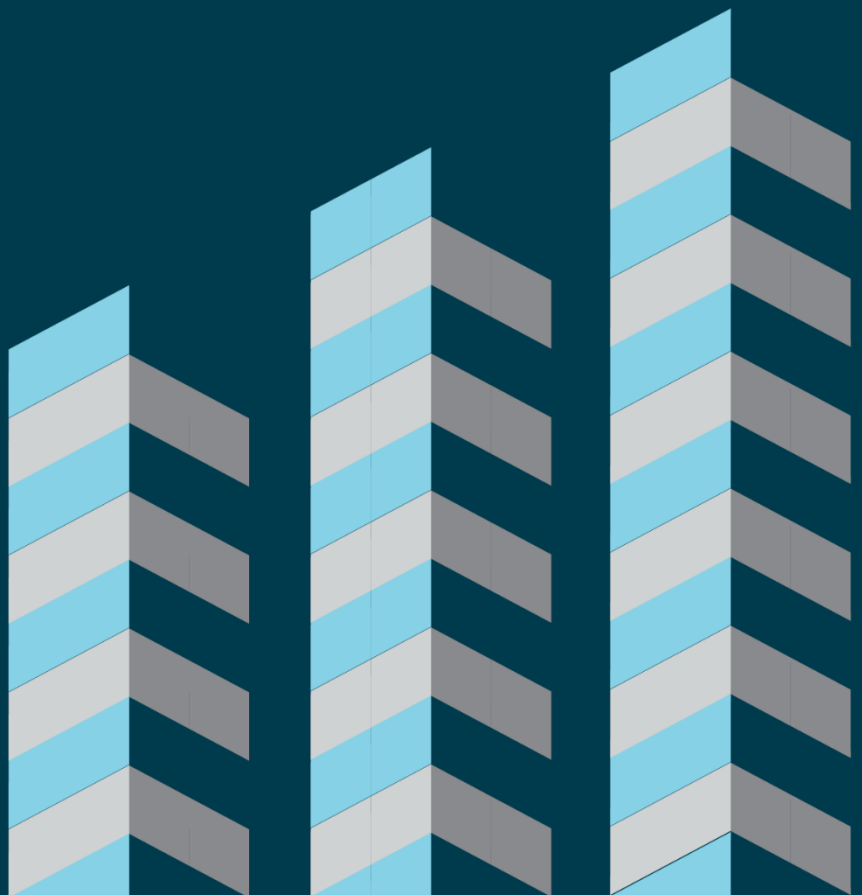


# DESITE BIM

## User Manual

Manage your 3D Building Model interactively.



The information and guidance in this manual is currently the most up to date as of January 10<sup>th</sup> 2024. Thinkproject assumes no liability whatsoever for the use of this software, infringement of patent rights or rights of any third party that result from the use of this software. Thinkproject reserves the right to make changes to this software, in order to bring about technical improvements in the software, at any time and without any prior notice. No part of this documentation may at any time or in any form whatsoever be reproduced (printed, copied or the like), edited or distributed in any electronic system or manner without the express prior written consent of Thinkproject Holding GmbH.

Release: January 10<sup>th</sup> 2024

### **Support Contact for DESITE BIM**

Telephone: +49 89 143 770288

Support: [Contact form](#)

## Document Versions

Date	Software Version	Editor	Comments
2024-04	3.4	ZM	Chapter 'Project Coordinates' added Chapter 'Shortcuts' modified Chapter 'Selection Sets' added
2024-02	3.4	ZM	Chapter 'Meaning of Check Run Status' added Chapter 'Quantity Take-off Sheet' added Chapter 'Tools' extended
2023-12	3.4	ZM, JB	Chapter 'Model Versions' added Chapter 'Project template' added Chapter 'Configuration' added Chapter 'Command line switches' added
2023-09-01	3.2	JB, ZM	Smaller corrections Chapter 'Bill of quantity' extended Chapter 'Clash detection' added Chapter 'Model check' added Chapter 'DESITE API' added Chapter 'Contacts' added Chapter 'Optimized Oriented Bounding Box' added
2023-02-13	3.2	JB	Smaller corrections  System requirements updated  Chapter 'Cutting Tool' added
2023-01-09	3.2	JB	Initial PDF version of DESITE user manual

1. Introduction.....	8
2. Installation .....	9
2.1 System Requirements .....	9
2.2 License.....	9
2.2.1 Named User .....	12
2.2.2 Concurrent User .....	12
2.3 Configuration .....	14
3. Getting Started .....	16
3.1 User Interface.....	16
3.2 Application Menu.....	16
3.2.1 Import Data.....	17
3.2.2 Project Template.....	19
3.2.3 Supported File Formats .....	21
3.2.4 Redirect Project Files .....	22
3.3 Customize User Interface.....	23
3.4 Navigation.....	24
3.5 Selection .....	26
3.5.1 Selection Settings .....	27
3.5.2 Selection and Visibility .....	28
3.6 Clipping Planes .....	29
3.7 Redlining .....	30
3.8 Measuring .....	31
4. Information Management .....	35
4.1 Domains and Models .....	35
4.1.1 Domain Hierarchy (tree structure) .....	36
4.2 Geometry Domain / Project Structure .....	37
4.3 Selection Sets .....	39
4.4 Building Structure.....	41
4.4.1 Building structure element tools .....	42
4.5 Rule based linking .....	44
4.5.1 Match types .....	46
4.6 Activities domain.....	50
4.6.1 Linking Activities to 3d Objects .....	51
4.6.2 4D Simulation.....	52



4.6.3	Customize 4D Visualisation.....	53
4.6.4	Export 4D Simulation .....	54
4.6.5	Creating Charts from Activities.....	55
4.7	Process Components .....	56
4.7.1	Calculation of schedules.....	58
4.7.2	Define Process Components .....	58
4.7.3	Restrictions .....	62
4.7.4	Structuring of schedules.....	63
4.7.5	Calendars .....	68
4.7.6	Process Component Properties .....	69
4.8	Bill of Quantities.....	70
4.8.1	Add Objects to BoQ Items .....	72
4.8.2	Calculate Quantity of BoQ Items .....	72
4.8.3	Quantity Take-off Sheet.....	73
4.8.4	Import/Export BoQ.....	75
4.9	Issues and Viewpoints.....	75
4.9.1	Redlining .....	77
4.9.2	3D Viewpoint.....	78
4.9.3	Issues.....	78
4.9.4	Issue Domain Configuration .....	80
4.9.5	Import and Export of Issues .....	81
4.10	Documents Domain .....	82
4.10.1	Place documents in 3D Space .....	84
4.10.2	Notes.....	86
4.10.3	Documents Domain properties .....	87
4.11	Types Domain .....	88
4.12	Clash Detection .....	91
4.12.1	Configure Clash Runs.....	93
4.12.2	Clash Detection Options .....	94
4.12.3	Defining objects for Clash Sets .....	102
4.12.4	Post Processing of Clashes.....	104
4.12.5	Printing clash reports.....	105
4.13	Model Check .....	107
4.13.1	Configure Check Runs .....	110
4.13.2	Check Run Options .....	112

4.13.3	Defining objects for Check Sets.....	112
4.13.4	Defining Check Run Rules.....	114
4.13.5	Meaning of Check Run Status.....	117
4.13.6	idPost Processing of Check Run results .....	118
4.13.7	Printing check result reports.....	119
5.	Project File Structure.....	122
6.	Working with Properties.....	125
6.1	Properties in the domain tree structure .....	125
6.1.1	Custom Properties Types .....	127
6.1.2	Filters.....	128
6.1.3	Formulas and Inheritance .....	132
6.1.4	Details View.....	137
6.2	Display Property Information.....	138
6.2.1	Quickinfo & Tooltips.....	138
6.2.2	Data Sheet .....	139
6.3	Selection Sets .....	140
6.4	Connecting to a Database.....	142
6.5	DESITE cp-Properties .....	144
6.5.1	General properties for all domains .....	144
6.5.2	Properties for Geometry domain.....	145
7.	DESITE API .....	151
7.1	Property Scripts.....	152
7.2	Scripts.....	156
7.3	WebForms.....	158
7.3.1	Creating own WebForm Modules.....	158
7.3.2	Navigation Bar .....	161
7.3.3	Entry points .....	162
7.3.4	Asynchronous API calls .....	164
7.3.5	Printing of WebForms.....	166
8.	Materials and Colors .....	170
8.1	Materials .....	170
8.2	Color schemes .....	171
8.3	Wire Mode Schemes .....	173
9.	Tools .....	175
9.1	Tool Functions .....	175

9.1.1	Optimized Oriented Bounding Box.....	175
9.2	Contacts & Clashes.....	178
9.2.1	Contacts.....	178
9.2.2	Clashes.....	180
9.3	Cutting .....	181
9.3.1	Using the Cutting Tool .....	181
9.3.2	Cutting Tool options.....	183
9.4	ID/Geometry/Attribute Check .....	185
9.4.1	ID Check.....	185
9.4.2	Geometry Check .....	185
9.4.3	Attribute Check.....	186
9.5	Model Versions .....	189
9.5.1	Activate Model Version Comparison .....	189
9.5.2	The Result of Comparison .....	190
10.	Command Line Switches.....	193
10.1	Passing Parameters .....	193
10.2	Parameter Reference.....	194
10.2.1	file.....	194
10.2.2	srvPort, cltPort .....	194
10.2.3	showWidget.....	194
10.2.4	webFormLocations .....	195
10.2.5	unlockWebForms .....	195
10.2.6	lang.....	195
10.2.7	scriptDebug .....	196
10.3	Availability in DESITE Applications.....	196

## 1. Introduction

DESITE md is an analysis and information software that facilitates the use of digital building models and improves collaboration between project participants. DESITE md helps you with your day-to-day work with building models and enables you to access the information in your BIM model.

DESITE md enables you to check your building models interactively and perform rule-based model checking using your own rulesets. You can also calculate quantities from model geometry and link schedules to the building model to further analyze and visualize your schedule. After all, md stands for 'manage data'.

With DESITE md, you can visualize, check, expand and analyze a 3D building model interactively.

From the very beginning.



## 2. Installation

### 2.1 System Requirements

The following system requirements apply to all applications of the DESITE line of products. Note that, while the given 'minimum requirements' suffice to run the software, the performance will heavily depend on your projects' size. Hence, even the 'recommended requirements' should be regarded as open-ended.

We suggest using two monitors when working with DESITE md or DESITE md pro because the increased screen space facilitates the software use and navigation. General screen requirements are not applicable to the tablet-oriented DESITE touch application.

It is highly recommended to install the specific graphics drivers provided by the manufacturer and keep them updated.

Please note that while it's possible to use DESITE BIM in virtualized environments, they are not officially supported.

	Minimum Requirements	Recommended Requirements
<b>Operating System</b>	Windows 10 64 Bit or newer	Windows 10 64 Bit or newer
<b>CPU</b>	64 Bit E.g. Intel Core i5 2500 / AMD Athlon 200GE	64 Bit E.g. Intel Core i9 9900 / AMD Ryzen 2700X
<b>GPU</b>	Integrated Graphics Card, OpenGL 3.0 compatible  E.g. Intel HD Graphics 2000, Nvidia 920 MX, AMD internal GPU with Vega 3 CU or newer	Dedicated Graphics Card, OpenGL 3.0 compatible  E.g. Nvidia GTX 1660 / AMD Radeon 570X
<b>Memory</b>	4 GB RAM	32 GB RAM
<b>Storage Space</b>	700 MB	700 MB
<b>Screen Resolution</b>	1920 x 1080 Full HD	2560 x 1440
<b>Other</b>	Internet connection required*	Internet connection required*

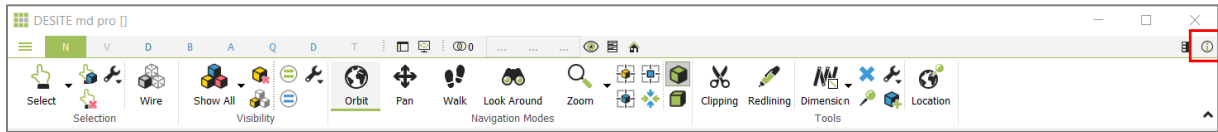
\* Depends on chosen license

### 2.2 License

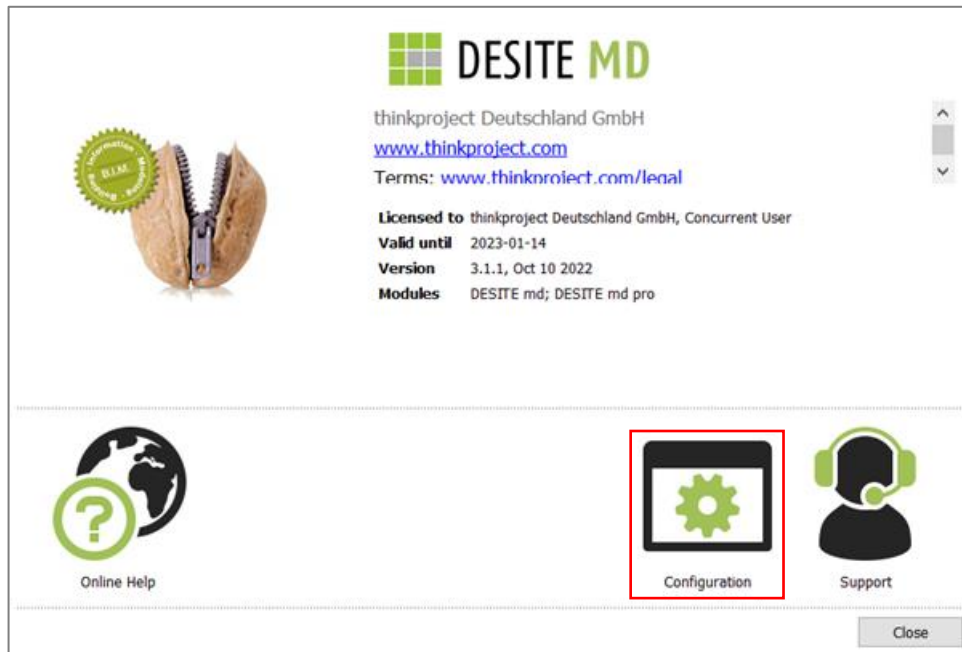
After the first installation of DESITE, the license window appears automatically at start-up.

If you are already working with DESITE and want to enter a new license key:

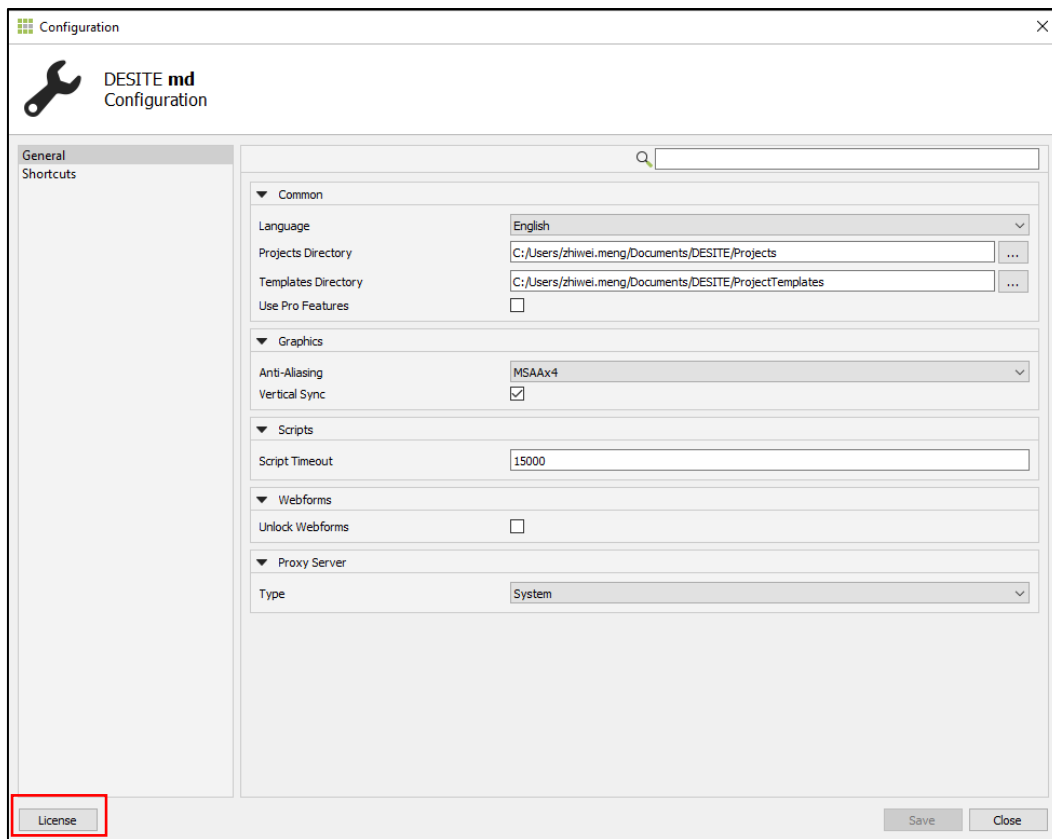
1. Click on the Info-icon in the upper right corner.



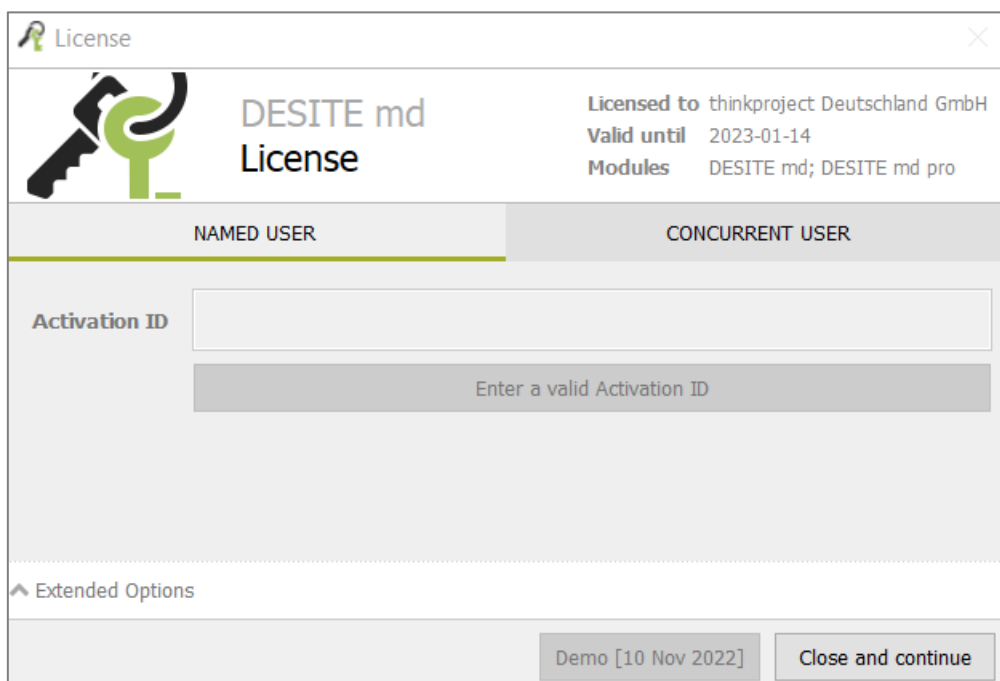
2. Then click on the Configuration button



3. Click on the license button on the bottom left



Depending on the type of license you have purchased, select the NAMED USER (user license) or CONCURRENT USER (network license) tab as appropriate and enter your license key.



If you would like to test DESITE first, click on the Demo button. This will generate a test version with a runtime of 14 days.

### 2.2.1 Named User

NAMED USER licenses are assigned to a specific user and may be registered on a maximum of two devices for this user.

The license is activated via an Activation ID. This consists of 32 characters and includes both letters and numbers. Every fourth character is separated by a hyphen.

To activate the license, an active Internet connection is required. Afterwards, no Internet connection is required to use the software.

The validity of the license depends on the duration of the contract. If the contract is extended, a new Activation ID is required.

### 2.2.2 Concurrent User

CONCURRENT USER licenses are network licenses. They are not assigned to a specific user. The number of concurrent users depends on the number of licenses purchased (at least two). For example, if ten CONCURRENT USER licenses are available in your company, ten employees can work with DESITE simultaneously.


The activation of the license is done by an Instance ID. This consists of 12 characters and contains both letters and numbers (without hyphen).


To activate the license or to start DESITE, an active internet connection is required.

The validity of the license depends on the duration of the contract. If the contract is renewed, a new instance ID is not required.

If you want to use the license without internet connection, you have the possibility to borrow a license ID. To do this, click on *Extended Options* on the CONCURRENT USER tab and then on Borrow. Specify the duration of the borrowing here.



 License



**DESITE md**  
**License**

Licensed to	thinkproject Deutschland GmbH
Valid until	2023-01-14
Borrowed until	2022-10-13 01:59
Modules	DESITE md; DESITE md pro

NAMED USER

CONCURRENT USER

Instance ID

License checked out

☐ Use a local (on-premise) license server

Extended Options

Proxy Settings

Borrow

Borrow (permanently check out) license until

12.10.22

Return borrowed License

Return License (Check In)

Borrow successful.

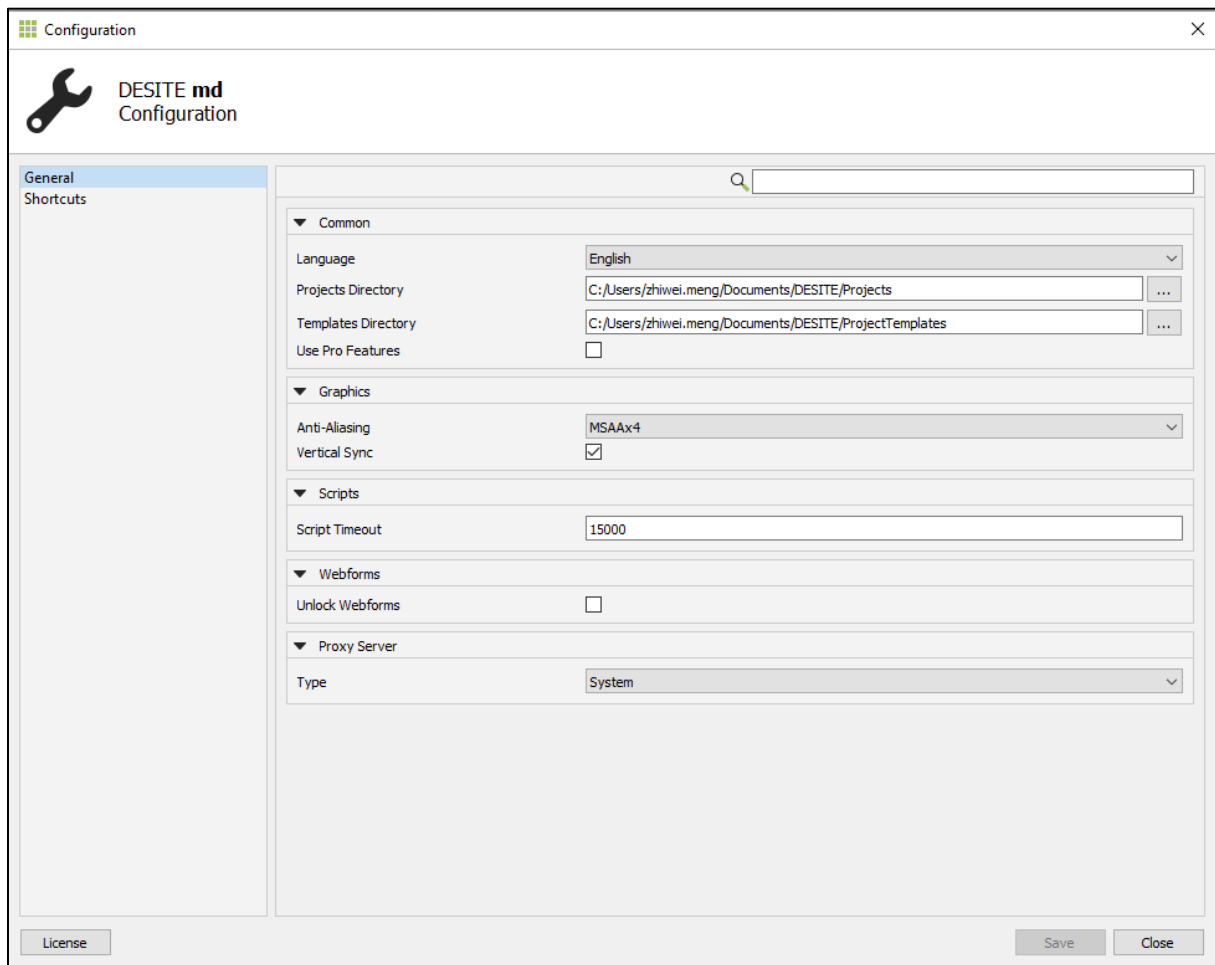
Demo [10 Nov 2022]

Close and continue

**Notes:**

A CONCURRENT USER license can be borrowed for a maximum of 14 days. Borrowing can only be done with an active internet connection.

## 2.3 Configuration



From version 3.4 the configuration panel consists of general and shortcuts. In the general configuration, it is possible to:

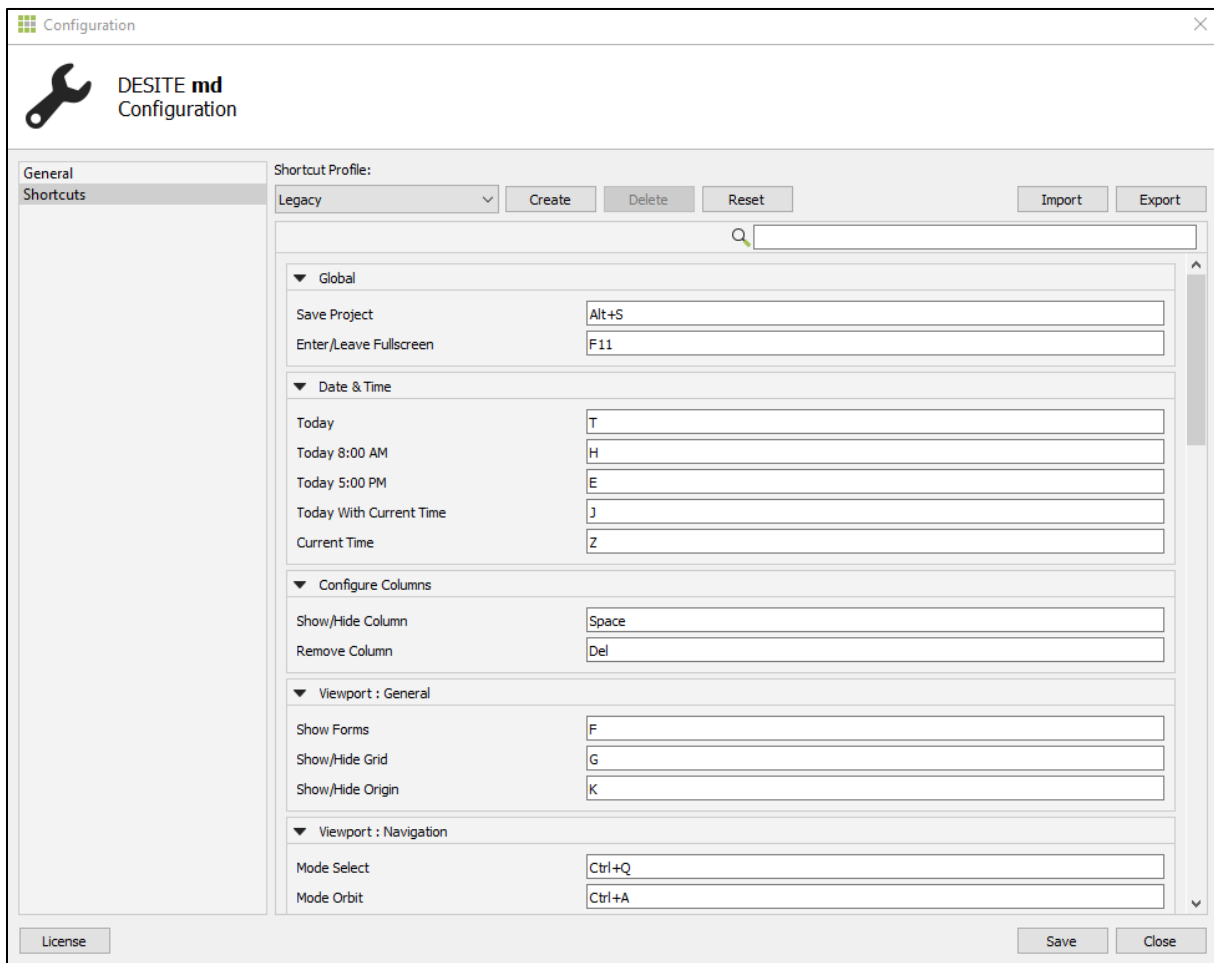
1. Choose a language from English, German or French
2. Define the default directory for projects as well as project templates
3. Choose whether to use the features of DESITE md pro or not
4. Configure anti-aliasing and vertical sync
5. Define script timeout (please refer to the tooltip for a possible range of the value)
6. Always unlock webforms
7. Configure proxy server

All the settings in configuration panel are automatically saved in a config file. There is a folder for the respective product, for example "desiteMD2", in which a json file named desite.config is located. The path could possibly be:

C:\Users\[username]\AppData\Roaming\thinkproject\desiteMD2. However it is not recommended to change or add any settings/tags in this file manually. Please always remember to remain a backup of original configuration file if you want to make any trial in the json file.

Each time after saving the settings, the configuration will not be updated until DESITE is restarted. If the settings need to be restored, you can just completely delete the file

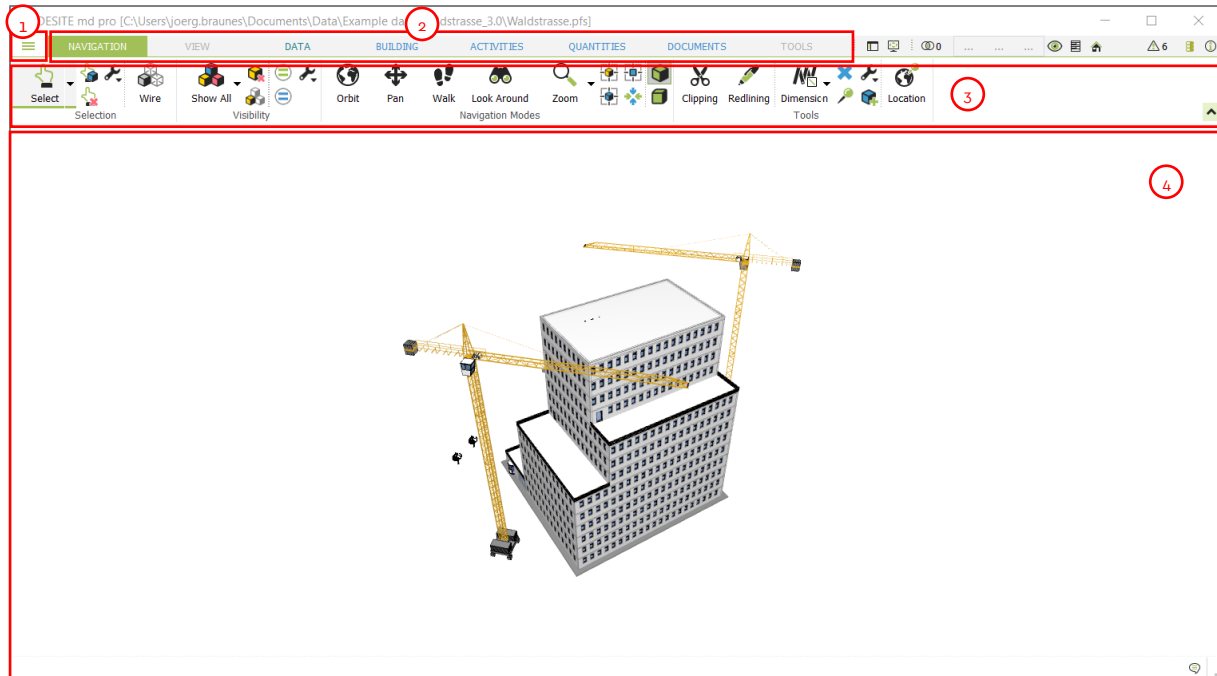
under the path which was mentioned above. But you need then to give again all the license information after the restart.



Whereas in the shortcuts, you can choose using legacy or default profile. In the DESITE version from 3.4, all the defined shortcuts are listed with editable text fields. You can easily edit on one of them and save the changes. Furthermore, you could also create a new shortcut or reset the definitions. To share the definitions with others, it is possible to export or import the file of shortcut profile.

## 3. Getting Started

### 3.1 User Interface



1. Application Menu
2. Ribbon Tabs
3. Ribbon Bar
4. Main Window shows a 3D view of the active project

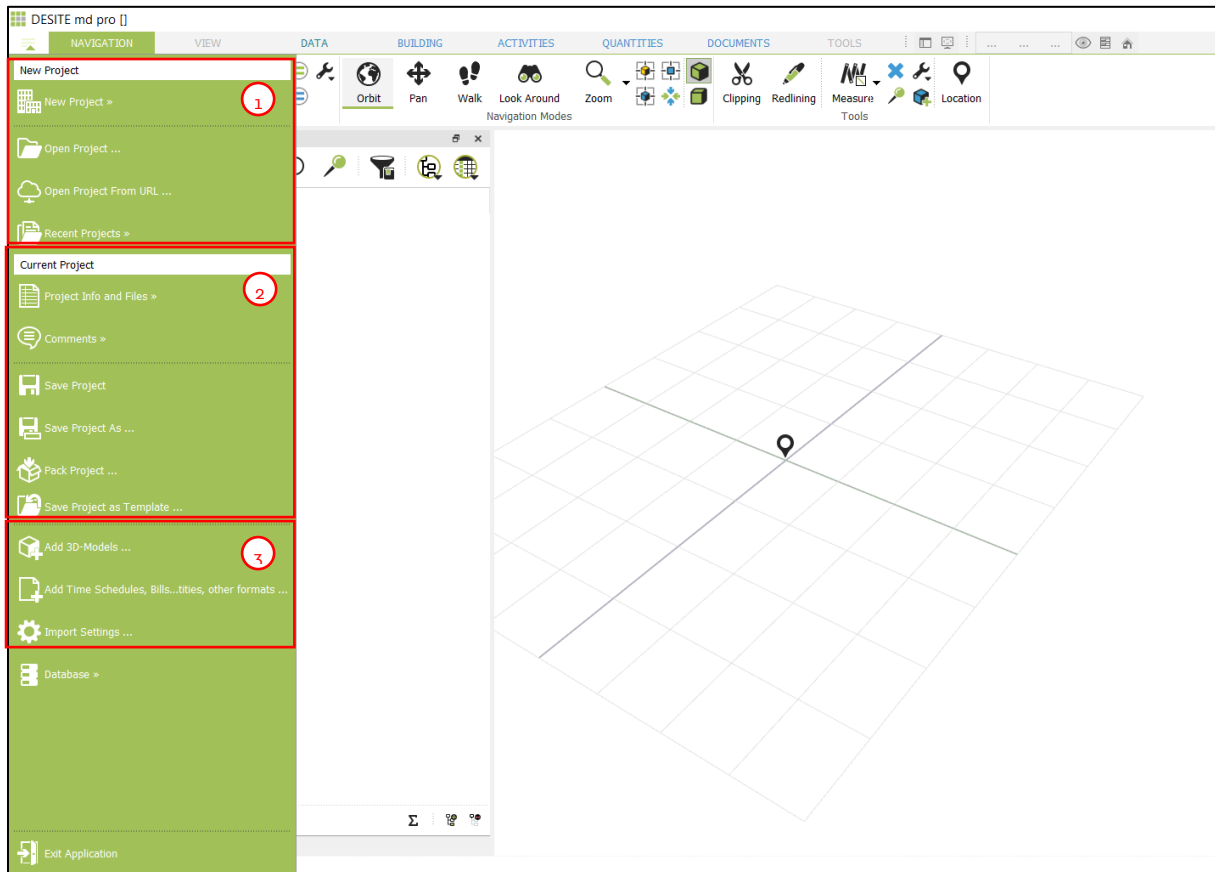
Ribbon Tabs:

- **NAVIGATION:** Navigation and Object Selection options; Redlining and Measurement tools
- **VIEW:** Issues and 3D Viewpoints, Materials and Color schemes, Visualization options
- **DATA:** Display and Entry of Attribute information corresponding to 3D model objects
- **BUILDING:** Project Structure, Building Structure and Model Export
- **ACTIVITIES:** 4D Simulation, Time Schedules and Process Components
- **QUANTITIES:** Bill of Quantities
- **DOCUMENTS:** Managing Documents and Linking to 3D model objects
- **TOOLS:** Model Checking, Clash Detection and Model Version Comparison

### 3.2 Application Menu

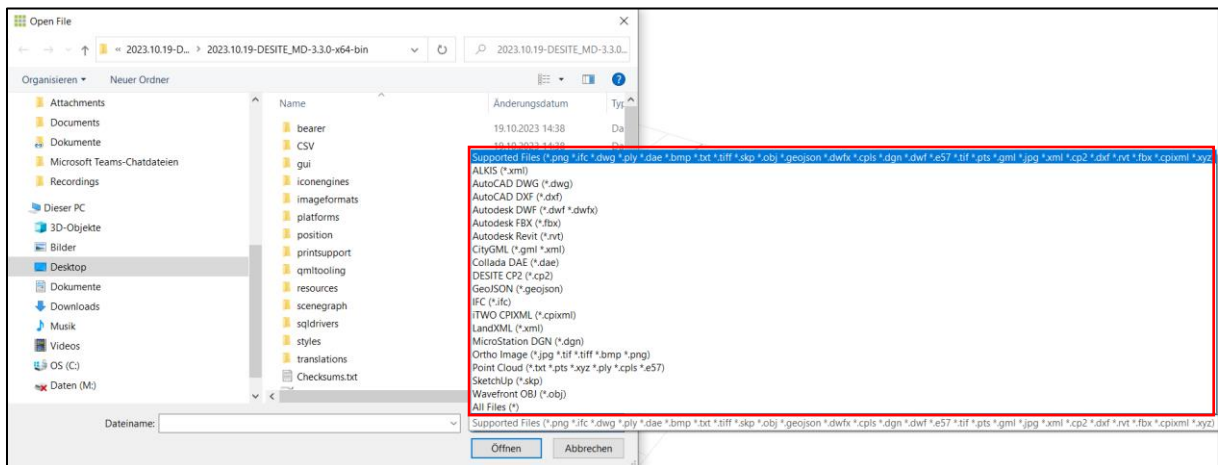
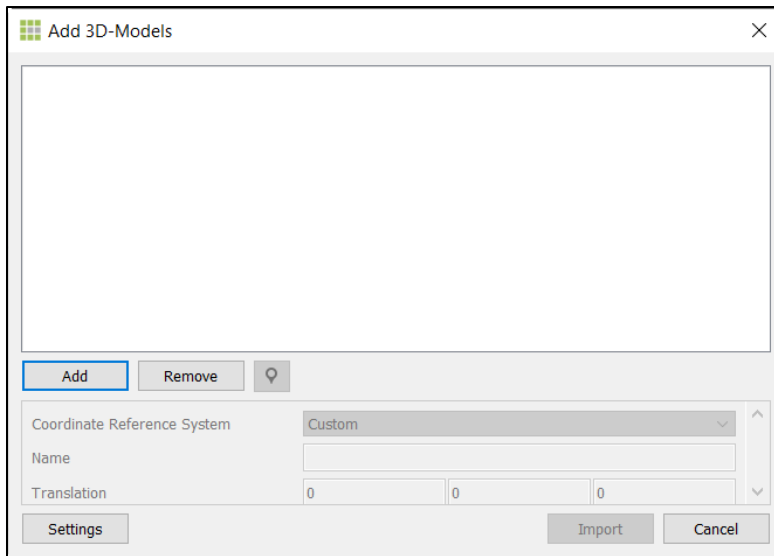
The Application Menu contains all the important functions for managing projects.

