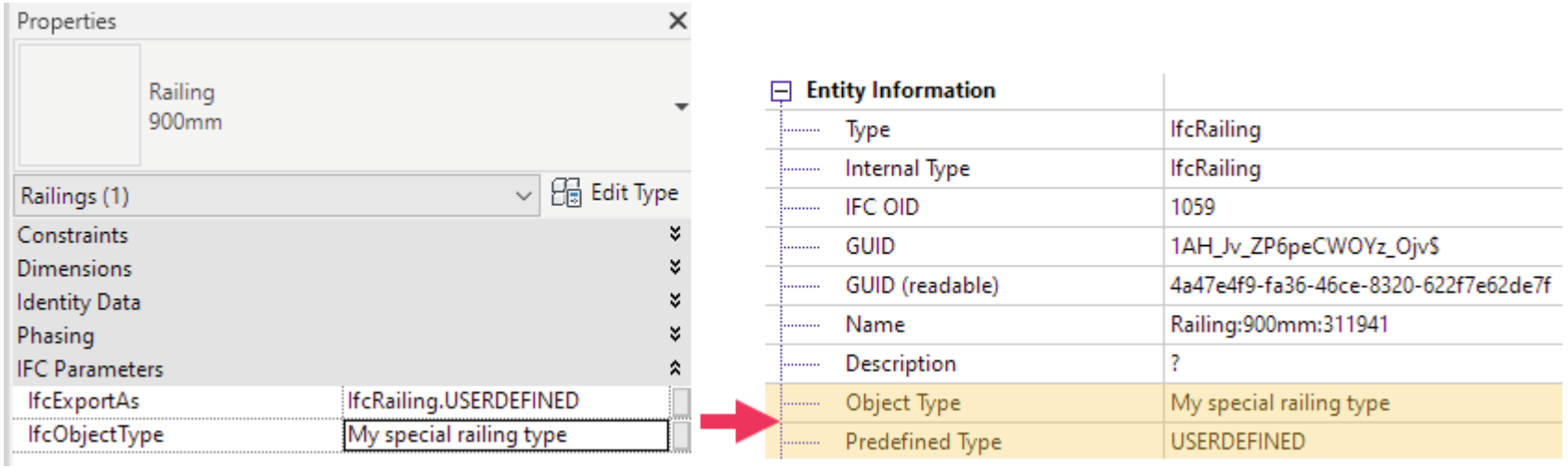
Pset_RailingCommon	
IsExternal	TRUE
Reference	Wall-Ext_215Bwk

Note: there are some restrictions to the mapping of more complex system families like curtain walls to other IFC classes. A link to an overview of all restrictions and possible mappings will be provided on Autodesk IFC resources.

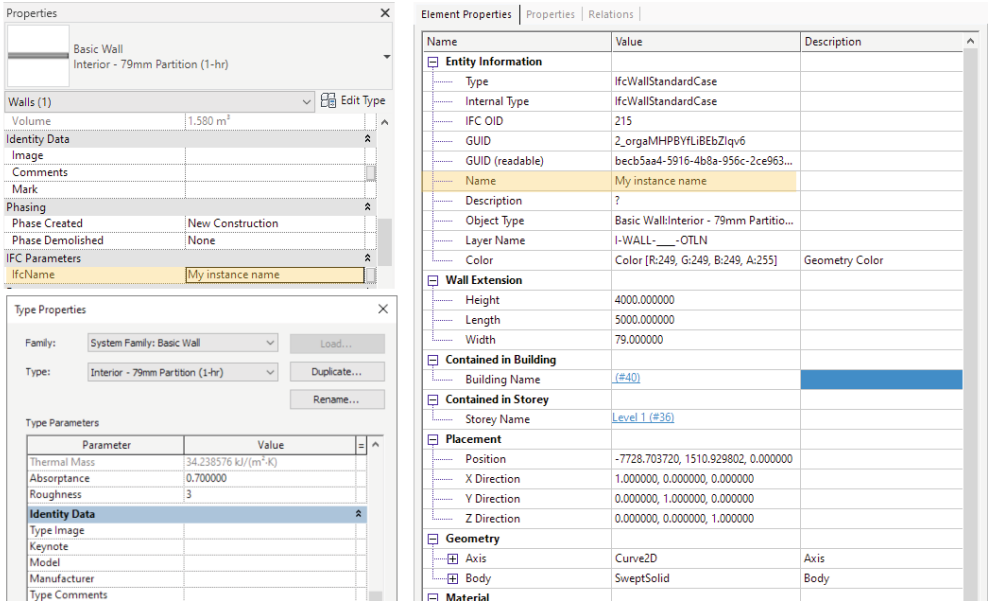
The IFC Schema also allows USERDEFINED types. The correct use of these is achieved by adding USERDEFINED as type and then specifying the type with the parameter IfcObjectType. This is an overview of types defined for IfcRailing as defined in the IFC 4 documentation:

Constant	Description
HANDRAIL	A type of railing designed to serve as an optional structural support for loads applied by human occupants (at hand height). Generally located adjacent to ramps and stairs. Generally floor or wall mounted
GUARDRAIL	A type of railing designed to guard human occupants from falling off a stair, ramp or landing where there is a vertical drop at the edge of such floors/landings.
BALUSTRADE	Similar to the definitions of a guardrail except the location is at the edge of a floor, rather then a stair or ramp. Examples are balustrates at roof-tops or balconies.
USERDEFINED	User-defined railing element, a term to identify the user type is given by the attribute IfcRailing.ObjectType.
NOTDEFINED	Undefined railing element, no type information available.

A userdefined type definition in Revit would look like this:

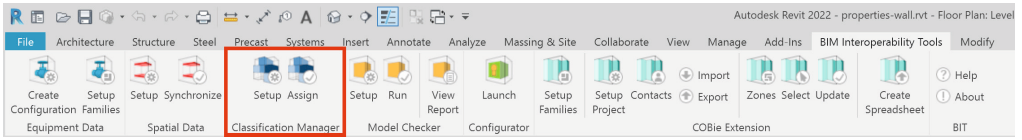


Further it is possible to export user defined Type names for IFC entities. Revit uses a special type-parameter “NameOverride” for changing the type name of a Revit element type. Together with the instance parameter “IfcName”, any naming convention according to project or office standards is possible.



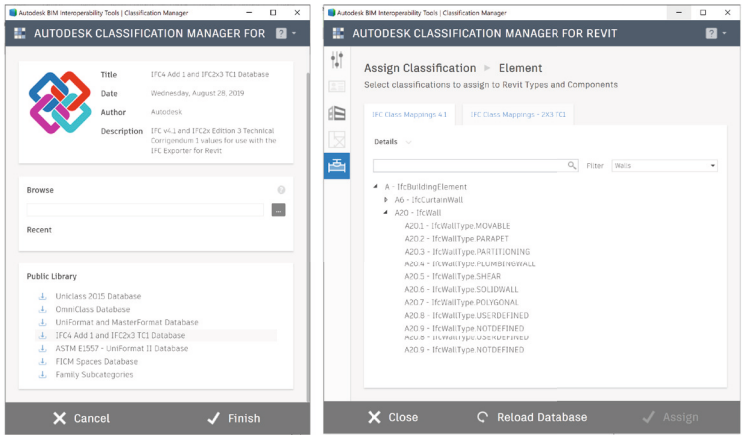
Autodesk Classification Manager for Revit

The Autodesk Interoperability Tools are a free collections of Add-ins available at https://interoperability.autodesk.com.



The classification manager comes with a set of predefined classification tables, among them also IFC2x3 and IFC4. The classification manager can be used to simplify the individual mapping of classes, as its dialog provides a selection list and also supports multiple selections of elements and categories for instance and type parameters.

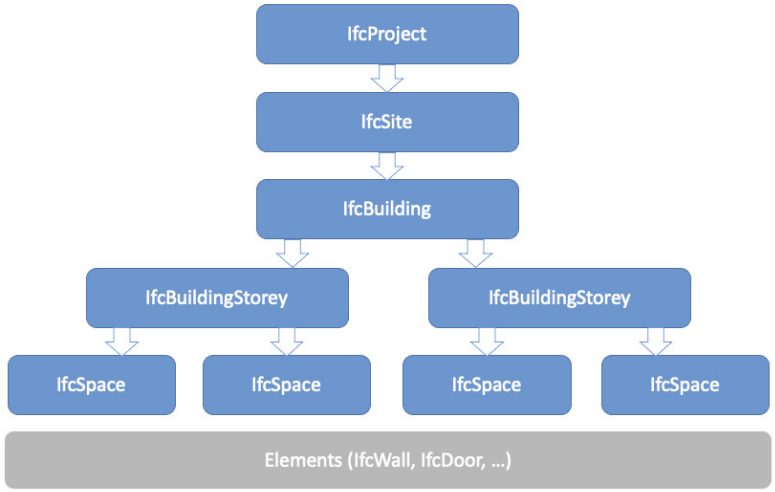
The predefined configuration will also create the IfcExportAs parameter as a type parameter if it is not already existing in the project. The configuration files are available for download in Excel format and also include instructions, so they can be adapted as needed.



Options for the export of IFC Files

Basic IFC structure

The structure of the IFC schema is complex and contains many abstract layers which are not visible to the end user. If we focus on the structure visible to use in the IFC viewers, we will notice the following hierarchy:

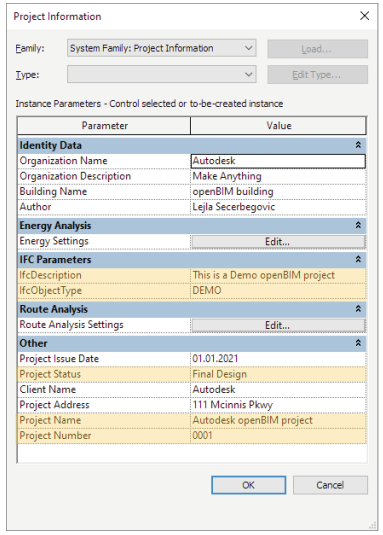


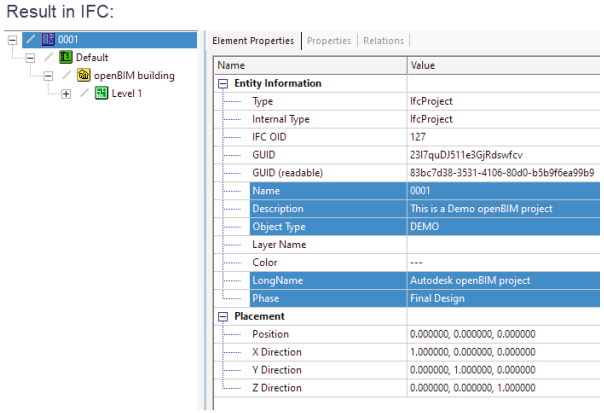
The top 3 entities (IfcProject, IfcSite and IfcBuilding) are represented only once per IFC file. The IFC schema itself allows the existence of multiple buildings per site, however it is not intended to have multiple buildings in one Revit project, therefore Revit can only export one building.

These entities are treated differently from other entities in Revit because they do not have a physical representation in Revit but are derived from the Project Information.

IfcProject

The top level entity is typically the main container in the tree structure of IFC viewers. It doesn't have any Psets defined, and it is not possible to attach custom PSets on this level, but the project has some properties which can be populated:

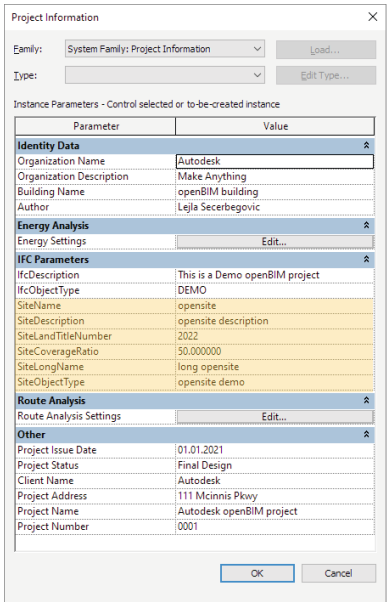




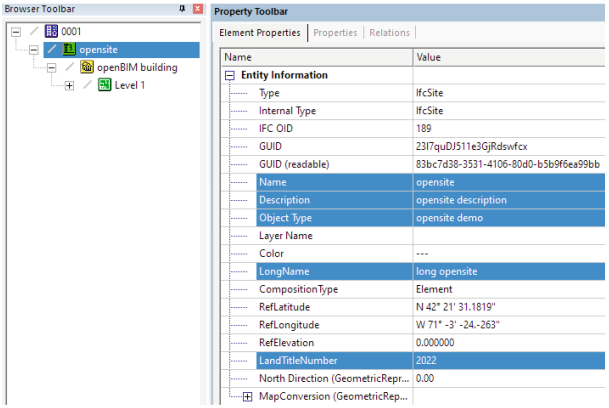
Note: The parameters grouped under IFC Parameters have been added manually and assigned as Instance parameters to the category Project Information.
Layer name / Color can only be relevant for entities which represent physical objects, while IfcProject is only a container and therefore has no physical representation in CAD software.

IfcProject with IfcSite

The second level represents the Site and is a bit more complex from the project, as it can also be associated with a topography object in Revit. In a scenario without topography, the main properties can also be added to Project Information from the shared parameter file (just search for all properties which start with “Site”):

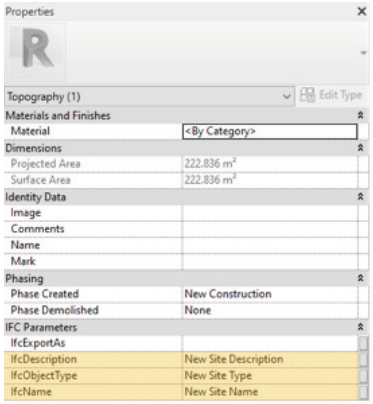


Result in IFC:

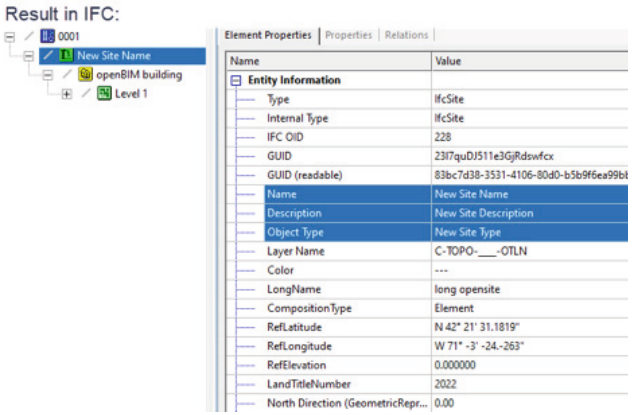


The RefLatitude and RefLongitude are derived from the Location set in the *Manage* tab in Revit.