1. You can create an empty New Project or open existing projects in native formats .cpa, .cpzip or .pfs. In this case, click on Open Project.
2. You can display Project Info for the current project, export individual models, save the entire project or save current project as a template.
3. Whether a 3D geometry model, time schedule, bill of quantities or database, you can import any project data into the project using the function Add 3D-Models and Add Time Schedule, Bills.... Also it is possible to adjust import settings and save it in the project.

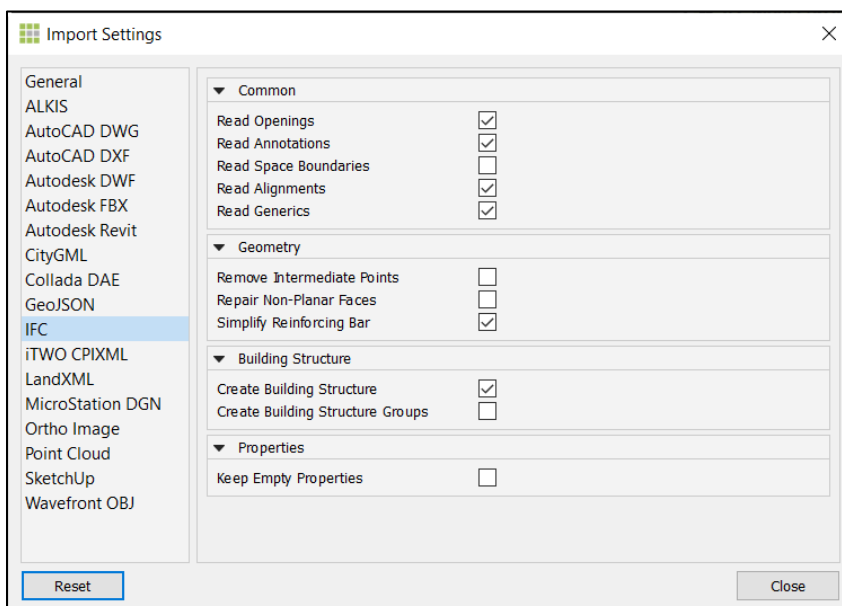


### 3.2.1 Import Data

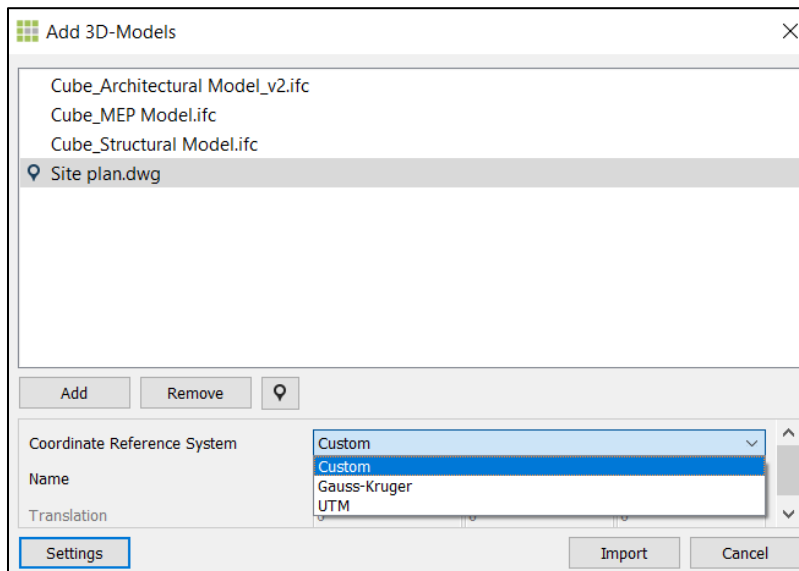
In order to import a file, you can either select Add 3D-Models... or Add Time Schedules, Bills... under Application Menu. You can choose the file format to be imported in the file dialog box that is opened.



When you click on the Open button, data will be imported into DESITE md. In the settings, you can select different options for various file format. Once the settings are defined, it would be applied by all the imported models. You can also save the settings into project templates and use it in other projects.



Multiple models of different types can be imported and put together in your project. When several 3d-models are imported, it is possible to choose one of them as basis for georeferencing and to select the coordinate reference system if it has been acknowledged, such as Gauss-Kruger or UTM system. If a Gauß-Kruger or UTM system is selected, the calculated GPS coordinates are available under "Navigation - Location". If the model does not use any of these systems, you can select "Custom" instead.



### 3.2.2 Project Template


With DESITE MD, you can save a project as a project template and create new projects from this template.

The advantage is that you only have to define macros, scripts, bookmarks, webforms, import settings, etc. in the template once, and these are available to you later in all projects created from this template.

To create a project template, you should first save the project, which you want to use as project template. You can freely choose the name and storage location. Then you can choose in the application menu "Save Project as Template".

In the dialog box that appears, you must give the template a name. The default path of project template is at C:/Users/[username]/Documents/DESITE/ProjectTemplates . Optionally, you can add a description. In the lower part of the dialog box, you can select which parts of the project should also be part of the template. Then click on "Save template".

Save as Template

 Saving Project as Template

Project Name: Waldstrasse

Template Path: C:/Users/.../Documents/DESITE/ProjectTemplates

Template Name: template


Description:

Files	Description
<input type="checkbox"/> Config.Max Bögl	
<input type="checkbox"/> My Documents	
<input type="checkbox"/> Originals	
<input checked="" type="checkbox"/> Waldstrasse.automation	
<input type="checkbox"/> Waldstrasse.clash	
<input type="checkbox"/> Waldstrasse.config	
<input checked="" type="checkbox"/> Waldstrasse.global	Global bookmarks and settings
<input type="checkbox"/> Waldstrasse.log	
<input type="checkbox"/> Waldstrasse.messages	Project update log
<input checked="" type="checkbox"/> Waldstrasse.prgs	Property Programs
<input checked="" type="checkbox"/> Waldstrasse.scripts	Scripts
<input type="checkbox"/> Waldstrasse.session.plani.kivindi	Session bookmarks and settings


Save Template Cancel

To create a new project from the template, click on " New project" in the menu. At the bottom of the " Template " section, you will see a list with the names of all defined templates. Click on the name of the desired template to select it. Hint : If no templates or not all templates are displayed, click on "Refresh".

New Empty Project

 Empty Project

New Project

 New Project [WARNING] Project Directory is empty

Number and Description

Project Number

Short Description

Path and Name

C:/Users/.../Documents/DESITE/Projects

Project Directory / Project Name

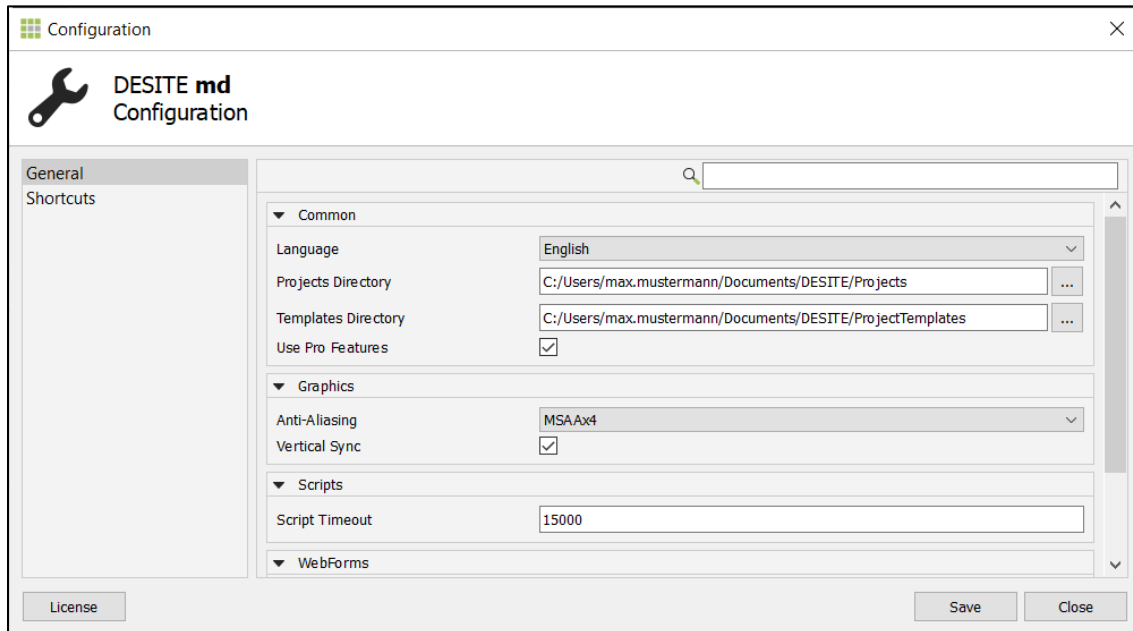
Template

Name	Description
waldstreet	C:/Users/.../Documents/DESITE/ProjectTemplates/waldstreet
zhm	No description.
zhmtest	

The "New project" button creates a new project using the template, but as you can see, the button is grayed out, i.e. inactive. Reason: unlike with the "Empty project", you must specify the storage location and name here when creating the new project so that the template files can be copied there. So please specify a project directory and a project name in the "Path and name" section. The "New project" button is activated when all the necessary information has been entered. The complete path to the new project is then displayed again. Click on the button to create a new project from the specified template at the specified location.



In the configuration, you can set the standard paths for your projects and project templates. The default paths are stored in the configuration file Config.desite.xml. To start the configuration, hold down the Shift key during the start of DESITE MD is started. If you want to change the default paths for saving your projects and project templates, you can do this by clicking on button next to the path specification.



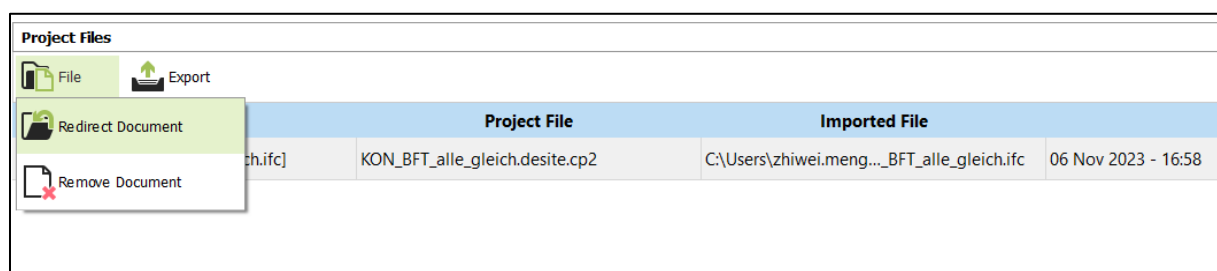
### 3.2.3 Supported File Formats

DESITE Project (*.pfs *.cpa, *.cpzip)
DESITE Binary Format 1 (*.cpb) / DESITE Binary Format 2 (*.cp2)
AutoCAD Drawing (*.dwg)
AutoCAD Exchange File (*.dxf)
Autodesk DWF (*.dwf *.dwfx )
Autodesk FBX (*.fbx)
Autodesk Revit (*.rvt)
CityGML 1.0/2.0 (*.gml, *.xml)
Collada DAE 1.5 (*.dae)
CPI XML (*.cpixml)
IFC 2x3, IFC 4, IFC 4.1, IFC 4.2, IFC 4.3 RC2 (*.ifc)
LandXML (*.xml)
MicroStation Design (*.dgn)
Polygon File Format (*.ply)

REB (*.d45 *.d49 *.d58)
SketchUp (*.skp)
Wavefront OBJ (*.obj)
DESITE Point Clouds (*.cpls)
E57 Point Clouds (*.e57)
General 3D Point Clouds (*.xyz)
Leica LeiserScan (*.txt *.pts)
ALKIS (*.xml)
GeoJSON (*.geojson)
DESITE Activities (*.tsxml)
Asta Powerproject (*.pp)
Microsoft Access (*.mdb *.accdb)
MS Project (*.mpp) / MS Project XML (*.xml)
pro-Plan (*.ppl)
Tilos Exchange (*.xml)
Primavera P6 (*.xer, *.xml)
DESITE Bill of Quantities (*.boq.xml *.modBOQ.xml)
GAEB XML X81-X86 (*.x81 *.x82 *.x83 *.x84 *.x85 *.x86)
GAEB 90 D81-D86 (*.d81 *.d82 *.d83 *.d84 *.d85 *.d86)
CPI CSV (*.csv)
SQLite DB (*.db)

### 3.2.4 Redirect Project Files

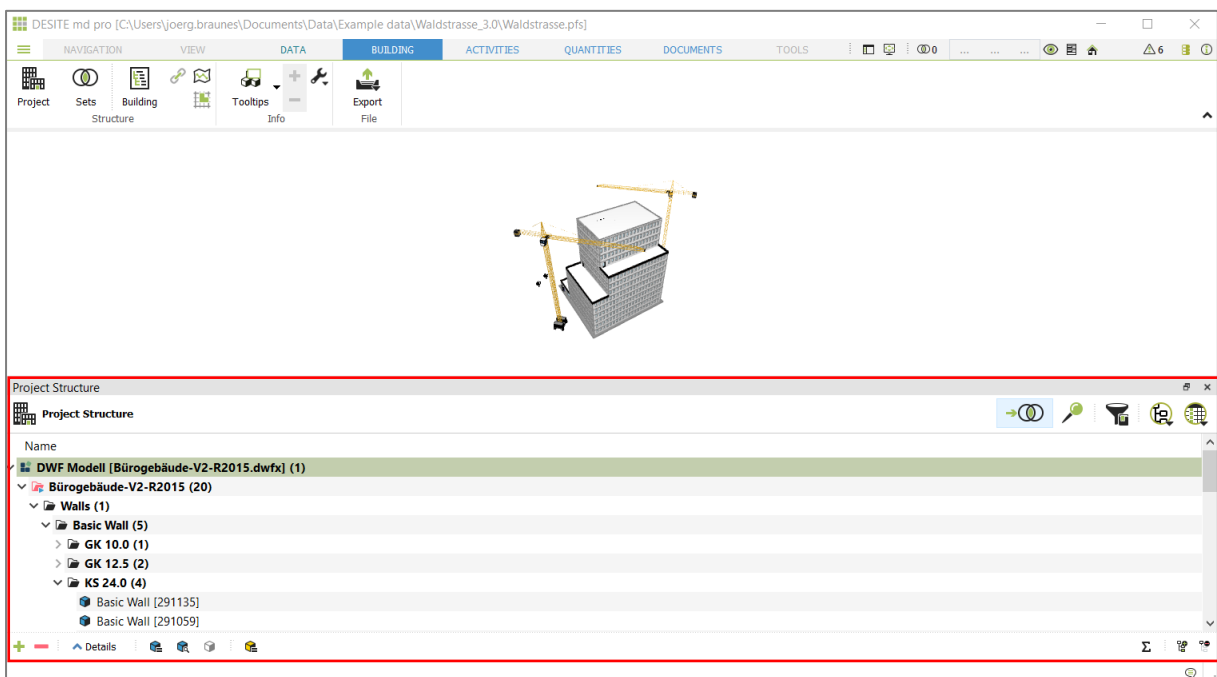
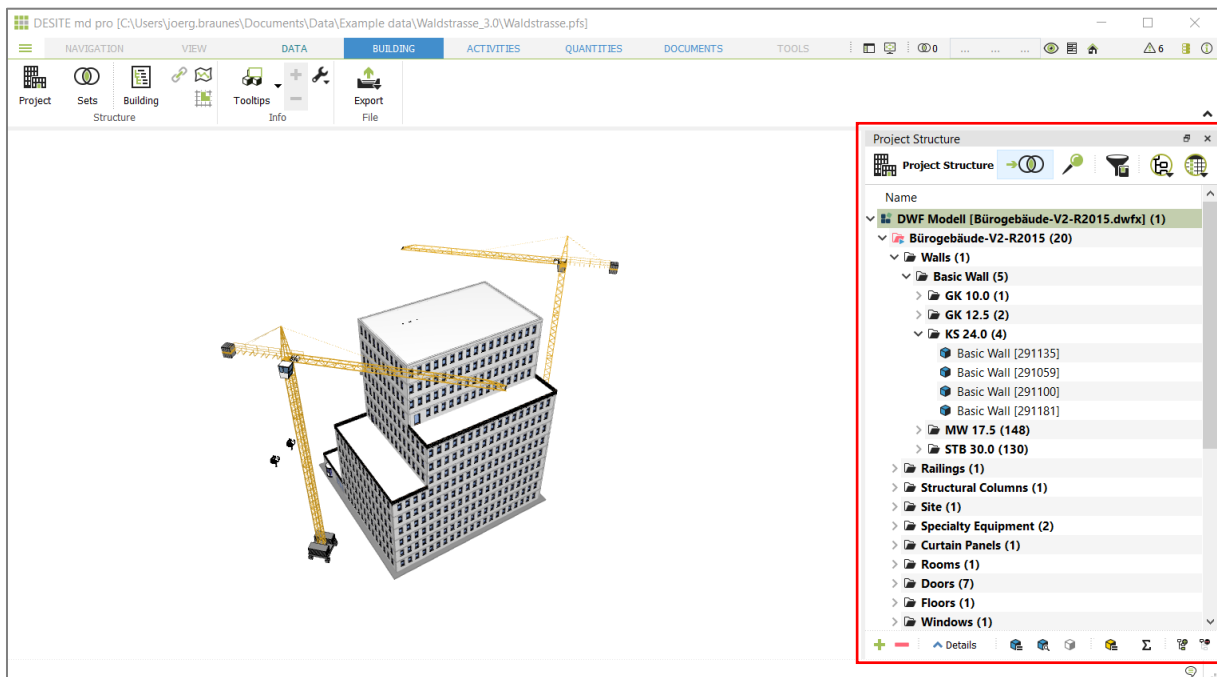
If you want to update one model or file which has already been included in the project, you can use the function of redirection. You can find it under "Application Menu – Project Info and Files". A list of all files in project is shown under "Project Files". There you can choose one file and click on the button for redirection:



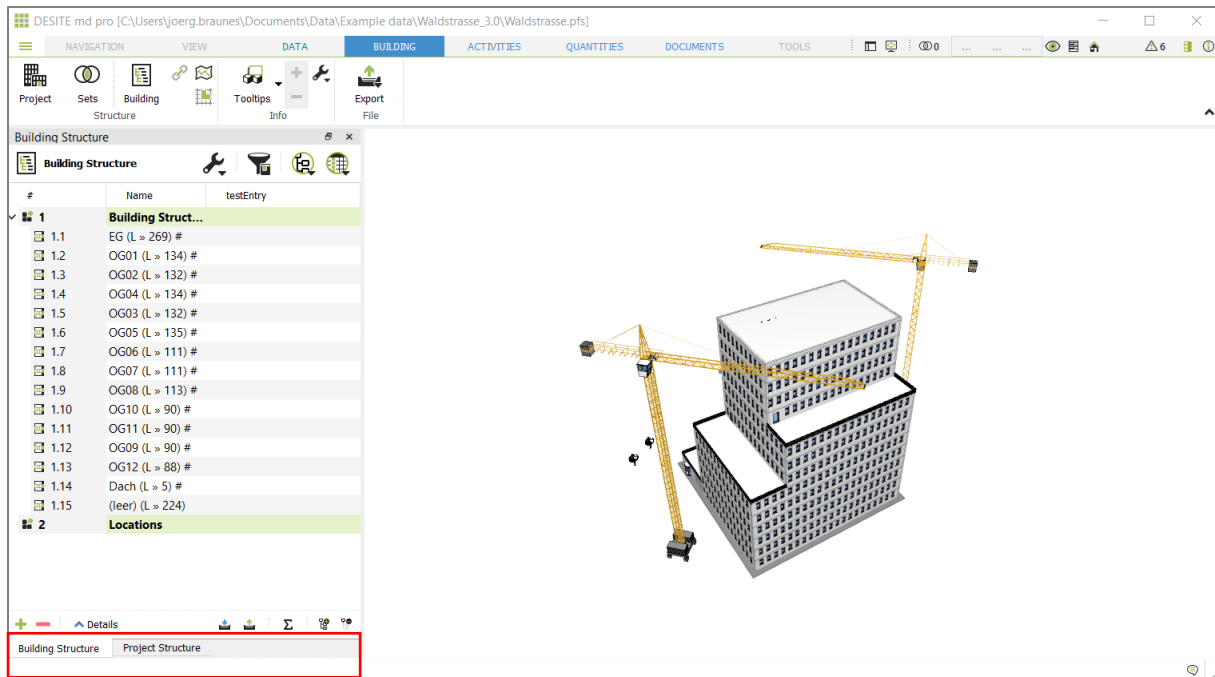
If the project is loaded the next time, DESITE will ask you, if you want to update the related model from the new file location. This updates the model with all its geometry and properties from the file, but keeps all existing links and additional custom properties etc. that were already saved in the project.

### 3.3 Customize User Interface

When a module (e.g. project structure) is selected, a new sidebar ('widget') will open. These sidebars can be moved and docked with drag-and-drop or used on a separate window (e.g. on a second screen) as desired:

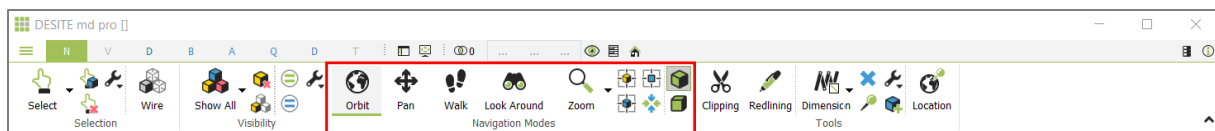


It's possible to place multiples widgets to the same sidebar. This creates tabs for widget:



### 3.4 Navigation

There are various Navigation Modes available to explore the model in 3D View:



In Orbit mode, you can rotate the camera around your model and view it from different angles. Press and hold the left mouse button and move the mouse to the left and right to rotate or up and down to tilt the camera.

**Tip:** You can left-click on an object to set the pivot point while rotating the model.



Pan moves the camera on screen level, i.e. left, right, up or down.

**Tip:** The arrow keys can be used to pan in any navigation mode. A similar effect can be achieved by pressing and holding the mouse wheel and dragging the mouse.



In Walk mode, you can explore the model in detail. Use the mouse to move back and forth and to turn to the left or right. The further you slide the mouse with the mouse button pressed, the faster you will move in the scene.



When using Look Around, your position remains fixed, and you can freely pan the virtual camera using the mouse.

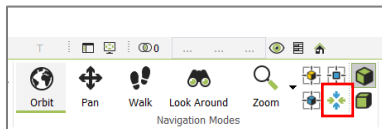
**Tip:** You can also look around in any navigation mode by using 'Ctrl' key and the left mouse button together.



You can use Zoom mode by clicking on the icon on the left side of the main window and moving the mouse back and forth. Alternatively, you can also use the mouse to draw a frame around the area to be zoomed in: left-click and hold on the center of the target area and drag the mouse in any direction until the frame is of the desired size.

**Tip:** In Orbit mode, other navigation modes can also be used temporarily pressing the mouse wheel and right mouse button or using left mouse button in combination with 'Ctrl' key.

Standard views are available for a practical and quick overview of the model.



Clicking on an arrow will set the view to the selected position.



In addition, following predefined views are also available:



View will be adapted to visible objects



View will be adapted to selected objects



Selected or visible objects will be displayed from above



Perspective projection (recommended)



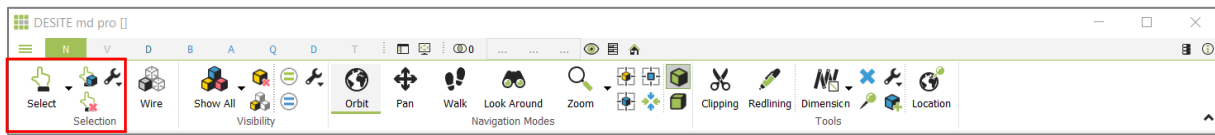
Orthographic projection

**Tip:** To switch between navigation modes more quickly, use the 'Ctrl' key in combination with the following keys:

- **Q:** 'Select' mode
- **W:** 'Pan' mode
- **A:** 'Orbit' mode
- **S:** 'Look around' mode
- **X:** 'Zoom' mode
- **Y:** 'Walk' mode

### 3.5 Selection

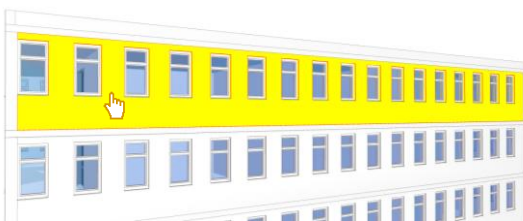
Selection allows the user to select 3D objects in the model in order to work on them.



Select objects in 3D view

There are three selection modes in the 3D View:

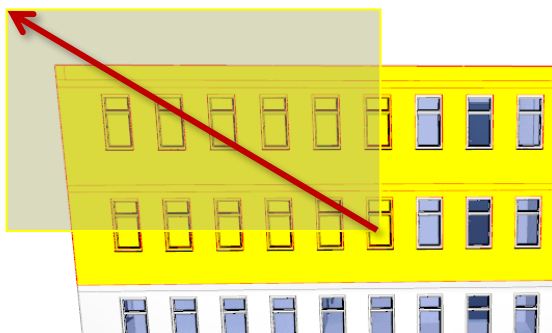
1. Individual selection by clicking on an object.



2. Box selection: **Left-click and hold, and drag the mouse down and to the right.** The selection box will be green. All objects which lie entirely within the box are selected, even if they are concealed by other objects and not visible.



3. Box selection: **Left-click and hold, and drag the mouse either to the left, upwards or both.** The selection box will be yellow. All objects which lie partly within the box are selected. Concealed objects are not selected.



### Tip:

You can select multiple objects one after another if you press and hold 'Ctrl' while clicking on objects.



Select all visible geometric elements



Clear selection

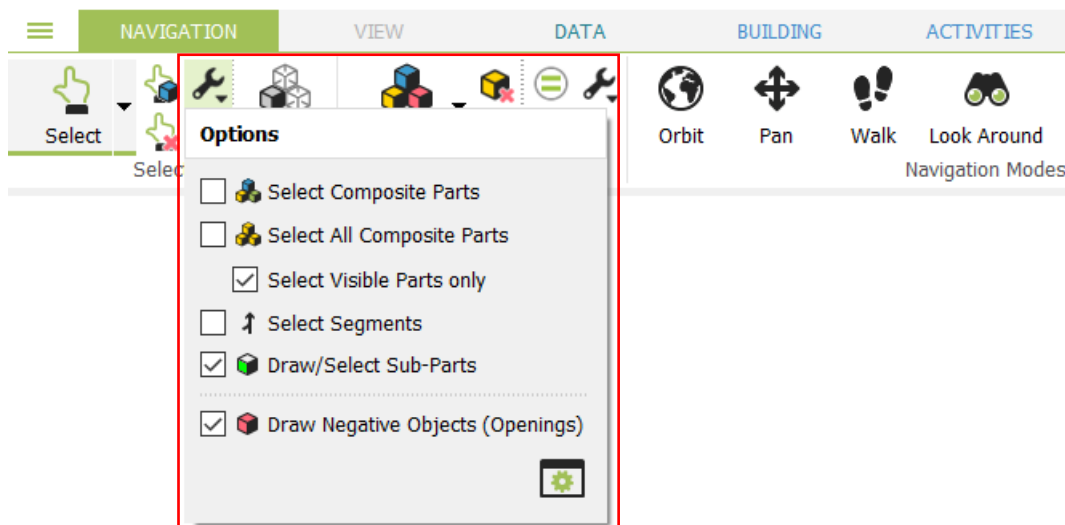
### 3.5.1 Selection Settings

Composite objects (assemblies) are distinguished in many geometry models.

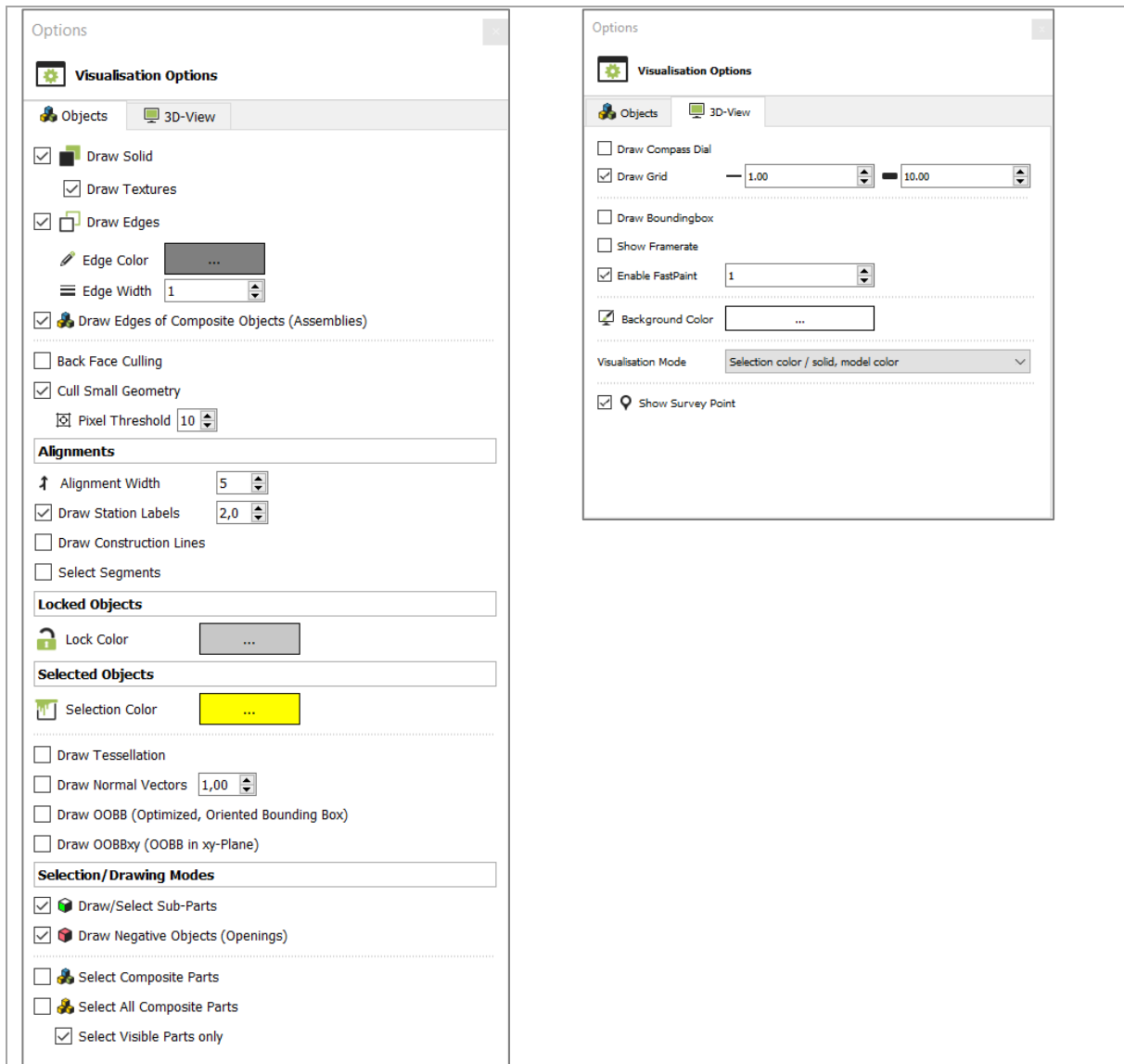


Selection type of such Composite objects can be specified in the navigation menu or in display options (under View / Options). Here you can also activate display of sub-parts and negative objects such as window or door openings (if they are present in the project).

**Note:** Sub-parts are components that have been divided by **Cutting** tool in DESITE md.

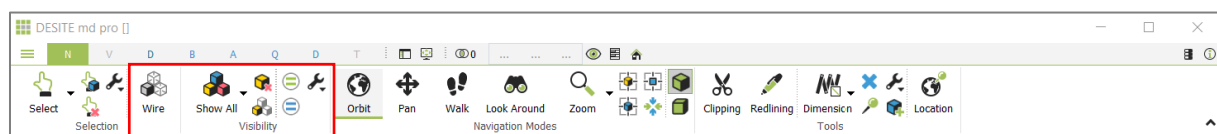


Via 'Open Options' button, you can select further display options in the tabs Objects and 3D-View.



### 3.5.2 Selection and Visibility

Display and visibility of objects can be changed depending on their selection status.

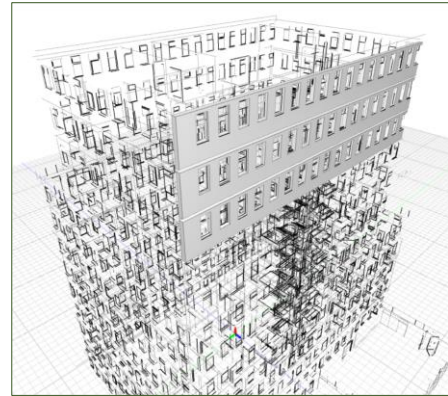


With *Wire mode*, only edges of objects are displayed in 3D view. Click on the button again to deactivate this display mode.





If objects are selected, they are still displayed with their faces. This allows a quick visual inspection of individual selected objects.



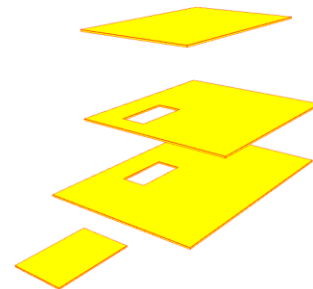
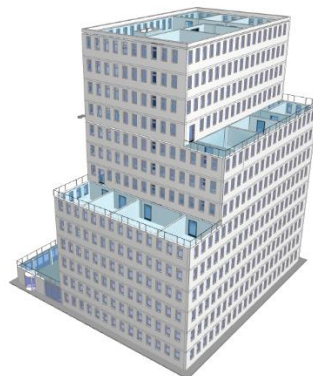
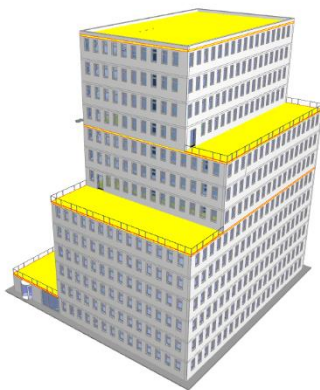
All 3D model objects in the project will be displayed. You can also press 'A' key as an alternative.



Selected objects will be hidden. You can also press '-' key as an alternative.

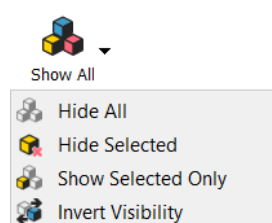
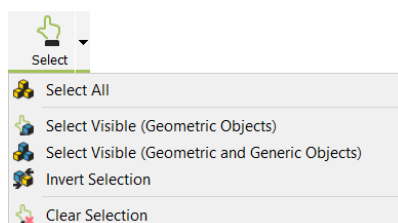


Only selected objects will be displayed. You can also press '#' key as an alternative.



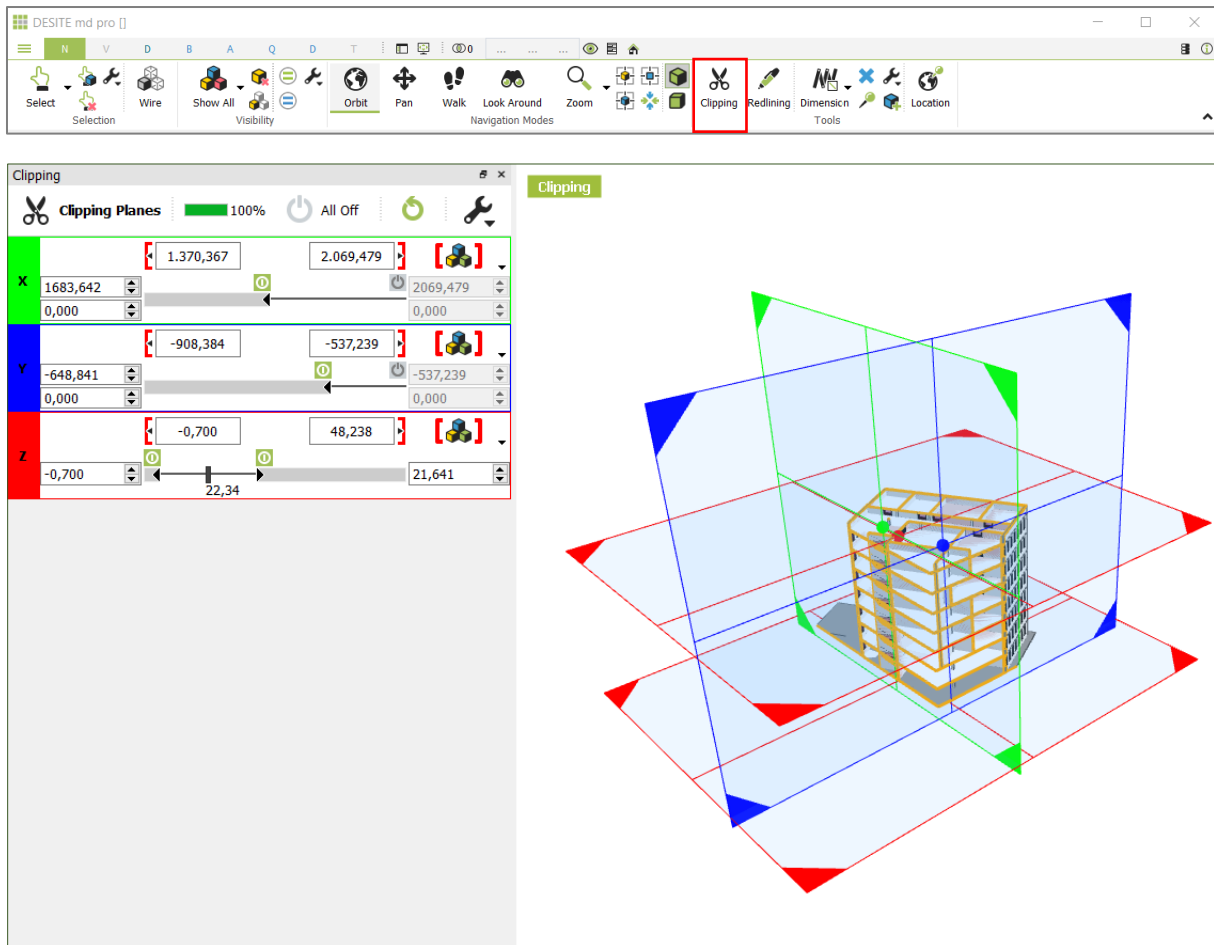
### Tip:

'Selection' and 'Visibility' of model objects can also be inverted. Selected objects can be hidden, displayed or isolated and vice versa. You find these additional options in the drop down of the *Select* and *Show all* buttons:



## 3.6 Clipping Planes

With Clipping function, parts of the models can be cut away up to six planes simultaneously.



Activate the desired clipping plane using On/Off symbol and move the section using the slider bar below it.



Alternatively, you can also move the clipping planes using the mouse cursor (scissors) by clicking on corners of each section or center point.



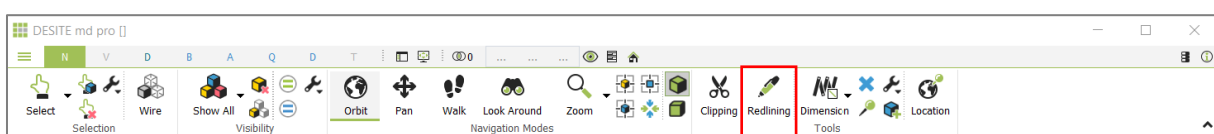
You can adjust display settings for the edges, colors and fills of cutting sections by clicking on the spanner icon.



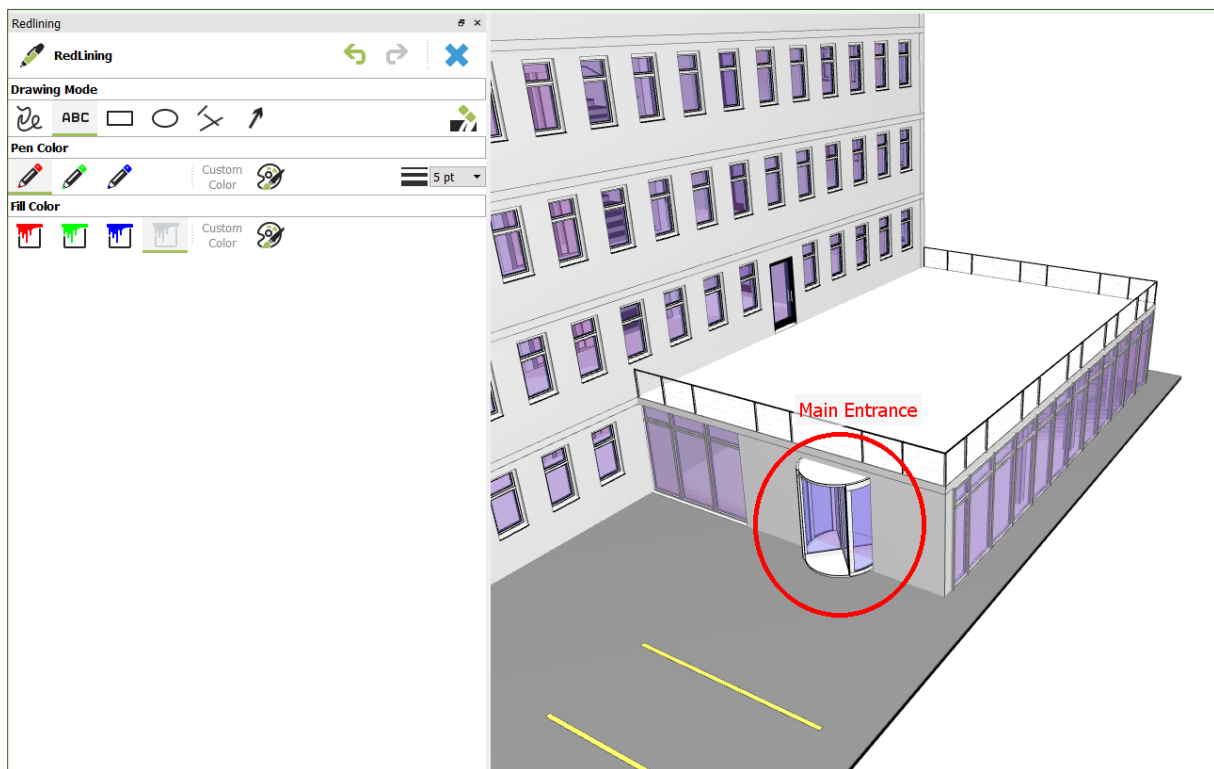
**Tip:** Bounding box of visible or selected objects can also be used to restrict the limits within which a clipping plane can be moved.

### 3.7 Redlining

Redlining tool provides functions for placing comments and annotations within the 3D model.



Click on the Redlining button in the Navigation toolbar in order to open the palette for markup tools. You can adjust settings for Drawing Modes, Pen color and Fill color and choose different colors, shapes or pen thickness values.



You can erase markups by using the eraser while redlining.



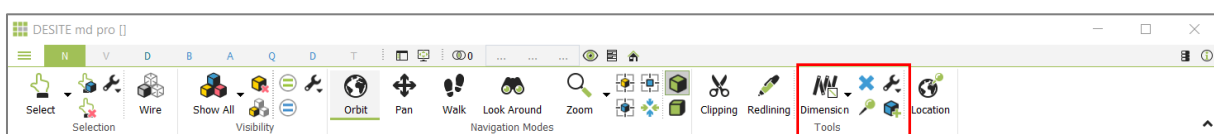
You can remove last markup item or restore the last removed item using arrow buttons.



You can remove all markup items by clicking on X button

**Tip:** Markups will be saved automatically in a 3D Viewpoint. You can also manually save visible and selected objects or clipping planes while creating 3D viewpoints as well.

### 3.8 Measuring



The Dimension tool allows the user to measure points, paths, polylines, 3D polygons, areas and distances.

User can switch between measuring modes using drop down arrow button.



In Points mode, you can display the coordinates of any number of points in the model as required.



In Lines mode, you can measure the distance between two points. Coordinates of selected start and end points will also be displayed.



In Polylines mode, you measure the paths of several consecutive lines. Angles between the lines as well as the total length of all paths are also displayed.



In 3D Polygons mode, you can set as many consecutive points as desired. Dimensions and angles are displayed corresponding to each line or area created.

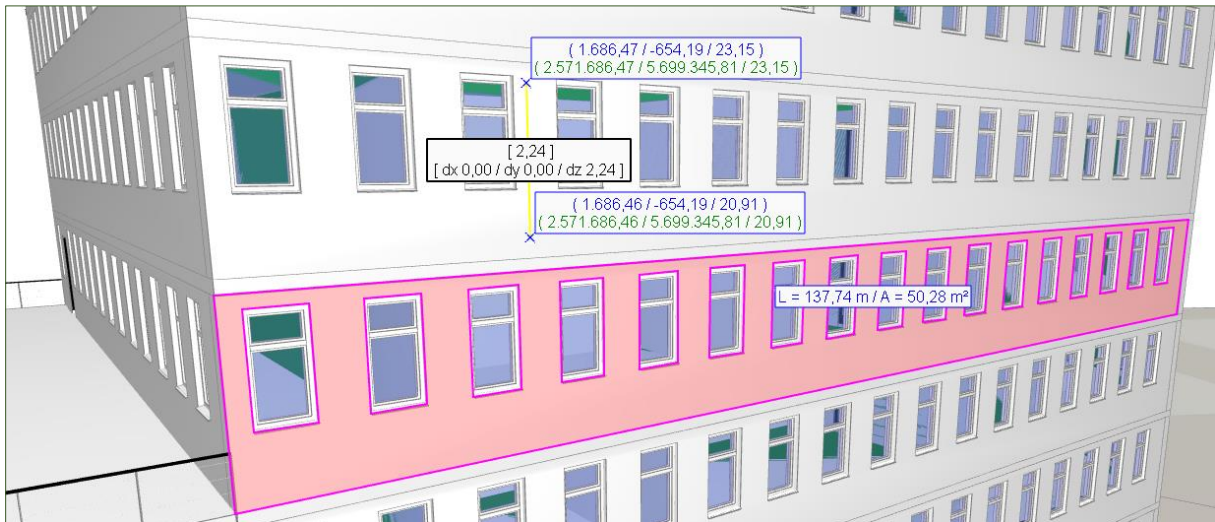


In Faces mode, you measure the areas of objects with just one click. Length of the border lines as well as measured area will be displayed.



In Distance mode, you can measure the perpendicular distance between two parallel paths or parallel surfaces.

Dimensions will be displayed as temporary objects in the 3D view:



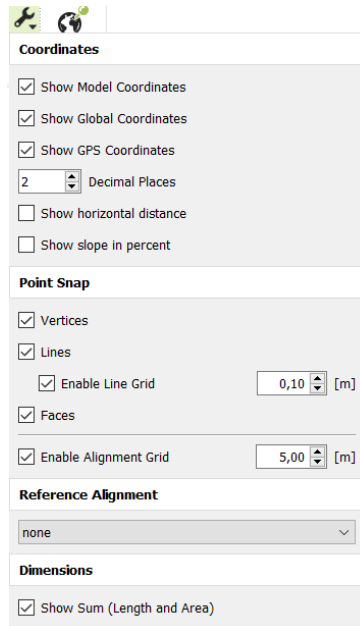
Clears the 3D view from all dimensions



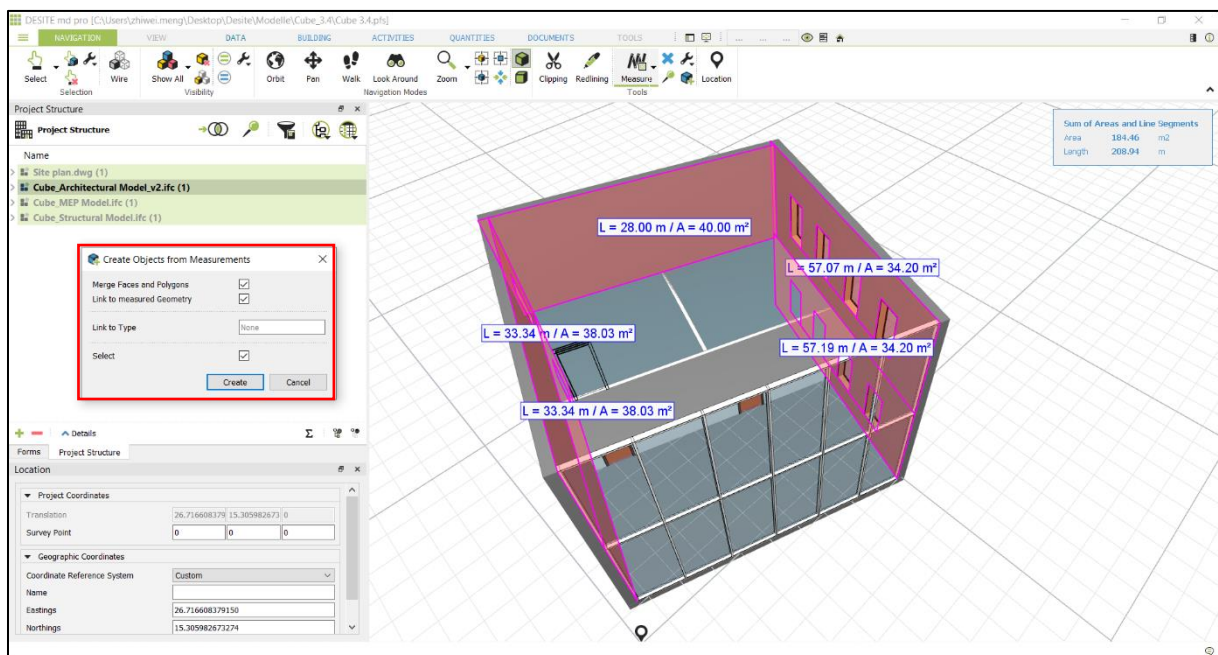
If you select the stick measurement item to 3D view option, you can navigate in DESITE md as usual and the dimension will always remain in the 3D view.



Adjust options for dimension objects (coordinate values, snapping, reference alignment)



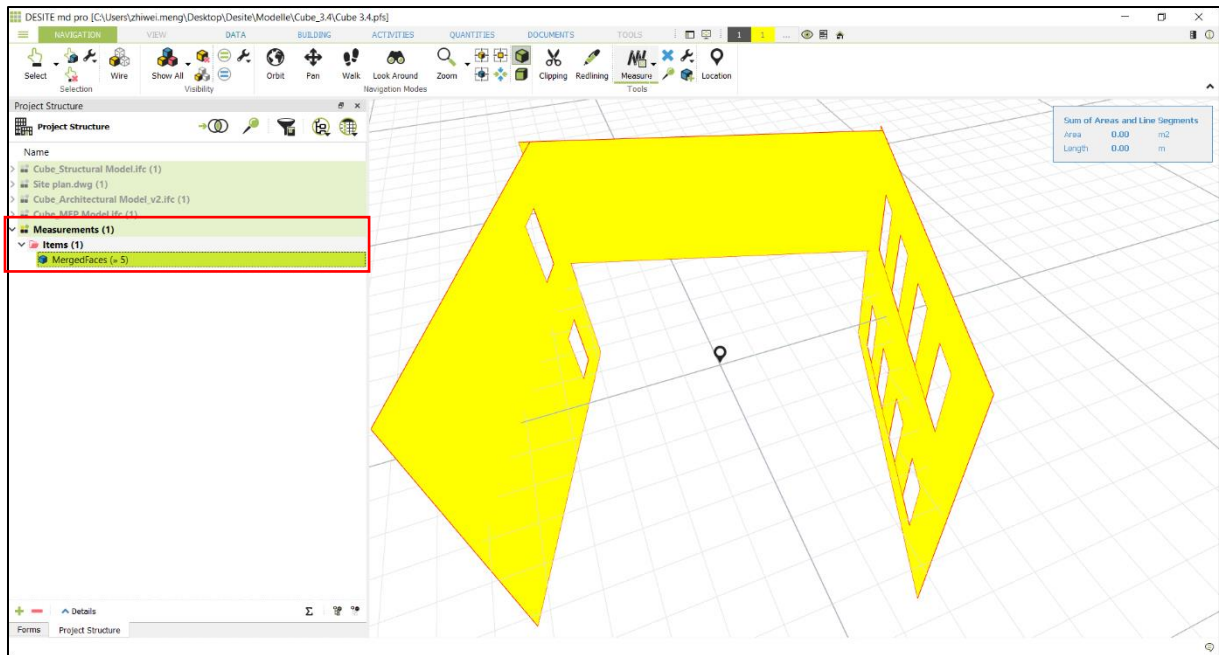
This option creates new geometry objects from all visible dimensions and stores them in the project structure.



Creates geometry objects from all visible dimensions with the selected options. The newly created objects will be stored in the project structure in folder 'DIMENSIONS' and can now be handled as any other 3D geometry object.

- **Merge Faces and Polygons:** merges all face and polygon dimensions into one single object
- **Link to measured Geometry:** measured geometry would be linked to the generated geometry in the project structure (number of linked objects is visible in brackets)
- **Link to Type:** enter a Type name to link the generated geometry to this Type (for more information on Types, see: **Types Domain**)
- **Select:** The newly created objects will be selected



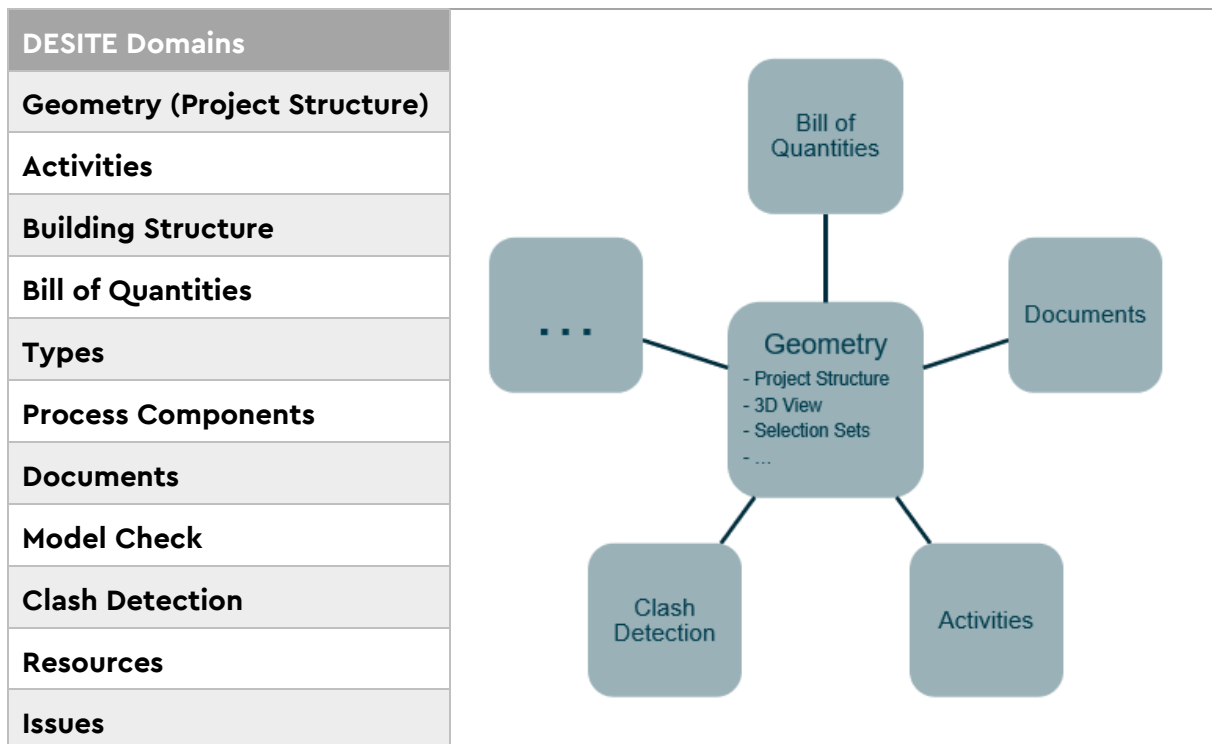


## 4. Information Management

### 4.1 Domains and Models

All information in a project is managed in different domains.

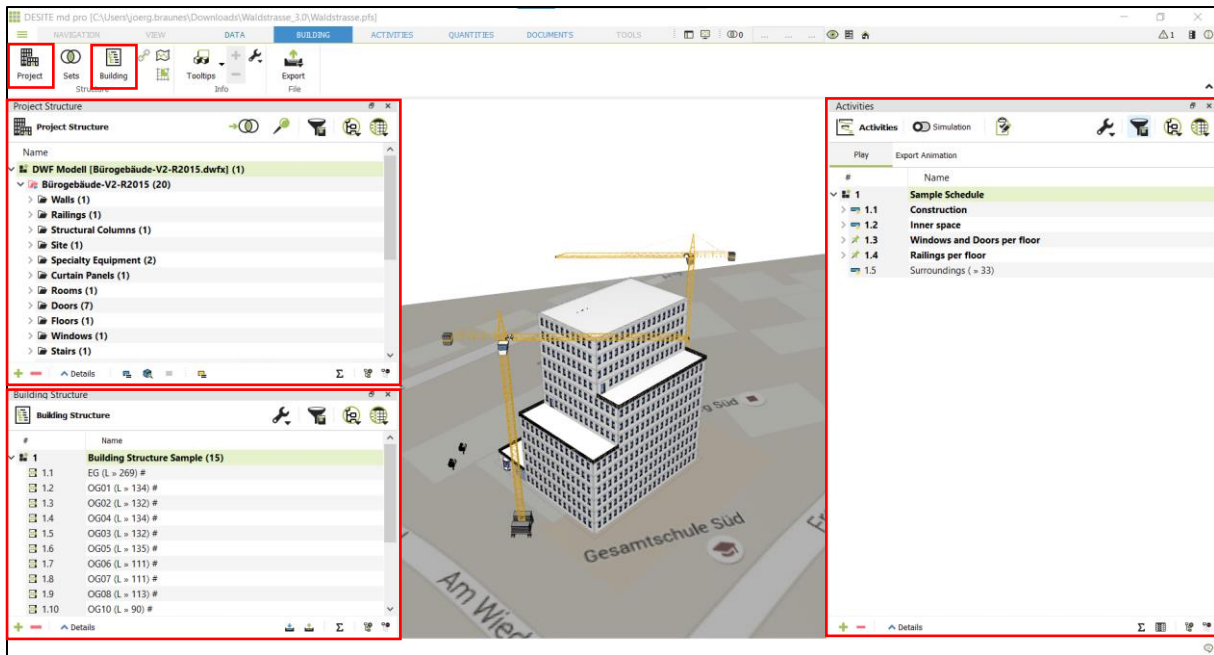
Each domain contains arbitrary number of models and domain-specific objects. The geometry model forms the central reference point. Objects from all domains can be linked to geometric objects, including geometric objects to each other.



Objects of domain models are represented as tree structures in most domains.

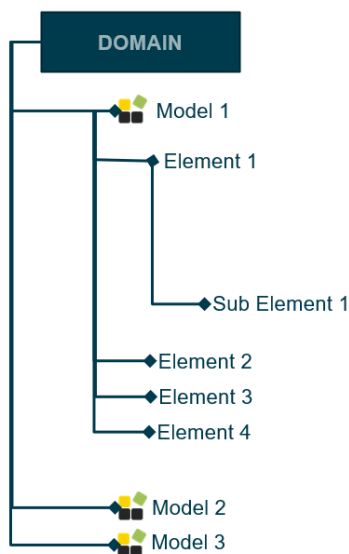
In the following screenshots, the Geometry, Building and Activities Domain tree structures are shown.

**Hint:** The Geometry Domain is also called 'Project Structure', as it represents the main structure of a project with all its geometric objects.



Key functionalities of the tree structure, like the creation of objects, handling of selections and element visibility, are very similar for each domain.

#### 4.1.1 Domain Hierarchy (tree structure)



The first element in any domain hierarchy is always a model, under which other elements are organized. A model can contain multiple instance elements and container elements. Container elements can contain both further container elements and instance elements.



Model



Root Container





Container



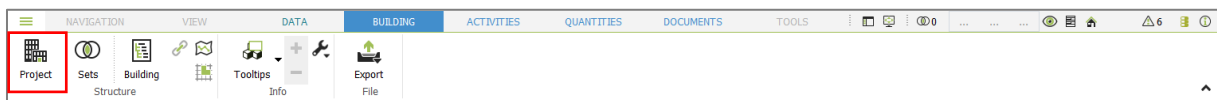
Element (in this case a geometry object). The element icons look different depending on their domain and the type of element.

### Hint:

An instance element is an element unit that does not contain any more child elements (there is no further nesting).

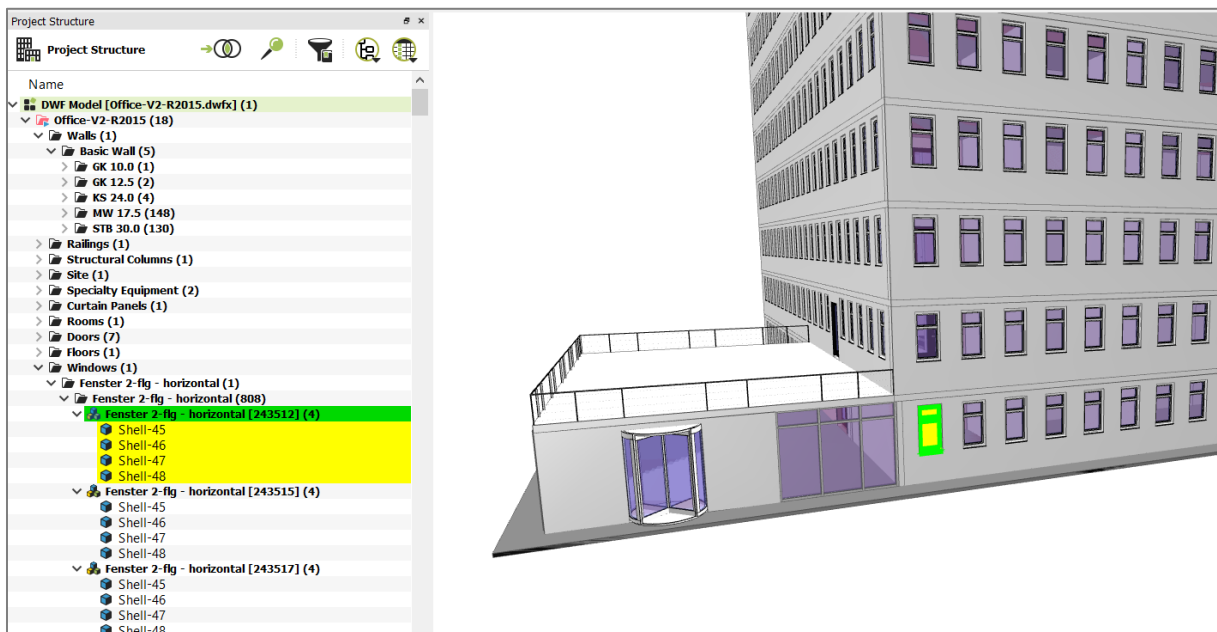
You can create new Models, Container or Elements with the '+' button. The available options differ depending on the domain. Element properties can be displayed in multiple columns next to the element names. For more information, refer to the chapter **Working with Properties**.

## 4.2 Geometry Domain / Project Structure














Contents of imported geometry models are displayed in a tree view under Project Structure. You can open project structure by clicking on the Project button in the Building toolbar.

Each model corresponds to an imported file. The structure of the tree is given by imported native file.



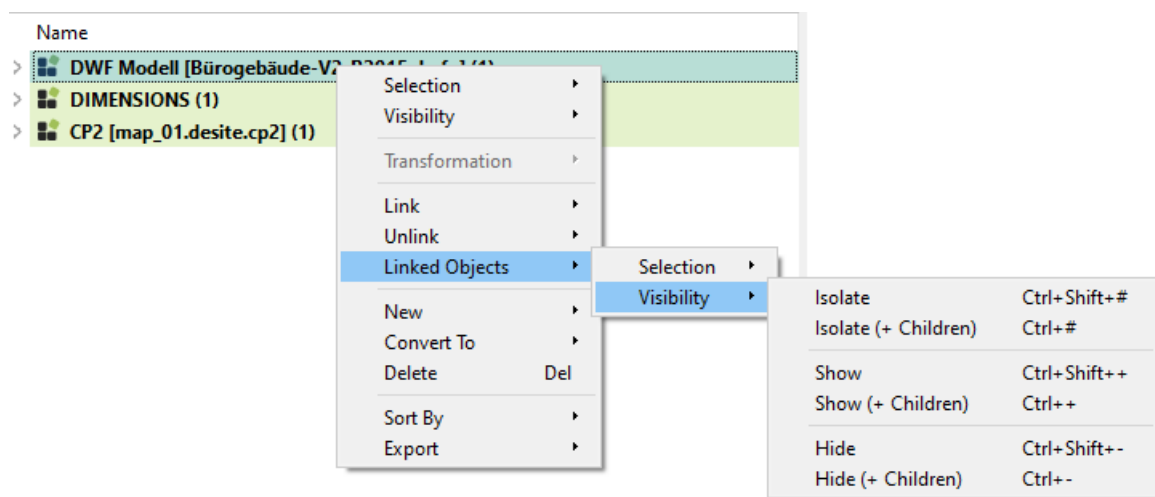
Tree structure is broken down into the following elements:

-  Model. In the Project Structure this represents an imported file. It's also possible to create new models and adding objects to it (e.g. by creating 3d objects from **Measuring**)
-  Root Container
-  Root Container with a translation applied
-  Container
-  Composite object (group of logically related objects, building component group)
-  Object
-  Opening or negative objects
-  Part of an object (this can be created with the cutting tool)
-  Lines
-  Points
-  Generic object

**Note:**

Objects are highlighted in yellow by default when selected, while Composite objects in green.

Visibility and selection of objects can also be handled in the Project Structure window. The context menu (opened by right-clicking on a tree element) offers many options:



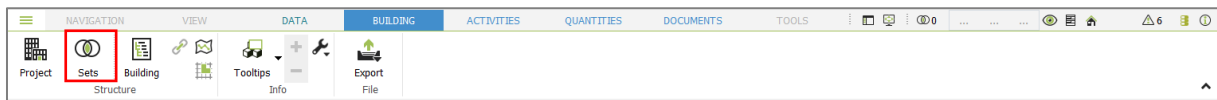
Alternatively, you can also use the following legacy shortcut keys:

- **Esc**: Cancel selection (use in the 3D view)
- **3**: Show selected only
- **-**: Hide selected
- **+**: Show selected
- **T**: Activate/deactivate tooltips

- **I**: Show Quick Info
- **A**: Show all
- **Z**: Zoom to visible

Please refer to the chapter **Configuration** for instruction of customizing shortcuts.

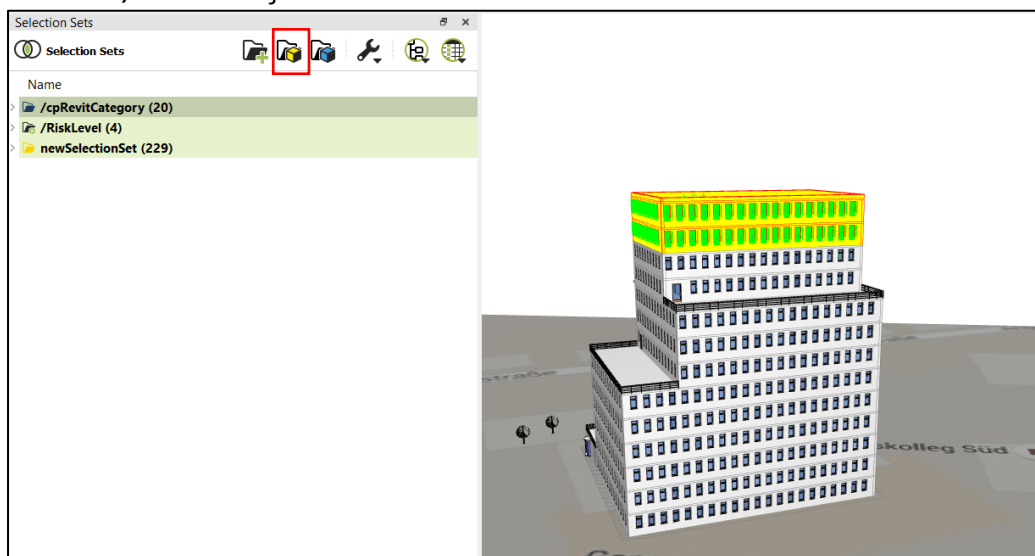
### 4.3 Selection Sets



The selection sets are used to create new object groups, which can later be used for e.g. clash detection, model checking, bill of quantities, etc. In large project, it could also be used to quickly view a part of the models or change the property value of a group of objects rapidly.

There are two approaches to create a selection set:

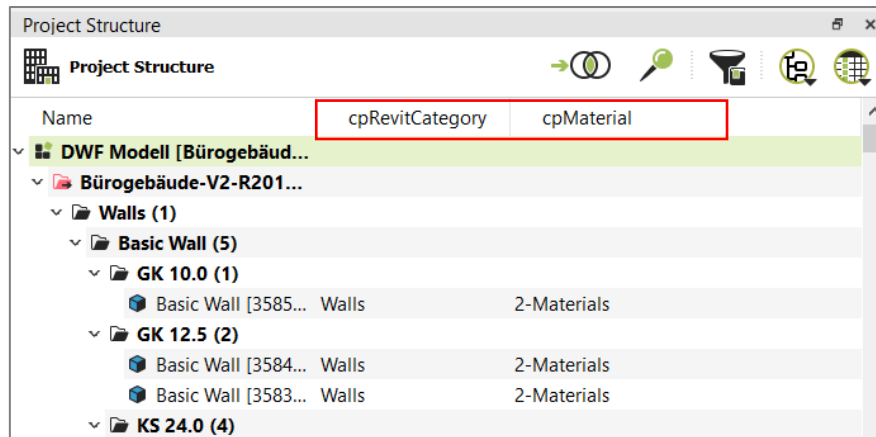
1. Select the objects which are to be grouped first in the 3D view, or set the target objects to visible. Then use the button "Create a new container with selected/visible objects" in selection set:



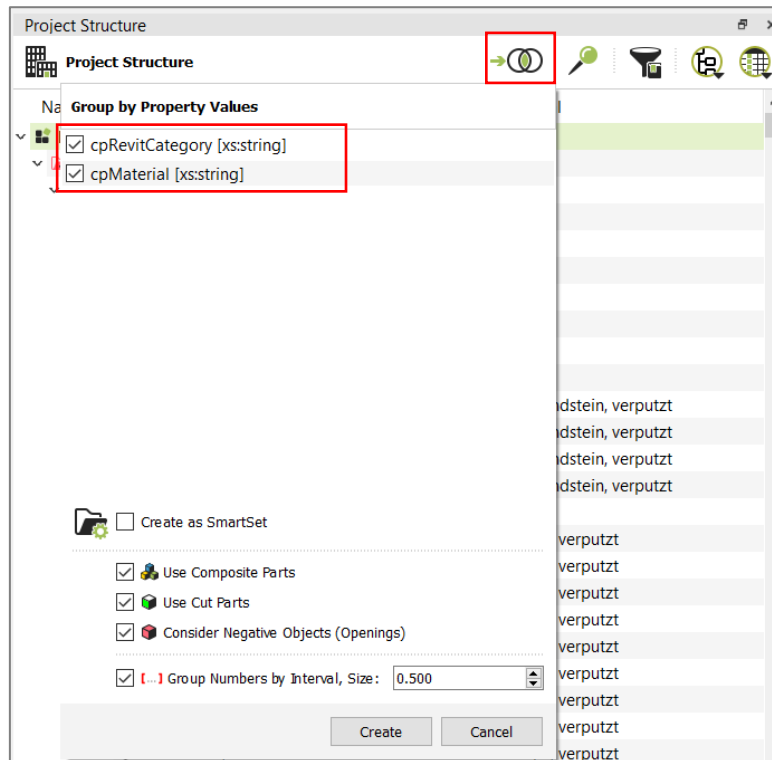
2. Use the option "create smart and selection sets from properties" in project structure. After clicking on the button, all shown properties in project structure are displayed, and you can choose the one/ones to create selection set. The order of properties in the list decides the hierarchy of created folders.

The following example shows how to create selection set from properties.

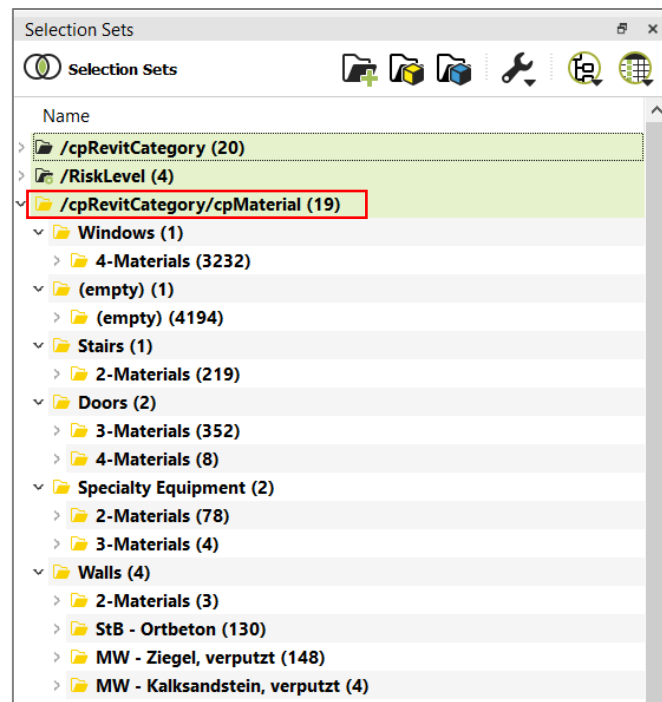
- Display the properties that are needed (in certain order)



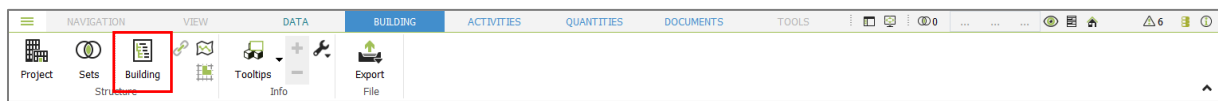
- Click on the button to set selection set, and you can still choose the needed properties. Please remember that the order of properties matters and can only be changed in the project structure under "configure columns".



- As the result, you can see the newly created folder under selection sets, with the name of properties. The first property "cpRevitCategory" is used for the first layer of hierarchy, such as "Windows, Stairs, Doors...". And the second property "cpMaterial" is used for the second layer "4-Materials, 3-Materials, 2-Materials,...".

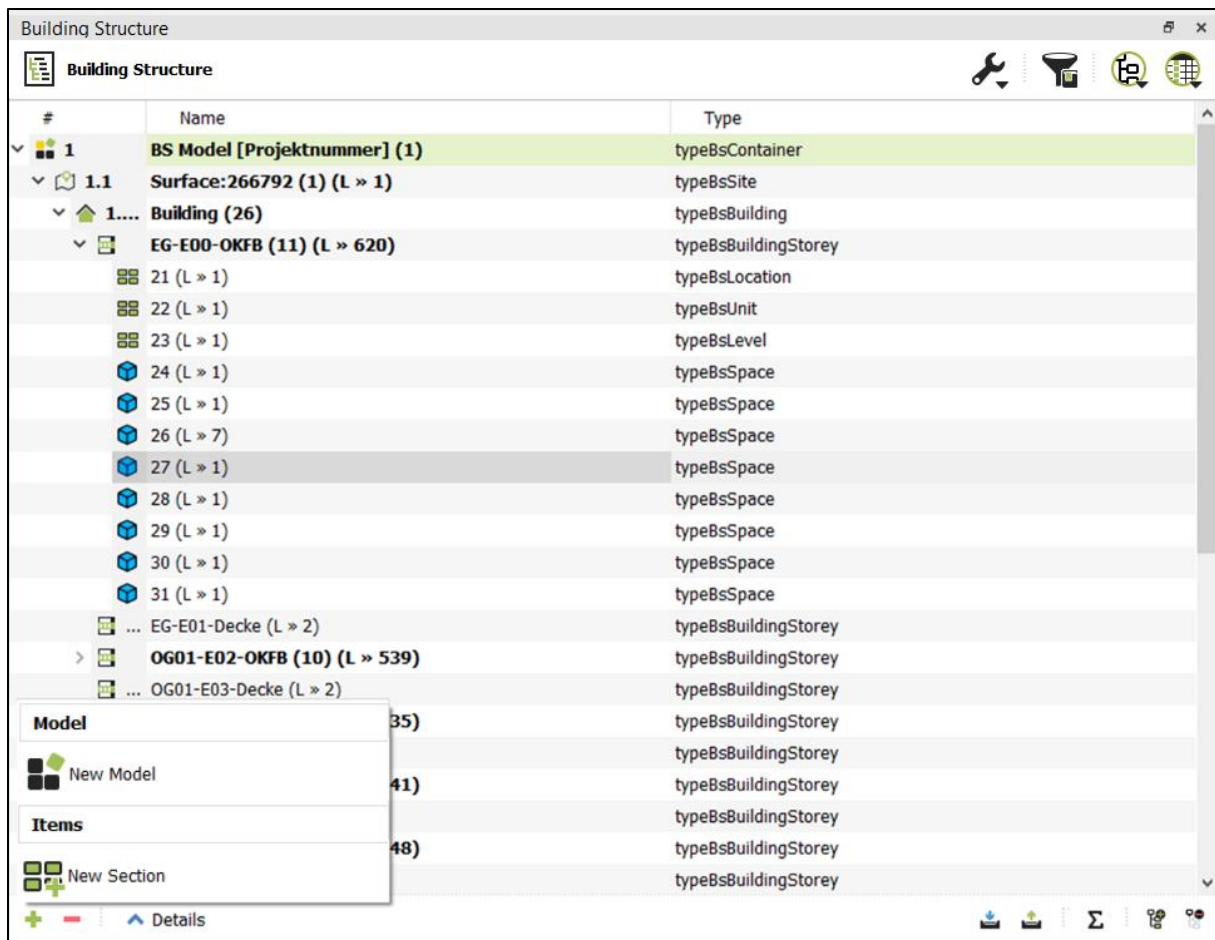


## 4.4 Building Structure



In the Building Domain, you can organize your project in a hierarchy according to the spatial structure (e.g. based on building storeys). For this purpose, the objects of the project are linked to the elements of the building structure.