If the project contains a Topography object, the IFC properties can also be assigned on this level and will override the previously shown properties specified in Project Information.



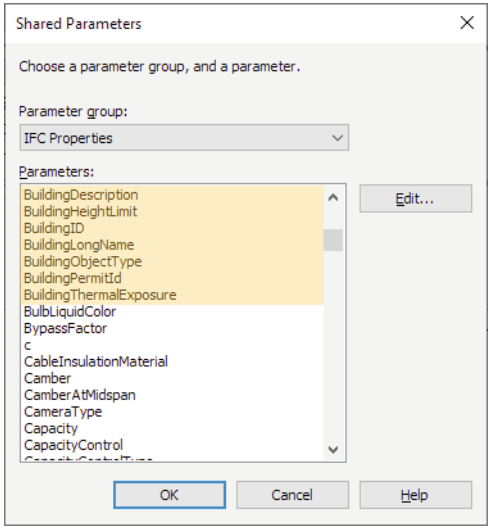
Result in IFC:

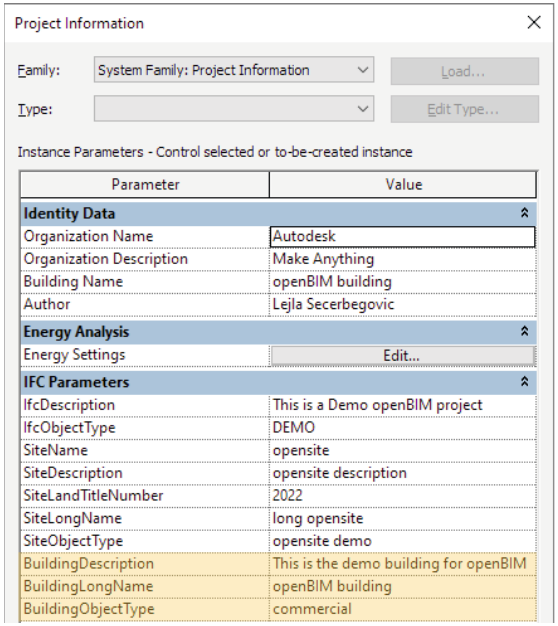


This can be done also with the other properties available, like LongName and LandTitleNumber. According to the IFC 4 RV documentation, IfcSite has two predefined Psets: Pset_SiteCommon and Pset_LandRegistration, which are both supported and included in the shared parameters file. Simply add the properties (either to the Project Information or to the Topography Category) and populate them.

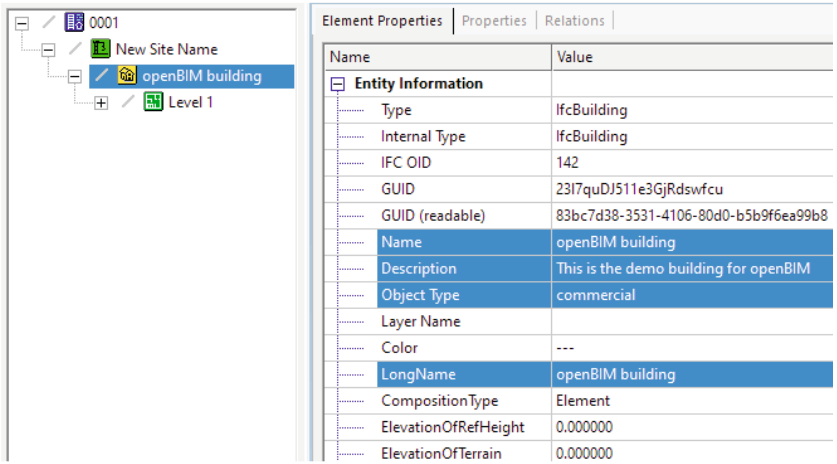
IfcBuilding

The third container is also the first spatial container and represents the building. It is also defined in the project information. You can add further supported properties from the Shared parameters file if you search for properties which start with “Building” and add them to the Project Information Category.





Result in IFC:



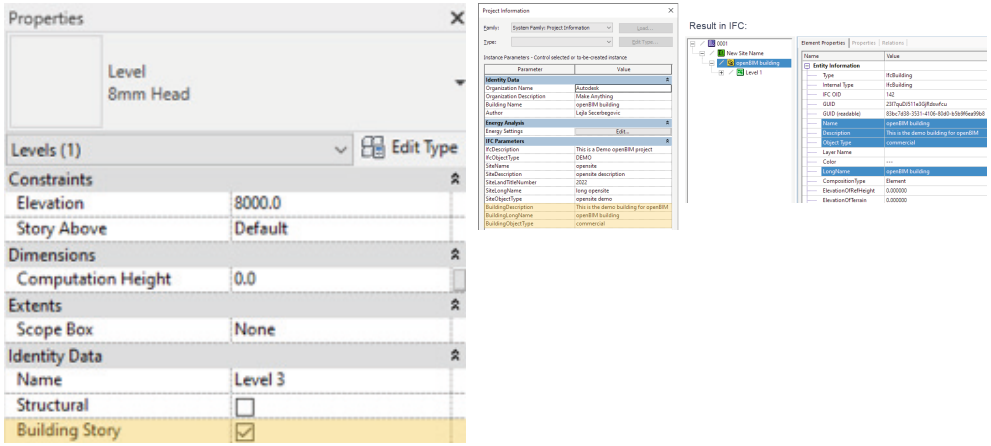
The Psets defined in the IFC schema are also automatically exported, if the properties are added from the shared parameter file and populated.

As already mentioned, the IFC schema supports multiple buildings, however Revit exports only one building per project due to its internal structure.

IfcBuildingStorey

The fourth container equivalent to the actual building stories and hosts the building elements, like Walls or Furniture. Since Revit often has many reference levels which do not represent the building structure, there is the option **Building Story** in the properties of each level which defines whether the level will be exported or not.

If this option is activated, the level will be exported to IFC, if not, it will be ignored. The elements which are in Revit assigned to a non-building story, will be automatically assigned to the next lower building story – if there is no lower building story, they are assigned to the next upper. Every project should have at least one Building Story.



Using IFC Shared Parameters

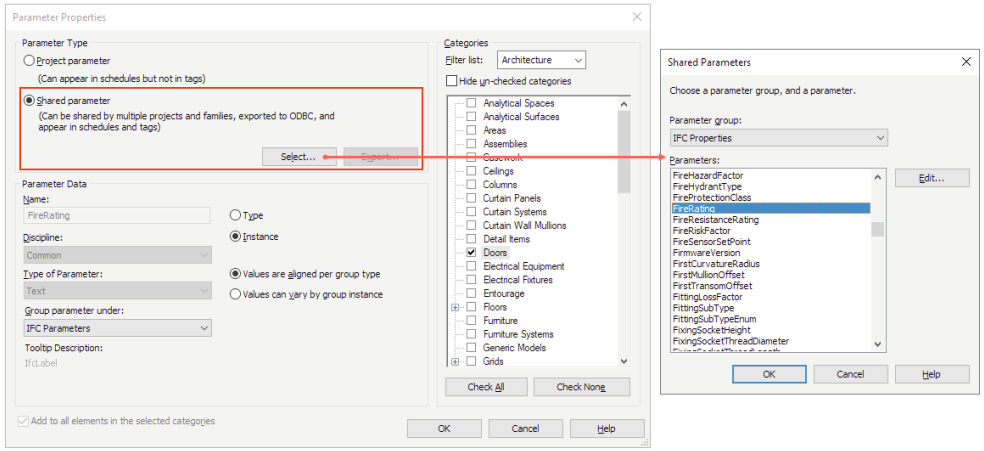
Not all properties defined in the IFC schema are part of Revit by default, as this would overload the projects. It is recommended to add only the parameters needed in a specific project. Frequently used parameters can be added to the project templates.

The Revit IFC Open Source ships with two shared parameter files which are stored in the following folder after the installation:

C:\ProgramData\Autodesk\ApplicationPlugins\IFC <Version>.bundle\Contents\

Alternatively, you can also download them from the Github repository mentioned in the previous chapter.
The two files are:
- IFC Shared Parameters-RevitIFCBuiltIn_ALL.txt
- IFC Shared Parameters-RevitIFCBuiltIn-Type_ALL.txt

The shared parameters are added to Revit using the Dialog found in Manage > Project Parameters and it is recommended to use the first file for adding the Instance parameters and the second one for Type parameters.



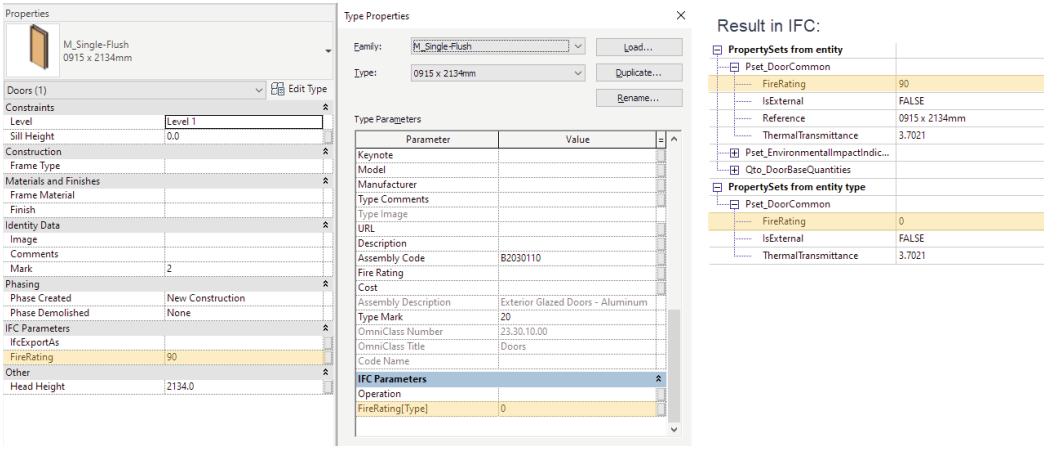
The reason for having two files is as follows: Like Revit, the IFC schema is based on types and instances. In IFC, however, the same parameter can be attached to instances and types (and can also be given different values), while Revit requires that the user choose between type and instance when assigning a parameter – it is not possible to select both.

Depending on the project requirements, you might need to attach certain properties to both IFC Instance and IFC Type levels. To achieve this, you can add the instance properties from the first file and the type properties from the second. The properties from the second file contain [Type] in the name in Revit, which will be removed during export.

To illustrate this, let's assume you need to deliver your doors with the Pset_DoorCommon containing different FireRating on Type and Instance. Steps:

- Add the Instance property according to the previous screenshot from *IFC Shared Parameters-RevitIFCBuiltIn_ALL.txt*, assign it to the Door Category and ideally group it under IFC Parameters (not obligatory but improves the overview).
- Add the Type property from *IFC Shared Parameters-RevitIFCBuiltIn-Type_ALL.txt* and take care to assign the to the Type this time (Instance is always default), select the Door Category and group under IFC Parameters.

The result should look like this:

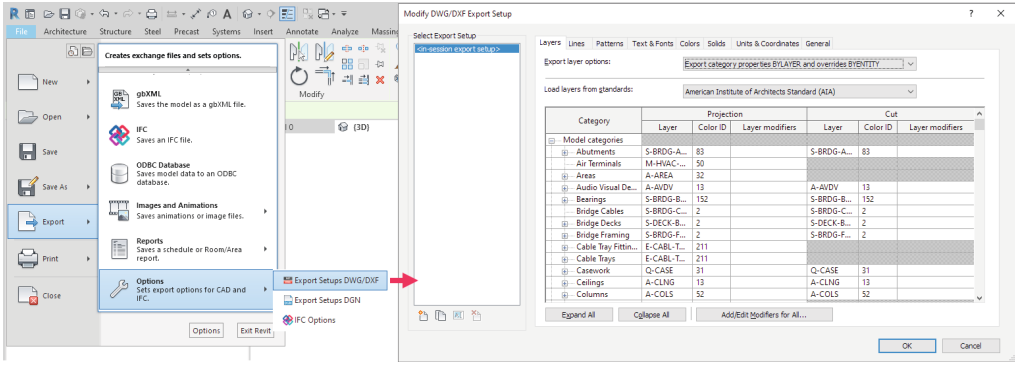


Whether this behavior makes sense for your project highly depends on your requirements, but it is useful to keep in mind that it is possible.

Export for layer-based Software

Some software products may require a layer structure added to the IFC classification. Revit automatically assigns the layer value according the default CAD (.dwg/.dgn) mapping file. The default configuration file is: C:\ProgramData\Autodesk\RVT 20xx\exportlayers-dwg-AIA.txt

The configuration found in this file can be adapted from the Revit UI when selecting Export > Options > Export settings DWG/DXF or manually using the syntax: <Revit Category Name><tab><tab><Layer Name>



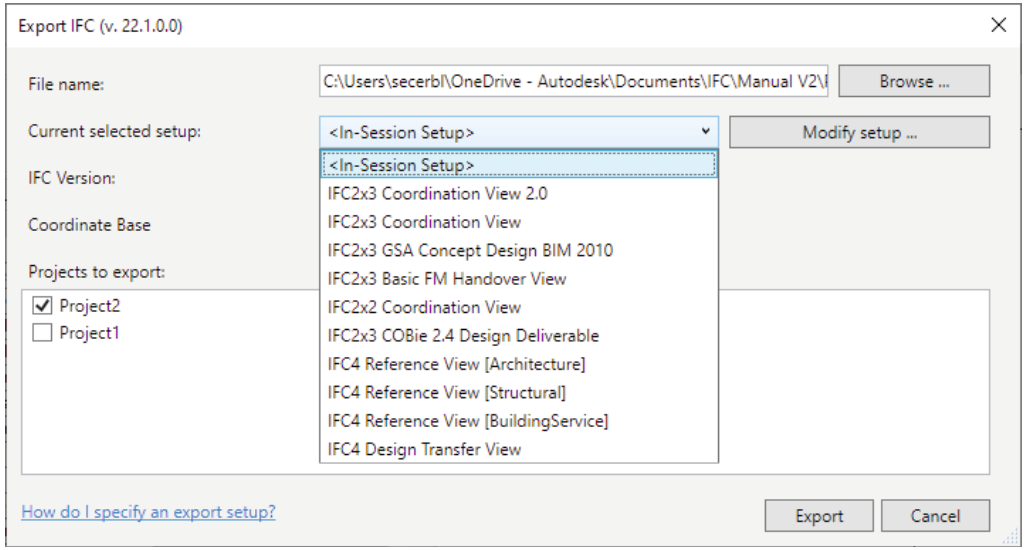
The reference to a custom layer reference file needs to be added to the Revit.ini file which can be found in the following location: C:\Users\<USER>\AppData\Roaming\Autodesk\Revit\Autodesk Revit 20xx

The full path to the layer reference file is added in the line starting with ExportLayersNameDGN=

For Example: ExportLayersNameDGN=C:\Users\<USER>\Documents\RevitLayers.txt
Just like with the class mapping, sometimes it is needed to assign the layer value on element level. For this, you can use the shared parameter *IfcPresentationLayer,* which is of course included in the official shared parameter files.

IFC Export Settings Dialog

The IFC export dialog in Revit is found by going to File > Export > IFC and it offers the direct selection of all built-in Model View Definitions (MVD) and also allows the export of all open projects, not only the active one:

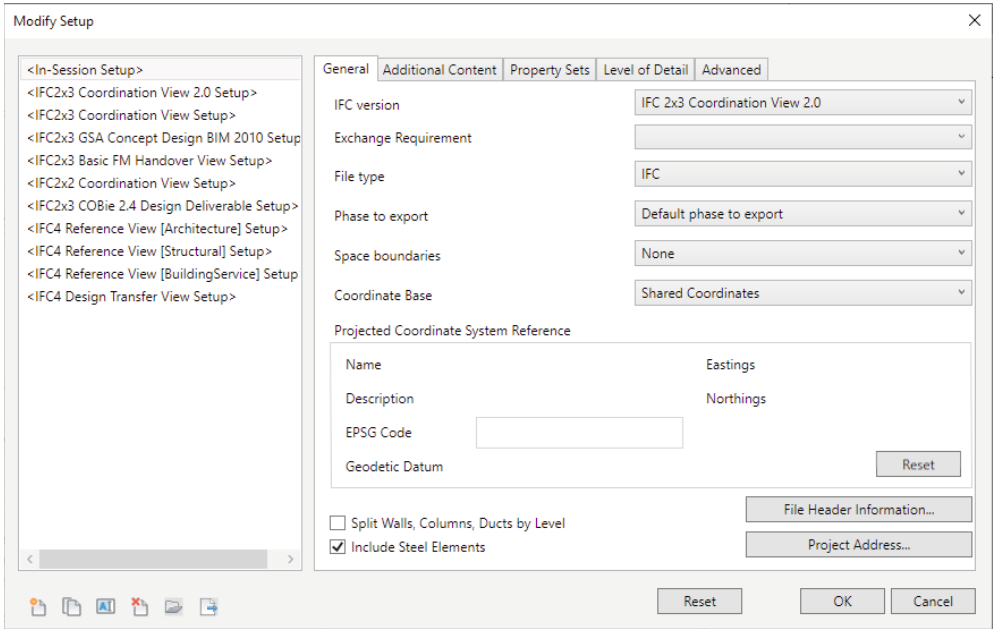


As already mentioned in this manual, the selection of the adequate IFC version and MVD is crucial for the quality of the content exported to IFC.
The most commonly used MVDs are IFC2x3 Coordination View 2.0 and IFC4 Reference View.

In addition, these settings can be modified by selecting Modify setup. The following pages are providing a detailed documentation about these settings.

General Settings

In this section, the settings for the In-Session Setup can be altered or a new Setup created by duplicating an existing one. It is not possible to change the predefined setups listed on the left between <>:



IFC Version allows the selection of the IFC specification and MVD, typically IFC2x3 Coordination View 2.0 or IFC4 Reference View. For more information, check the first chapter of this manual.

Exchange requirement is only valid when using IFC4, as here buildingSMART has defined different usecases for the certification for Architectural, Structural and MEP exchange.

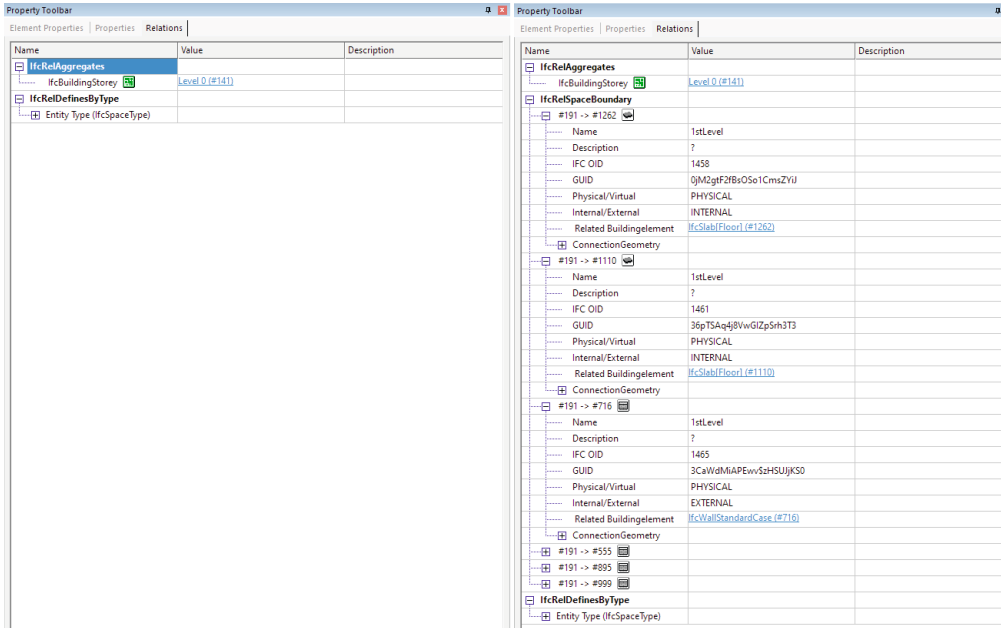
File Type allows the selection of alternative types such as .IFCXML or the zipped versions of .IFC / .IFCXML. The same results are achieved by exporting an .IFC and zipping it and .IFCXML is only by specific application. Most of the time the default setting .IFC should be your first choice.

Phase to export allows the selection of a specific phase in the project to be exported. The default phase to export is the last phase in the project. If „Export only elements visible in view“ is selected, the phase of the view will be used and this option greyed out.

Space boundaries defines the level of room/space boundaries exported:

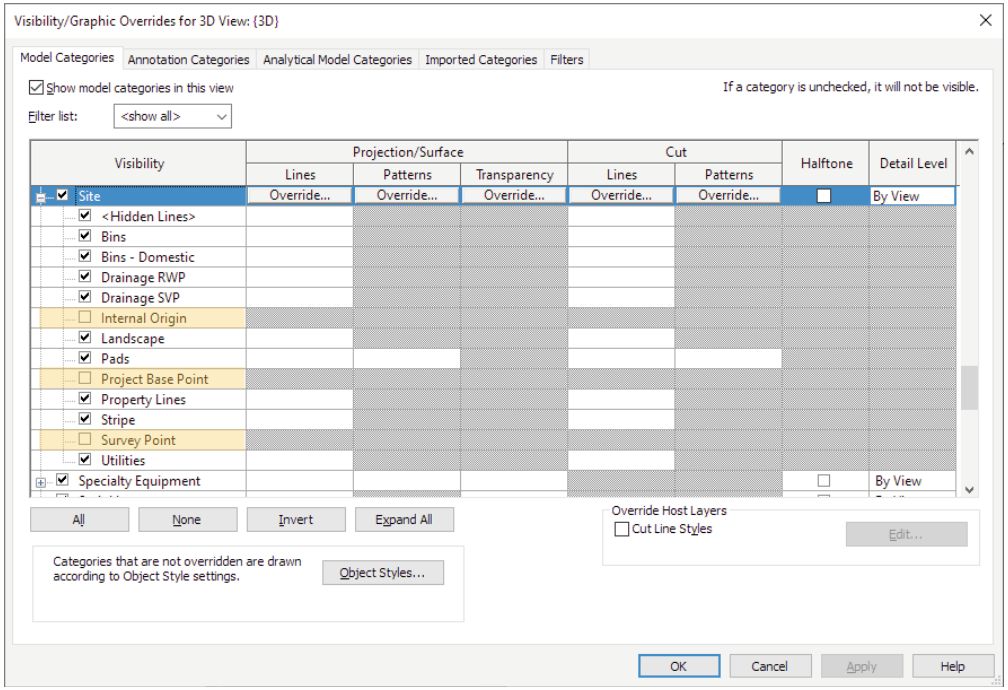
- None - room/space boundaries are not exported.
- 1st level - the room/space boundaries are included but are not optimized to split elements with respect to spaces on the opposite side of the boundary.
- 2nd level - the room/space boundaries are included and are split with respect to spaces on the opposite side of the boundary. A second level space boundary considers the material of the building element and the adjacent spaces behind it, providing thermal properties for further analysis.

The information is attached to spaces as well as to room boundary objects like walls and can be viewed in most viewers (example FZK viewer, left Level None, right 1st Level):



Coordinate base allows the selection between Shared Coordinates, Internal Origin, Project Base Point and Survey Point.

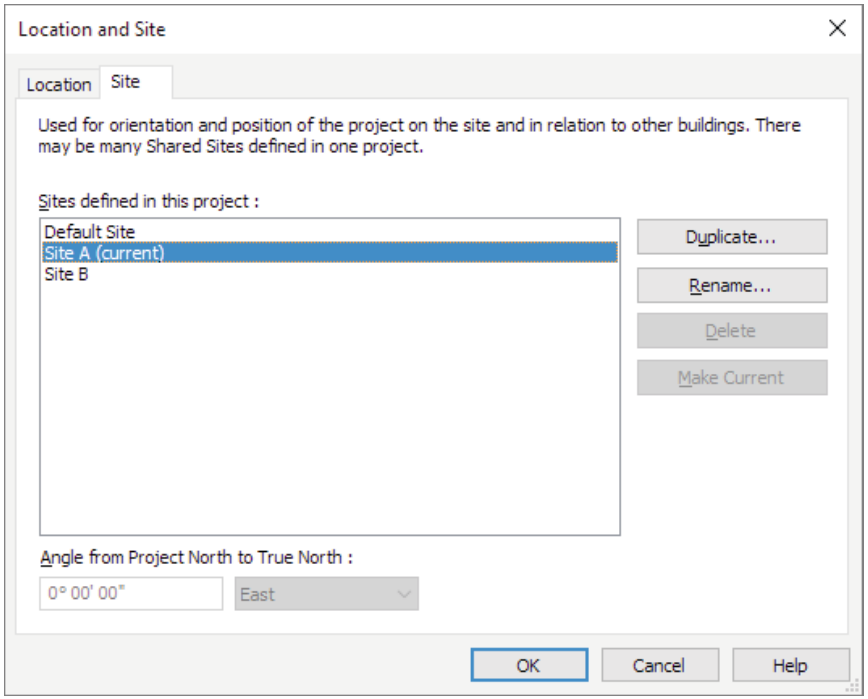
Every Revit project has initially three origins, which are usually hidden by default, but can be shown by going to Visibility Settings of your View > Site:



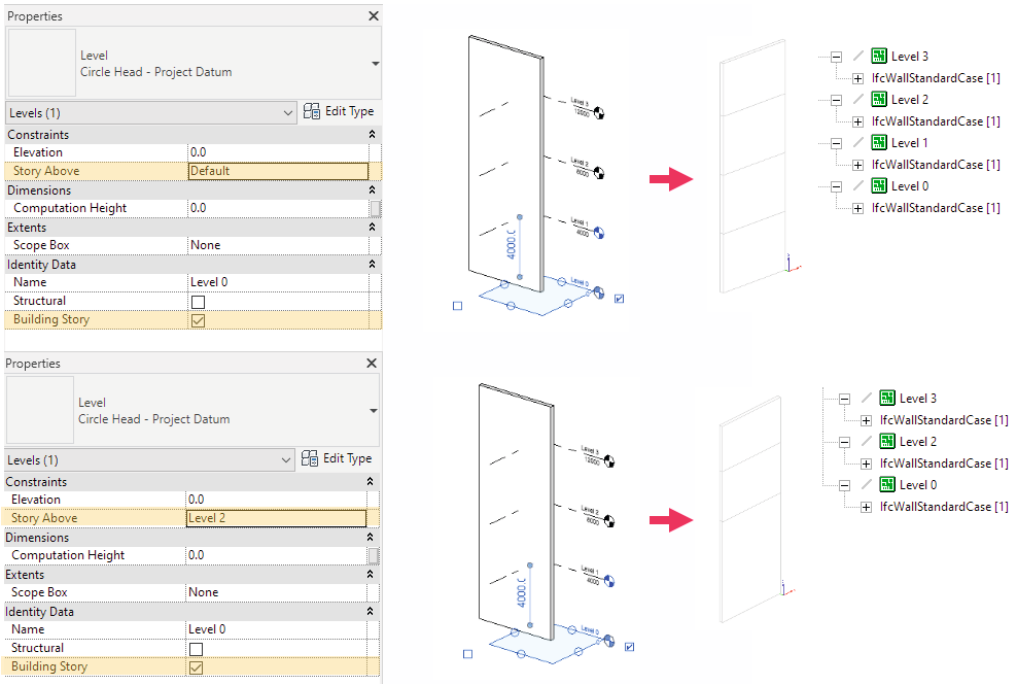
- **Internal Origin** cannot be moved and also represents the center of the 20 mile region in which Revit tolerates Geometry. Any kind of Geometry created beyond this region will cause error messages and are to be avoided.
- **Project Base Point** defines the project coordinates and is usually placed at a grid intersection or a building corner on the ground level. Typically, all point coordinates and heights in the project will be referenced to this point. This point can be moved (manually or by typing in coordinates) to a desired position, but won't move the project (unless the Project North, which is also visible in the Project Base Point, is changed). Prior to Revit 2020 the Project Base Point also had a clipped state, however this has been removed. The Project Base Point from Revit 2020 on is always unclipped.
- **Survey Point** marks a relevant point in the real world and can be clipped on unclipped. Moving a clipped survey point will actually change the shared coordinate system of the model, while moving the unclipped survey point can be moved (either manually or by typing in coordinates) without any effect on the shared system, similar to the Project Base Point.