Compared to other domains, the Building Structure contains no entity other than models and structural elements. The identity of a structural element is defined by its 'Type' property:

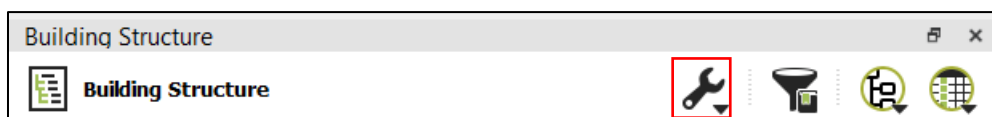


To manually assign geometric objects to the Building Structure, right click on the appropriate structural element and choose one of the options for manual linking.

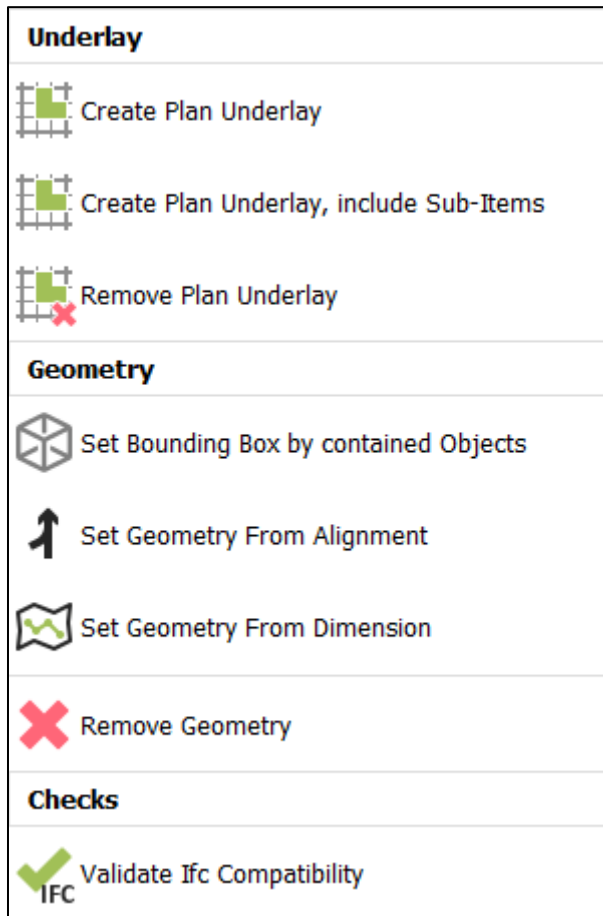


You can also use **Rule based linking** to assign geometry objects to the Building Structure.

#### 4.4.1 Building structure element tools

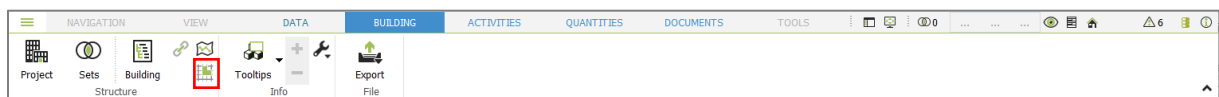


Access additional options and tools for working with the Building Structure:



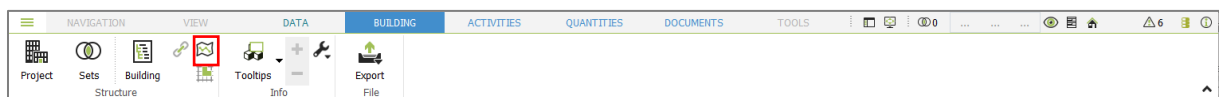
## Underlay

Tools to create 2D plan underlays from the linked geometry object. The underlay itself can be shown/hidden with the appropriate button in the Ribbon bar of the Building domain.



## Geometry

Tools to create a geometric representation that is directly assigned to a structural element. The resulting geometric representation can be shown/hidden with the appropriate button in the Ribbon bar of the Building domain.



## Checks

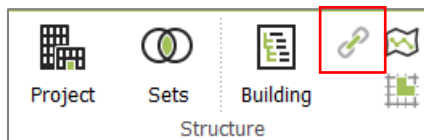
Validate IFC compatibility checks the selected Building Structure model, if it can be used as basis for an IFC export.

## 4.5 Rule based linking

In DESITE, links between 3D model elements and objects from other domains can be created based on user defined rules.

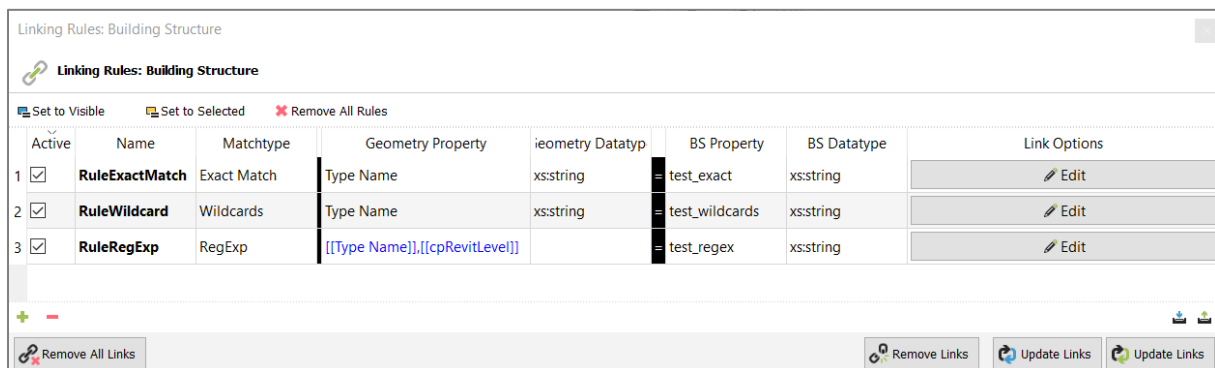
For example, Activities in a time schedule, Bill of Quantity items or object types in domain Types can be linked automatically to geometry objects. Rule-based linking is performed based on attributes of geometry objects and attributes of the objects of the respective domain.

Rule-based linking editor can be accessed via chain icon next to domain widget button in the respective domain i.e. Building Structure:



New rules can be created via '+' button at the bottom left corner in the editor window. Assign an appropriate name to each rule and define which Geometry Properties i.e. which 3D object attributes are to be compared with corresponding Building Structure Properties. If property values of both domains match together, objects are linked to activities automatically.

The following screenshot shows an example of three different rules to automatically link geometry objects to a Building Structure:



(Button text in the lower right is modified in the daily)

Options to create rules and links:

- **'+' Button:** Create a new Linking Rule
- **'-' Button:** Delete the selected Linking Rule
- **Remove All Links:** Remove all links between geometry objects and domain sections. This considers links created via linking rules as well as manually created links.
- **Remove rule-based Links:** Removes only the links which are created via linking rules
- **Update Links (visible Sections):** Creates or updates rule-based links for only the visible sections in domain. This only considers the active rules.

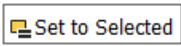
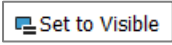


- **Update rule-based Links:** Creates or updates all rule-based links in the domain. This only considers the active rules.

Defining rules:

- **Active:** activate or de-activate this rule
- **Name:** name of the rule
- **Matchtype:** select between 'Exact Match', 'Wildcard' or 'RegExp' ( Regular Expression)
- **Geometry Property:** The name or names of the Geometry properties that are used for the linking. You can specify one property or multiple properties by placing multiple property names between double square brackets '[[...]]'. You can place a separator (e.g. ',') between the property names to make the concatenated properties more readable. Any other separator or no separator can also be used as desired.
- **Geometry Datatype:** Data type of the geometry property (if only one is used). If more properties are defined, this field is empty.
- **Equal sign '=':** visual separation between the definition of the properties in the geometry domain (on the left site) and the definition of the properties in the domain where the objects should be linked to (on the right site; in this example the Building Structure domain).
- **[domain] Property:** The name or names of the domain properties that should be linked to.
- **[domain] Datatype:** Data type of the domain property (if only one is used) Data type must match on left and right side.

To assign a linking rule to objects in the domain, you can either:

1. Select the related objects (e.g the Section in the Building Structure) in the domain and click 
2. Isolate the related objects (e.g the Section in the Building Structure) in the domain and click 
3. Drag and drop the rule to the name of the object in the domain (preferred option)

To see which Linking Rules are applied to a domain element, set the standard DESITE attributes 'cpLinkRule' and 'cpLinkRuleValue' as visible in the columns. These parameters contain the name of the assigned rule as well as the value of the parameter defined in that linking rule.

Building Structure						
#	Name	cpLinkRule	cpLinkRuleValue	test_exact	test_wildcards	test_regex
1	Building Structure Sample (15)					
2	Example Structure (3)					
2.1	newSection - exact match	RuleExactMatch	ks24.0	KS 24.0		
2.2	newSection - wildcard	RuleWildcard	gk*		GK*	
2.3	newSection - RegExp	RuleRegExp	^(stb.+og02.+)			^(STB.+OG02.+)

### 4.5.1 Match types

The Match type defines how the property values are evaluated in a Linking Rule.

#### Exact Match

Geometry objects and domain objects will be linked together, if the defined property values in the Linking Rules are identical.

#### Wildcards

Wildcards are placeholders represented by a single character, that can be used in filters and Linking Rules.

Example:

Building Structure						
#	Name	cpLinkRule	cpLinkRuleValue	test_exact	test_wildcards	test_regex
1	Building Structure Sample (15)					
2	Example Structure (3)					
2.1	newSection - exact match	RuleExactMatch	ks24.0	KS 24.0		
2.2	newSection - wildcard	RuleWildcard	gk*		GK*	
2.3	newSection - RegExp	RuleRegExp	^(stb.+og02.+)			^(STB.+OG02.+)

The property 'test\_wildcards' has the value 'GK\*'. The related Linking Rule defined this value as a 'Wildcard', so all geometry objects with a property value that begins with 'GK' will be linked to the related domain object.

Linking Rules: Building Structure							
Linking Rules: Building Structure							
Set to Visible Set to Selected Remove All Rules							
Active	Name	Matchtype	Geometry Property	Geometry Datatype	BS Property	BS Datatype	Link Options
1 <input checked="" type="checkbox"/>	RuleExactMatch	Exact Match	Type Name	xs:string	= test_exact	xs:string	Edit
2 <input checked="" type="checkbox"/>	RuleWildcard	Wildcards	Type Name	xs:string	= test_wildcards	xs:string	Edit
3 <input checked="" type="checkbox"/>	RuleRegExp	RegExp	[[Type Name]], [[cpRevitLevel]]		= test_regex	xs:string	Edit

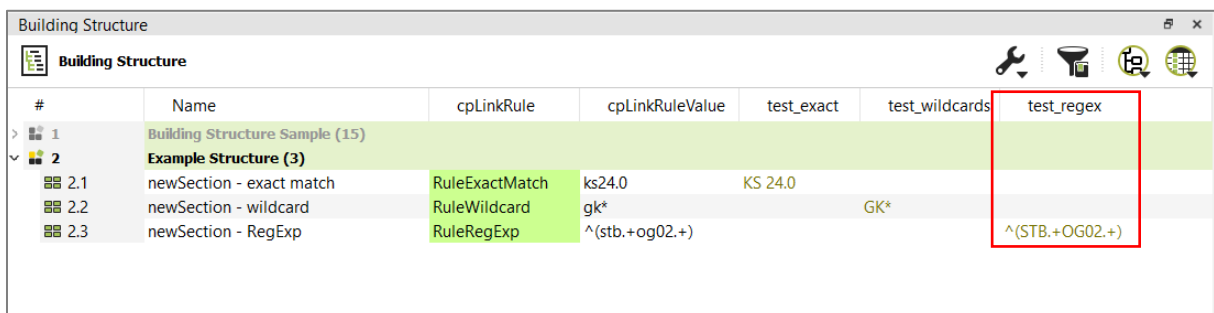
In DESITE the following Wildcards can be used for Linking Rules and also for Filters:

Search term	Description
Wa*	Text that begins with 'Wa', such as 'Wall'
*wall*12.*	Text that contains the elements 'Wall' <i>and</i> '12.', such as 'BasicWall_12.5'
Concrete Steel Masonry	Text is either 'Concrete' <i>or</i> 'Steel' <i>or</i> 'Masonry'
'Basic Wall XY'	Text equals 'Basic Wall XY' (without quotation marks, this would mean 'Basic Wall' <i>or</i> 'XY')
!Wa*	Text that does <i>not</i> begin with 'Wa'
Wa* Wind* (space in between)	Text that begins with 'Wa' or 'Wind'
> C	All text that begins with D, E, F etc.
< 0	All negative numbers
!*	Property is not present/defined
# Wa*	Remove all applied filters (i.e. no AND relation) and then show all objects which the property value begins with 'Wa'.
H[ua]nd	Finds Hund and Hand

## Regular expressions

A regular expression is a pattern of characters that can be used in Linking Rules to define a matching patter.

Example:



#	Name	cpLinkRule	cpLinkRuleValue	test_exact	test_wildcards	test_regex
> 1	Building Structure Sample (15)					
2	Example Structure (3)					
2.1	newSection - exact match	RuleExactMatch	ks24.0	KS 24.0		
2.2	newSection - wildcard	RuleWildcard	gk*		GK*	
2.3	newSection - RegExp	RuleRegExp	^(stb.+og02.+)			^(STB.+OG02.+)

The property 'test\_regex' has the value '^ (STB.+OG02.+)'.

The related Linking Rule defined this value as a 'Regular Expression', so all geometry objects with a property value that matches this RegEx will be linked to the related domain object.

Linking Rules: Building Structure

Linking Rules: Building Structure

Set to Visible Set to Selected Remove All Rules

Active	Name	Matchtype	Geometry Property	Geometry Datatype	BS Property	BS Datatype	Link Options
<input checked="" type="checkbox"/>	RuleExactMatch	Exact Match	Type Name	xs:string	test_exact	xs:string	Edit
<input checked="" type="checkbox"/>	RuleWildcard	Wildcards	Type Name	xs:string	test_wildcards	xs:string	Edit
<input checked="" type="checkbox"/>	RuleRegExp	RegExp	[[Type Name]],[[cpRevitLevel]]		test_regex	xs:string	Edit

Remove All Links Remove Links Update Links Update Links

In this example two Geometry Properties are used, 'Type Name' and 'cpRevitLevel'. The RegEx matches with all values (as a combination of 'Type Name' and 'cpRevitLevel') that:

- starts with 'STB' (Quantifier '^')
- followed by any character with one or more repetitions (Metacharacter '.' and Quantifier '+')
- followed by 'OGO2' (Letters and Digits)
- followed by any character with one or more repetitions (Metacharacter '.' and Quantifier '+')

### Quick reference

The following summary of Regular Expressions is just an excerpt and not a complete reference. To learn more about Regular Expressions (RegEx), countless web resources and tutorials are available. Some examples<sup>1</sup>:

- <https://www.regular-expressions.info/tutorial.html>
- <https://regexone.com/>
- [https://www.w3schools.com/jsref/jsref\\_obj\\_regexp.asp](https://www.w3schools.com/jsref/jsref_obj_regexp.asp)

**Brackets:** Brackets are used to find a range of characters

Expression	Description
[abc]	Find any character between the brackets
[^abc]	Find any character NOT between the brackets
[0-9]	Find any character between the brackets (any digit)
[^0-9]	Find any character NOT between the brackets (any non-digit)

<sup>1</sup> Disclaimer:

The mentioned links partly lead to external websites of third parties, on whose contents we have no influence. Therefore, we cannot assume any liability for these external contents. For the contents of the respective provider or operator of the pages is always responsible for the content of the linked pages. The linked pages were checked for possible legal violations at the time of linking. Illegal contents were not recognizable at the time of linking. However, a permanent control of the contents of the linked pages is not reasonable without concrete evidence of a violation of the law. In the case of infringements become known, we will remove such links immediately.

(x y)	Find any of the alternatives specified
-------	--

### Quantifiers

Quantifier	Description
n+	Matches any string that contains at least one n
n*	Matches any string that contains zero or more occurrences of n
n?	Matches any string that contains zero or one occurrences of n
n{X}	Matches any string that contains a sequence of X n's
n{X,Y}	Matches any string that contains a sequence of X to Y n's
n{X,}	Matches any string that contains a sequence of at least X n's
n\$	Matches any string with n at the end of it
^n	Matches any string with n at the beginning of it
?=n	Matches any string that is followed by a specific string n
?!n	Matches any string that is not followed by a specific string n

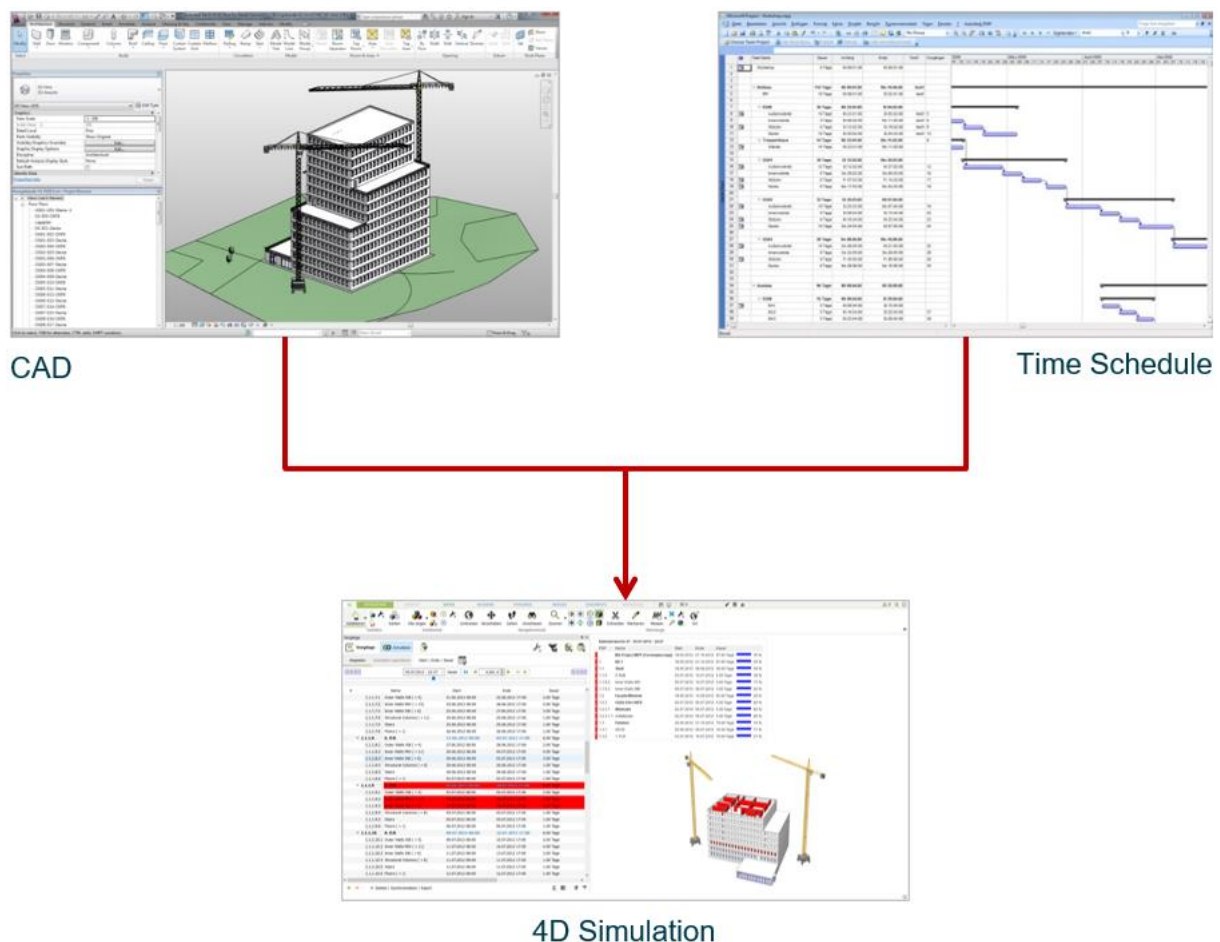
**Metacharacters:** Metacharacters are characters with a special meaning:

Metacharacter	Description
.	Find a single character, except newline or line terminator
\w	Find a word character
\W	Find a non-word character
\d	Find a digit
\D	Find a non-digit character
\s	Find a whitespace character
\S	Find a non-whitespace character
\b	Find a match at the beginning/end of a word, beginning like this: \bHI, end like this: HI\b
\B	Find a match, but not at the beginning/end of a word
\0	Find a NULL character
\n	Find a new line character

\f	Find a form feed character
\r	Find a carriage return character
\t	Find a tab character
\v	Find a vertical tab character
\xxx	Find the character specified by an octal number xxx
\xdd	Find the character specified by a hexadecimal number dd
\udddd	Find the Unicode character specified by a hexadecimal number dddd

## 4.6 Activities domain

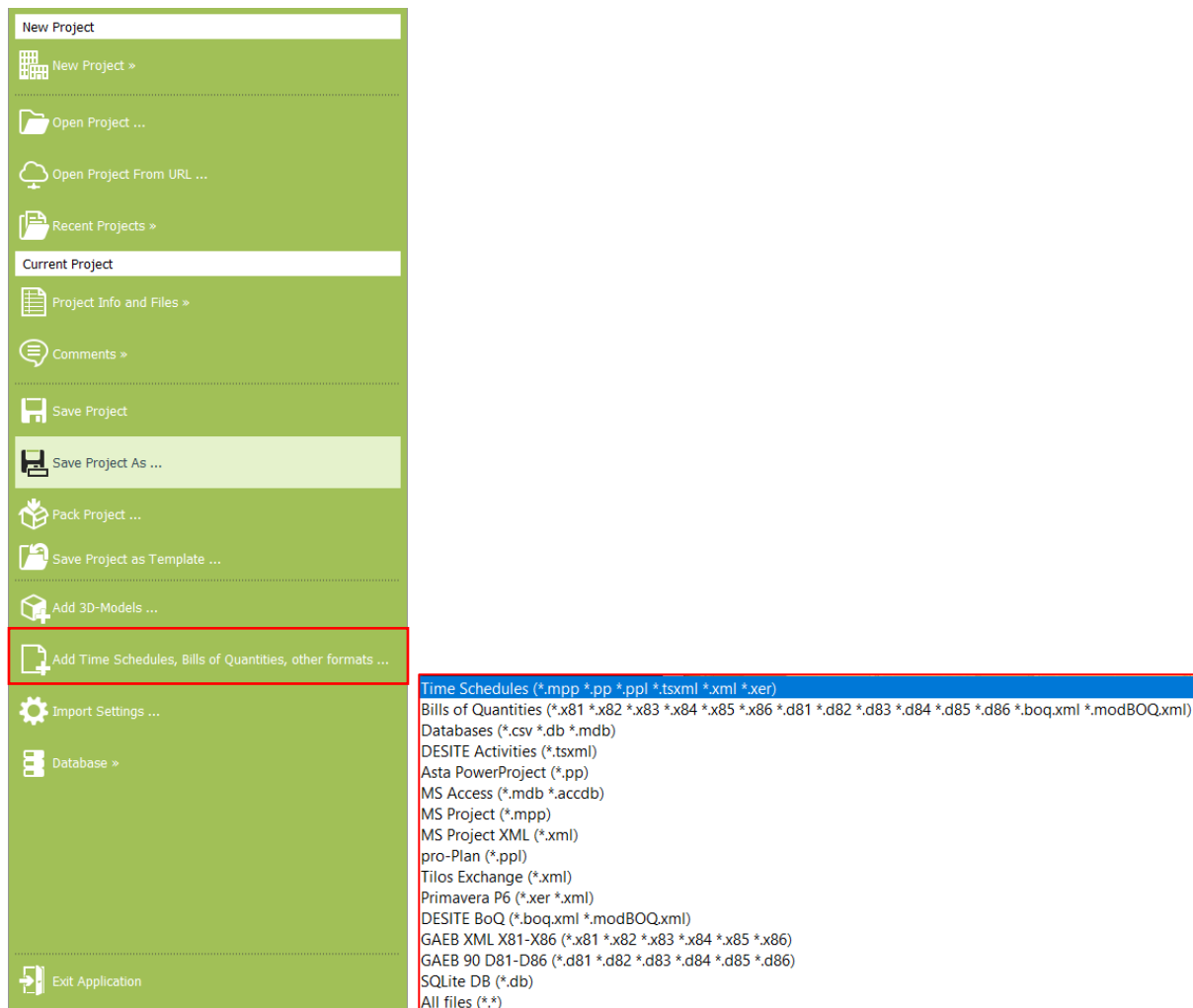
You can also import time schedules into DESITE md/md pro or create them manually. If you link the activities of the time schedule to the elements in the geometry, you create a 4D BIM coordination model.



This 4D BIM model can later be used for various use cases i.e. for visual time schedule controlling, for feasibility analysis as well as time-related quantity analysis.

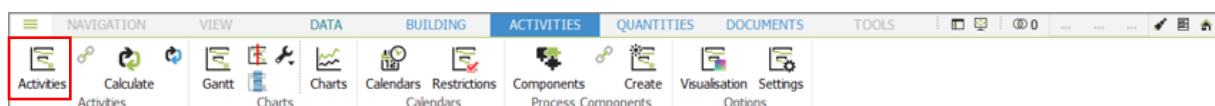
If you would like to import an existing time schedule into your DESITE project, click on Add Time Schedule / ... under Application Menu and select time schedule format to be

imported. Microsoft Project (\*.mpp), Asta Powerproject (\*.pp), Tilos (\*.xml), pro-Plan (\*.ppl) and native DESITE time schedule format (\*.tsxml) are currently supported.

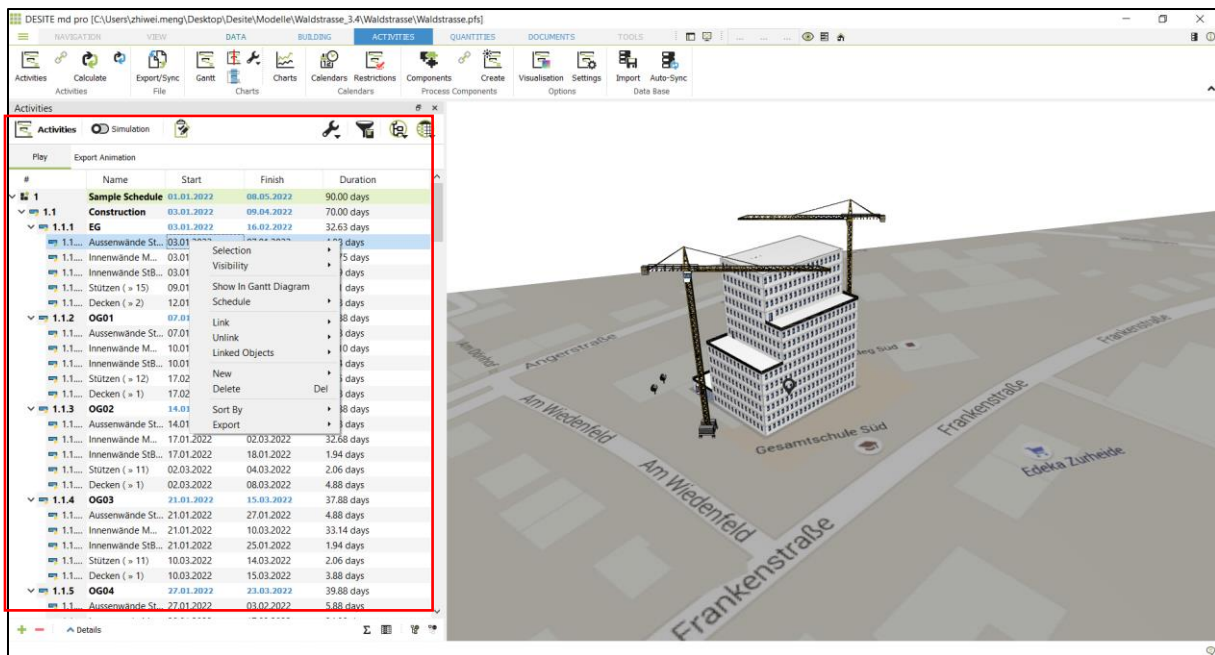


#### 4.6.1 Linking Activities to 3d Objects

You can create and run 4D simulations under Activities menu tab. Click on Activities button in the toolbar to open the widget and access all available time schedules that have been imported or manually generated in the project.



In order to link your time schedule and 3D model together, most simply you can manually select model objects in 3D view and then assign those to related activities in your time schedule. It is suggested to use selection sets in order to filter and select model objects in an easier way.



It is generally more effective to link 3D objects to activities automatically by using **Rule based linking** functionality.

#### Advantages:

- Rules can be saved and reused in other projects.
- Linking rules can be reapplied if a geometry model version or time schedule has been imported and thus links between model objects and activities can easily be updated.

#### Disadvantage:

- Activities and 3D (geometry) objects must define the same property types as in the DESITE projects, these rules have been saved from.
- Also property values matching the rules must have been set for the rules to actually function.

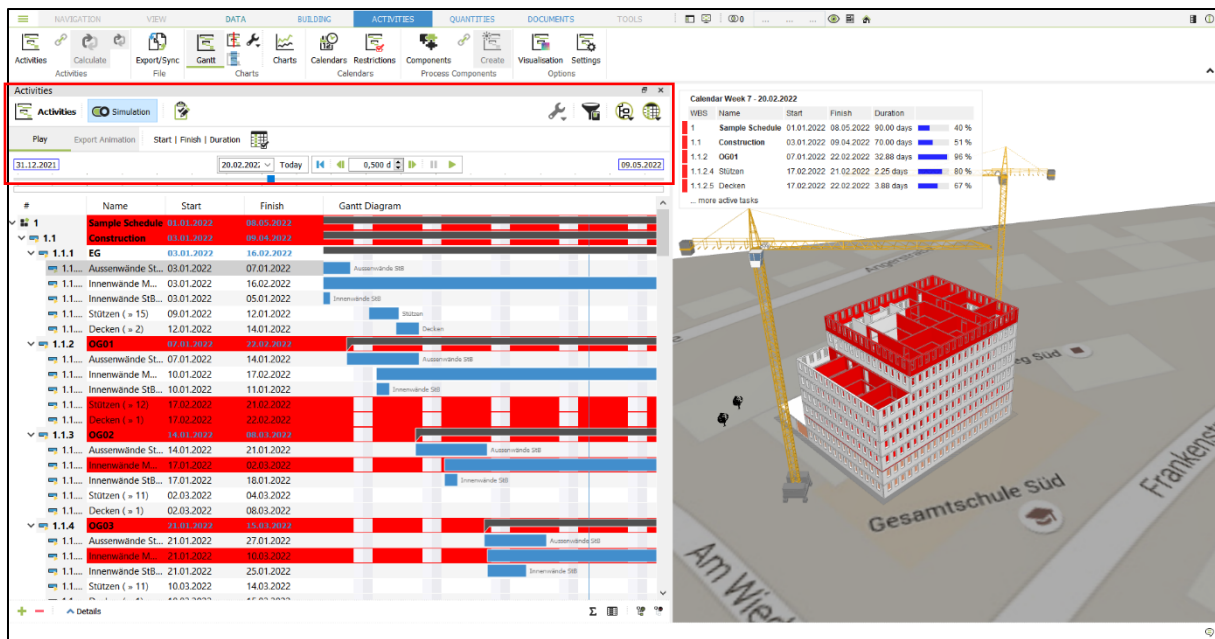
It is also possible to create time schedule automatically based on the **Building Structure** together with **Process Components**.

### 4.6.2 4D Simulation

Click on Simulation slider button in order to switch on 4D simulation mode. A new section with a timeline will be displayed.

You can specify a Start Date and Step Size for simulation in the toolbar. Use the corresponding buttons to play or pause the simulation. Single steps are also possible.





**Tip:** Activate the domains properties Start, Finish and Gantt Diagram in the Activities domain table (see also Properties in the domain tree structure).



Simulation Settings: Define options for the 4D Simulation and the legend that is displayed in the 3D view during the simulation.

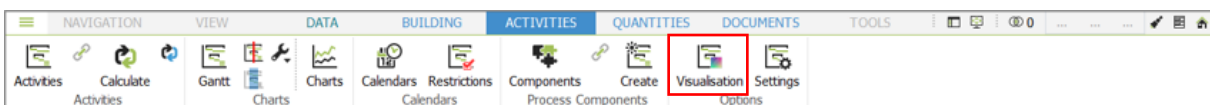


Activity tools: Multiple operations to work with tasks and corresponding Process Components.

**Note:** Simulation options and settings can only be accessed and adjusted when 4D simulation mode is switched off.

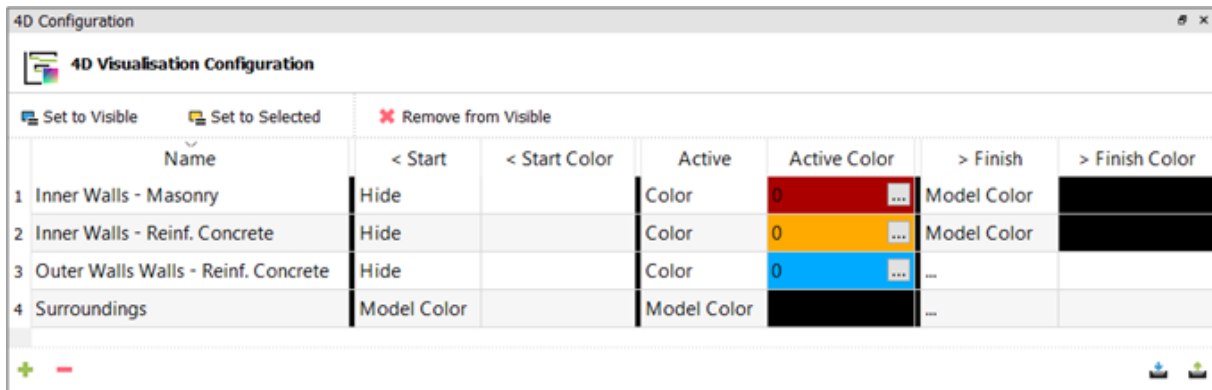
### 4.6.3 Customize 4D Visualisation

You can customize how model objects linked to corresponding activities are displayed before, during and after their execution while running a 4D simulation. For that, click on Visualisation tool under the ACTIVITIES menu tab.



Click on '+' button in order to create a new visualization rule, and assign a name to it (Simulation mode needs to be switched off). Start and Start color define corresponding activities and how model objects linked to them are displayed before they are executed.

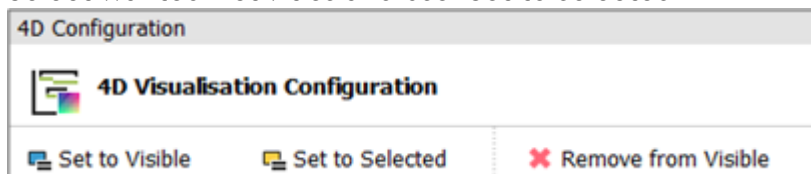
Active and Active color define corresponding activities and how model objects linked to them are displayed during their execution. Finish and Finish color define corresponding activities and how model objects linked to them are displayed after they have been completed.



**Tip:** As a rule of thumb, usually Start is set as hidden so that building components do not appear in 3D view before they are created. Building components are frequently displayed in their model colors after Finish.

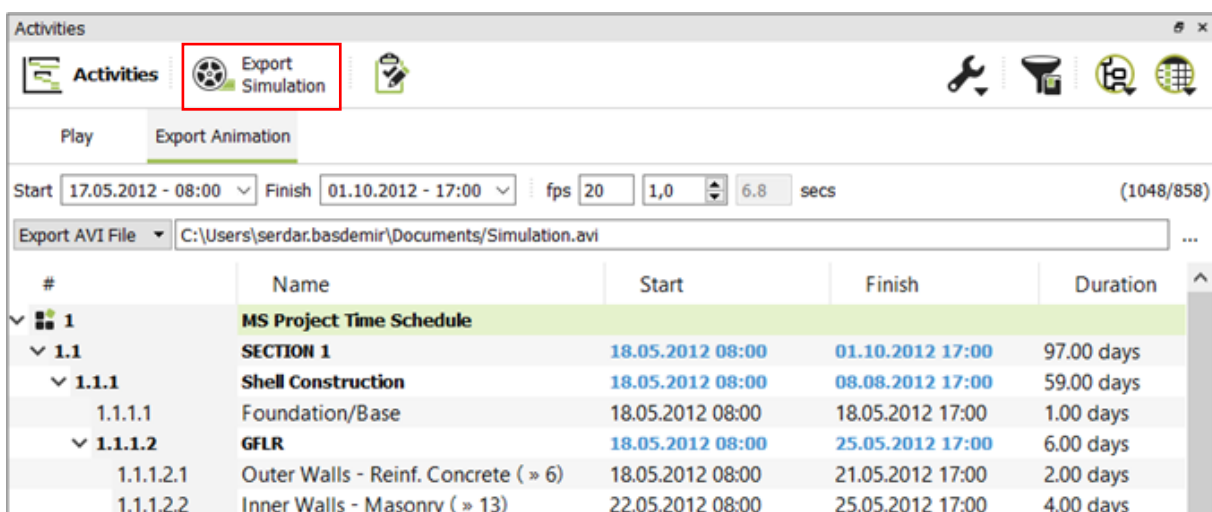
In the next step, created visualization rules must be assigned to the activities or containers that include activities. Assignment is carried out in following ways:

- Drag the rule and drop it on the Activity or a Container that contains Activities
- Make wanted Activities visible and use 'Set to Visible'
- Select wanted Activities and use 'Set to selected'



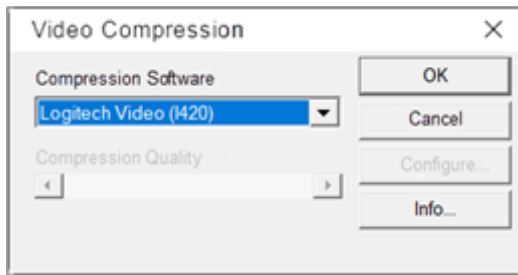
#### 4.6.4 Export 4D Simulation

When 4D simulation mode is switched off, click on Export Animation button first in simulation toolbar in order to export a simulation. A new section with export options will be displayed and Export Simulation option will appear in the Simulation toolbar.