In the default templates all points should be located at the same spot and should be adjusted according to the project agreement.

Shared site is an additional concept used to set up the relation between linked models. One Revit project can contain multiple shared sites and this option will refer to the currently selected one:



Split Walls, Columns, Ducts by level splits all elements crossing multiple building storeys automatically when exporting. When using this option, it is important to check levels defined as *Building Story* and also review the option *Level Above – Default* will use the next higher Building Storey to cut all elements assigned to the current level, unless another level is selected explicitly. The elements created by splitting will be assigned to the levels they were cut by.

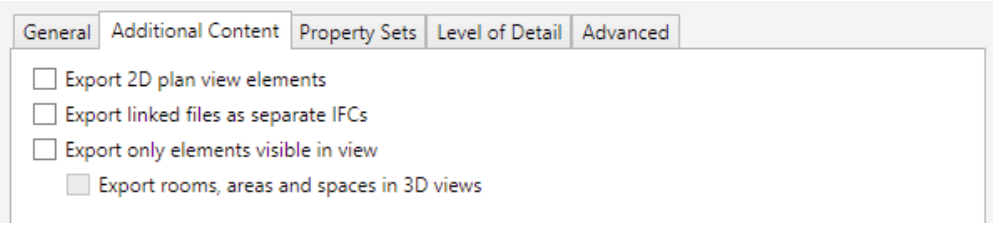


Include Steel Elements exports structural steel incl. steel connections.

File Header Information allows the definition of Author's name, E-Mail, Organization and Authorization in the file IFC header.

Project Address overwrites the address set in the project information either for buiding and/or site when exporting and also pushes this information back to Revit if *Update Project information* is selected.

Additional Content



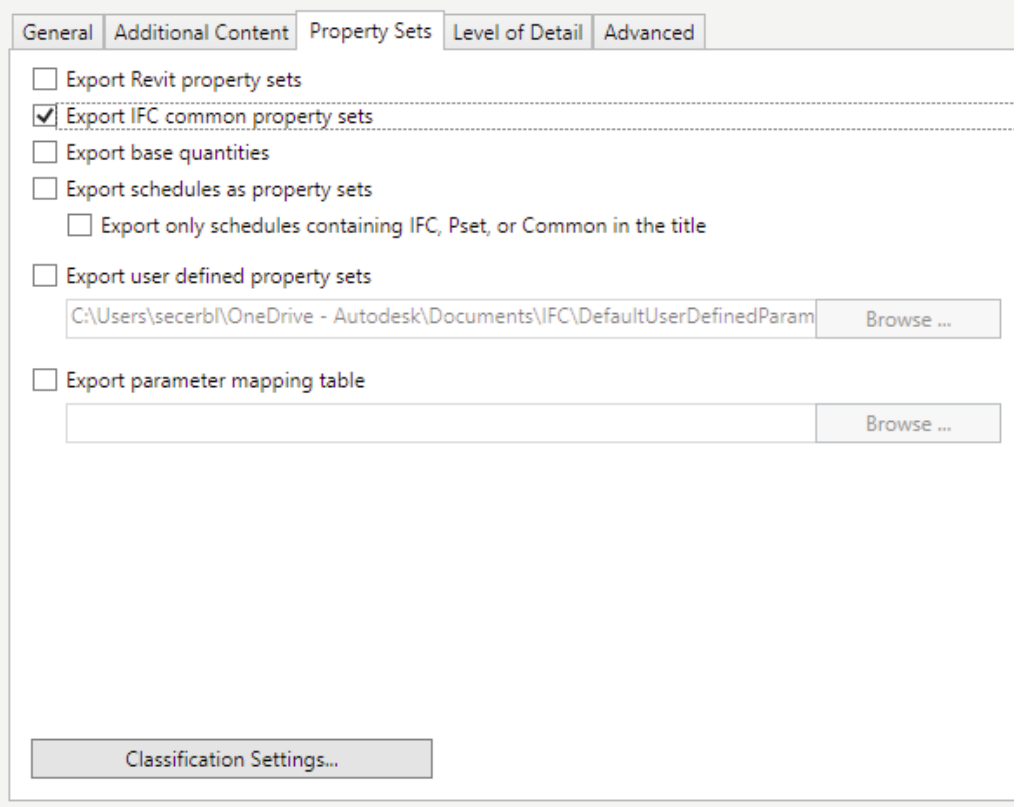
Export 2D plan view elements allows the export of the 2D elements supported by the IFC schema, such as notes and filled regions. Grids are considered as 3D elements and can be exported by assigning the Grids Revit Category to the *IfcGrid* class. It should be noted that IFC is a 3D oriented schema and only a limited number of 2D elements are supported in general, therefore PDF is still commonly used for 2D documentation.

Export linked files as separate IFCs uses the same settings to exportany linked files as separate IFCs. It is not possible to merge multiple Revit projects to one IFC when exporting from Revit, but the files can then be visualized together again in Autodesk Navisworks or most IFC viewers.

Export only elements visible in view will use the currently active view to evaluate which elements to export. As 3D views in Revit do not show rooms, areas and spaces, it is possible to include these using the second option **Export rooms, areas and spaces in 3D views.**

Property Sets

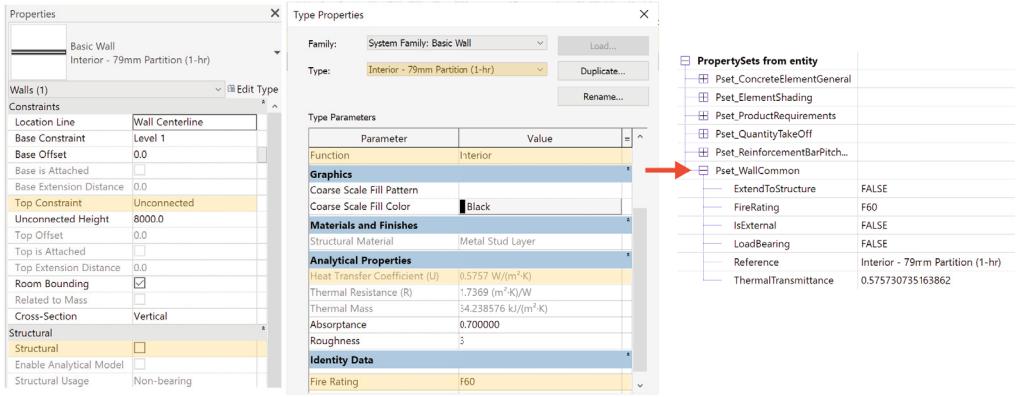
The property sets are carrying the information defined in the model and are therefore next to the correct classification the most important export setting. Note that in general *empty properties won't be exported.*



Export Revit property sets is deactivated by default, as this option will export all Revit properties according to their internal grouping. This will include a lot of obsolete information in the IFC and will also significantly increase the file size. It is recommended to use this option with caution and for testing purposes only.

Export IFC common property sets exports the default properties defined in the IFC schema and is activated by default. Existing Revit properties are automatically mapped to the IFC properties.

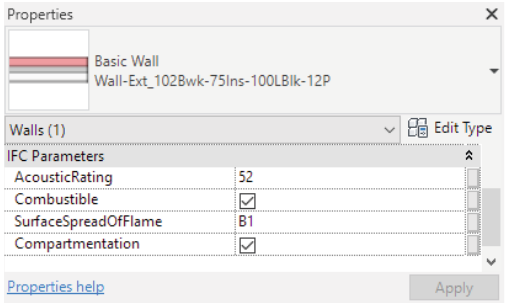
The common property sets can be recognized by the Pset_ prefix after the export:

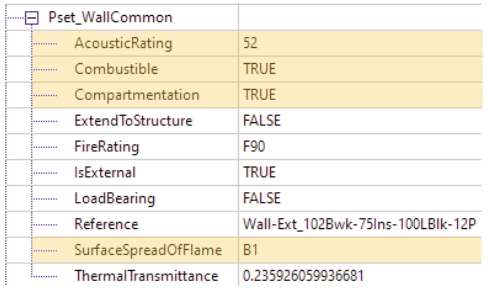


As the IFC schema contains many properties which are not commonly used in all projects and therefore not included in Revit by default, only a subset of the properties defined in a Pset will be exported using this option. The complete Pset_WallCommon includes several properties not existing in Revit by default:

Property	Description
Reference	Component type (type name)
AcousticRating	Sound insulation class
FireRating	Fire-resistance class (type parameter)
Combustible	Combustible material
SurfaceSpreadOfFlame	Fire behavior
ThermalTransmittance	U-value (type parameter)
IsExternal	Exterior component (type parameter, given as yes/no)
ExtendToStructure	Fixed on top (behavior)
LoadBearing	Load bearing (instance parameter)
Compartmentation	Fire compartment-defining component

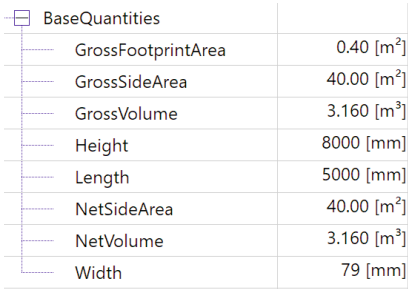
There are several options to add these properties. The first and the most simple one is by adding the properties with the same name and data type as defined in the IFC schema to Revit. The easiest way to accomplish this is by using the IFC shared parameters file already introduced in this manual (see: Using IFC Shared Parameters). This will make sure that the spelling and the data type are correct. Once these properties have been added and populated, they will be automatically added to the Pset upon export:





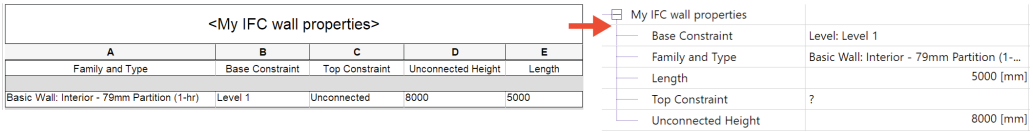
Alternatively, it is possible to map other properties (as long as they have the same data type) to the corresponding IFC properties.

Export base quantities will include another type of property sets defined in the IFC schema as well, which are meant specifically for estimation and QTO purposes.
For a wall these quantities typically look like this:



Export schedules as property sets allows the creation of user defined property sets through Revit schedules. All properties which are not part of the standard property sets defined in the IFC schema can be added to custom property sets. As Revit projects can have many schedules, it is also possible to limit this option to **Schedules containing IFC, Pset or Common in the title.**

All properties are collected in the schedule and can be found in the IFC upon export:



Note: only the official property sets defined in the IFC schema are allowed to start with “Pset_ “.

The advantage of this workflow is that it is not necessary to worry about data types or configuration files, however the schedules are not that easily transferrable between projects, which is why there is also a second option to create user defined property sets.

Export user defined property sets is the equivalent to exporting schedules as property sets, however it uses a text file as a configuration file instead. The default template file can be found here: C:\ProgramData\Autodesk\ApplicationPlugins\IFC 20xx.bundle\Contents\20xx
and includes detailed instructions and examples.

Basic structure:

```
# Format:
#    PropertySet:   <Pset Name> I[nstance]/T[ype]   <element list separated by ','>
#   <Property Name 1>   <Data type> <[opt] Revit parameter name, if different from IFC>
#   <Property Name 2>   <Data type> <[opt] Revit parameter name, if different from IFC>
```

Everything between <> is replaced:

<Pset Name>: Name of the Pset, don't use Pset_ as Prefix as this is reserved for standard IFC Psets

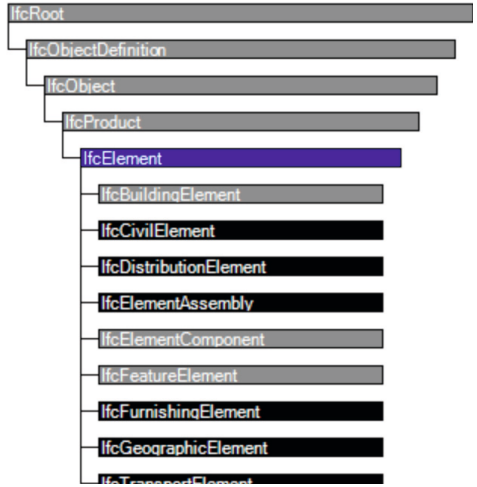
I[nstance]/T[ype]: used to specify Instance or Type properties, obsolete in current versions as the selection happens automatically, use either I or T

<element list separated by ','>: this is the place where either one or more IFC classes for which this Pset will be applied are listed, e.g. IfcWall, IfcSlab, IfcColumn. If the Pset should be applied to all elements, use the next higher entity (with - IfcBuildingElement building elements like walls, doors etc. or IfcElement to include also Civil and Distribution elements. This can be checked in the IFC documentation by searching for Entity inheritance.

<Property Name>: property name as displayed in Revit

<Data type>: the supported IFC data types are listed in the template file, the most commonly used are Text, Integer, Real, Length, Volume, Boolean.
There are currently 40 IFC property types that are supported in the Revit IFC export. Not every property type in Revit can be mapped directly to the IFC type, as IFC uses a different way of specifying some of the units. When mapping Revit data type that does not have a direct mapping to the IFC data type, it can be mapped to a primitive type, e.g. Real or Integer. This will export the value unconverted using Revit internal units.

Entity inheritance

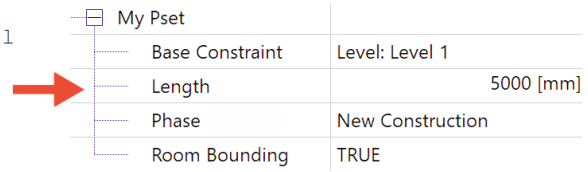


<[opt] Revit parameter name, if different from IFC> is an optional field and can be omitted, if the name of the Revit property should also be used for the IFC property. If the IFC property should have a different name, it can be entered here.

Note: All entries are separated by a <TAB> and the file should be saved in the UTF-8 format.