Once you have configured the necessary settings for start, end, step size etc. as desired, choose a directory to save the simulation and then click on Export Simulation button.

Then you will be asked which compression software to be used and export the simulation into the specified folder.



#### 4.6.5 Creating Charts from Activities

Click on Charts button in Activities toolbar in order to create property and time-related 4D charts. This allows you to do different time-based analyzes, e.g. a quantity analysis with a time-variation curve.



Show list of properties: Set only the properties whose data is to be analyzed visible in the Property List, i.e. 'cpVolume'

Charts							
Filter Properties							
	W	AW	BW	Name	Data Type	Domain	
31	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	cpSurfaceContactAreaOOBB_LW_Max	xs:double	Objects	
32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	types:Tapezierung Kosten	xs:double	Objects	
33	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	cpSurfaceArea	xs:double	Objects	
34	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cpVolume	xs:double	Objects	
35	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Roughness	xs:int	Objects	
36	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Stringer Thickness	xs:double	Objects	
37	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Länge Waschbecken	xs:double	Objects	
38	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	cpTopAreaPartMin	xs:double	Objects	
39	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Landing Carriage Height	xs:double	Objects	
40	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Einbautiefe (von außen)	xs:double	Objects	
41	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Use Landing Height Adjustment	xs:int	Objects	
42	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	cpLinesLength	xs:double	Objects	
43	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	cpLateralArea	xs:double	Objects	
44	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	cpSurfacePartAreaMin	xs:double	Objects	
45	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Höhe unterer Flügel	xs:double	Objects	
46	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Roughness	xs:long	Objects	
47	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Temperature Loss	xs:double	Objects	
48	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	RB:Wandfläche	xs:double	Objects	
49	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Function	xs:int	Objects	

There are four options to select for each property:

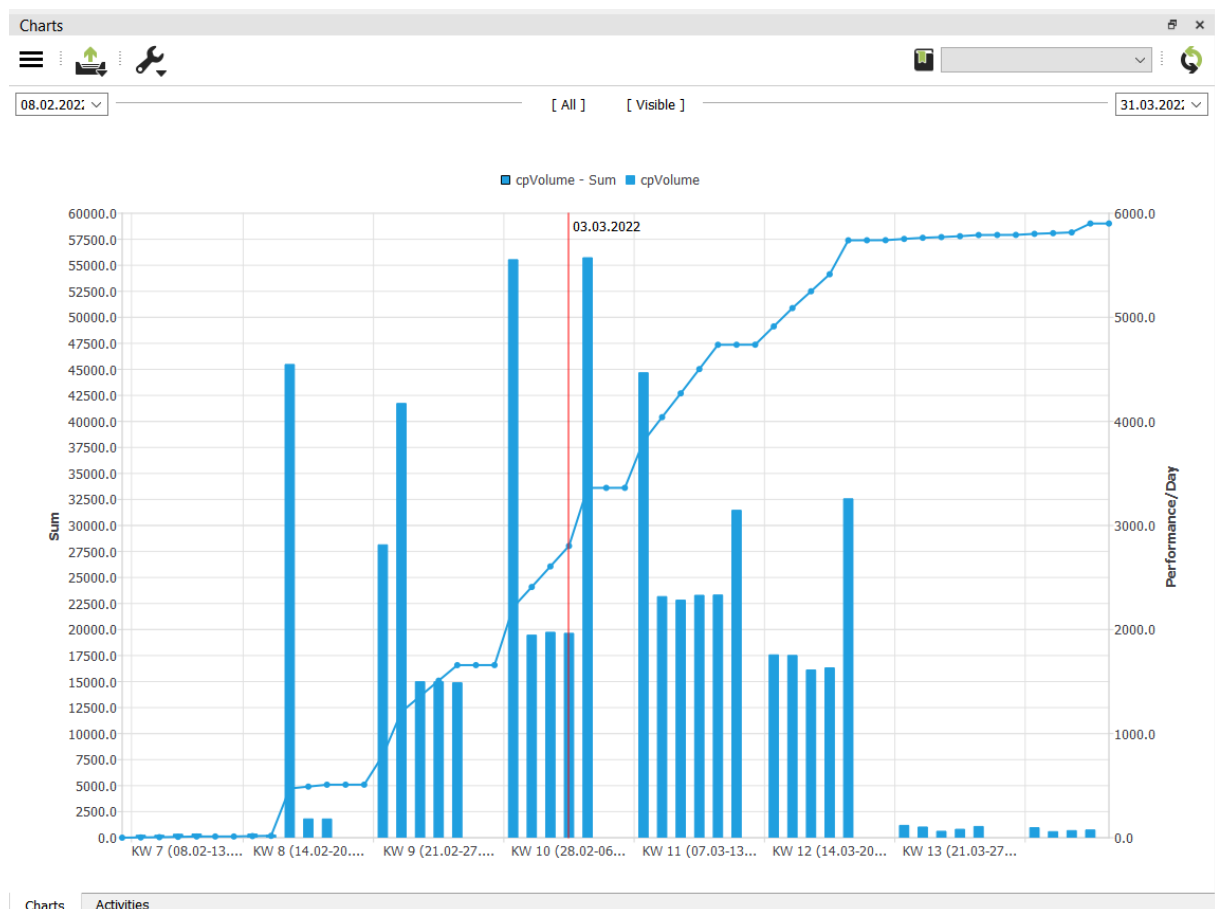
- Visible
- W (Work)
- AW (Actual Work)
- BW (Baseline Work)



In the configuration menu (spanner icon), set the chart display to Visible activities.



Update the chart by clicking on Refresh button at the upper right corner.



Copy the current Chart to clipboard or export a list of values

## 4.7 Process Components

Time schedules can be automatically created on the basis of a building model. This requires Process Components and the **Building Structure**.

The Process Components specify which activities are to be executed to create a building component, thus they define the work content, sequence of the work, and the required duration based on the production rate and resource of linked geometry.

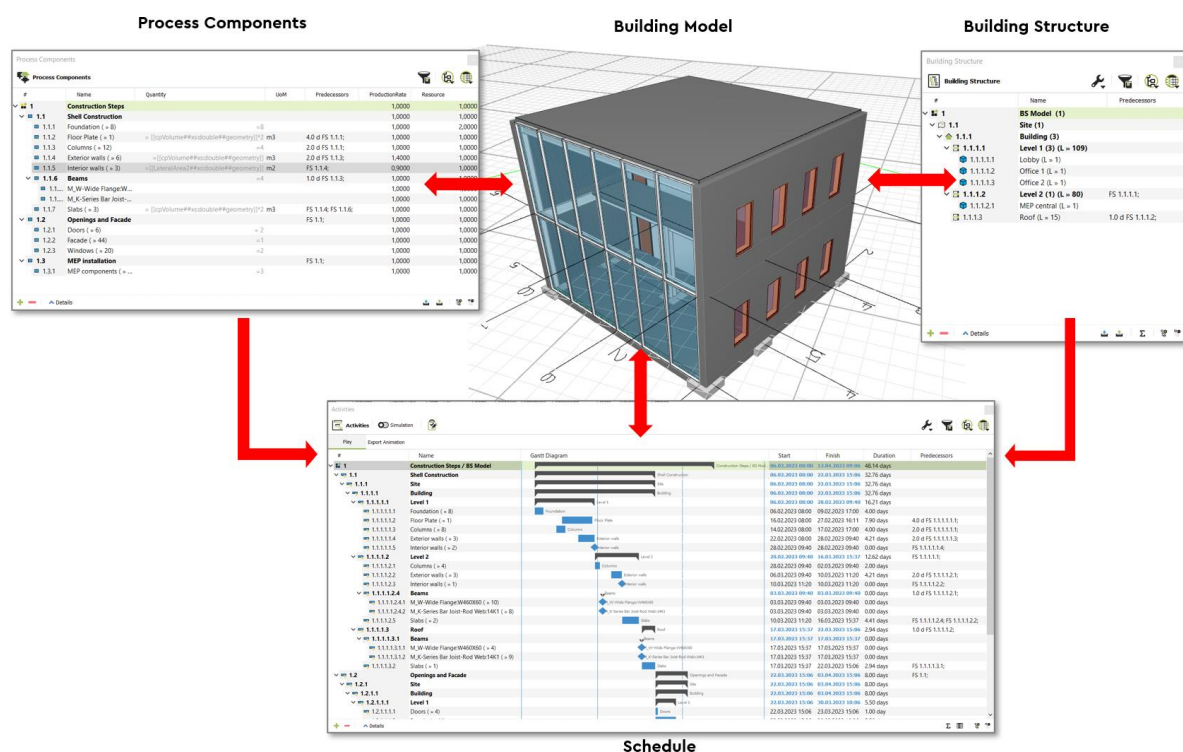
A Process Component contains:

- Name
- An effort value in hours per unit of measure and resource
- A formula for quantity calculation and a unit of measure
- Relationships to other Process Components
- Settings for the visualization in the 4D simulation
- If necessary, a Process Component can contains other Process Components

The Building Structure defines the location where an activity is to be carried out. The elements of the building structure are arranged hierarchically, they define the topological structure of the building.

The calculation of a schedule is performed in the following steps:

1. The objects of the building model are linked to their Process Components.
2. The objects of the building model are linked to the elements of the Building Structure.
3. The entry nodes for the Process Components and the Building Structure are selected.
4. The schedule is calculated with
  - Start, finish, duration of tasks
  - Task dependencies
  - Quantities per task
  - Visualization parameters



#### 4.7.1 Calculation of schedules

To calculate the schedule, open the 'Create time schedule' tool from the Activities Ribbon bar.



For the calculation of the schedule, the entry nodes in the Process Components (1) and the Building Structure (2) must be defined.

 A screenshot of the 'Create Time Schedule' dialog box. The dialog has a title bar and a close button. Below the title bar, there's a section with a gear icon and the text 'Create Time Schedule', followed by a green play button labeled 'Create Time Schedule'. The main area contains three sections: 
 1. 'Process Components Root' (circled with a red box and a red circle with the number 1): It includes an icon of a puzzle piece and two input fields: 'ID' with the value '1e0b4cd1-b53e-4577-bc56-a0afd64ea755' and 'Name' with the value 'Example Order Keys'.
 2. 'Building Structure Root' (circled with a red box and a red circle with the number 2): It includes an icon of a document with a list and two input fields: 'ID' with the value 'bs::2N\_MsFeQT7wA6JSEh3b9nO-oh' and 'Name' with the value 'BS Model'.
 3. 'Project' section: It contains 'Start Date' set to '01.02.2023 08:00' and 'Calendar' set to '... (Default Calendar)', both with dropdown arrows.

The entry nodes define which substructures of the Process Components and Building Structure are to be taken into account. You can just drag-and-drop the appropriate entries of these two domains into the specific fields of the creation dialog.

Now define the start date for the schedule and one of the **Calendars** that should be used.

Start the calculation with the 'Create Time Schedule' button. As a result, a new model (that schedule) is created in the **Activities domain**.

The next chapters explain in detail, how to define Process Components and the automated structuring of the schedule.

#### 4.7.2 Define Process Components

The define Process Components, click on the appropriate button in the Activities Ribbon bar.

NAVIGATION	VIEW	DATA	BUILDING	ACTIVITIES	QUANTITIES	DOCUMENTS	TOOLS
Activities	Calculate	Garit	Charts	Components	Create	Visualisation	Settings
Activities	Activities	Charts	Calendars	Restrictions	Process Components	Options	

#	Name	Quantity	UoM	Predecessors	ProductionRate	Resource
1	<b>Construction Steps</b>				1,0000	1,0000
1.1	<b>Shell Construction</b>				1,0000	1,0000
1.1.1	Foundation ( » 8)	=8			1,0000	2,0000
1.1.2	Floor Plate ( » 1)	= [[cpVolume##xs:double##geometry]]*2		4.0 d FS 1.1.1;	1,0000	1,0000
1.1.3	Columns ( » 12)	=4		2.0 d FS 1.1.1;	1,0000	1,0000
1.1.4	Exterior walls ( » 6)	= [[cpVolume##xs:double##geometry]] m3		2.0 d FS 1.1.3;	1,0000	1,0000
1.1.5	Interior walls ( » 3)	= [[cpLateralArea2##xs:double##geometry]] m2		FS 1.1.4;	0,5000	1,0000
1.1.6	<b>Beams</b>	=4		1.0 d FS 1.1.3;	1,0000	1,0000
1.1.7	Slabs ( » 3)	= [[cpVolume##xs:double##geometry]]*2 m3		FS 1.1.4; FS 1.1.6;	1,0000	1,0000
1.2	<b>Openings and Facade</b>			FS 1.1;	1,0000	1,0000
1.2.1	Doors ( » 6)	= 2			1,0000	1,0000
1.2.2	Facade ( » 44)	=1			1,0000	1,0000
1.2.3	Windows ( » 20)	=2			1,0000	1,0000
1.3	<b>MEP installation</b>			FS 1.1;	1,0000	1,0000
1.3.1	MEP components ( » 41)	=3			1,0000	1,0000

Process Components are organized in a hierarchy. This hierarchy (level of detail) is applied to the schedule during the calculation. You can also create folders to group Process Components. These folders are not considered in the calculation of the schedule. Thus, folders help to reduce the hierarchies in activity models.

### Duration of tasks:

The final duration of a task in the schedule is calculation from the parameters 'Quantity', 'ProductionRate' and 'Resource'. The parameter 'UoM' (Unit of Measure) specifies what kind of quantity is meant, e.g. to build a concrete slab, the quantity is measured in  $m^3$ , the installation of multiple windows is measured in pieces.

The quantity is either calculated with a formula or specified with a fixed value. If the expression is preceded by an equal sign, the expression is interpreted as a formula, evaluated for each linked geometry object and summed up at the calculation. If a number is entered, this value is taken as a fixed quantity for the calculation.

### Example:



Process Components					
#	Name	Quantity	UoM	ProductionRate	Resource
> 1	Construction Steps			1,0000	1,0000
✓ 2	Example			1,0000	1,0000
2.1	fixed value for quantity ( » 4)	2,0000		1,0000	1,0000
2.2	fixed value per object ( » 4)	= 2		1,0000	1,0000
2.3	formula ( » 1)	= [[cpVolume##xs:double##geometry]]*2 m3		1,0000	1,0000

- **Component 2.1:** The quantity on the resulting operation is '2'. Regardless of the number of linked objects.
- **Component 2.2:** The quantity for each object is 2, in the sum of four linked objects the value is 8.
- **Component 2.3:** The quantity is calculated from the volume of each object, multiplied by 2 and summed over all linked objects

A variable is marked with [...]. A variable is defined by the name, the data type (xs:double, xs:int, xs:string, ...) and the domain (geometry, activities, pc, type, ...).

**Tip:** If you start typing a property name (e.g. cpVolume) into the quantity field, a list with all available properties appears.

The production rate describes how much hours one resource would need to fulfill this task (create one quantity unit). The resource indicates how many people are available for this work. You can adjust these number based on the real production rate and the available resources.

In the resulting schedule, the duration of the work is calculated as:

$$Duration = Quantity * Production Rate / Resource$$

This is the resulting schedule from the Process Components example above:

Activities			
#	Name	pc:Qty	Duration
> 1	Construction Steps / BS M...		10.13 days
✓ 2	Example / BS Model		2.56 days
2.1	Site		2.56 days
2.1.1	Building		2.56 days
2.1.1.1	Level 1		2.56 days
2.1.1.1.1	fixed value for quantity ( » 4)	2,0000	0.25 days
2.1.1.1.2	fixed value per object ( » 4)	8,0000	1.00 day
2.1.1.1.3	formula ( » 1)	20,5213	2.56 days

The property 'pc:Qty' shows the resulting quantity for this task based on the value or formula in the Process Component. The 'Duration' is the result of the quantity multiplied by the production rate (quantity unit per hour) and divided by the number of resources.



In this example a working day includes 8 hours. This can be defined in the **Calendars** section.

### Predecessor / successor relationships

The predecessors of a Process Component are defined in the 'Predecessors' property (see also the usage of the property 'Order Flag' in chapter **Structuring of schedules**) These dependencies are transferred to tasks that are generated from the Process Components.

#	Name	Predecessors	Quantity	UoM	Productic	Resource
1	Construction Steps				1,0000	1,0000
2	Example Quantity formulas				1,0000	1,0000
3	Example Order Keys				1,0000	1,0000
4	Example Dependencies				1,0000	1,0000
4.1	Example task A ( » 1)		= [[cpVolume#xs:double##geometry]]*2	m3	1,0000	1,0000
4.2	Example task B ( » 6)	3.0 d FS 4.1;	= [[cpVolume#xs:double##geometry]]	m3	1,0000	1,0000

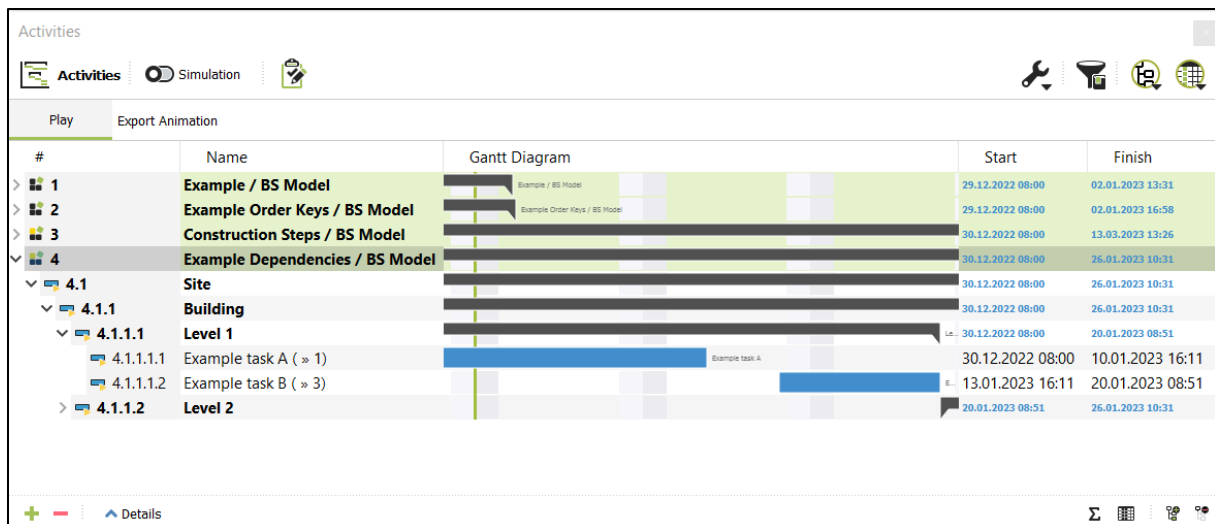
Enter dependencies in the form: *Buffer time // SSFF // Dependent element number*

Following keys are usable:

Key	Description
min, h, d, w	Buffer time in minutes, hours, days or weeks
SS	Start-Start dependency Both activities start together
SF	Start-Finish dependency The start of the first one is the end of the second one
FS	Finish-Start dependency The end of the first one is the beginning of the second one
FF	Finish-Finish dependency Both activities finish together.

### Example:

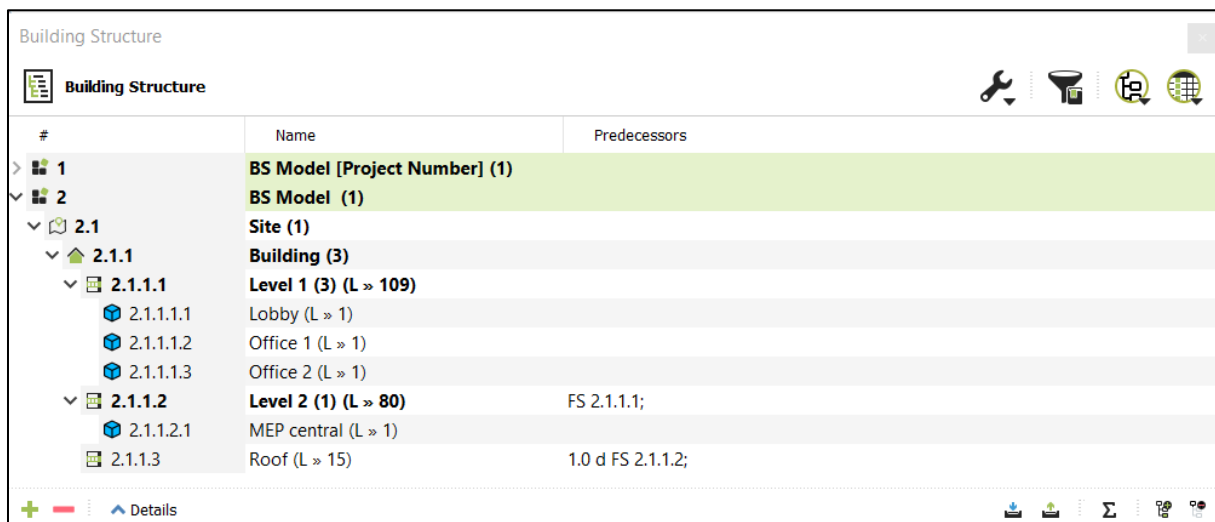
In the screenshot above, the Component 'Example task B' has the dependency '3,0 d FS 4.1' In the resulting schedule the task will start 3 days after the task for Component 'Example task A' has finished:



Dependencies are also defined in the **Building Structure**, which are transferred to tasks that are generated from these elements of the building structure. For this there is also in the building structure the property 'Predecessors'.

### Example:

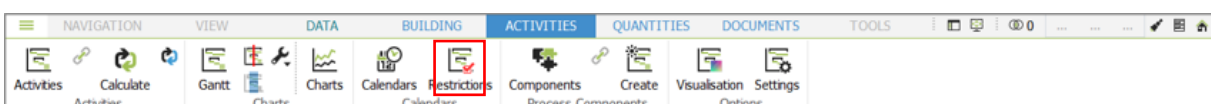
For the 'Level 2' a dependency is defined, so that all work will start after the 'Level 2' (with order number 2.1.1.1) is finished. And the work on 'Roof' starts 1 day after the work on 'Level 2' (order number 2.1.1.2) has finished.









### 4.7.3 Restrictions





It is possible to assign Restrictions to Process Components, to control whether a task should start on a certain day, at a certain time or should have a fixed duration (time).

Open the 'Restrictions' window from the Activities Ribbon bar.



Task Restrictions					
 <b>Task Restrictions</b>					
 Set to Visible  Set to Selected  Remove from Visible					
	Name	Comment	Start on Day	Start at Time	Fixed Time (Days)
1	Monday		Monday	09:00	
2	Fixed Time	...	Wednesday		5,0000 [d]
 					

With the '+' button, you can create new Restrictions. To assign a Restriction to a Process Component, drag and drop the Restriction to the appropriate Process Component. The name of Restriction is now visible in the property 'TaskRestrictions':

Process Components					
 <b>Process Components</b>					
#	Name	Predecessors	Quantity	UoM	TaskRestrictions
> 1	Construction Steps				
> 2	Example Quantity f...				
✓ 3	Example Order Keys				
3.1	Create Walls ( » 9)		=[[cpVolume##... m3		Monday
> 4	Example Dependen...				
   Details					

All tasks that are created from this Process Component will be defined based on this Restriction.

#### 4.7.4 Structuring of schedules

When calculating the schedule, a hierarchy is created in the Activities, which results partly from the Process Components and partly from the Building Structure:

		Process Component /	
		Tasks	
		Building Structure	
Part 1	PC: 'OrderFlag' = true	1 Process Components Sample / Building Structure Sample	298.94 days
Part 2		1.1 Construction	105.83 days
		1.1.1 Building A	56.46 days
		1.1.1... Level 1	41.28 days
Part 3	PC: 'OrderFlag' = false	1.1.1... Level 2	32.57 days
		1... Exterior walls concrete ( » 4)	4.72 days 63,0024 m3
		1... Interior walls brickwork ( » 12)	28.10 days 280,9691 m2
		1... Interior walls concrete ( » 6)	1.94 days 19,4292 m3
		1... Columns ( » 12)	2.25 days 12,0000
		1... Slabs ( » 1)	3.47 days 138,9051 m3
		1.1.1... Level 3	37.15 days
		1.1.1... Level 4	39.37 days
		1.1.1... Level 5	37.61 days
		1.1.1... Roof	2.14 days
		1.1.2 Building B	44.36 days
		1.2 Inner Space	298.94 days
		1.2.1 Building A	147.03 days
		1.2.1... Level 1	147.03 days
		1.2.1... Level 2	128.83 days
		1.2.1... Level 3	141.49 days
		1.2.1... Level 4	146.91 days
		1.2.1... Level 5	142.65 days
		1.2.2 Building B	146.91 days

The first part of the hierarchy results from the Process Components, the second part from the Building Structure. And the third part again from the Process Components.

The change between the first and the second part is defined in the Process Components by the property 'OrderFlag'.

Process Components				
#	Name	OrderFlag	Quantity	UoM
1	Process Components Sample	true		
1.1	Construction	true		
1.1.1	Exterior walls concrete ( » 54)	false	= [[cpVolume##xs:double##geometry]]	m3
1.1.2	Interior walls brickwork ( » 148)	false	= [[Area##xs:double##geometry]]	m2
1.1.3	Interior walls concrete ( » 76)	false	= [[cpVolume##xs:double##geometry]]	m3
1.1.4	Columns ( » 112)	false		1 Stk
1.1.5	Stairs ( » 73)	false		1 Stk
1.1.6	Slabs ( » 14)	false	= [[cpVolume##xs:double##geometry]]	m3
1.2	Inner Space	true		
1.2.1	Rooms ( » 122)	false		

To further subdivide the structure in the tasks, tasks can be created, based on a certain property value of the linked objects. For this purpose, the 'OrderKeys' property is used for extended grouping of objects.

For each value of the property that the linked objects have, a task is created and the objects are assigned with the corresponding value. The tasks are sorted by the value of the property.

These tasks are not associated with each other, which results in a parallelization of the activities.

### Example:

If the Process Component of the creation of walls is defined like this:

#	Name	OrderKeys	Quantity	UoM	P
1	Construction Steps				
2	Example Quantity formulas				
3	Example Order Keys				
3.1	Create Walls ( > 9)		=[cpVolume##xs:double##geometry]]	m3	

The resulting task will look like this (for a Building Structure with two storeys):

#	Name	Duration
1	Construction Steps / BS Model	10.13 days
2	Example / BS Model	2.56 days
3	Example Order Keys / BS Model	9.74 days
3.1	Site	9.74 days
3.1.1	Building	9.74 days
3.1.1.1	Level 1	5.11 days
3.1.1.1.1	Create Walls ( > 5)	5.11 days
3.1.1.2	Level 2	4.64 days
3.1.1.2.1	Create Walls ( > 4)	4.64 days

If you define a value for 'OrderKeys' based on the name property of the linked objects:

#	Name	OrderKeys	Quantity	UoM
1	Construction Steps			
2	Example Quantity formulas			
3	Example Order Keys			
3.1	Create Walls ( > 9)	[[cpName##xs:string##geometry]];	=[cpVolume##xs:double##geometry]]	m3

The resulting tasks will look like this. A sub-task is created for each linked object, sorted by the names:

Activities		
<div> <div>Activities</div> <div>Simulation</div> <div></div> </div> <div> <div>Play</div> <div>Export Animation</div> </div>		
#	Name	Duration
> 1	Construction Steps / BS Model	10.13 days
> 2	Example / BS Model	2.56 days
✓ 3	Example Order Keys / BS Model	3.00 days
✓ 3.1	Site	3.00 days
✓ 3.1.1	Building	3.00 days
✓ 3.1.1.1	Level 1	1.50 days
✓ 3.1.1.1.1	Create Walls	1.50 days
3.1.1.1.1.1	Basic Wall:Generic - 300mm:299076 ( » 1)	1.50 days
3.1.1.1.1.2	Basic Wall:Generic - 300mm:299077 ( » 1)	1.43 days
3.1.1.1.1.3	Basic Wall:Generic - 300mm:299079 ( » 1)	1.28 days
3.1.1.1.1.4	Basic Wall:Interior - 135mm Partition (2-hr):302842 ( » 1)	0.59 days
3.1.1.1.1.5	Basic Wall:Interior - 135mm Partition (2-hr):302958 ( » 1)	0.31 days
✓ 3.1.1.2	Level 2	1.50 days
✓ 3.1.1.2.1	Create Walls	1.50 days
3.1.1.2.1.1	Basic Wall:Generic - 300mm:299518 ( » 1)	1.50 days
3.1.1.2.1.2	Basic Wall:Generic - 300mm:299519 ( » 1)	1.43 days
3.1.1.2.1.3	Basic Wall:Generic - 300mm:299520 ( » 1)	1.28 days
3.1.1.2.1.4	Basic Wall:Interior - 135mm Partition (2-hr):303266 ( » 1)	0.42 days

It is also possible to extend the structure of the tasks on Building Structure level with the property 'TaktKey'. This makes it possible, to classify objects into tacts and automatically generate tacts in the schedule from them.

This property is defined as variable based on a property in the **Geometry Domain / Project Structure**.

### Example

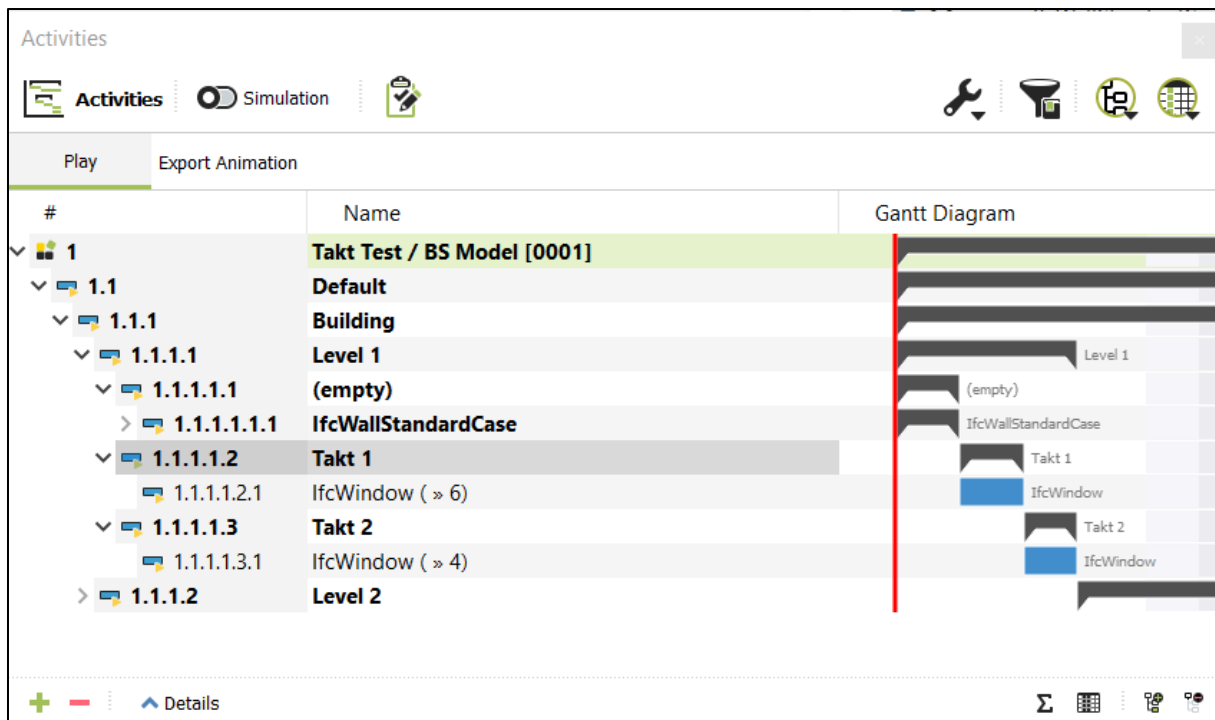
The property 'TaktName' was created in the Project Structure and a corresponding value was assigned to each window:

Project Structure		
Name	TaktName	ifcType
IFC Model [Cube_Architectural Model.ifc] (1)		
0001 (1)		IfcProject
Default (1)		IfcSite
Building (2)		IfcBuilding
Level 1 (14)		IfcBuildingStorey
Basic Wall:Generic - 300mm:299076		IfcWallStandardCase
Basic Wall:Generic - 300mm:299077 (1)		IfcWallStandardCase
Basic Wall:Generic - 300mm:299079 (4)		IfcWallStandardCase
Curtain Wall:Exterior Glazing:299276 (44)		IfcCurtainWall
M_Fixed:0915 x 1830mm:299891 (2)	Takt 1	IfcWindow
M_Fixed:0915 x 1830mm:300008 (2)	Takt 2	IfcWindow
M_Fixed:0915 x 1830mm:300062 (2)	Takt 1	IfcWindow
M_Fixed:0915 x 1830mm:300110 (2)	Takt 2	IfcWindow
Floor:Generic 300mm:302608		IfcSlab
Basic Wall:Interior - 135mm Partition (2-hr):302842 (2)		IfcWallStandardCase
Basic Wall:Interior - 135mm Partition (2-hr):302958		IfcWallStandardCase
M_Fixed:0915 x 1830mm:303024 (2)	Takt 1	IfcWindow
M_Single-Flush:0915 x 2134mm:303070 (2)		IfcDoor

In the Building Structure, the property 'TaktName' was assigned with the variable '[[TaktName##xs:string##geometry]]' to read the appropriate values from the Project Structure during the creation of the schedule:

Building Structure			
#	Name	TaktKey	Predecessors
1	BS Model [0001] (1)		
1.1	Default (1)		
1.1.1	Building (2)		
1.1.1.1	Level 1 (L » 79)	[[TaktName##xs:string##geometry]];	
1.1.1.2	Level 2 (L » 30)	[[TaktName##xs:string##geometry]];	FS 1.1.1.1;

In the resulting schedule, the structure includes now additional nodes for the different tacts (the tact 'empty' contains all objects of the Project Structure in the same hierarchy, but without a value in the property 'TaktName'):



#### Hint:

Tasks created from Process Components with the property 'OrderFlag' = true, are linked with dependencies defined in the property 'Predecessor' in the Process Component.

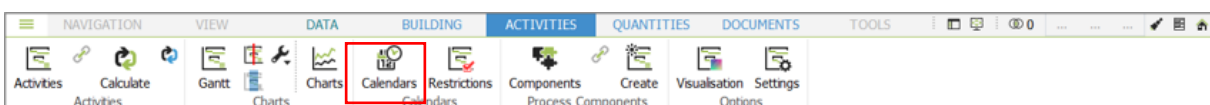
For tasks calculated from Process Components with the property 'OrderFlag' = false apply the following rules:

- Order relations, which are defined at the corresponding Process Component in the property 'Predecessors', are set in relation within the same hierarchy level of the Building Structure.
- Order relations, which are defined at the corresponding Process Component in the property 'PredecessorsLocation', are related to the tasks within the previous hierarchy level of the Building Structure.

For tasks created from the property 'TaktKey', the corresponding sub-tasks are linked with an finish-to-start order relation.

#### 4.7.5 Calendars

For the automated schedule calculation, you can define a calendar with the available working days and working hours per day. Open the Calendar dialog from the Activities toolbar.





Calendars

Calendars

Calendar List

	Name	Comment
1	... (Default Calendar)	
2	MyCalendar	custom calendar

+

-

Working Time

Calendar Options

January 2023

	Mo	Di	Mi	Do	Fr	Sa	So
52	26	27	28	29	30	31	1
1	2	3	4	5	6	7	8
2	9	10	11	12	13	14	15
3	16	17	18	19	20	21	22
4	23	24	25	26	27	28	29
5	30	31	1	2	3	4	5

Default-Week

Day

User Defined Days / Holidays

	Date	Comment
1	2022-12-31	New Year's Eve

+

-

Week / Day view

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
05:00 - 06:00							
06:00 - 07:00							
07:00 - 08:00							
08:00 - 09:00							
09:00 - 10:00							
10:00 - 11:00							
11:00 - 12:00							
12:00 - 13:00							
13:00 - 14:00							
14:00 - 15:00							
15:00 - 16:00							
16:00 - 17:00							
17:00 - 18:00							
18:00 - 19:00							

Set

Remove

- Calendar list:** You can define multiple calendars. Create new entries with the '+' button and delete existing entries with the '-' button.
- Calendar view:** Select specific days to define the working time. On the tab *Calendar Options* you can define default values for working hours per days, working days per week etc.
- User Defined Days / Holidays:** Define specific days, where the working time deviates from the default values. Create new entries with the '+' button and delete existing entries with the '-' button.
- Week / Day view:** Define the working hours for a default week and for selected days from the *User Defined Days* list. To do so, click on the columns in the table and then on 'Set' to add this hour to the working time, or on 'Remove' to remove it from the working time.

## 4.7.6 Process Component Properties

If a task in the schedule is calculated from Process Components, some properties that are defined for the Process Component, will be automatically written into the appropriate property of the task. The following table gives an overview of these properties:

69

Description	Process Component Property	Task Property	Value
Task is created from Process Component		pc:SourcePc	ID of the Process Component
Tasks is created from Building Structure		pc:SourceBs	ID of the Building Structure element
Visualisation in 4D Simulation	TaskVisualisationID	cpVisualisation	ID of the configuration rule
Restrictions	TaskRestrictionsID	cpRestrictions	ID of the Restriction element
Trade	Trade	pc:Trade	User input
Unit of measure	UoM	pc:QtyUom	User input
Quantity, forecast	Quantity	pc:Qty	Calculated by formula
Quantity, target	Quantity	pc:BaselineQty	Calculated by formula
Expense value, forecast	ProductionRate	pc:ProductionRate	User input
Expense value, target	ProductionRate	pc:BaselineProductionRate	User input
Resource value, forecast	Resource	pc:Resource	User input
Resource value, target	Resource	pc:BaselineResource	User input

## 4.8 Bill of Quantities

Bill of Quantities (BoQ) domain in DESITE allows you to import and display your BoQ tables easily and interactively, link 3D model objects to corresponding BoQ items and assign quantities calculated from model geometry. You can create your own BoQ table manually in DESITE or import from an external software via GAEB file (GAEB XML). Bill of Quantities domain can be accessed under Quantities menu tab.