Example:

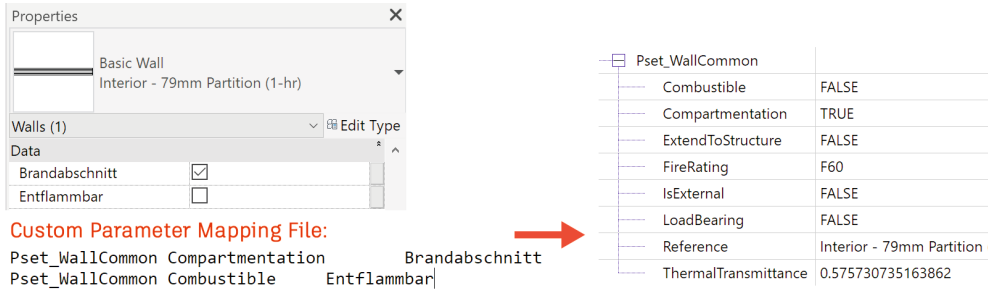
```
PropertySet:	My Pset	I	IfcWall
	Phase Created	Text	Phase
	Base Constraint	Text
	Room Bounding	Boolean
	Length	Length
```



Export parameter mapping table allows the mapping of custom Revit properties to the standard mapping properties, as long as they have the same data type. Similarly to the user defined property sets, this is accomplished with a text based mapping file. There is no default template included for this file, however the syntax is fairly simple:

IFC Common PropertySet Name <TAB> IFC Property Name <TAB> Revit Property Name

By using this method, Revit properties can be named according to the project or company standards and will be mapped according to the correct IFC terminology upon export.

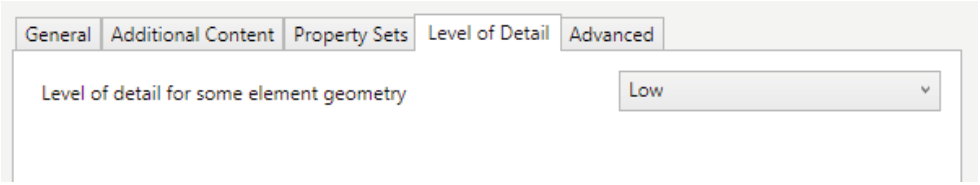


Classification settings is the last option in this section which allows the input of the main information about the classification system used in the model.

More information about classifications can be found in the chapter Using classifications in Revit.

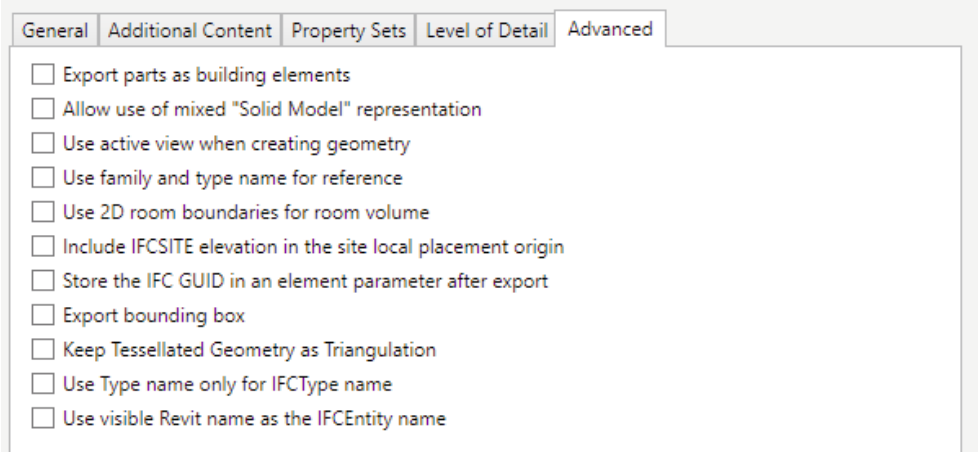
Level of Detail

This option lets you select the level of detail for tessellated geometry. By default, the level of detail is set to “Low”. As the Level of detail has impact on file-size and data-quality it is recommended to evaluate this option before export.



Advanced

This tab offers advanced option which can be used when needed:

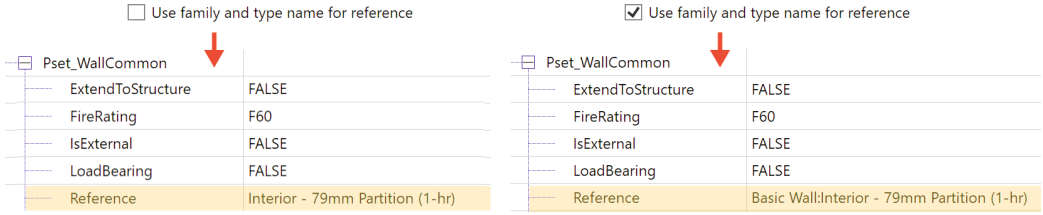


Export parts as building elements is relevant when working with parts. The default settings will export the original element only and by activating this option it is possible to export the parts themselves as separate elements.

Allow use of mixed “Solid Model” representation enables the export of combined swept solid and B-rep models. A geometric object in an IFC data model is normally generated from either one or several swept solid objects, or from B-rep objects alone. The combination of these two types of representation is not enabled by default in the IFC schema. For more complex components, in particular, this leads either to a larger file size or incorrect presentation, as elements are wholly represented B-rep objects. Solid model representation combines the two types of representation within a single class, which can mean better geometric results at a smaller file size for complex models. It should be noted, however, that the IFC file exported using this setting no longer complies with the default IFC schema and must therefore be accepted as such by all those involved in the project. For certain areas of use, it may be necessary to have an unaltered default schema for export.

Use active view when creating geometry will use the level of detail level of the current view (Coarse / Medium / Fine) and will export all objects according to the way they are displayed in Revit.

Use family and type name for reference will influence the way the naming of the reference in IFC. By default, the Revit Type name is used for the IFC reference. By activating this option, the family name will be used along with the type name:



Use 2D room boundaries for room volume simplifies the calculation of the room volume based on two-dimensional spatial boundaries. Using the default settings, the Revit room geometry is used to determine the volume in IFC.

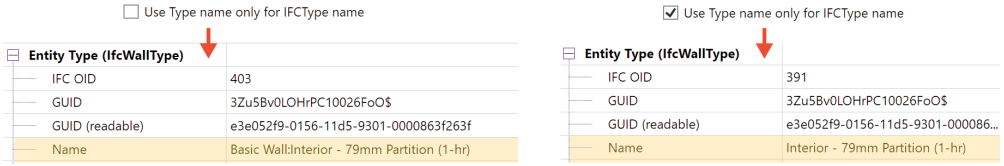
Include IFCSITE elevation in the site local placement origin: Select this option to include the elevation from the Z offset of the IFCSITE local placement. Clear the option to exclude it.

Store the IFC GUID in an element parameter after export: Select this option to store the generated IFC GUIDs in the project file after export. This will add „IFC GUID“ parameters to elements and their types, and Project Information for Project, Site, and Building GUIDS.

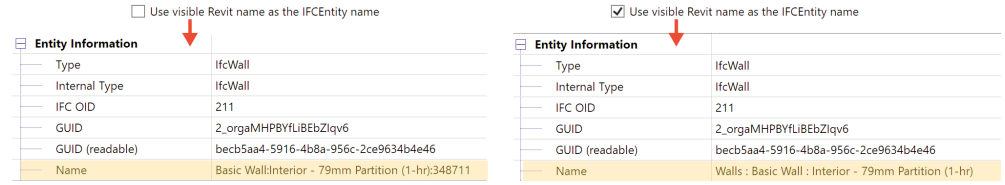
Export bounding box allows the export of bounding box representations.

Keep Tessellated Geometry as Triangulation: Select this advanced option to use a triangulation method that is compatible with older IFC 4 Reference View viewers.

Use Type name only for IfcType name excludes the family name for the IFC type name:



Use visible Revit name as IFC entity name: influences the generation of the element name in IFC:



Using classifications in Revit

Classification basics

Classifications help group and classify BIM data in a simple and efficient manner. In addition to the standard IFC classification according to component classes, there are various international and national classification systems available, for example:

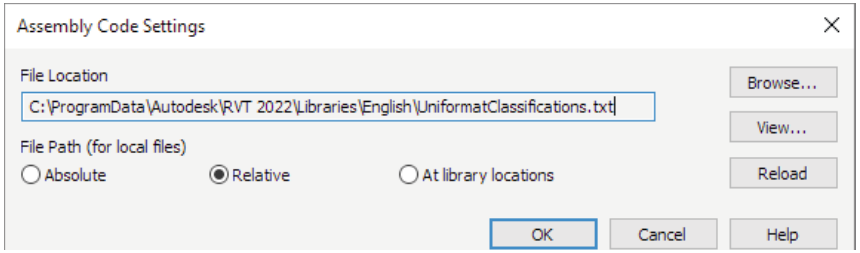
- Uniclass2015
- Omniclass / Uni Format / Master Format
- ASTM E1557
- FICM

Revit writes and reads IFC data and thus supports the IFC classification of the respective IFC schema. To export correct IFC classifications, it is sufficient to select the correct „mapping table“.

Uniclass 2015

Uniclass 2015 is a unified classification system for all sectors of the UK construction industry. It contains consistent tables classifying items of all scales. It was first releases in 1997 allowing project information to be structured to a recognized standard.

The now valid version of Uniclass is compatible with BIM processes.



Assembly Code „Manage -> Additional Settings -> Assembly Code”

The default – type based - classification system used in Revit is the Uniclass system. It is distributed as a text file shipped with every Revit license. In a default installation this file can be found at:

C:\ProgramData\Autodesk\RVT 20XX\Libraries\<your localization>\ UniformatClassifications.txt

The Uniclass classification is type-based and assigned to the “Assembly Code” parameter.

For exporting the Assembly Code no further action is necessary, it is automatically exported as IFCClassification [11]

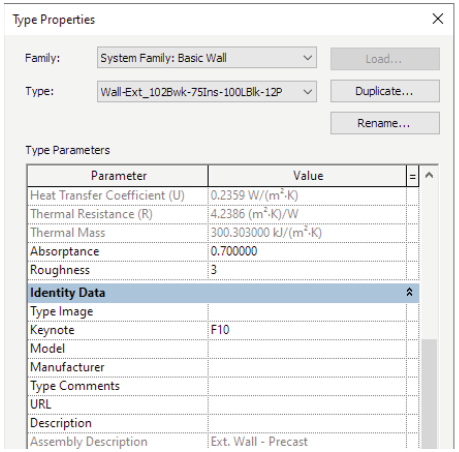


Figure 11: Assembly Code assigned to a System Family of type wall

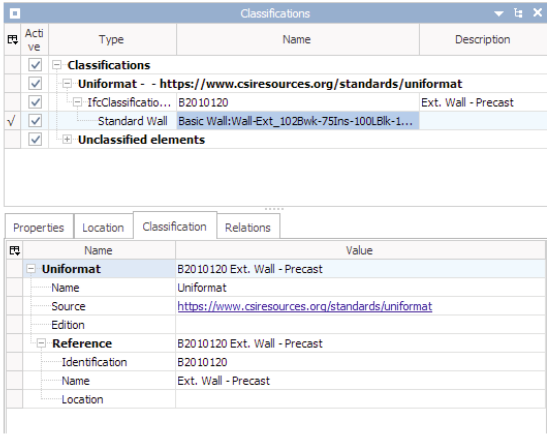


Figure 12: Assembly Code as Uniformat Classification for IFC entity

11. https://standards.buildingsmart.org/IFC/RELEASE/IFC4_1/FINAL/HTML/schema/ifcexternalreferenceresource/lexical/ifcclassification.htm

OmniClass®

Classification according to OmniClass® is a comprehensive classification system for the construction industry published by the Construction Specifications Institute (CSI) providing a classification structure for electronic databases and software through the full project life cycle. The default path to classifications in Revit is: [12]

C:\Users\<current username>\AppData\Roaming\Autodesk\Revit\<product name and release>

For exporting OmiClass® classifications to Revit objects manually, the IFC export – option *Modify Setup -> Property Sets -> Classification Settings...* must be selected. In figure 13 the required data is provided. The resulting classification is presented in figure 14.

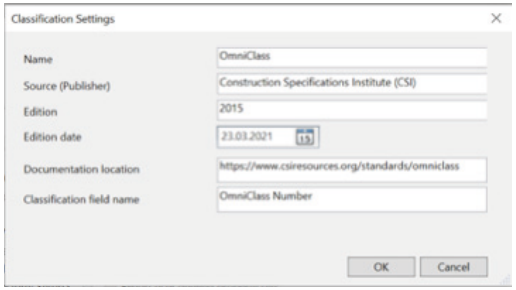


Figure 13: Revit Classification Settings

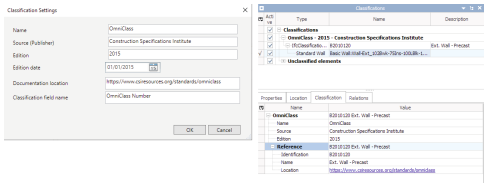


Figure 14: Column OmniClass classified – result as IFC

Classifications with the Autodesk Classification Manager for Revit

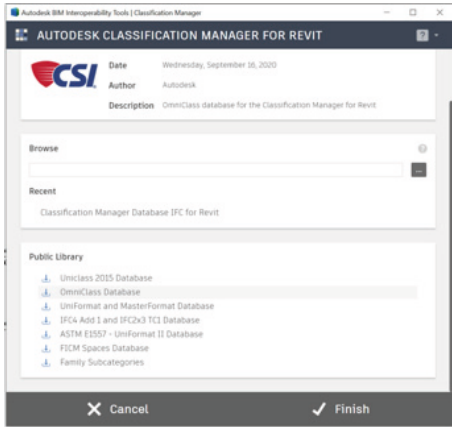


Figure 15: Revit Classification Manager

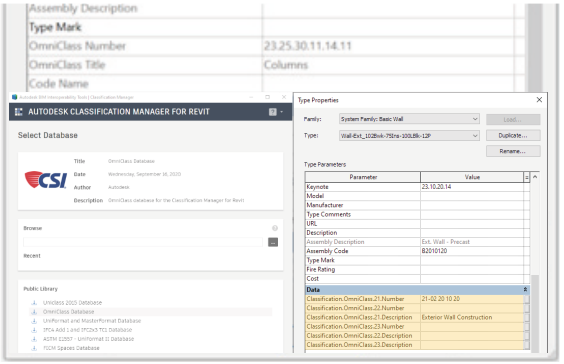


Figure 16: Classification Manager's shared parameters

Another method for classifiing Revit elements is the Classification Manager for Revit.