In order to create your own BoQ table manually, click on '+' button at the bottom left corner and select the desired item from the menu that opens. Here you can generate a BoQ table or add further entries such as BoQ items, additional texts, take-off sheets etc. in an imported BoQ file.

You can rename the entries with a double click on values on Name field.

**Bill of Quantities**

WBS	Name
LV	desite MWM Musterprojekt Hochbau
01	Hauptgebäudeturm
01.01	Baustelleneinrichtung
01.01...	Baustelleneinrichtung einrichten, vorhalten, räumen
01.01...	Dokumentation der Bauabrechnung
01.02	Erdarbeiten
01.02...	Erdarbeiten nach DIN 18300
01.02...	Boden Baugrube lösen, lagern Tiefe bis 2,8 m Bodenklasse 3
01.02...	Boden Graben Kanal Tiefe bis 1,45 m lösen, lagern Bodenklasse 3
01.02...	Oberboden abtragen, lagern d= 30 cm Bodengruppe 4
01.02...	Betonpflaster L/B 100/100 mm H 80 mm grau
01.02...	Betonpflaster L/B 100/100 mm H 80 mm in MWM-blau
01.02...	Betrieb Wasserhaltungsanlage
01.03	Maurerarbeiten
01.04	Betonarbeiten
01.05	Nachträge
02	Nebengebäude

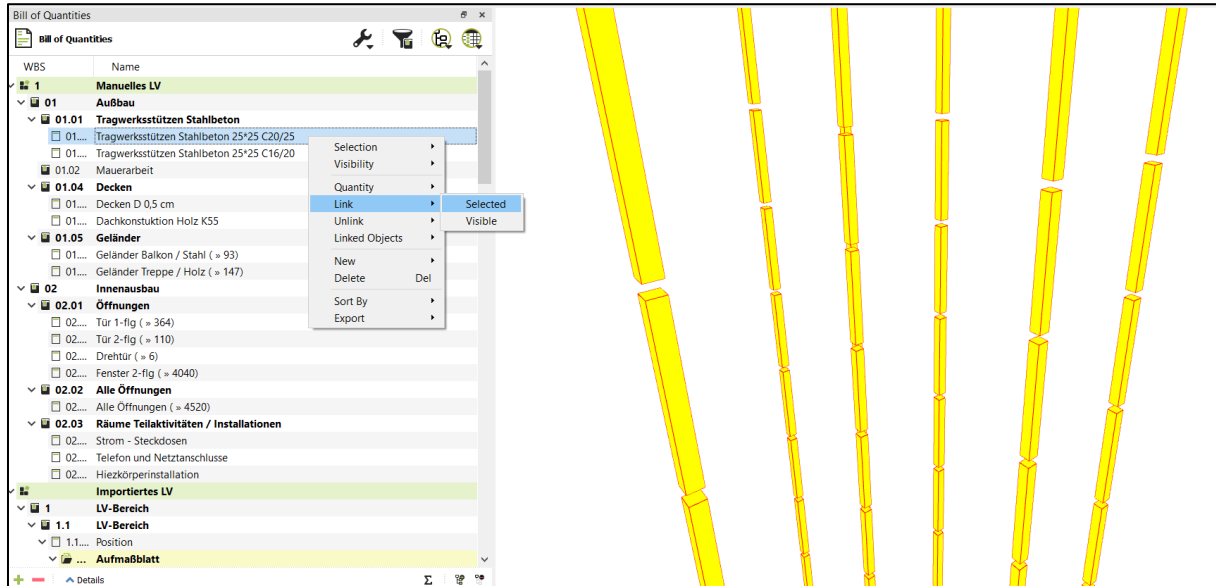
  

OZ	Menge	ME	EP in EUR	GP in EUR
01.02.0050	Betonpflaster L/B 100/100 mm H 80 mm in MWM-blau	0 m <sup>2</sup>	22.41	0.00

Pflasterdecke ZTV P-StB, Förderweg bis 4 km, aus Pflastersteinen aus Beton DIN EN 1338 blau, max. Differenzen J, Witterungswiderstand B, Abriebwiderstand H, gefast, Maße L/B 100/100 mm, Höhe 80 mm, in Reihen, in Fußgängerzonen, Bettung aus Kiessand, Körnung 0/4, Dicke 3 bis 5 cm, Pflasterfugen verfüllen mit Sand.

### 4.8.1 Add Objects to BoQ Items

With a right-click on the BoQ item, there are several options to add building components to it. One efficient way to do so is to first categorize the objects within Selection Sets (see [Selection Sets](#)).



### 4.8.2 Calculate Quantity of BoQ Items

In the following example, you will see how to calculate area of the floors. First of all display the property "Quantity" in the tree structure. For the calculation of the sum over the linked objects the function: `=SUM_LINKED(...)` is available, while the properties of the 3D objects to be evaluated in the formula are entered with two square brackets (how to use formulas, refer to [Formulas and Inheritance](#)).

WBS	Name	Quantity
1	Manuelles LV	
01	Außbau	
01.01	Tragwerksstützen Stahlbeton	
01.02	Mauernarbeit	
01.04	Decken	
01.04...	Decken D 0,5 cm ( » 14)	<code>=SUM_LINKED( [[cpVolume#xs:double#geometry]] )</code>
01.04...	Dachkonstruktion Holz K55	
01.05	Geländer	
02	Innenausbau	
	Importiertes LV	

Afterwards you can click on the sum logo to activate the calculation and show the result.

Bill of Quantities		
WBS	Name	Quantity
1	Manuelles LV	
01	Außbau	
01.01	Tragwerksstützen Stahlbeton	
01.02	Mauerarbeit	
01.04	Decken	
01.04...	Decken D 0,5 cm ( » 14)	1.690,8545
01.04...	Dachkonstruktion Holz K55	
01.05	Geländer	
02	Innenausbau	
	Importiertes LV	

Also, mathematical formulars are allowed, such as:

`=SUM_LINKED( 0.5 * [[cpVolume##xs:double##geometry]] )`

Furthermore, you can calculate with more than one attribute:

`=SUM_LINKED([[cpVolume##xs:double##geometry]] *  
[[Bewehrungsgrad##xs:double]])`

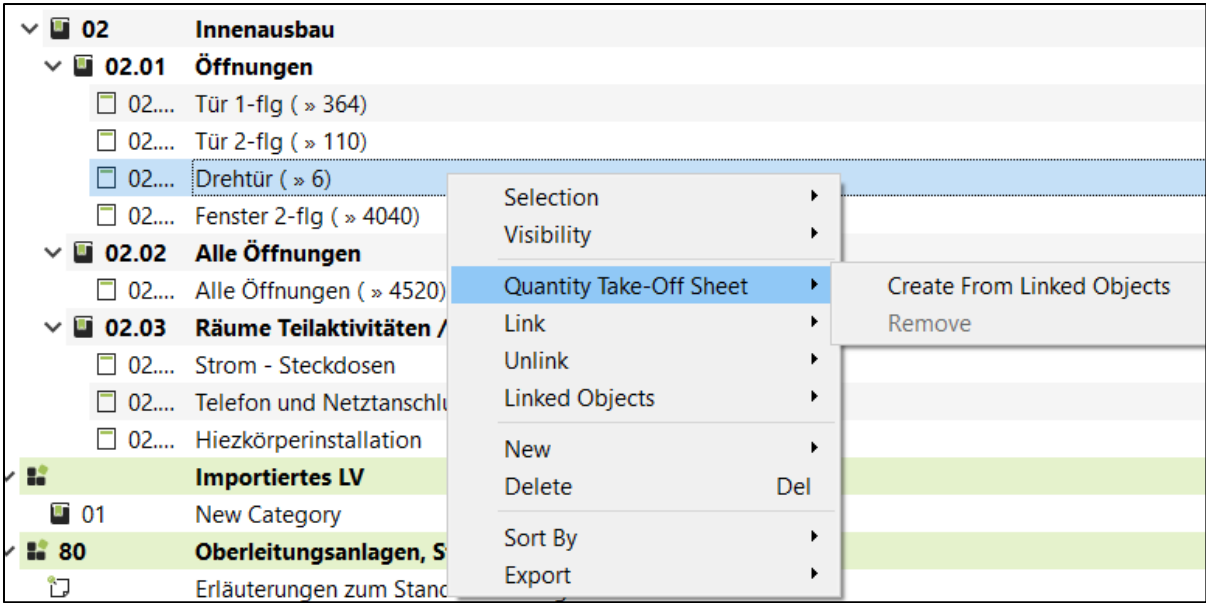
### 4.8.3 Quantity Take-off Sheet

For each BoQ item, one can create quantity take-off sheet ( "Aufmaßblätter" in German) according to the standard of GAEB X31 or DA11. It can represent the amount of work that has been done and is used e.g. to create the invoice for partial performance on the construction site.

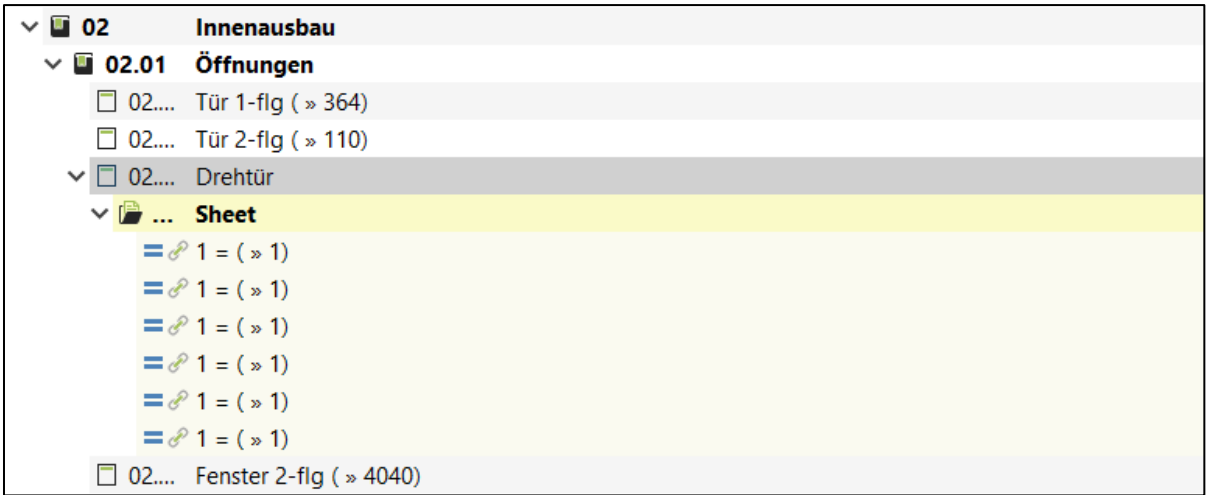
To create the quantity take-off sheet, you should first choose one BoQ item, for instance:

02	Innenausbau
02.01	Öffnungen
02....	Tür 1-flg ( » 364)
02....	Tür 2-flg ( » 110)
02....	Drehtür ( » 6)
02....	Fenster 2-flg ( » 4040)
02.02	Alle Öffnungen
02....	Alle Öffnungen ( » 4520)

You can then find the option for creating the sheet via context menu (open by right-click):



Select "Create from linked objects" and you will see that a new sheet is created under the item, in which the objects linked to the item are listed one after the other. This means that the number of objects listed is identical to the number that was in the brackets of the BoQ item.



To calculate quantity of each object in the sheet, you can directly enter the formular in the name of the object, such as:

WBS	Name	qto:Quantity
02.01	Öffnungen	
02.01.0001	Tür 1-flg (> 364)	
02.01.0002	Tür 2-flg (> 110)	
02.01.0003	Drehtür	=SUM
0001	Sheet	
A0	29+57 (> 1)	
A1	+100= (> 1)	186,0000
A2	2*[[cpVolume##xs:double##geometry]]= (> 1)	0,8992
A3	1= (> 1)	1,0000
A4	1= (> 1)	1,0000
A5	1= (> 1)	1,0000

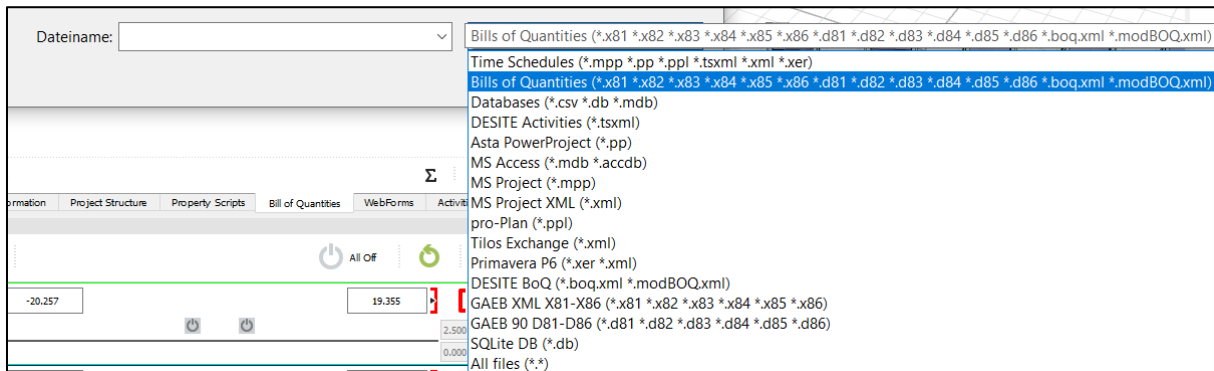
Or you can enter the formular in the "qto: Approach" of BoQ-item, before the quantity take-off sheet is created. And by creating the quantity take-off sheet from linked objects, the formular will be applied to each quantity-calculation of the objects:

WBS	Name	qto:Quantity	qto:Approach
02.01	Öffnungen		
02.01.0001	Tür 1-flg ( » 364)		
02.01.0002	Tür 2-flg ( » 110)		
02.01.0003	Drehtür ( » 6)	=SUM	[[cpVolume##xs:double##geometry]]*1.1=
02.01.0004	Fenster 2-flg ( » 4040)		

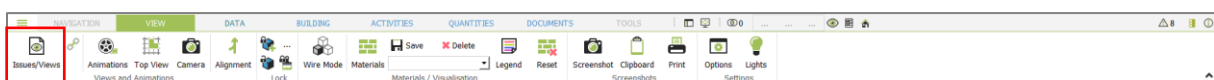
WBS	Name	qto:Quantity	qto:Approach
02.01	Öffnungen		
02.01.0001	Tür 1-flg ( » 364)		
02.01.0002	Tür 2-flg ( » 110)		
02.01.0003	Drehtür	=SUM	[[cpVolume##xs:double##geometry]]*1.1=
0001	Sheet		
A0	[[cpVolume##xs:double##geometry]]*1.1= ( » 1)	0,4216	
A1	[[cpVolume##xs:double##geometry]]*1.1= ( » 1)	0,0000	
A2	[[cpVolume##xs:double##geometry]]*1.1= ( » 1)	0,4945	
A3	[[cpVolume##xs:double##geometry]]*1.1= ( » 1)	0,4216	
A4	[[cpVolume##xs:double##geometry]]*1.1= ( » 1)	0,4945	
A5	[[cpVolume##xs:double##geometry]]*1.1= ( » 1)	0,0000	

#### 4.8.4 Import/Export BoQ

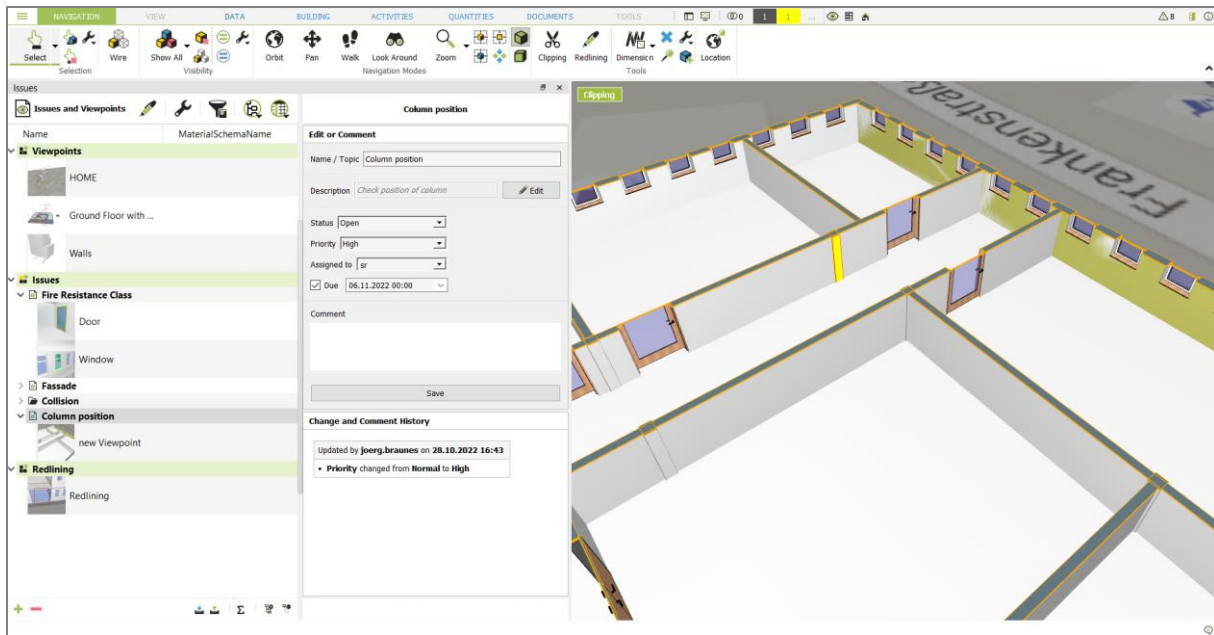
To import BoQ item, you can either choose the option "Add Time Schedules, Bill of Quantities,..." under application menu, or click on button "Import" in BoQ domain. The common formats which are prescribed by GAEB, such as X81-86, D81-86, X31, DA11, as well as native format of DESITE \*.boq.xml are supported.



#### 4.9 Issues and Viewpoints



Issues and 3D Viewpoints are displayed in a tree view as any other domain objects. You can open tree view by clicking on the Issues/Views button in the View toolbar.



Tree structure is broken down into the following elements:



Issue/Viewpoint Model. The Model can contain multiple Issues and 3D Viewpoints.



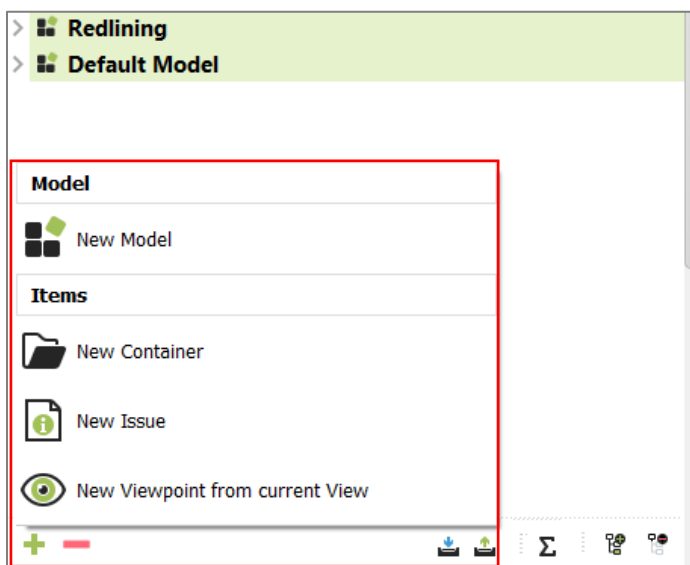
Container. A Container can be used to organize Issues and 3D Viewpoint in different groups



Issue.

*Preview* 3D Viewpoint

To create a new element for the tree structure, click the '+' Button and choose one of the options:

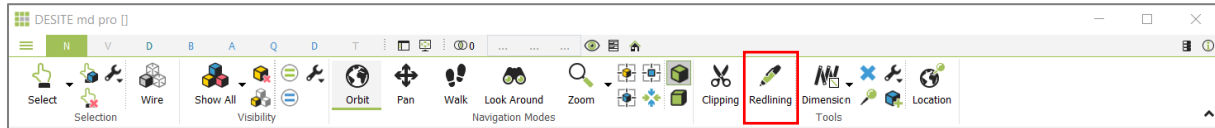


To delete an element of the tree structure, select the desired element and click the '-' button.

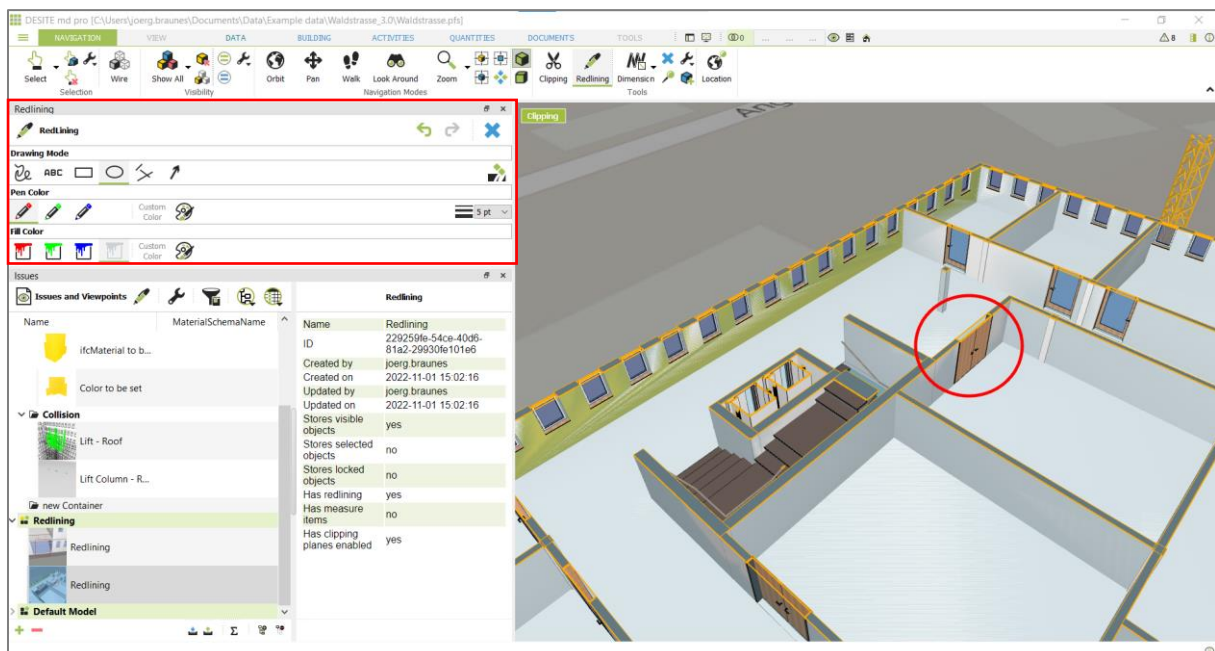


## 4.9.1 Redlining

Redlining tool provides functions for placing comments and annotations within the 3D model. Click on the Redlining button in the Navigation toolbar or in the Issues and Viewpoints Widget to open the palette for markup tools.



You can adjust settings for Drawing Modes, Pen color and Fill color and choose different colors, shapes or pen thickness values.



You can erase markups by using the eraser while redlining.



You can remove last markup item or restore the last removed item using arrow buttons.

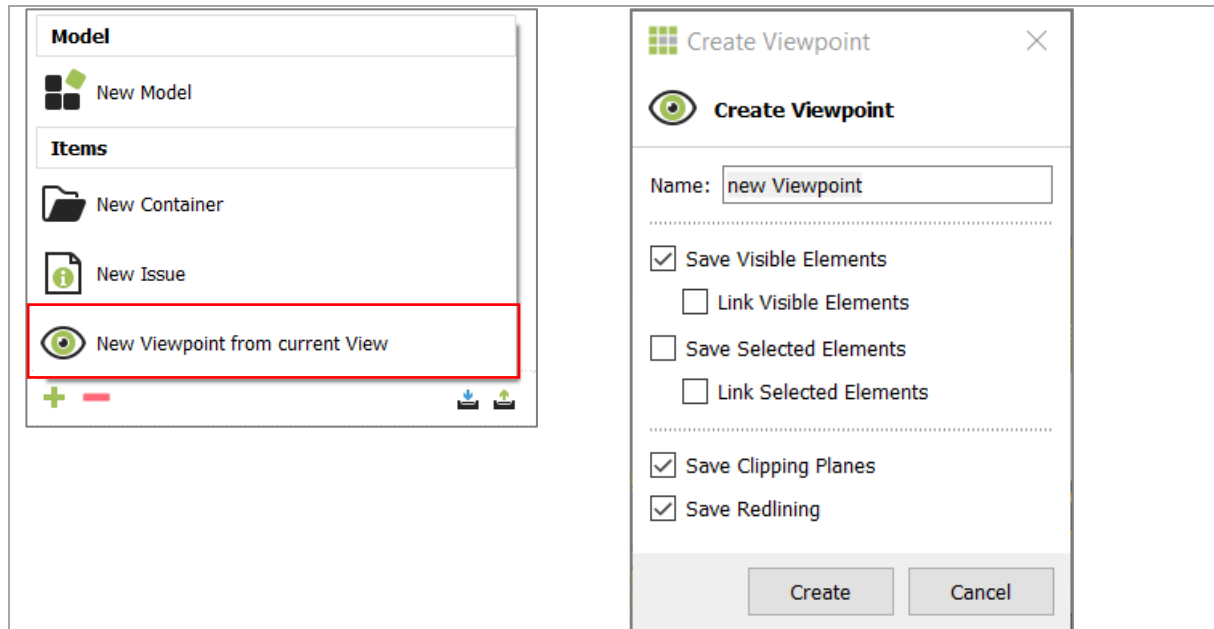


You can remove all markup items by clicking on X button

A 3D viewpoint will automatically be created and saved in the tree view when you create a markup.

### 4.9.2 3D Viewpoint

If you want to create a 3D viewpoint manually, set the desired view in the model, add comments or annotations with redlining as required, click on the '+' button and choose the viewpoint option. A new dialog appears, where you can define what information should be saved together with the viewpoint:



The 3D Viewpoint is saved in the tree structure with a small preview image of the current view. Saved viewpoints can be restored with a double click on the element in the tree structure. Changing the name if a 3D Viewpoint is possible by changing the value of the 'cpName' property, with the a click on 'F2'.

### 4.9.3 Issues

Issues can be used to save observations in the model, share them with other project members and track progress on solving these issues. You can create new issue by clicking on the '+' button and choose 'New Issue'. This opens a new widget in the side panel of the view/issues widget:

The screenshot shows the 'Issues' application interface. On the left is a tree view titled 'Issues' with a sub-header 'MaterialSchemaName'. The tree contains several categories: 'Viewpoints', 'Issues' (expanded), 'Fire Resistance Class', 'Facade' (expanded), 'Collision' (expanded), 'Redlining' (expanded), and 'Default Model'. Under 'Issues', there are items like 'ifcMaterial to b...' and 'Color to be set'. Under 'Collision', there are 'Lift - Roof' and 'Lift Column - R...'. Under 'Redlining', there are two 'Redlining' items. At the bottom of the tree is 'newIssue'. On the right is a 'New Issue' form. It has a 'Name / Topic' field with 'newIssue'. Below it are dropdowns for 'Status' (Open), 'Priority' (Normal), and 'Assigned to'. There is a checkbox for 'Due' with a date '02.11.2022 00:00'. Below these is a 'Description' text area. A checkbox labeled 'Attach (reparent) currently active Viewpoint to this Issue' is checked. At the bottom of the form is a 'Save' button.

1. **Name / Topic:** Define a meaningful name for the issue.
2. **Status:** Define the status of the issue. Options in the drop down are customizable in the Domain Configuration.
3. **Priority:** Define the priority of the issue. Options in the drop down are customizable in the Domain Configuration.
4. **Assigned to:** Define a name who should take care of the issue. Available names are customizable in the Domain Configuration.
5. **Due Date:** Define a date until the issue should be solved. This is optional
6. **Description:** Add details about the issue
7. **Attach active Viewpoint:** If this option is selected, the active viewpoint will be attached as child to the issue. If the active viewpoint already exists in the tree structure as an element, this viewpoint element will be moved under the issue.
8. **Save:** Save the issue with the current options and descriptions in the project.

To edit an existing issue, click on the element in the tree structure. Now you can edit all options. As soon as you click on save, the changes will be added as entries into the Change and Comment History:

The screenshot shows the 'Issues' application interface. On the left is a tree structure with categories like Viewpoints, Issues, Fire Resistance Class, Fassade, Collision, Redlining, Door, and Default Model. The 'Door' category is selected, showing a list of issues. On the right is a detailed view for the 'Door' issue. The 'Edit or Comment' section includes fields for Name / Topic (Door), Description (Please ch...ems wrong), Status (Closed), Priority (Normal), Assigned to (sr), and Due (02.11.2022 00:00). There is an 'Edit' button next to the description. Below this is a 'Comment' text area. A 'Save' button is highlighted with a red box. Below the 'Save' button is the 'Change and Comment History' section, which shows two entries: one where the status changed from 'In Progress' to 'Closed' and another where it changed from 'Open' to 'In Progress'.

#### Tip:

Issues can also be created automatically from Clash Detection and also from Model Checks.

#### 4.9.4 Issue Domain Configuration

Domain Configuration allows you to customize options for viewpoints and issues.

Issues

Issues and Viewpoints

Issues and Viewpoints Configuration

Users Status Priority Preview Defaults

This is a list of allowed Status values on Issues.

Status values currently in use by at least one Issue are shown in *italics*. Modifying one of these will also modify the Issue.

Status	Default	Mapped alias values
Open	<input checked="" type="checkbox"/>	New
In Progress	<input type="checkbox"/>	Active;Assigned
Closed	<input type="checkbox"/>	Ignored;Resolved
Re-Opened	<input type="checkbox"/>	

When creating, importing or updating an Issue with a Status not in the list:

☒ Extend the list of allowed Status values

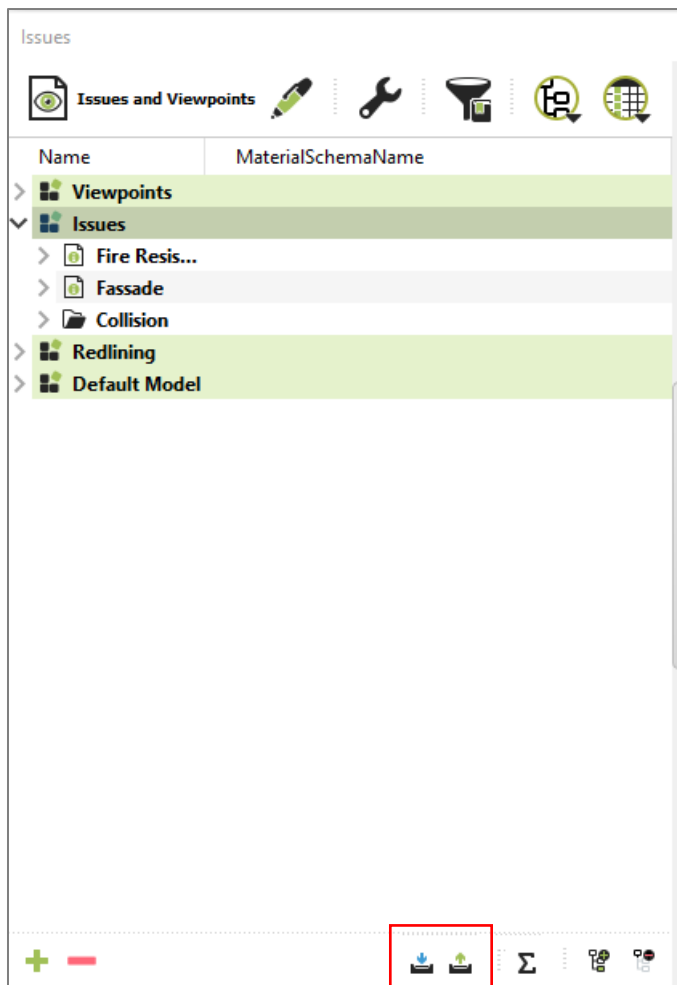
☐ Use the default value (on new Issues) or reject the change (on edit)

OK Cancel

- **Users:** Define a list of user names that will be used as creators and assignees for Issues
- **Status:** A list of statuses that will be used for Issues. You can also define the behaviour if Issues are imported from BCF and contain different status than defined in DESITE.
- **Priority:** A list of priorities that will be used for Issues. You can also define the behaviour if Issues are imported from BCF and contain different priority values than defined in DESITE.
- **Preview:** Define how the preview image and thumbnail for 3d viewpoint will be created.
- **Defaults:** Define the default settings for new 3d viewpoint that will be created together with Issues.

#### 4.9.5 Import and Export of Issues

Issues and Viewpoints can be imported to the project and also exported in order to share with other project stakeholders.



Exports the selected issues or viewpoints as combined or individual BCF files or creates a PDF report. For the BCF export a dialog appears where additional export settings can be defined.



Import issues and viewpoint as \*.bcf or \*.bcfzip or the legacy native format \*.vpxml. If the ID of an imported issue already exists in the project, the issue will be updated with the imported information.

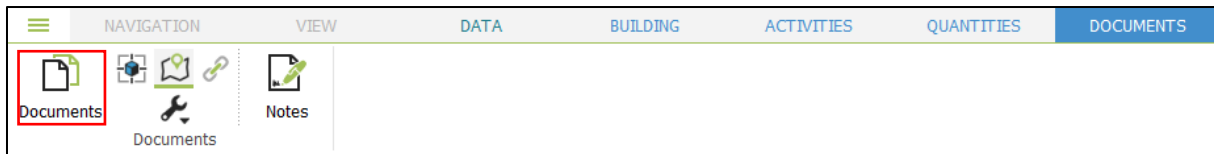
## 4.10 Documents Domain

In the Documents domain, external documents from various sources can be added and managed:

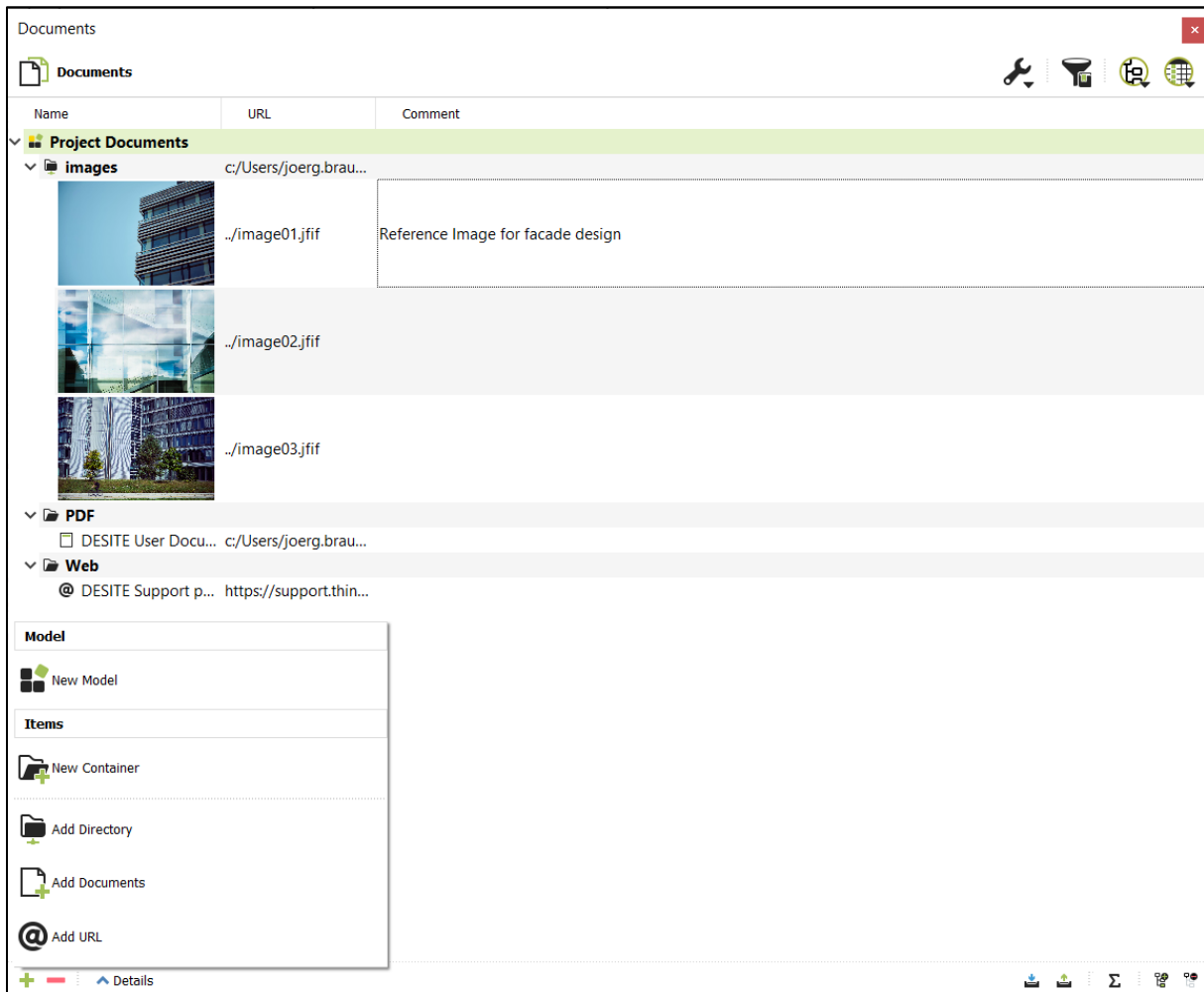
- individual files like images or PDF documents,
- directories from the file system,
- directories from a Webservice
- URLs

Documents can be linked to geometric objects with manual linking or **Rule based linking** and also placed in the 3D space.

Open the Documents dialogue from the Documents Ribbon tab:



Similar to other domains, Documents are organized in a hierarchical structure with models, containers and elements, which represent the external documents.