With this plugin Revit elements can be classified interactively. The IFC export works according to figure 13, only the shared parameter name has to be adopted.

Further information can be found at: https://interoperability.autodesk.com/

12. **Keynote** Table file can be located directly in Revit: Annotate/Keynote/Keynoting Settings. Keynotes are a means of annotating model elements. Revit is prepared to do this, in fact you can directly create a Keynote Legend filtered by sheet, that means that if you insert that legend in a sheet it will only list keynotes defined in that sheet, so the intention to use it as an annotate tool is clear. Keynote Table refers to **Masterformat**, which is another classification list also published by CSI. Last version is the one based in Masterformat 2004. Masterformat criteria, as well as OmniClass, is about listing work results. It also incorporates construction practices.

Advanced / multiple Classifications

Basically, classifications in Revit are restricted to one classification system per file. Using following shared parameters allows to add multiple classification systems to one model.[13]

Names for multiple classification shared parameters are [14]:

ClassificationCode
ClassificationCode(2)
ClassificationCode(3)
ClassificationCode(4)
ClassificationCode(5)
ClassificationCode(6)
ClassificationCode(7)
ClassificationCode(8)
ClassificationCode(9)
ClassificationCode(10)

The syntax for establishing a classification is:

[ClassificationName]Code:Title
Example:
[Maturity]01:STATUS

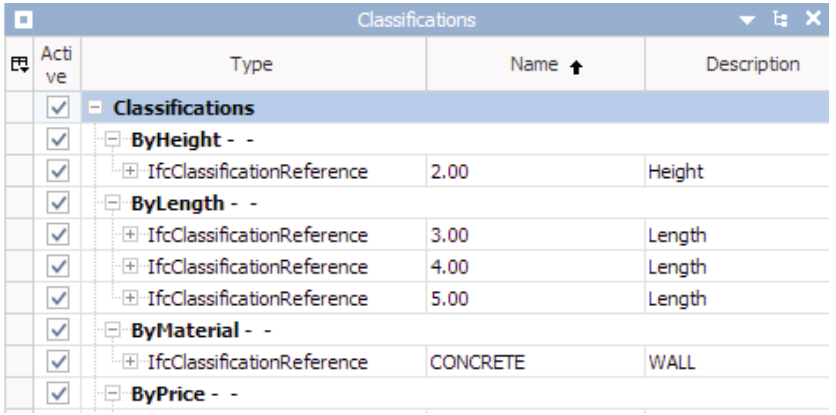


Figure17: Multiple Classifications in IFC

Data	
ClassificationCode	[Maturity]01:STATUS
ClassificationCode(2)	[ByMaterial]CONCRETE:WALL
ClassificationCode(3)	[ByHeight]2.00:Height
ClassificationCode(4)	[ByLength]5.00:Length
ClassificationCode(5)	[ByPrice]Low:Price
ClassificationCode(6)	
ClassificationCode(7)	
ClassificationCode(8)	
ClassificationCode(9)	
ClassificationCode(10)	

Figure18: Multiple Classifications in Revit

13. Now the use of multiple classifications is restricted. IfcClassification attributes, including source, edition, edition date, name, description, location, and reference tokens are not supported.
14. ClassificationCode(1) is not functional.

Further use cases and tips

Exporting Floors to IFC

Revit floors are mostly modeled using two separate elements: a load bearing slab for the level and the floor structures for each room.

For IFC export all slabs are assigned to the IFCSlab class by default. In terms of IFC this might be a wrong classification as slabs shall export as IFCSlab class and floors as IFCCovering class mainly because of different PropertySets associated.

To accomplish that, floors in Revit are specified as IFCExportAs “IFCCovering” and IFCExportType “FLOORING”. Alternatively, both class and and type can be assigned to IFCExportAs using the syntax: IFCCovering.FLOORING.

Default:

Entity Information	
Type	IfcSlab[Floor]
Internal Type	IfcSlab[Floor]
IFC OID	325
GUID	0sVQDJH5bAmuG$chlJzfHc
GUID (readable)	367da353-4459-4ac3-843f-9a...
Name	Floor:Floor-Grnd-Bearing_65...
Description	?
Object Type	Floor:Floor-Grnd-Bearing_65...
Predefined Type	FLOOR
Layer Name	A-FLOR-___-OTLN
Color	Color [R:165, G:42, B:42, A:255]

Customized:

IFC Parameters	
IfcExportAs	IfcCovering.FLOORING

or

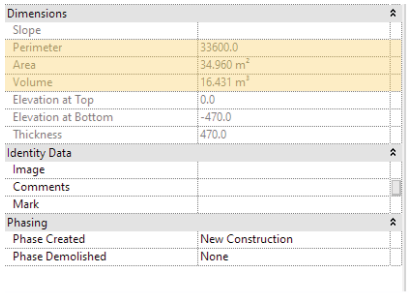
IFC Parameters	
IfcExportAs	IfcCovering
IfcExportType	FLOORING

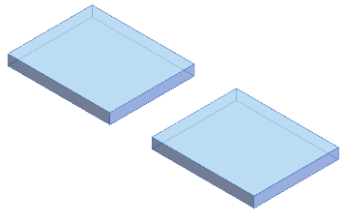
Entity Information	
Type	IfcCovering
Internal Type	IfcCovering
IFC OID	209
GUID	0sVQDJH5bAmuG$chlJzfI2
GUID (readable)	367da353-4459-4ac3-843f-9a...
Name	Floor:Floor_Timber_22Cbd-2...
Description	?
Object Type	Floor:Floor_Timber_22Cbd-2...
Predefined Type	FLOORING
Layer Name	A-FLOR-___-OTLN
Color	Color [R:127, G:127, B:127, A:...

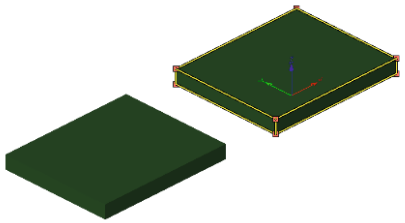
Figure 19: IFCEntities and predefined Types for floors

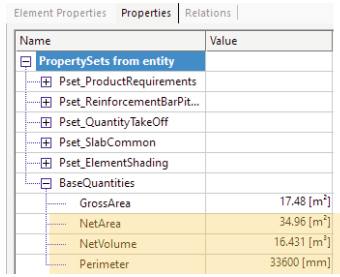
Modelling Slabs for IFC export

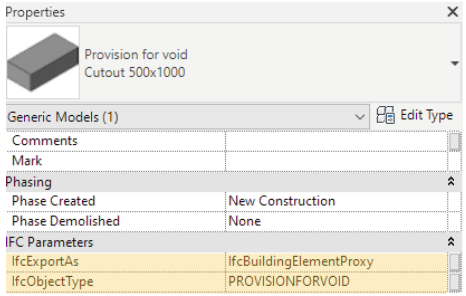
Even though Revit allows the creation of floor/sketched ceiling geometries from un-connected polygons, they are to be avoided in the models because when exporting to IFC those connected Revit objects are considered independent elements in IFC and all values of properties are assigned to each resulting IFC object.

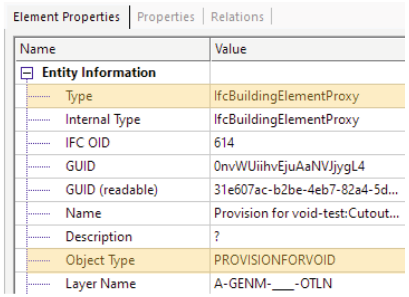












Cut openings

The use of proxy objects has largely been established in the preliminary design and coordination of cut openings in an integrated design process. In IFC those objects are called “provision for void” objects and are exchanged between domain models together with alphanumeric information and dimensions.

Proxy elements origin either from native Revit aperture elements or are simple families with a void.

To accomplish the export of provision for void objects, the native Revit object is specified as IFCExportAs “IFCElementProxy” and IFCObjectType “PROVISIONFORVOID”.

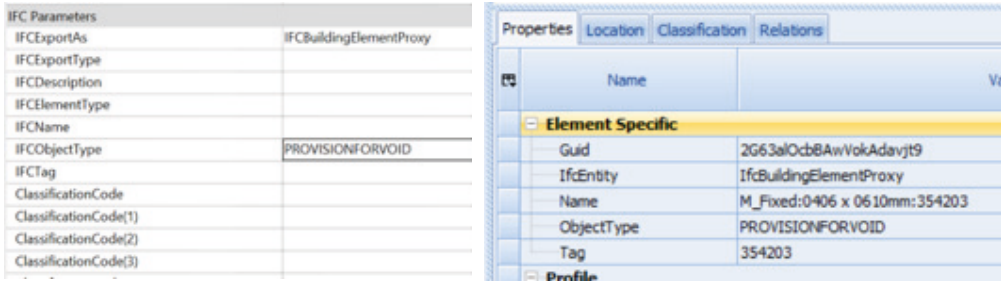
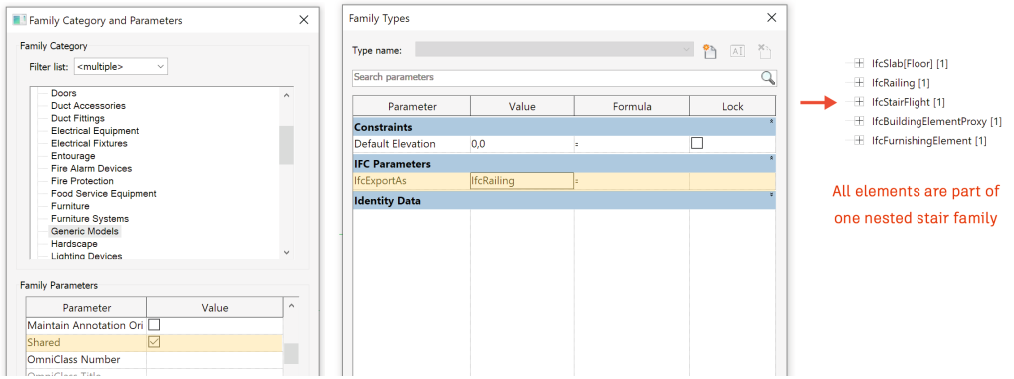


Figure 20: Provision for void

Nested Families

When exporting nested families, all elements will be assigned to one class / entity by default. It is however possible to classify nested families separately as own entities. For this, these families need to be shared and also have an own IfcExportAs parameter:



Assigning assemblies

Assemblies are important for the higher-level grouping of components like structural beam systems, beam grids and reinforcement cages. Unlike Revit groups, assemblies are exported to IFC as IFCElementAssembly classes with higher-level properties assigned.

To accomplish the export of element assemblies, the native Revit object is specified as IFCExportAs “IFCElementAssembly” and IFCObjectType “RIGID_FRAME”.

15. RIGID_FRAME is just one item of the IFCElementAssembly ENUM

Zones

The export of IFCZones from Revit is realized through a set of shared parameters that are assigned to room objects.

Zones in IFC are an aggregation of spaces that can be classified. From Revit the export of zones classifications is restricted to one classification per model.

The Revit parameter for zone classification is “ZoneClassificationCode”. The syntax is the same as for advanced / multiple classifications.

ZoneClassificationCode: [ZoneClassificationName]Code:Title

Room Name and Classification

A	B
Name	ClassificationCode(3)
Room	[ROOMS]01.01.01:Single Appartment
Room	[ROOMS]01.01.02:Double Appartment
Room	[ROOMS]01.01.02:Double Appartment

Zone Classification

C
ZoneClassificationCode
[ZONE]01:ZoneClass
[ZONE]02:ZoneClass
[ZONE]02:ZoneClass

Zone Name, ZoneDescription, ZoneObjectType

D	E	F
ZoneName	ZoneDescription	ZoneObjectType
TOP1	TOP 01	Small
TOP2	TOP 01	Medium
TOP3	TOP 01	Big

G	H	I
ZoneName 2	ZoneDescription 2	ZoneObjectType 2
Appartment 01	Appartment 01 in Building 01	Single-Appartment
Appartment 02	Appartment 02 in Building 01	Double-Appartment
Appartment 02	Appartment 02 in Building 01	Studio

Zone Name 2, ZoneDescription 2, ZoneObjectType 2

J	K	L
ZoneName 3	ZoneDescription 3	ZoneObjectType 3
Site 01	Building 01 at site 01	Family Home
Site 02	Building 01 at site 02	Family Home
Site 02	Building 01 at site 02	Practise

Zone Name 3, ZoneDescription 3, ZoneObjectType 3

M	N	O
IFCDescription	IFCName	IFCObjectType
Room Description A	Room Number	Room-Object1
Room Description B	Room Number	Room-Object2
Room Description C	Room Number	Room-Object3

Room Parameters

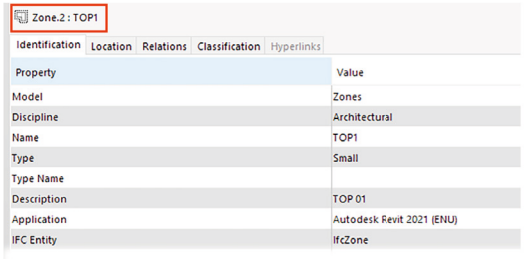
Zone related parameters allow more detailed informations on zones. In the figure above the exportable revit parameters are listed.

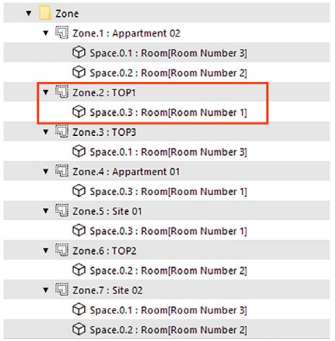
Room Name and Classification are assigned to the rooms in IFC.

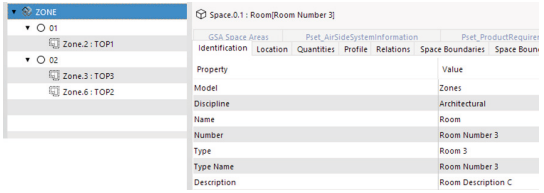
The ZoneClassificationCode is the classification parameter for zones.

ZoneName, ZoneDescription and ZoneObjectType define zones objects . Three independent zone-definitions are available (ZoneName, ZoneName 2 and ZoneName 3).

Note: The IFCName parameter is mapped to Number, IFCDescription is mapped to IFCSpace – Description.







Appendix

Dynamo and IFC

In this appendix you will find some “Dynamo” examples for preparing or enhancing IFC Data.

Adding Classifications to Revit

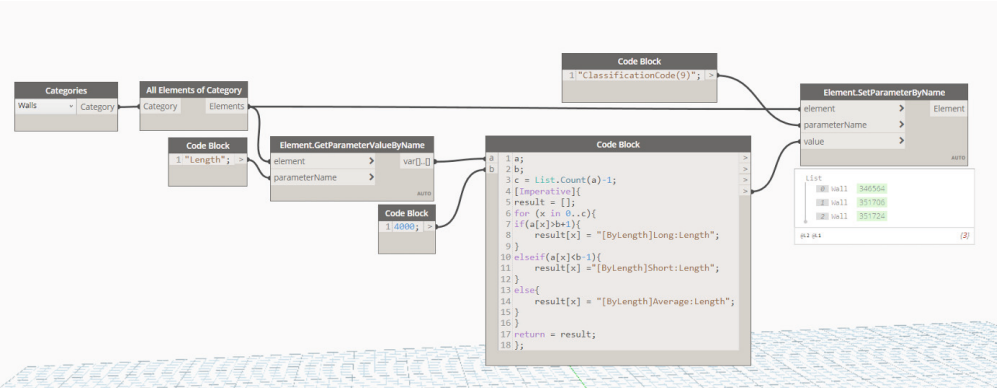
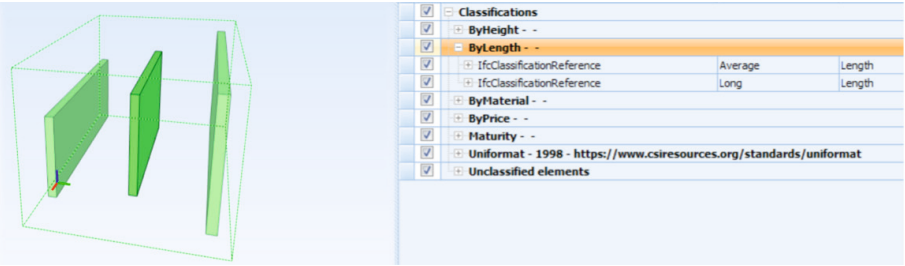


Figure 4: Dynamo script for classifying walls

Description:

Select elements from Revit model; in the imperative Code Block evaluate the classification’s result – remember **[ByLength]** is the name of the classification, the title and **Long / Short/Average** the respective code.

The result is pushed into the parameter ”ClassificationCode(9)”.

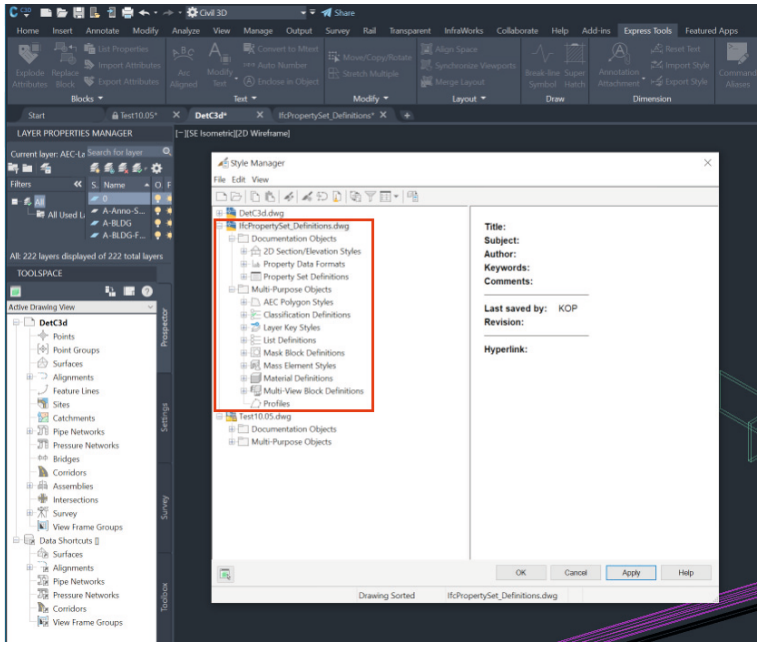


IFC Export for AutoCAD based products

For exporting data from AutoCAD based products like Civil 3D, AutoCAD MEP, … to IFC some basic considerations are relevant.

The AutoCAD data must be structured for the IFC export. This is done in the “Style Manager” (AutoCAD command: “STYLEMANAGER”). This command opens a dialog for creating and editing styles defining the appearance of objects in a drawing and most importantly for IFC export.

Styles are used in AutoCAD to define objects (such as walls, pipes, windows, …), documentation objects (such as 2D sections/elevations, property data formats and property set definitions), and multi-purpose objects (such as layer keys, classification settings, material definitions, …).

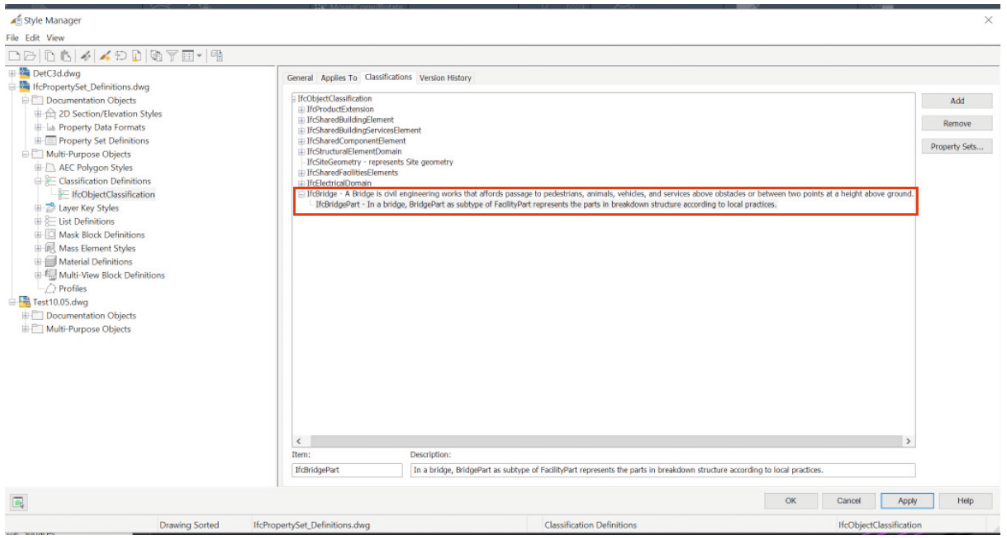


Creating IFC classes and assigning

First objects are selected to whom the classifications apply to. After that – or before classifications are created. In the top right corner of the “Classifications” window there are buttons for adding and removing classes or assigning property sets to classes.

The structure for IFC classes is corresponding to the respective IFC schema. Subclasses can be created by selecting a parent class.

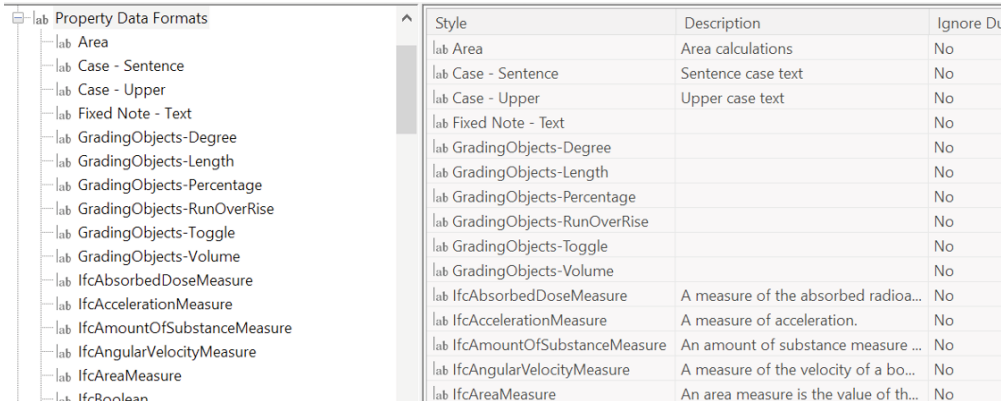
Now for each class selected property sets can be assigned.



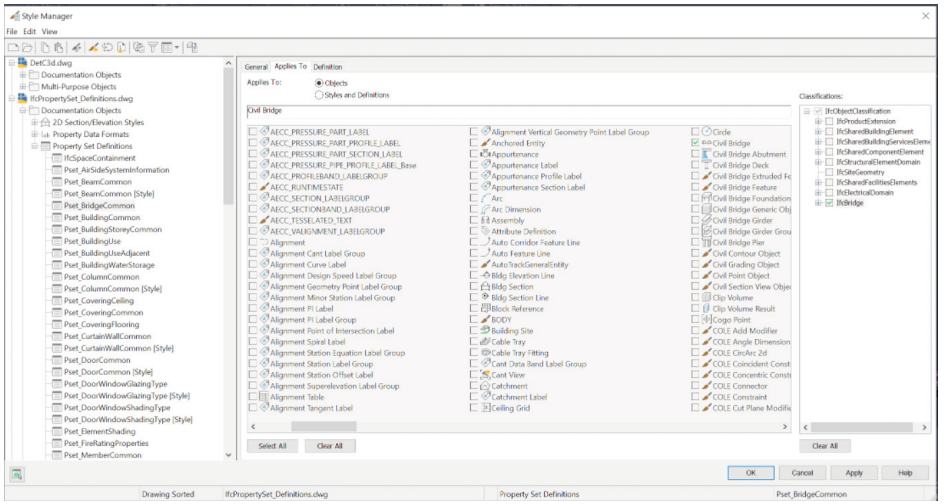
Properties, Property Data Formats and PropertySets

The creation of properties follows strict rules.

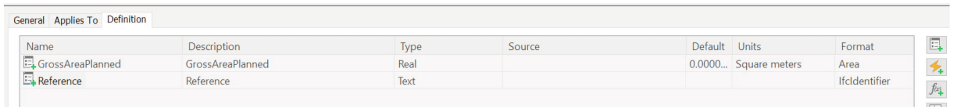
Before creating a new property, one must check if the required data format for this property already exists. If not, a new style must be created in “Property Data Formats” (context menu -> New) .[16]



Now a new property set is created, classes and properties are assigned to it. (for this manual it is “Pset_BridgeCommon).

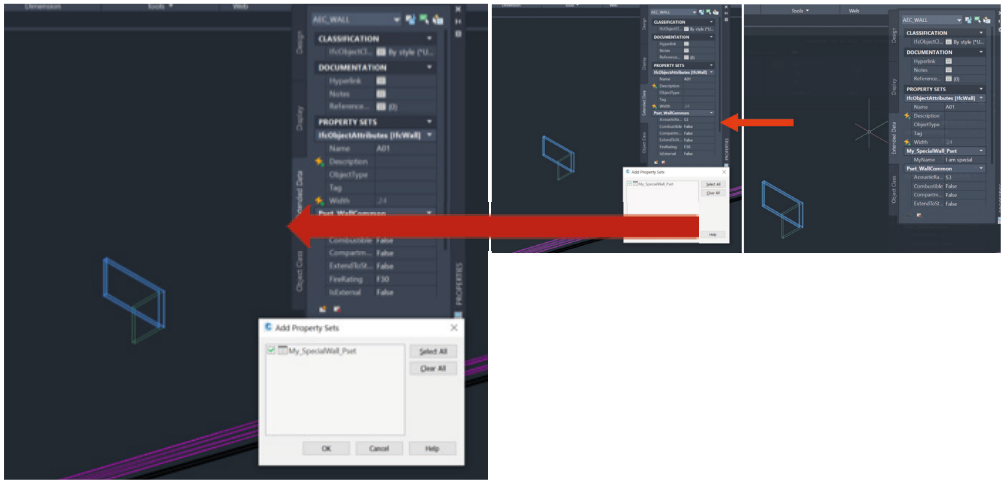


- General: Define name of Pset, add description
- Applies To: Assign objects (Civil Bridge)
- Classification: Select IFC class (IFCBridge
- Definition: Add properties.



16. For this manual the file “IfcPropertySet_Definitions.dwg” was used as prototype drawing for the use in Style Manager.

Now those properties can be assigned to AutoCAD objects by clicking the “add property” icon – marked with red arrow in fig. and the selecting Pset (here: My_SpecialWall_Pset).



Left side: Assignment of Pset My_Special_Wall, right: Pset My_Special_Wall assigned, value “I am special”

Now the data can be exported to IFC.

During export
- the respective IFC schema is selected
-Object types are selected for export
- Ressources and Assignment are selected

Finally, the selected data is exported.

Digital Quality Management for IFC projects by Tobias Schmidt, TÜV SÜD

The use of IFC is particularly interesting for those appointing parties – or building owners - who want to rely on a universal project implementation of BIM. The universal tactic of an IFC application can be triggered through various project strategies: a short-term project approval is incapacitating the appointing party to formulate an individual BIM strategy, or the technical procurement has identified the best project feasibility and the highest project attractiveness when different software solutions are implemented, or the Project Information Manager has defined an information model to rely on a generally recognized standard.

For both appointing and appointed parties in a BIM project, IFC as a data medium has the potential to streamline the entire information management process: information models that have already been created from a software application can also be used by other systems without investing a lot of manual effort in duplicating, repairing or completing information models. This quality measure, in turn, is achieved when the overall project strategy and the entire Information Management is adapted to fully support IFC as a deliverable and the matter of open BIM as a project work culture.

TUV SUD has recognized, that “I want IFC” from the appointing party and that a click on “Export to IFC” on authoring and coordination level from appointed parties, such as technical consultants and contractors, is not a sufficient quality measure to achieve a very best IFC. Looking into the Information Management described in ISO 19650, it turns out that IFC is not just a data format. IFC means a well-structured, aligned, and synchronized work culture among all stakeholder, across all trades and along the entire project or asset lifecycle.