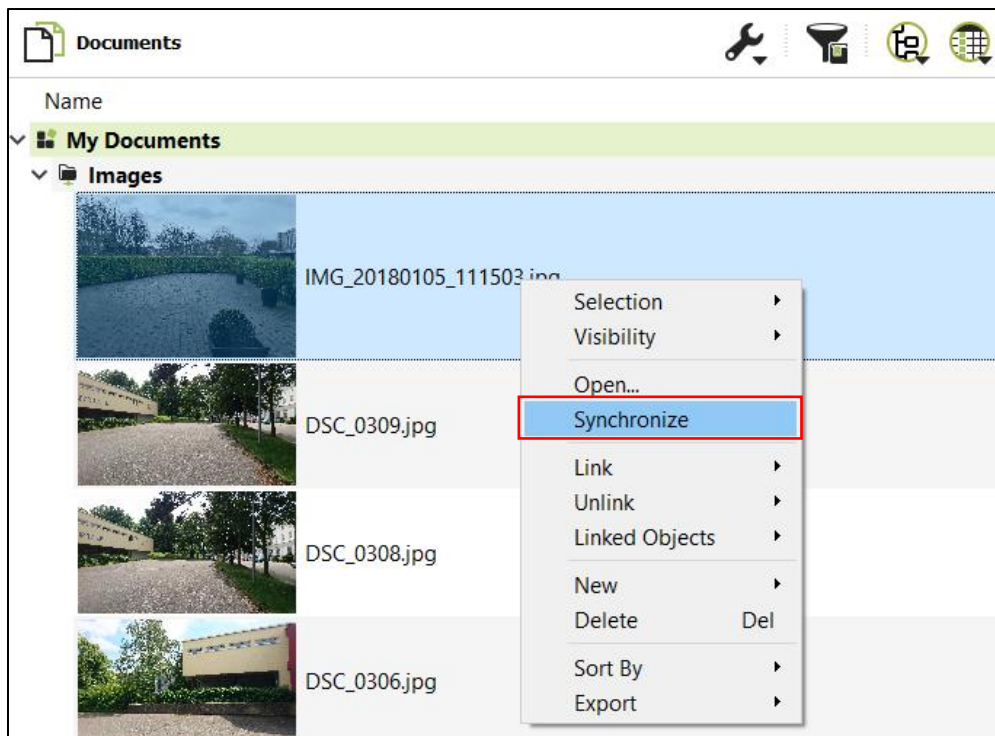


With the '+' button, you can add new elements to the tree structure.

Double click on the element name or the preview image to open the linked Document in related Windows application.

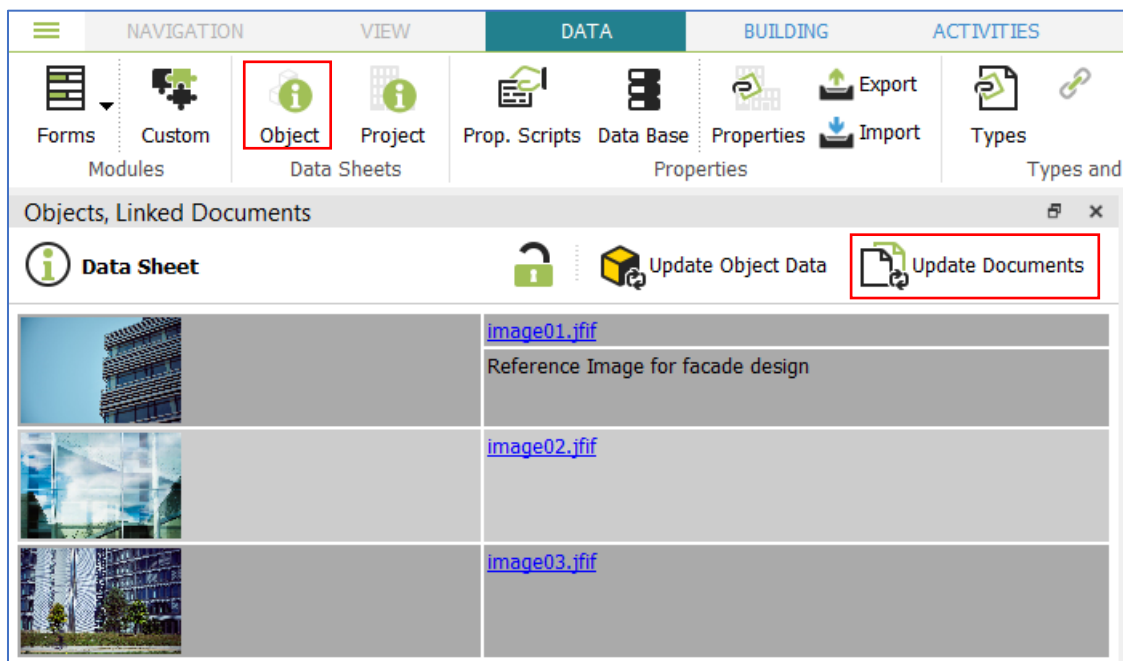
If you add a 'URL' Document, display the property 'URL' and enter the appropriate address as property value.

If a directory is added as a Container (e.g. the Container 'Images' in the screenshot above), all including files are added automatically as child elements. Right click on the Container and choose 'Synchronize Containers' to update the elements based on the current file status in the source folder:



### Tip:

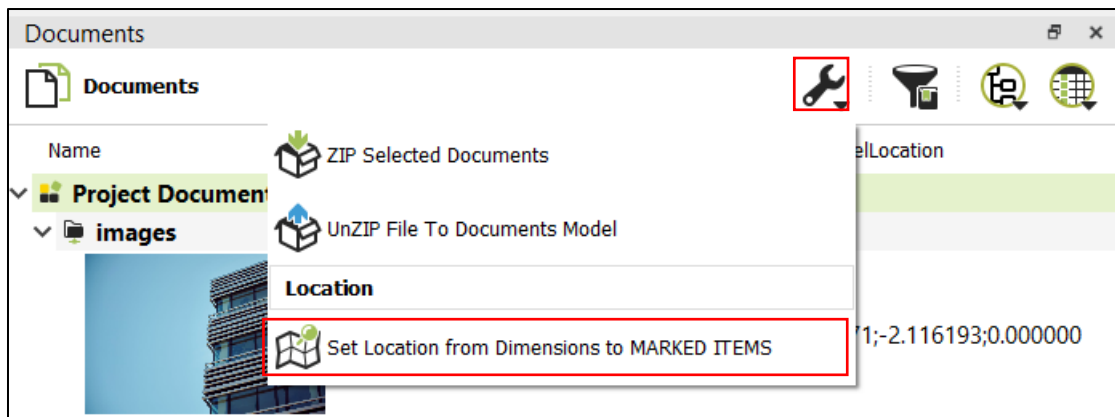
If a Document is linked to a geometry object, this Document can also be listed in the **Data Sheet**:



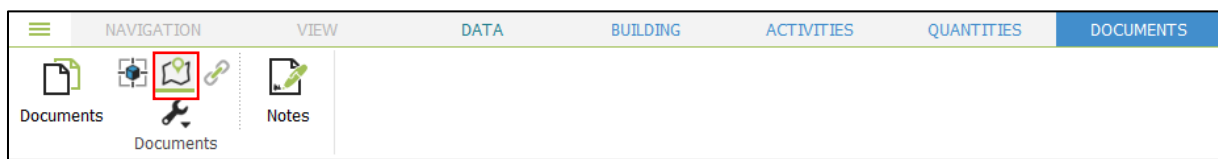
### 4.10.1 Place documents in 3D Space

It's possible to assign a coordinate in the 3d space to a Document. You can either enter the coordinate manually into the property 'ModelLocation' in the form '[x];[y];[z]', or you add a measurement point (see **Measuring**) to the model and choose the option 'Set Location from Dimensions to MARKED ITEMS' from the Documents tools:

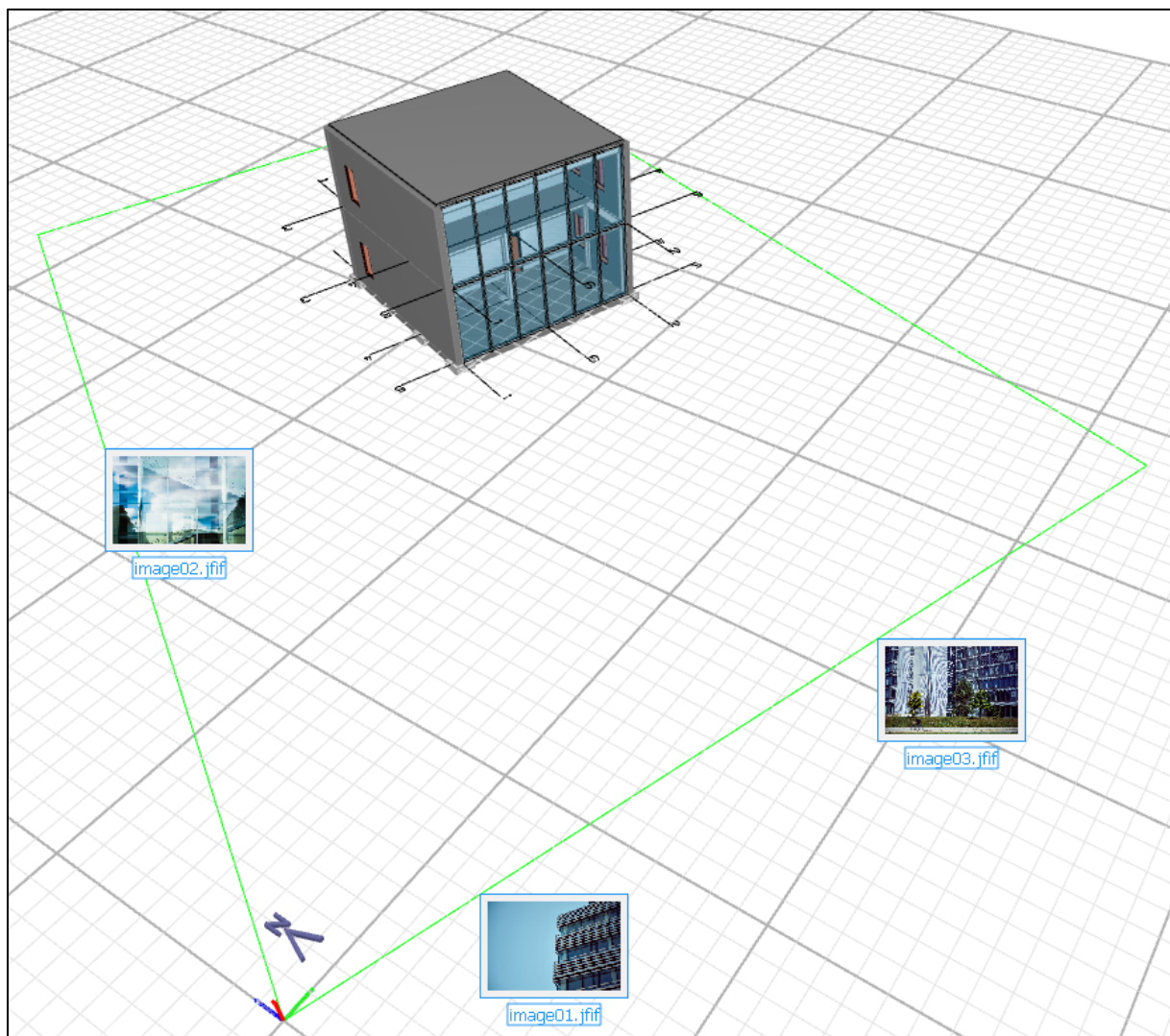




To display a preview of the located Documents in the 3D View, active the 'Show Location of Documents' option from the Ribbon bar:

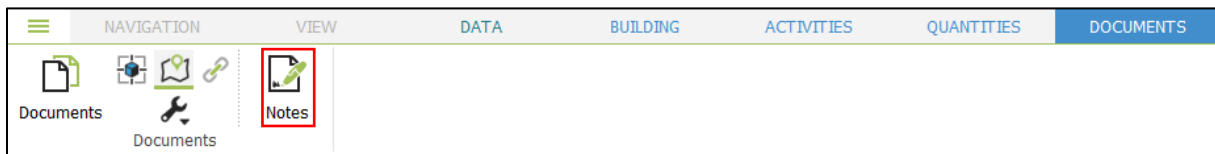


The Documents are now visible on the related geometry point in the 3D View:

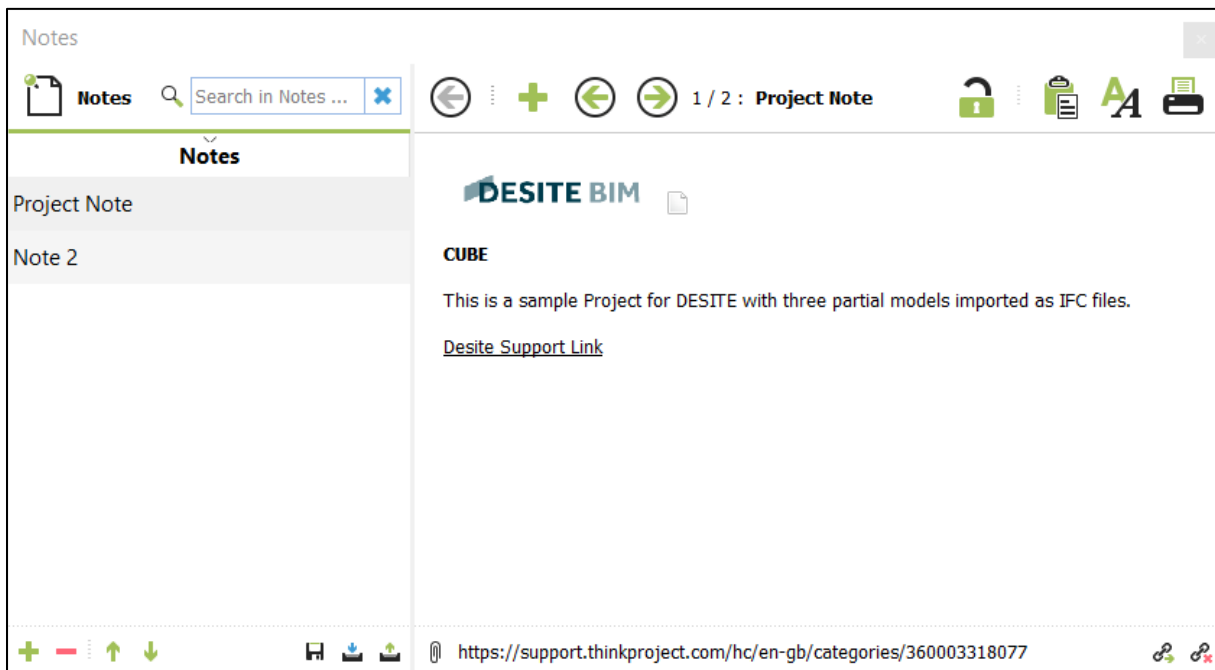












#### 4.10.2 Notes

The Notes tool provides functions for creating general notes and descriptions about the project. Open the Notes window from the Documents Ribbon tab:



You can create new notes and edit, save, print, delete, import or export existing notes.



-   Show / hide Notes list
-  Create a new Note
-   Previous / next Note
-  Add link to a local file to marked text
-  Add URL to marked text
-  Remove URL / file link of marked text
-  Set Note editable yes / no
-  Insert options for:
  - Image link
  - Document link
  - Current user and date stamp



Formatting options for paragraph, text, images



Print, PDF export

### Options for Notes list



Add / remove Note



Move Note in list up / down



Save Note



Import / export Note

### 4.10.3 Documents Domain properties

Following domain specific properties are used in the Documents domain:

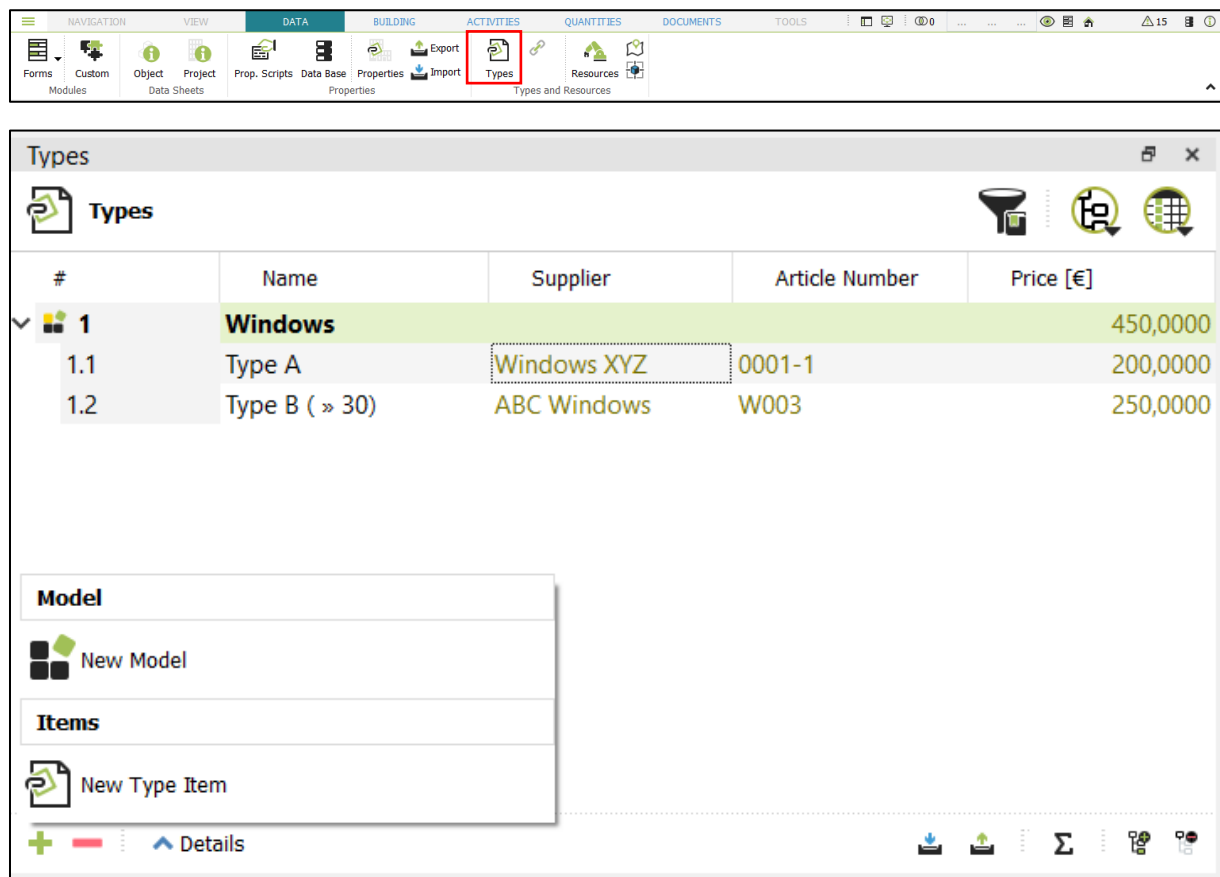
Name	Data type	Description
URL	xs:anyURI	For URL elements: the web address; for documents and directories: relative path in the project directory
GlobalURL	xs:anyURI	For URL elements: the web address; for documents and directories: absolute path in the project directory
RemoteURL	xs:anyURI	The web address of the directory added by a WebService.
ModelLocation	xs:string	Model coordinates of the point assigned to the document
GpsLocation	xs:string	GPS coordinates of the point assigned to the document
GpsLocationJSON	xs:string	GPS coordinates (in JSON format) of the point assigned to the document
GpsDirectionJSON	xs:string	For photo documents: GPS data (in JSON format) for the direction in which the photo was taken.
RefLocationID	xs:IDREF	ID (in the project structure) of the alignment linked to the document or ID of the section of the building structure linked to the document.
RefLocationFrom	xs:double	Start value of the stationing related to an alignment linked to the document.

RefLocationTo	xs:double	End value of the stationing related to an alignment linked to the document.
doc:HEAD	xs:string	Only for documents created from a .cp.json or .cpjson file and containing JSON with the key 'HEAD': the HEAD value
doc:DATA	xs:string	Only for documents created from a .cp.json or .cpjson file and containing JSON with the key 'DATA': the DATA value

## 4.11 Types Domain

The domain Types allows you to group objects with the same property values into types, in order to manage and edit them more effectively.

Similar to the other domains, Types are organized in a hierarchical structure. Open the Types dialogue from the Data Ribbon tab:



The screenshot shows the 'Types' ribbon tab in the software interface. The ribbon includes buttons for 'Forms', 'Custom', 'Object', 'Project', 'Prop. Scripts', 'Data Base', 'Properties', 'Export', 'Import', 'Types', and 'Resources'. The 'Types' button is highlighted with a red box. Below the ribbon, the 'Types' dialog box is open, displaying a table of types and a sidebar with options to create new models and items.

#	Name	Supplier	Article Number	Price [€]
1	<b>Windows</b>			450,0000
1.1	Type A	Windows XYZ	0001-1	200,0000
1.2	Type B ( » 30)	ABC Windows	W003	250,0000

**Model**

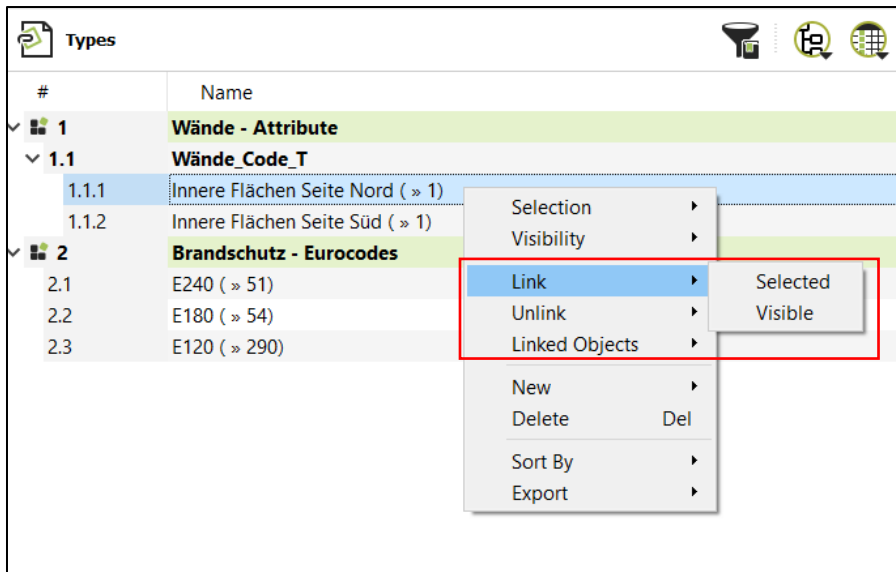
- New Model

**Items**

- New Type Item

At the bottom of the dialog, there is a '+' button and a 'Details' button.

You can add new Types models and Type items with the '+' button. To manually assign geometric objects to the Types, right click on the appropriate Type element and choose one of the options for manual linking:

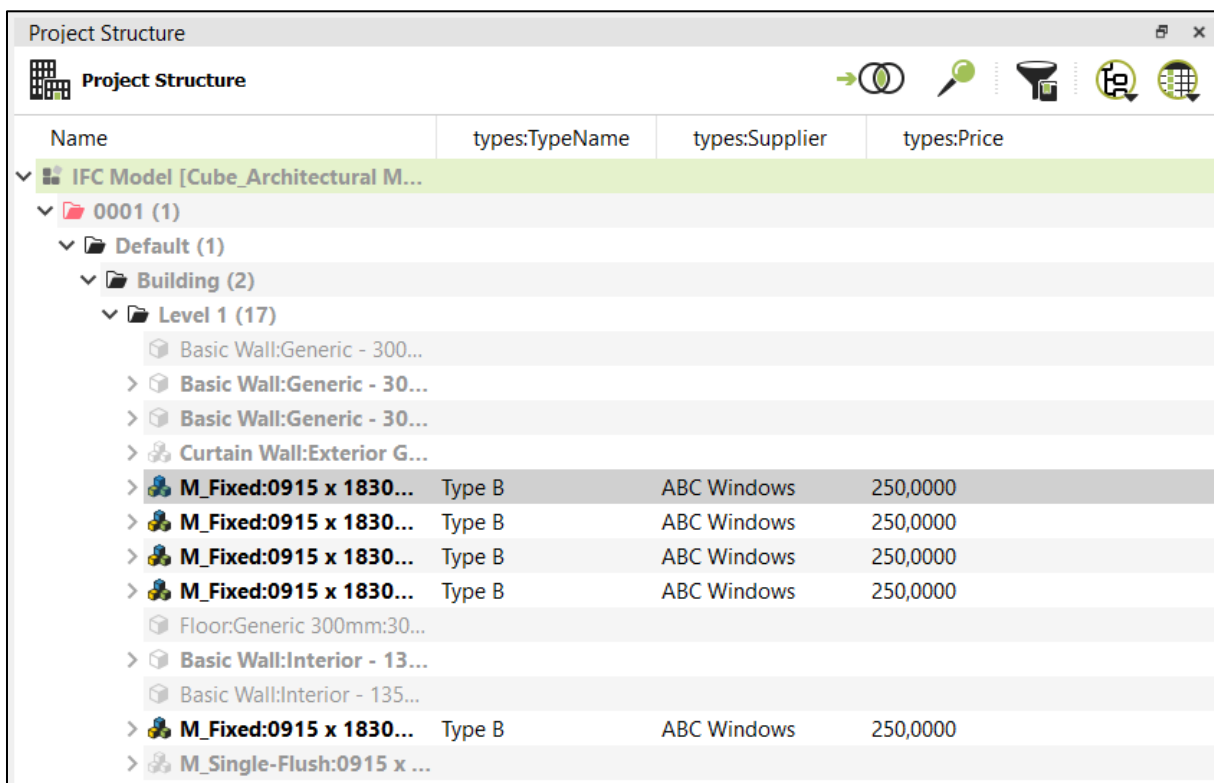


You can also use **Rule based linking** to assign geometry objects to the Types.

### Example:

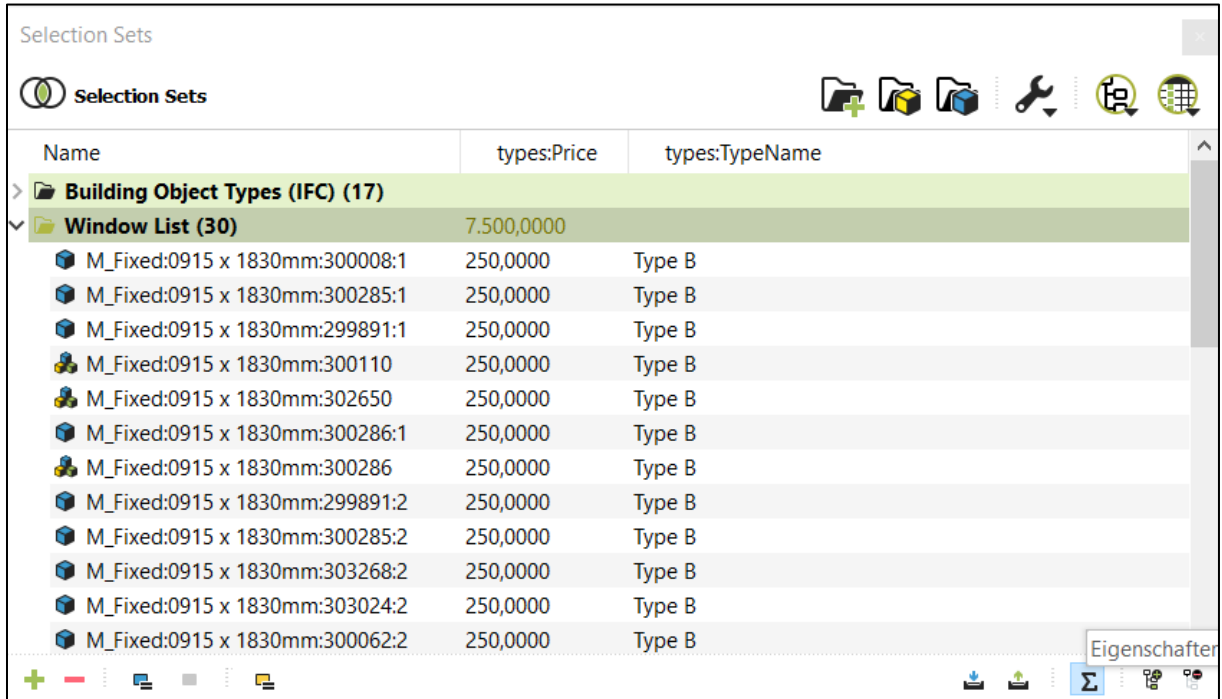
In the screenshot above, a new Type model with two Types ('Type A' and 'Type B') were defined. For these Types custom properties for supplier, article number and price were defined (see also: **Custom Properties Types**).

The property values for each Type are now also available for the linked geometry objects in the Project structure. The prefix 'types:' indicates, that these properties are derived from the linked Type:



If a property value in the Types definition is changes (e.g. a new price), this value is immediately visible in the Project structure.

In the following screenshot, a **Selection Sets** was created that contains all windows in the project. With the type property 'types:Price' visible, it is now easily possible to calculate the total price for all windows with the formula '=SUM' (see also: **Formulas and Inheritance**) in the top container:

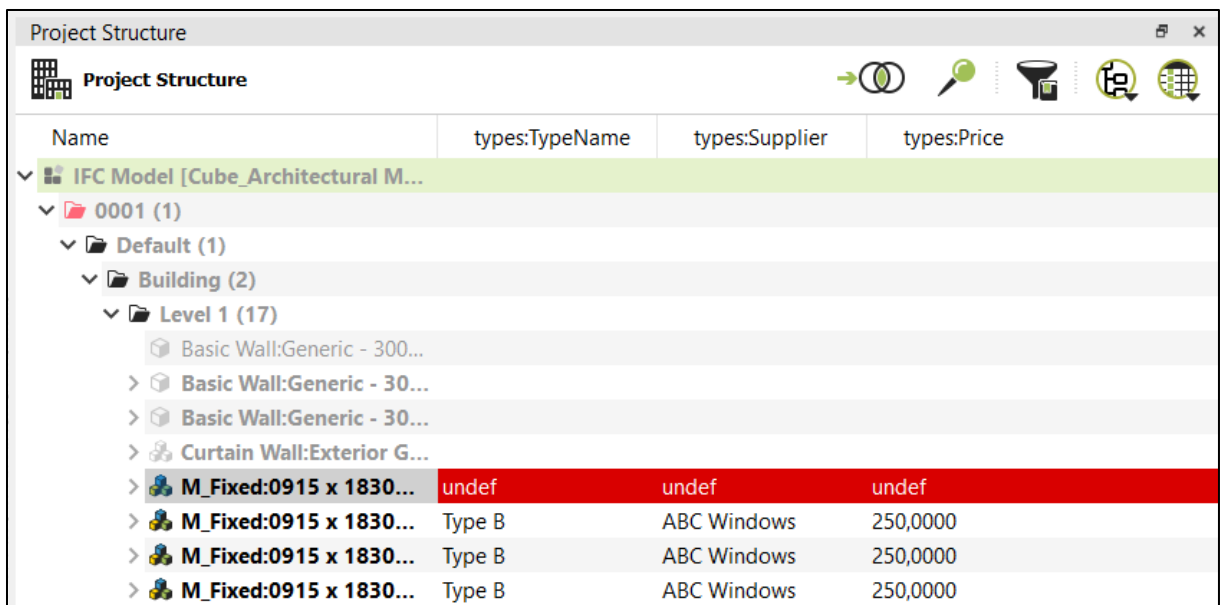


The screenshot shows the 'Selection Sets' window with a table of window types. The table has three columns: 'Name', 'types:Price', and 'types:TypeName'. The 'Window List (30)' is expanded, showing 12 rows of window types, all with a price of 250,000 and type 'Type B'. The total price for the list is 7,500,000.

Name	types:Price	types:TypeName
<b>Building Object Types (IFC) (17)</b>		
<b>Window List (30)</b>	7.500,0000	
M_Fixed:0915 x 1830mm:300008:1	250,0000	Type B
M_Fixed:0915 x 1830mm:300285:1	250,0000	Type B
M_Fixed:0915 x 1830mm:299891:1	250,0000	Type B
M_Fixed:0915 x 1830mm:300110	250,0000	Type B
M_Fixed:0915 x 1830mm:302650	250,0000	Type B
M_Fixed:0915 x 1830mm:300286:1	250,0000	Type B
M_Fixed:0915 x 1830mm:300286	250,0000	Type B
M_Fixed:0915 x 1830mm:299891:2	250,0000	Type B
M_Fixed:0915 x 1830mm:300285:2	250,0000	Type B
M_Fixed:0915 x 1830mm:303268:2	250,0000	Type B
M_Fixed:0915 x 1830mm:303024:2	250,0000	Type B
M_Fixed:0915 x 1830mm:300062:2	250,0000	Type B

### Hint:

A geometry object should only be linked with one Type element. If you link a geometry object with multiple Types, the derived values show a warning with 'undef' in the data tables:



The screenshot shows the 'Project Structure' window with a tree view of the project hierarchy. The 'Level 1 (17)' is expanded, showing a list of objects. The object 'M\_Fixed:0915 x 1830...' is highlighted in red, indicating a warning. The table below shows the data for this object, with 'undef' values for 'types:Price' and 'types:Supplier'.

Name	types:TypeName	types:Supplier	types:Price
<b>IFC Model [Cube_Architectural M...]</b>			
<b>0001 (1)</b>			
<b>Default (1)</b>			
<b>Building (2)</b>			
<b>Level 1 (17)</b>			
Basic Wall:Generic - 300...			
Basic Wall:Generic - 30...			
Basic Wall:Generic - 30...			
Curtain Wall:Exterior G...			
M_Fixed:0915 x 1830...	undef	undef	undef
M_Fixed:0915 x 1830...	Type B	ABC Windows	250,0000
M_Fixed:0915 x 1830...	Type B	ABC Windows	250,0000
M_Fixed:0915 x 1830...	Type B	ABC Windows	250,0000

## 4.12 Clash Detection

The Clash Detection module can be used to identify and analyze geometric conflicts in the model. It covers not only the model itself (self-intersection), but the collisions between discipline specific models to integrate an error-free coordination model.

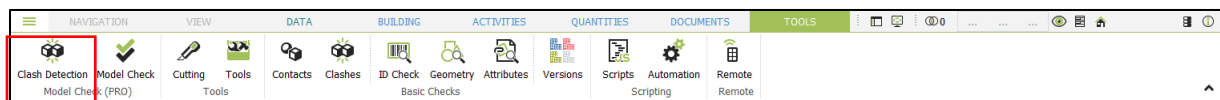
To organize the check, several clash runs can be defined that check different sets of objects with different options.

Furthermore, there is the possibility to define arbitrary functions in a post-processing with a script that sets property values for a check result based on user-defined criteria.

### Hint:

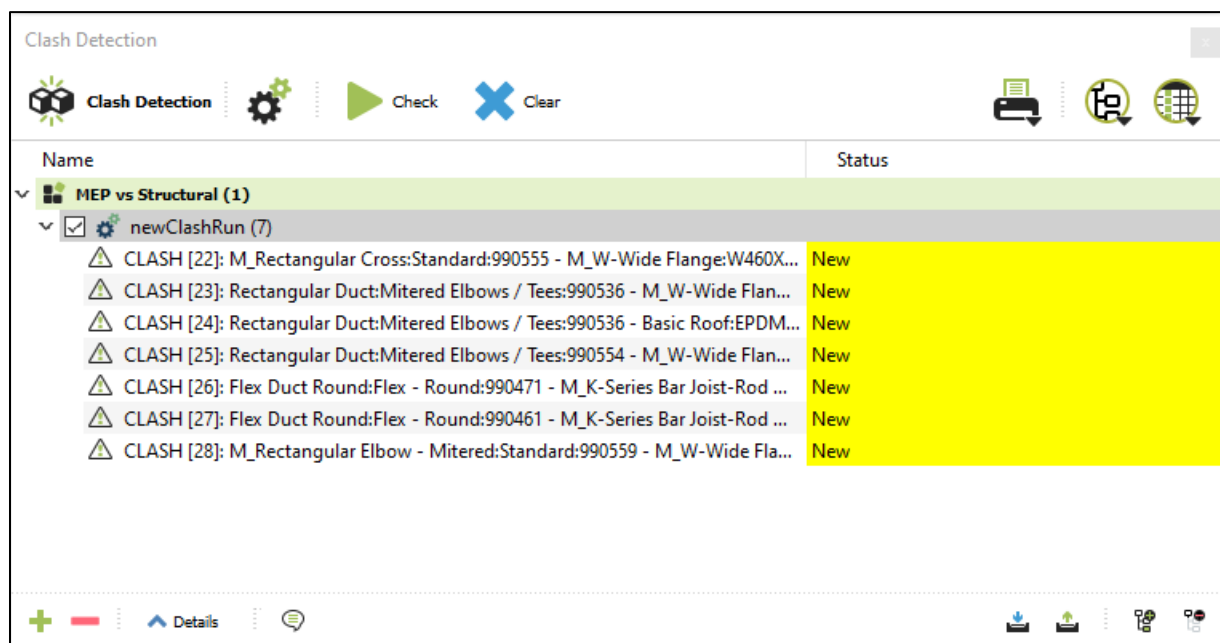
The module Clash Detection is only available for DESITE MD PRO.

Open the Clash Detection dialogue from the Tools Ribbon tab:



Similar to the other domains, Clash Detection is organized in a hierarchical structure containing:

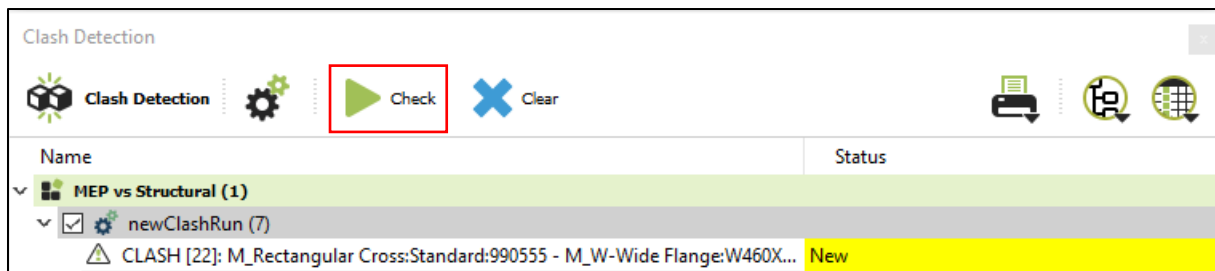
- Clash Models
  - Clash Runs
    - Clash Results



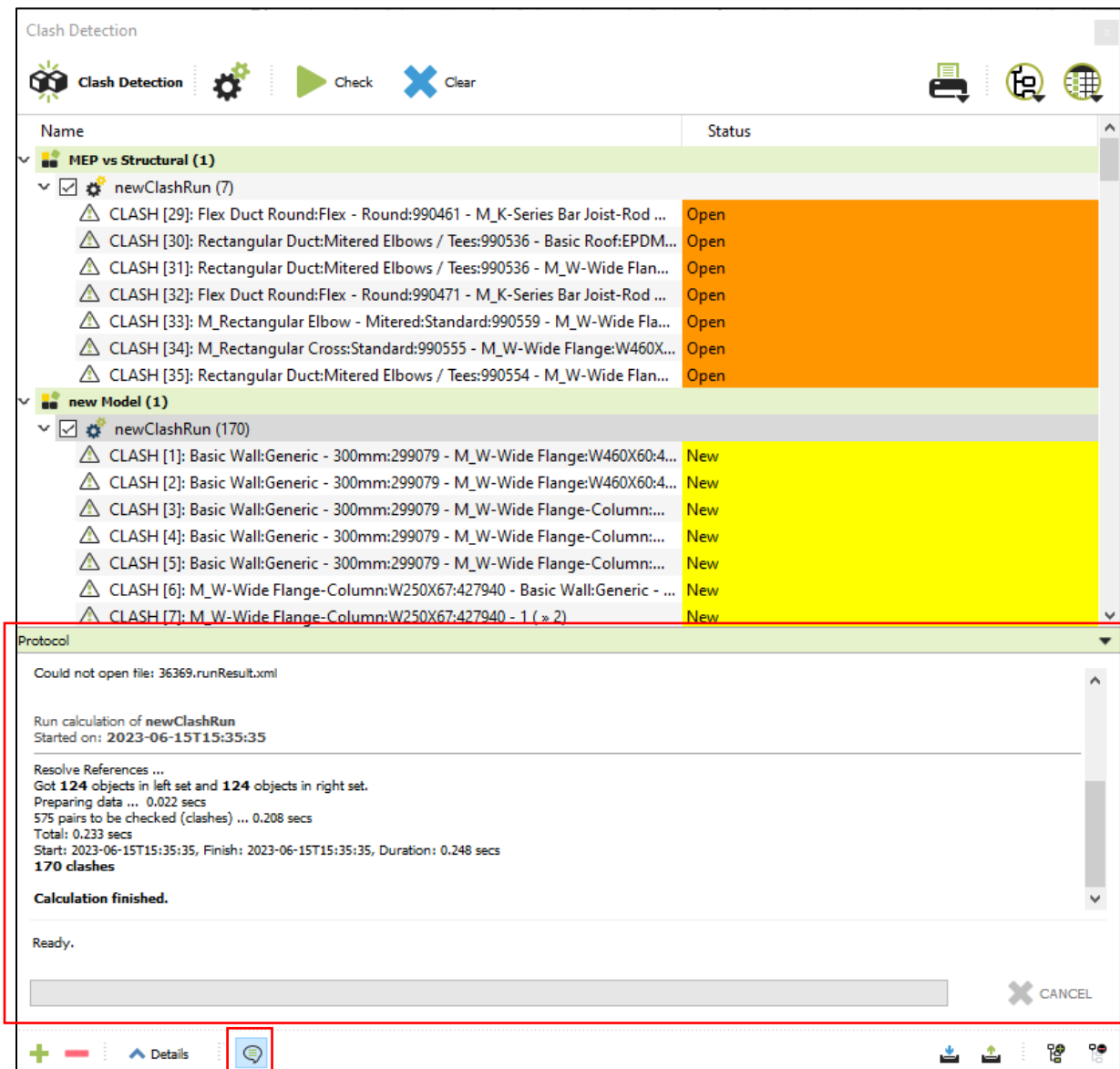
You can add new Clash Models and Clash Runs with the '+' button.