Three important items towards project work cultural and implementation of IFC from the view of TUV SUD

The successful IFC use is guaranteed if the building owner, as the appointing party, and all the appointed parties jointly set up a solid information management in BIM projects that supports IFC quality:

- **Clearly define overall IFC requirements at the start of the project:** because the IFC standard has now grown to a large „data ecosystem“ with many options and different characteristics, appointing parties shall develop and integrate Exchange Information Requirements to define which use cases for the project and for the building documentation, using IFC, should be realized; the Model View Definitions developed by BuildingSmart (see „BuildingSmart MVD Database“) gives an insight into which project topics IFC can optimally support; Model View Definitions are part of every solid IFC project ‘ Exchange Information Requirements, because these MVDs log for appointed parties which elements from the various trades and specialist models are really required; MVDs create a very lean, clearly structured information requirement and avoid the need to transfer, manage and update all (and thus also unnecessary) information from all the involved trade models; both appointing and appointed parties benefit equally from an objective IFC model through defined MVDs, because less information in better quality strengthens everyone involved in the project

- **Set up the IFC modelling in a jointly coordinated manner**: for a well-coordinated, aligned information Model authoring, coordination and handover via IFC, the BIM Execution Plan (BEP) and the Master Information Delivery Plan (MIDP) play an essential role; by means of the BEP and MIDP, the appointed parties take on the organizational and procedural BIM topics of the Exchange Information Requirements and document on a technical level, among other aspects, how all trades and planning participants create a jointly coordinated ‘federatable’ IFC model and work with it; BEP and MIDP also promote the coordination of all

those involved in the planning before the model is created with regard to specific settings and processes (e.g. BIM coordination) in order to ensure that every trade and every party contributes to receiving a quality-optimized IFC export for the best possible overall IFC model; here are particularly important:

- Jointly agreed project settings and modelling approaches in the respective native formats, which have a direct effect on the IFC trade or technical model quality by means of which the MVDs are implemented
- decide about export settings that are coordinated with one another so that each IFC trade model can be optimally integrated into the overall model in a time-saving manner and with the best possible data completeness (e.g. for collision checks, quantity and cost calculations, AsBuilt BIM documentation, etc.)

Common Information Management, instead of "mutually assigning errors": during project processing, IFC "lives" primarily through joint creation, coordination and use of an IFC-based information model; it is important that all trades work together on the "common denominator" of IFC, both at the technical level and at the overall project level, so that the various appointed parties and involved specialists support each other to achieve the goals of an optimal IFC project; For appointed and appointing parties, when using BIM, the focus is on feasibility, quality in added value and implementation, as well as better productivity and the highest possible data completeness. ISO 19650 speaks about

- cyclical, defined information model submission from the appointed parties to the appointing party, for the purpose of appointing party acceptance
- yclical availability checks of reference information and of shared resources; generating information; complete quality assurance checks; review information (models) and approve for sharing

With these three 'IFC best practices', appointing parties and appointed parties can create the foundation for a solid, joint IFC application in projects. It is important that basic parameters such as IFC version (IFC 2.3, IFC 4.X), the Model View Definitions and the dedicated Use Cases including the relevant joint BIM model export settings are coordinated among all trades and project phases, so that the best possible IFC at both the technical layer and the overall project workflow level is enabled.

IFC application in projects – TUV SUD "IFC Quality Essentials" for self-checking BIM models

From the experience of the BIM team at TUV SUD, having audited and consulted on IFC projects across the globe, a total of three checking categories for the best possible IFC quality and "IFC Quality Essentials" can be derived. If these are observed jointly in the project, important - but of course not all - aspects for a real open BIM culture are properly been implemented:

1. Model Structure and Model Integrity

A project-specific, uniform model structure across all trades is important because, especially when using IFC, this item is the basis for all trade models to be coordinated with one another, e.g. for the creation of federated models as basement to perform cross-trade use cases, such as quantity take-offs, clash detection etc. Only If the model structure, including the naming of the parameters (IFC PSets), of all technical models involved in the project is uniform and consistent in accordance with ISO 16739 and BuildingSmart nomenclature, the federated models can be created with as little data loss as possible.

Risk values in this area have the effect that the IFC models cannot be used for automated design reviews and for technical applications, e.g. fire protection, calculations of pipe and sewage networks, energy calculations etc.

Here are some IFC checking best practices from TUV SUD to ensure your IFC models are set up for a project-specific, uniform model structure across all trades:

- Identical Common Project Basepoint: each discipline model should have the same global positioning. This is reflected by the model‘s Longitude, Latitude, Bottom elevation and rotation to True North; a common project base point the very first quality item and most essential towards the coordination and ‘checkability’ of a discipline model
- There shall be only one – and not several – IFCsite instance in each project; if a project is defined by more than one IFCsite instance, it cannot be guaranteed that the trade models are coordinated by one physical measurement point
- Ensure that there are only unique GUIDs in all the trade models and that there is no doubled GUID in one of the IFC models, which would indicate doubled elements, ending e.g. in false quantity take-offs and in unclear responsibilities e.g. towards clash cleaning
- When it comes to geometric integrity, check that there are no 2D objects integrated (or left) in the IFC models, as 2D elements do not accurately represent the geometry of the individual elements and also 2D elements are not exhibited during the clash detection
- Check the Grid Lines: Each discipline model should contain grid lines; trade models that are not standardized by a single grid system cannot guarantee cohesion
- No ProxyElements as component should be specified and found as IfcBuildingElementProxy; please consider a proper IfcEntity instead, to enable that further use case, such as fire concepts, pipe/duct calculations and cost counting can be properly executed

2. Modelling Guidelines

Harmonized modelling guidelines across all project IFC models are important, as this area is the foundation for proper engineering reviews that require a homogenous IFC setup to be passed on to manufacturing and engineering.

Risk values in the area of the modelling guidelines arise when the respective trade models of a project are structured differently, which leads to inconsistent, incoherent IFCs, so that continued use of the IFC models, e.g. for the construction phase and for operation, is unsuccessful.

With the following few checks, it is easy to create a cross-trade common IFC quality on modelling level:

- Reasonable offset to host storey: check that all components are created within a reasonable offset to its host storey, which you can easily check when specifying and code checking with a project-relevant setting
- Validate that all hosted components have a geometry: components which are decomposed by other components must have a geometrical representation
- Check that host component may not have geometry: components which decompose into other components may not have a geometrical representation
- Storey heights within limits (customized assets per each project) are also a criteria to check for proper Model Integrity, as checking distances between intermediate slabs (= storey height) is recommended to see whether slabs, selected by using the IFC Entities classification, indicate that the project is really modelled floor-wise; a very relevant general VDC item
- Check the Sum of Material Layer Thicknesses (Total Component Thickness); this check ensures that the sum of the material layer thicknesses is equal to the total component thickness; if the total material layer thickness of components is not equal to the geometrical thickness of the components, there may be problems in original modeling of the components, or in exporting of the component.
- Avoid bulky and too detailed models: check that the geometrical representation is not too detailed, to ensure that the project does not include components with overly detailed geometry which is indicated by a too much detailed LoD (Level of Development) which results in a very slow authoring or coordination effecting in a weak project productivity; you can set a maximum number of polygones suitable for your project and then let Model Checks run through each component to detect too many Polygons per Object Component
- Check that the Material of Decomposed Components are defined (only) on component level to indicate decomposed components (assemblies); that is important to extract correct quantity take-offs and correct material definitions
- Analyse that MEP components within the IFC model/s are connected to at least one other MEP component and that any MEP component is part of a system; this rule checks if all MEP components are connected to at least one other MEP component, which indicates that there are no undetermined or non-connected items, which would effect on Quantity Takeoffs and indicate that there are elements in the IFC models that are not (yet) part of a well-coordinated functional system
- Architectural model should have spaces: check that the architectural model/s contain/s space components and that every ‘space’ has a unique identifier; that avoids doubled or overlayed spaces which, in turn, would effect on false spatial quantities and incorrect room books later
- Openings in Complex Walls should be related to the wall, not to one element; openings in an IFC modell that do not completely cut through a multilayer wall run the risk of creating uncoordinated openings

3. Information Requirements

Uniform and well-structured information requirements are the basis for reliable information transfer between trades and to further life cycle phases, e.g. for BIM-based tenders, maintenance optimization or Design for Maintainability, Schedule Management, etc.

Quality errors occur the risk that uncoordinated, missed or non-aligned information leads to misinterpretations, duplications and incorrect information especially for BIM use cases that involve several disciplines and that are relevant for numerous life cycle phases, such as Design-to-Construction use cases or those ones for Construction-to-Operation.

To get a basement of IFC quality in the area of Information requirements, start a check for the following items and extend the checklist by project specific additional validations:

- Correct PSets: check that each element of the IFC trade model/s is defined with its correct PSet and that – initially - no individual property nomenclatures or property contents are added or overwritten; PSets as defined in the original BuildingSmart IFC documentation ensure that BIM projects start smoothy and well-coordinated, to avoid that some trade models are initially developed by BuildingSmart PSets, while other might already contain unique property structures or contents, which would then disable general information exchange and information processing on federated model level; as a help, check if components contain default ProperySets starting with „Pset_“ and take a closer look to all those items missing “Pset_” at the start
- Enable that each component is be defined by an IfcEntity, as that is important to properly work with the IFC Classifications according to ISO 16739 later on; in terms of IFC, layers and classification are not properties but actually ‘entities’; any entity is associated to other entities like IfcBoiler, IfcBuilding or IfcSpace through those important relationship
- Check that each component is be defined by an IfcType, as wrong or undefined types disable most of the BIM use cases
- Ensure that every component has a property “IFCAsset”; elements that are not defined by IFC Asset ID parameter/s are not identifiable to the Facilities Management
- Validate that each component is classified according to the IFC Type classification of BuildingSmart
- On attribute level, ensure that each component has a name, a Type, and a Material information, which adds usability of the IFC Project Information Models through clear human and machine-readable information, which is important to automate workflows e.g. with other programs or with Model Checkers
- Cross-check the Exchange Information Requirements and the BIM Execution Plan of the project with the applied generic IFC properties so that every required IFC property is present and also properly filled, e.g.
    - AcousticRating
    - FlammabilityRating
    - ThermalTransmittance
    - LoadBearing
    - FragilityRating
    - FireRating
    - etc.
- For accurate Quantity Takeoffs, check that the relevant IFC QuantitySets are present in every trade model and every relevant element, and also that the content is of the QuantitySets is accurately defined by the authoring tool (and not by human hand!); as an example, to extract proper Quantity Takeoff for walls right form the model, the following setup shall be checked: Pset_WallCommon. LoadBearing = TRUE and Pset_WallCommon.IsExternal = TRUE; also, check for the following consistencies:
    - Consistent Component Properties
    - Component Thickness Must Be Consistent
    - Component Profiles Must Be Consistent
    - Door and Window Dimensions Must Be Consistent
    - Door and Window Top Elevation Must Be Consistent
    - Wall Height Must Be Consistent
    - Column Length Must Be Consistent
    - Component Elevation Must Be Consistent
    - etc.

- Check that all project relevant Pset_BuildingStoreyCommon properties are onboard: as a basic Virtual Design & Construction (VDC) measure, every IFC model shall be developed storey-wise, to drive both design analytics and documentation use cases forward; take into account that several building attributes of Pset_BuildingStoreyCommon are handled directly at the IfcBuildingStorey instance; examples of important Pset_BuildingStoreyCommon properties are
    - EntranceLevel
    - AboveGround
    - GrossAreaPlanned
    - NetAreaPlanned
    - SprinklerProtection
    - SprinklerProtectionAutomatic
    - Pset_BuildingStorey BaseQuantities
    - NominalHeight
    - GrossFloorArea
    - NetFloorArea
    - GrossVolume
    - NetVolume
- Enable all relevant IFC models contain Compartmentation: checks that the components have property Compartmentation; missing Properties indicate ...

About the Author:
Tobias Schmidt is a renowned expert and BIM Director at TÜV SÜD. TÜV SÜD provides BIM Consulting and Advisory via a global network of experts who combine technical building know-how, business and process consulting expertise, and technology experience. BIM Consulting and Advisory by TÜV SÜD helps you to define the best feasible and profitable BIM strategies to implement proper Exchange Information Requirement (EIR) and BIM Execution Plan (BEP) as well as to optimise CAPEX and OPEX of your building.

EIR and BEP[17] by Peter Kompolschek

EIR and BEP are the core documents for successfully tendering and implementing BIM in a project.

Before analyzing ordering processes, some basic terms must be clarified:[18]

- Appointing party - as the receiver of information.
- Appointed party - as provider of information.[19]
- Appointment – as agreed instruction for the provision of information.

Usually, the appointment of information delivery is a three-step process, see fig. 1

Figure 1: workflow of tender process

Invitation to tender

The appointing party establishes exchange information requirements (EIR) for each lead appointed party's appointment, considering, where appropriate, the organizational information requirements (OIR), asset information requirements (AIR) and project information requirements (PIR).

An EIR is issued to each prospective lead appointed party being invited to tender for the relevant appointment.

Tender response

The prospective lead appointed parties respond to the EIR with a BIM execution plan (pre-appointment).

Appointment

When the lead appointed party is selected, the lead appointed party confirms the BIM execution plan and provides a defined set of information about the execution of the deliverables within its perimeter of responsibility.

EIR (Exchange Information Requirements)

The appointing party establishes the exchange information requirements to list all applicable information requirements. The exchange information requirements are provided to the potential appointed parties.
Where information requirements can state why, what, when, how produced, and for whom the information is needed (organization (OIR), asset (AIR) or project (PIR) information requirements). Informing the appointed party why the information is needed will allow them to innovate in the method of information production and delivery for the appointing party's business needs. Further information requirements should be a brief description of the purpose, the desired outcome and/or delivery of the appointing party's business need and information need.

BEP (BIM Execution Plan)

The BIM Execution Plan documents are updated by the Lead Appointed Party in agreement with appointing party and appointed parties to confirm the specifics that shall be used for the specific project. An information delivery strategy should reflect the lead appointed party's approach to meeting the information requirements as specified in the EIR. Also, the delivery team structure (overview of appointed parties) and/or the breakdown of delivery team into task teams are part of the delivery strategy and therefore should be specified here. The delivery team's information delivery strategy should also contain a set of objectives/goals for the collaborative production of information.

About the Author:
Peter Kompolschek is an Architect and a renowned BIM expert based in Austria. Next to his work as BIM advisor and manager for large architecture and infrastructure companies he is also an active member of several standard bodies, like the Austrian Standards, CEN and CELEC.

17. From "Guideline for the implementation of BIM Execution Plans (BEP) and Exchange Information Requirements (EIR) on European level based on EN ISO 19650 1 and -2"
18. All terms and concepts are in accordance with EN ISO 19650-1 and -2
19. A lead appointed party should be assigned for each delivery team.

AUTODESK®
REVIT

AUTODESK