To perform a Clash Detection, click on 'Check':





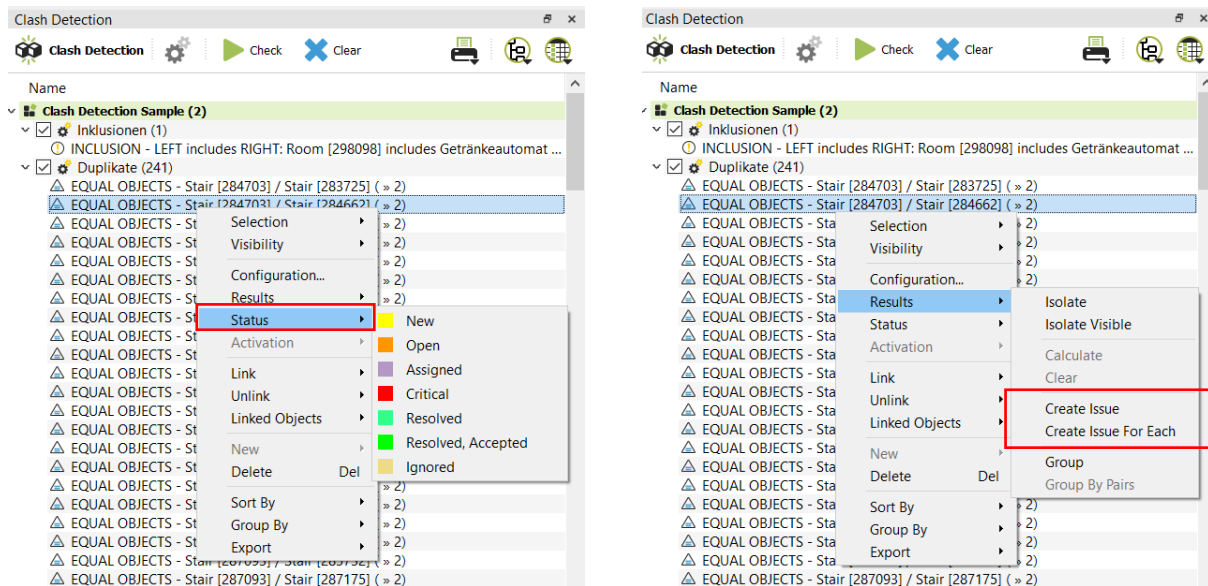
Now all currently active Clash Runs will be calculated based on their current settings. Progress and status of the calculation is visible in the 'Protocol' area:



The resulting clashes of the calculation are added as sub items in the tree hierarchy.

If the calculation is performed a second time on the Clash Run, the 'status' property shows an updated status. The status can also be set manually in the context menu:





You can also create 'Issues' from the results to manage them in the **Issues and Viewpoints** domain.

Within the options in the context menu, you can also start active Clash Runs, show objects of a clash result, group results etc.

### Dialog options



Opens **Configure Clash Runs** dialog



Starts calculation of all active Clash Runs (active Clash Runs are marked with a Checkmark in the tree view)



Clears all results from all Clash Runs



Print, PDF export



Add / remove Models and Clash Runs



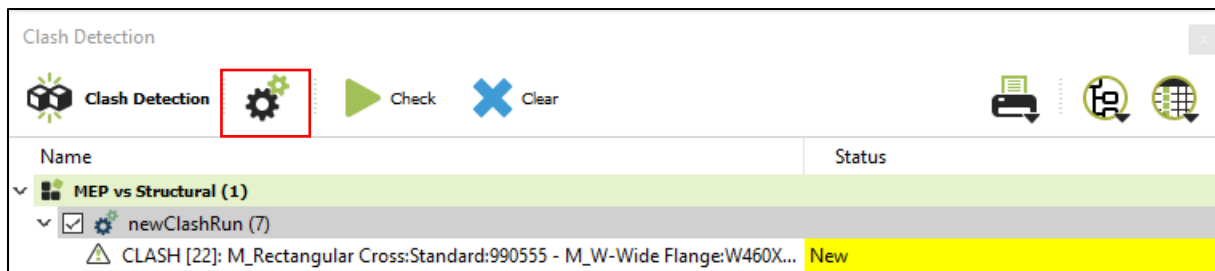
Show protocol



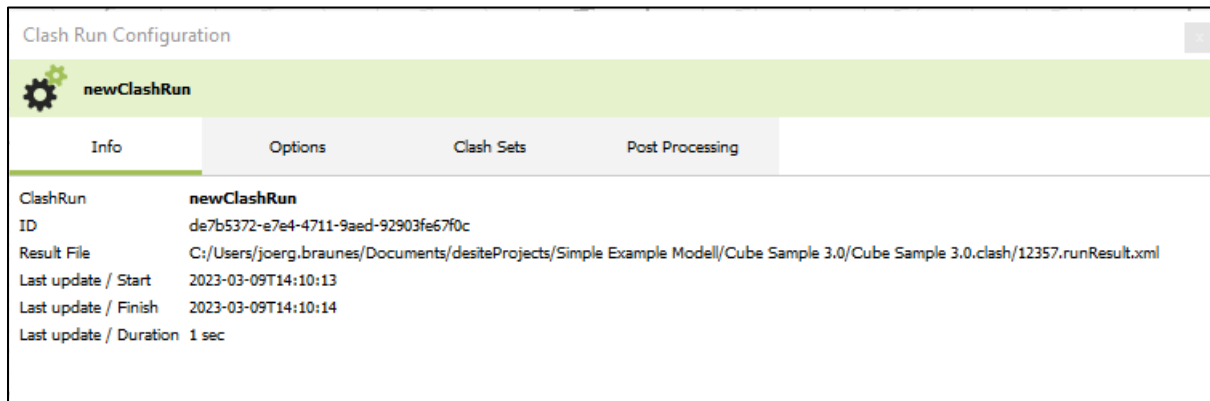
Import / export Clash Runs.

#### 4.12.1 Configure Clash Runs

To define the settings for the Clash Run, select the dedicated entry in the hierarchy and click on 'Configure Clash Run':



The Configuration dialogue opens:



### Info Tab

The information tab shows details about the clash run itself like start and finish date / time and last update.

### Options Tab

In this area the categories and parameters for the calculation can be defined (for a detailed description see [Clash Detection Options](#)). An active category is marked with a blue check mark.

### Clash Set Tab

In this area, the objects for which the test is to be performed are defined. A distinction is made between two groups, the right and left test sets. All objects of the right test set are tested against the objects of the left test set. For a detailed description how to define the sets, see [Defining objects for Clash Sets](#).

### Post Processing

Optionally, it is possible to write a post-processing in JavaScript for the collisions found. For a detailed description see [Post Processing of Clashes](#).

## 4.12.2 Clash Detection Options

The Clash Detection can be performed based on different categories. At least one category has to be active for a clash run. An active category is marked with a blue check mark.

## Category: Clashes

This category checks if objects clash with each other.

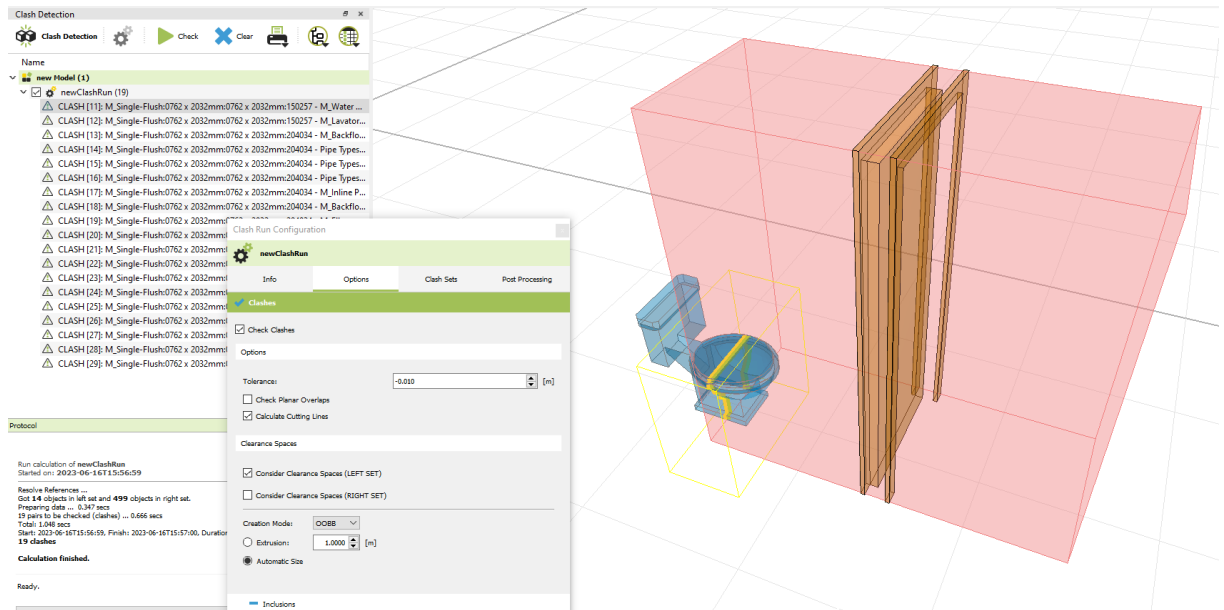
The screenshot shows the 'Clash Run Configuration' dialog box for 'MEP vs. Structural'. The 'Clashes' tab is selected. The 'Check Clashes' checkbox is checked. Under 'Options', 'Check Planar Overlaps' is unchecked and 'Calculate Cutting Lines' is checked. Under 'Clearance Spaces', both 'Consider Clearance Spaces (LEFT SET)' and 'Consider Clearance Spaces (RIGHT SET)' are unchecked. The 'Creation Mode' is set to 'OOBBxy'. The 'Extrusion' is set to '1.0000 [m]' and 'Automatic Size' is selected.

- Tolerance:** If the tolerance is negative, one object can penetrate another by the specified value without this being calculated as a collision. Here, collisions are omitted where the objects overlap only slightly.  
 If the tolerance is positive, a proximity is considered as a collision if the distance between the objects is less than the specified value, even if they do not touch.



- Calculate Cutting Lines:** This create a line object at the intersection of two objects
- Clearance Space:** With this option you can consider clashes also with the 'Clearance Space' of an object (e.g. the opening range of a door or window). In this case an optimized oriented bounding box will be calculated for the object (either for the object in the left set or right set) and extended to cover the assumed 'Clearance Space'.

**Example:** This Clash Run checks doors (defined in the left set) against the plumping (defined in the right set). A clash was found between the 'Clearance Space' (marked with a red box) of the door and a toilette:



## Category: Inclusions

This category checks if objects from one set are included in objects from the other set (the volume of the first object completely encloses the second object).

Clash Run Configuration

newClashRun

Info

Options

Clash Sets

Post Processing

Clashes

Inclusions

☒ Object in LEFT set includes object in RIGHT set

☐ Object in RIGHT set includes object in LEFT set

Options

Tolerance: 0.001 [m]

Contacts

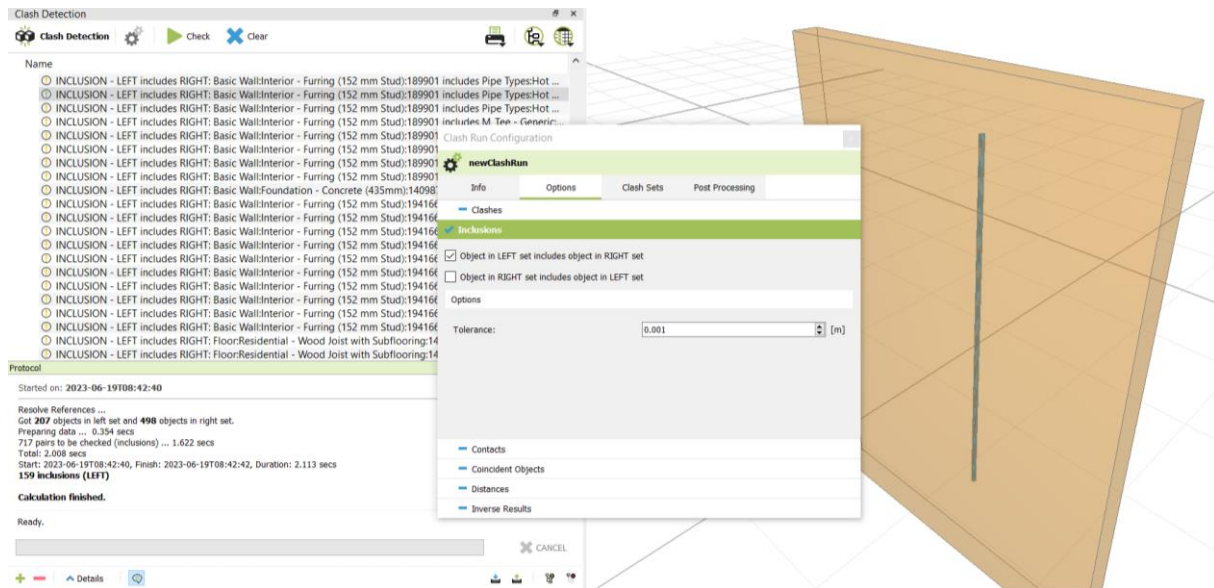
Coincident Objects

Distances

Inverse Results

You can specify in which direction the check should be performed: objects from the left set includes objects from the right set or vice versa.

**Example:** This Clash Run finds pipes that are included in a wall (without a wall recess):



## Tip:

If the check should be performed in both directions, it's recommended to define both check in two separate clash runs.

## Category: Contacts

This category calculates contact areas between objects in the two clash sets.

Clash Run Configuration

Contacts

Info

Options

Clash Sets

Post Processing

Clashes

Inclusions

Contacts

☒ Check Contacts

Options

☐ Create Contact Areas as 3D-Objects

Distance tolerance:

0.0050

[m]

Max. Angular Deviation of Normals:

5.00

[°]

☒ Check only Opposite Faces

☒ Check at top

☐ Check at bottom

☐ Check lateral

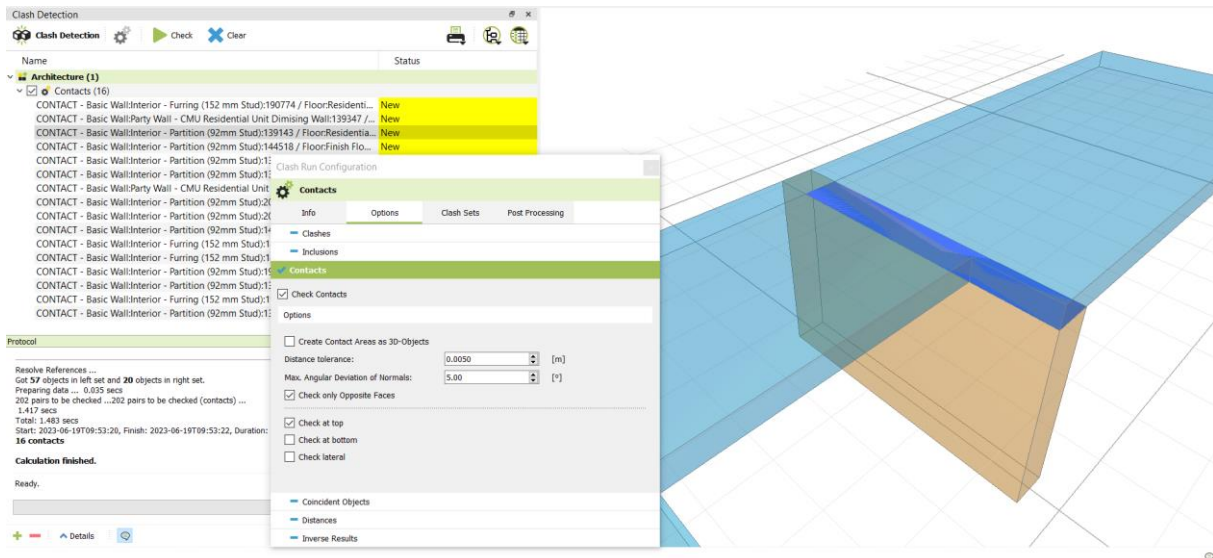
Coincident Objects

Distances

Inverse Results

- **Create Contact Areas as 3D-Objects:** In addition to the clash result, the contact area is also created as an individual 3d-object and added to the **Geometry Domain / Project Structure**.
- **Distance tolerance:** Values up to which distance of the surfaces a contact is detected.
- **Max. Angular Deviation of Normals:** Contacts are detected even if the surfaces are not exactly parallel. The value for this specifies the tolerance as the angle of the surface normal.
- **Check only opposite Faces:** Only if the surface normals of the objects in the right and left set point in the opposite direction, this is recognized as contact.
- **Check at top / bottom / lateral:** The check is performed only for the designated directions. Here, the left set determines what is tested for. For example, if you select a ceiling element in the left set and a wall underneath it in the right one, then 'Check at top' would not find this contact while 'Check at bottom' would find this contact.

**Example:** All walls are defined in the left set and all slabs are defined in the right set. The Check Run finds all contacts at the top of each wall with the slabs:



### Category: Coincident Objects

This test allows a check for similar objects, e.g. to find duplicates in your project.

### Clash Run Configuration

**Contacts**

Info

Options

Clash Sets

Post Processing

Clashes

Inclusions

Contacts

**Coincident Objects**

☒ Check Coincident Objects

Options

Tolerance of Bounding Box:
  [m]

---

☐ Compare Volume with Tolerance:
  [m<sup>3</sup>]

☐ Compare Surface Area with Tolerance:
  [m<sup>2</sup>]

Distances

Inverse Results

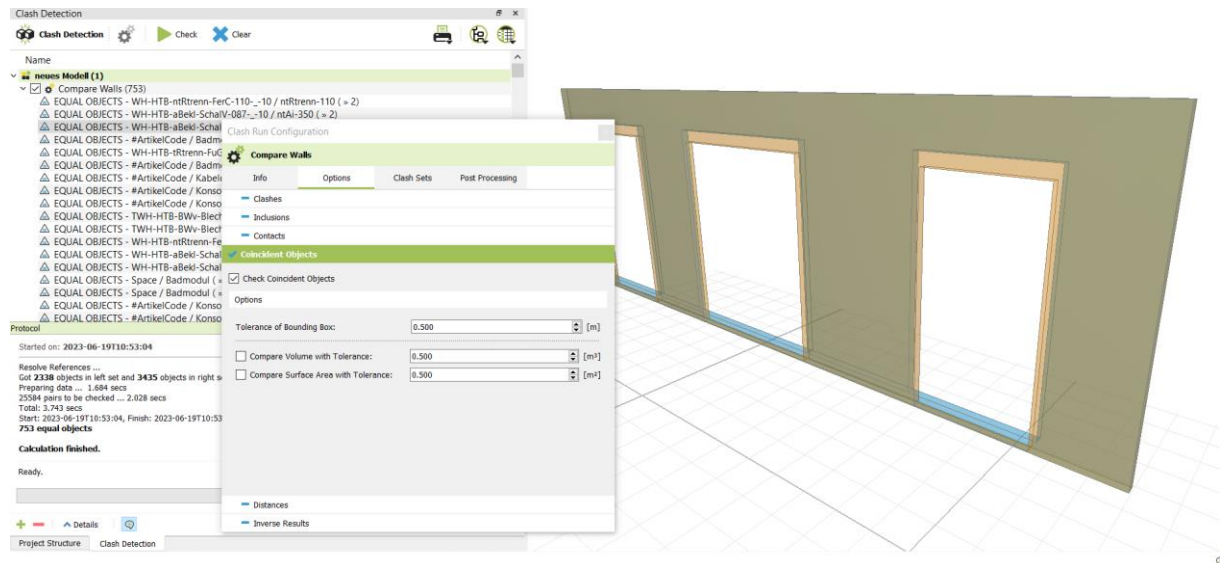
The check is performed in three steps:

- The bounding boxes of the object in the left set is compared with the object in the right set. You can specify a position tolerance for this check



- (optional) The volume of the object in the left set is compared with the volume in the right set. You can specify a volume tolerance for this check.
- (optional) The surface area of the object in the left set is compared with the surface area in the right set. You can specify an area tolerance for this check.

**Example:** This test compares the walls in two different partial models: the architectural model and the structural model. The check should show that the walls were modeled with the same position and size in both models. However, window openings of different sizes were allowed in detail. Therefore, the check was only applied to the bounding box.

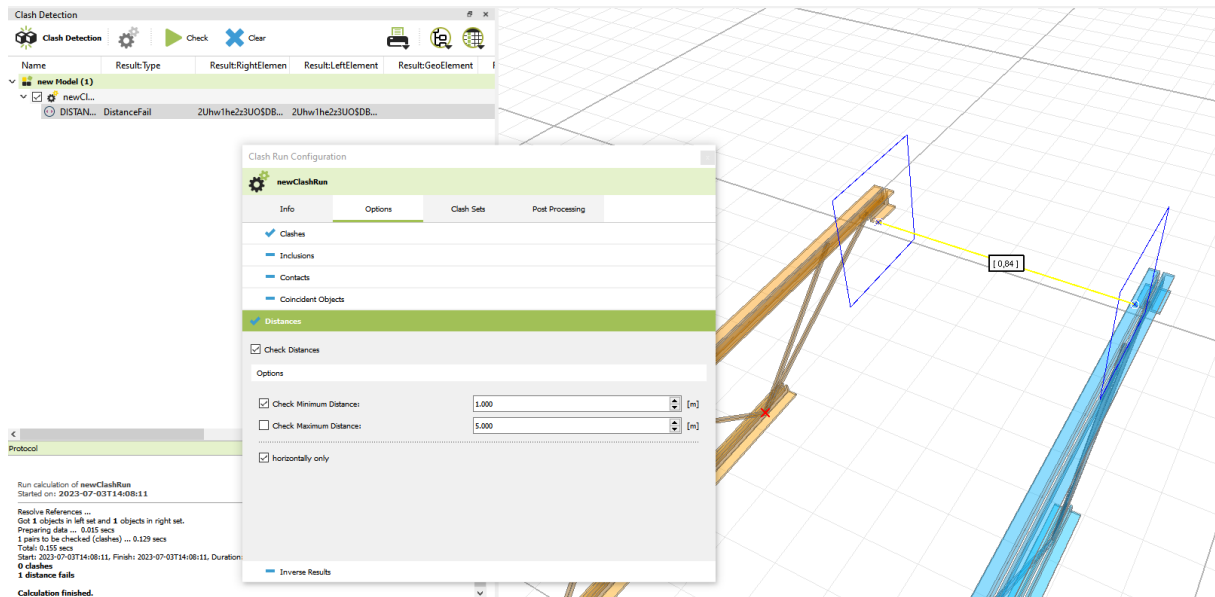


## Category: Distance

This test allows to check the minimal or maximal allowed distance between the objects in the left and right set.

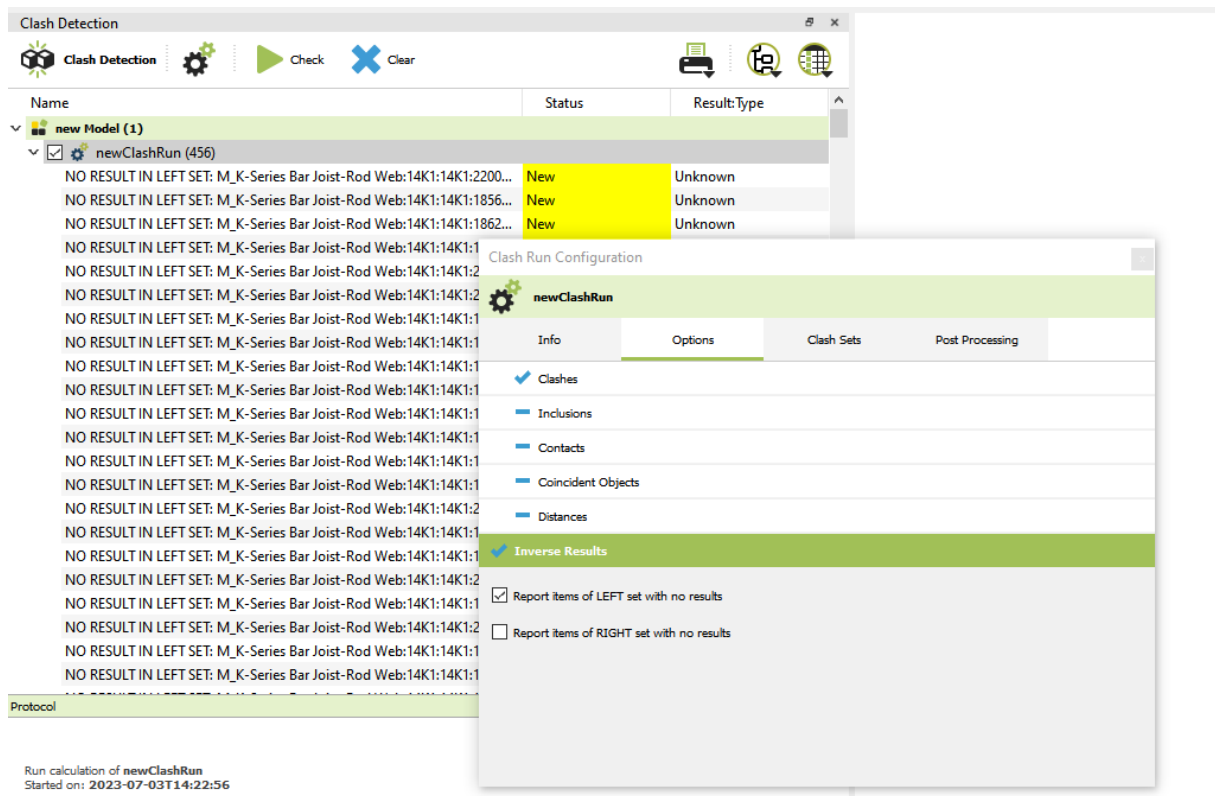
- **Check Minimum Distance:** The check is passed if the distances are above the set value.
- **Check Maximum Distance:** The check is passed if the distances are below the set value.
- **Horizontal only:** Only horizontal distances will be determined

**Example:** This test checks if the distance of the two beams are within the given minimal range (as an illustration, the distance is also shown with a **Measuring** item in this example):



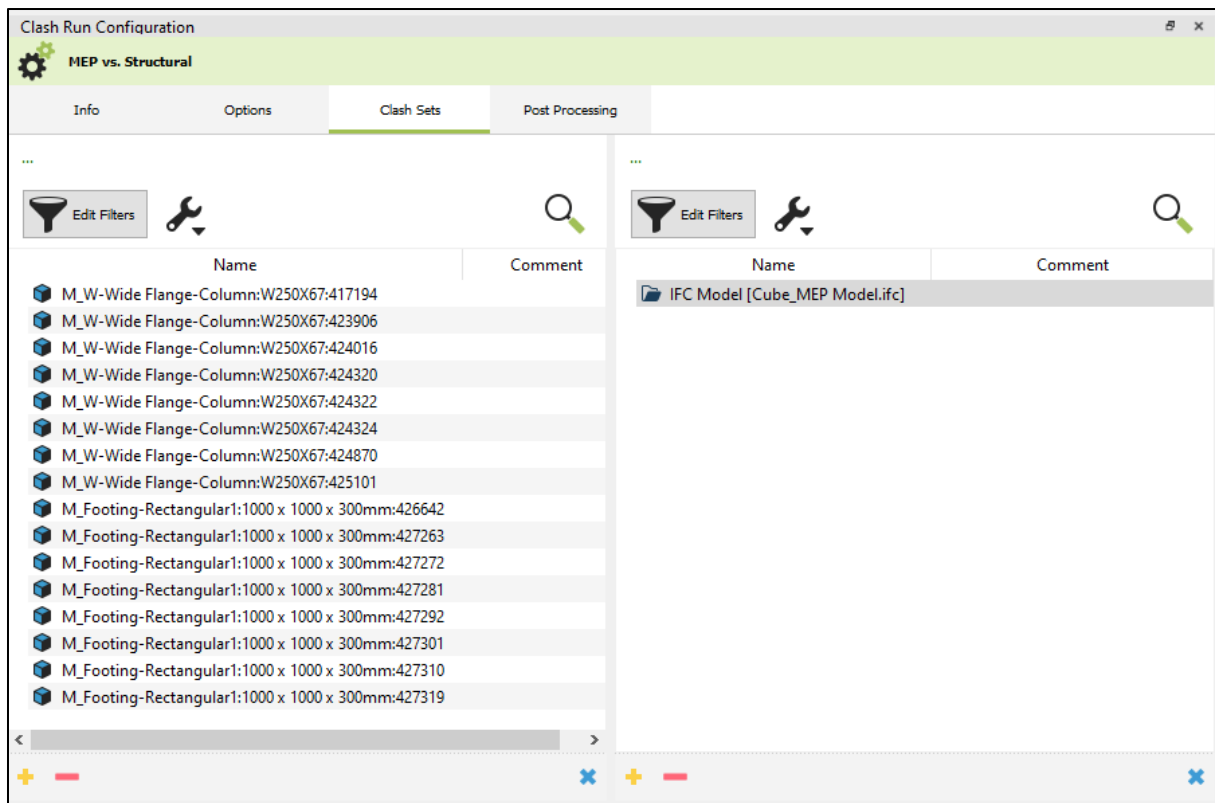
## Category: Inverse Results

While all other categories create a result in the Clash Run if the check fails, this category allows to create also results as a cross check if all other checks are passed. These 'inverse results' are named with 'NO RESULT IN LEFT / RIGHT SET':









### 4.12.3 Defining objects for Clash Sets

You can define two sets of objects that are checked against each other (left set and right set):



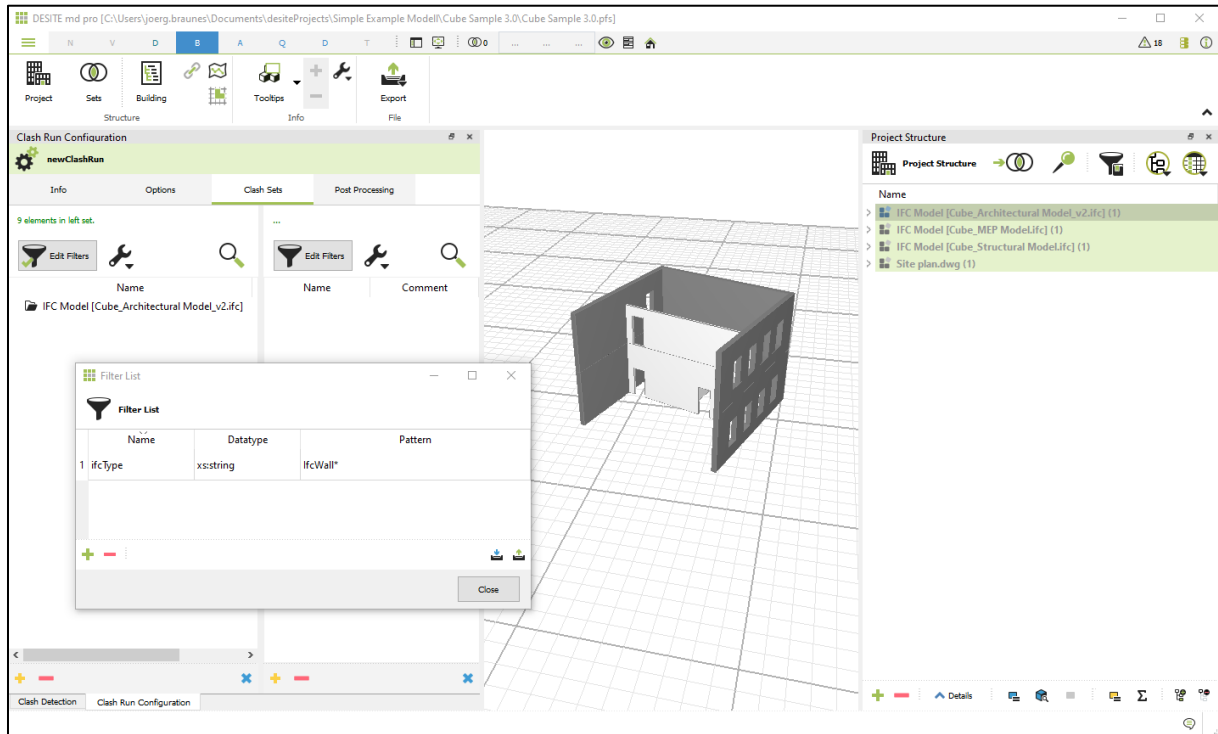
### Hint:

If no specific objects are added to the lists and no filter is defined, the calculation will be performed on all 3d objects in the project.

-  The desired objects can be added to the check set lists either by drag & drop from a tree view (for example, from **Selection Sets** or the **Geometry Domain / Project Structure** or **Building Structure**) or by manual selection from the 3D view and confirmation with the yellow '+' button at the bottom left of each list. If a folder is inserted, the test sets are updated before each test run.
-  Remove selected objects from the list
-  Remove all objects from the list
-  Isolates all objects from the respective list in the 3d view
-  With the filter option, you can further reduce the number of objects for the check. The filter is applied to all folders and objects that are currently part of the list. If the list does not contain any objects or folders, the filter is applied to all geometry objects in the project. If filters are defined, the icon shows a green check mark on the bottom left.
-  Additional options how specific object types are handled.

### Example for using filters:

For the left set the model 'IFC Model [Cube\_Architectural Model\_v2.ifc]' was added with drag & drop to the clash set. In order to reduce the number of objects for the calculation, an additional filter was applied, which filters for the property values that starts with 'IfcWall' in the property 'IfcType':



To check if the filters works correctly, click on the 'magnifier' icon in order to isolate all objects of the current list in the 3d view.

#### 4.12.4 Post Processing of Clashes

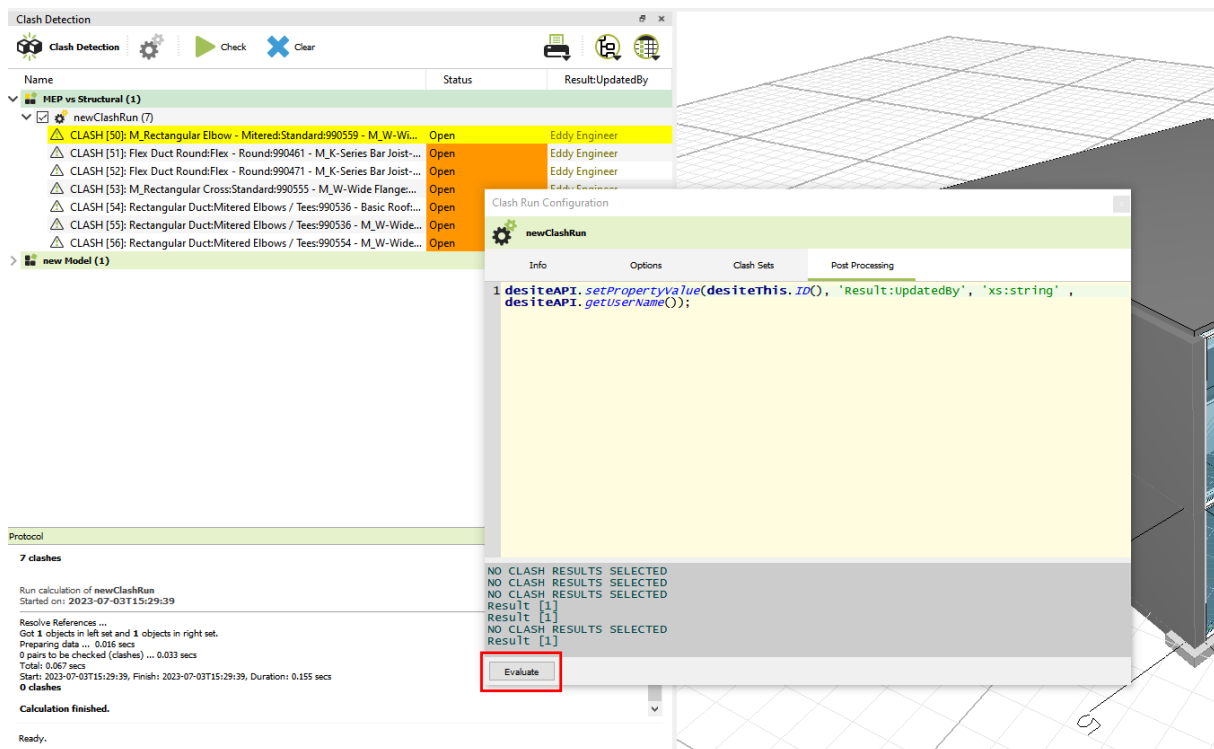
It is possible to define a post-processing step for a detected clash result in JavaScript.

**Hint:** For a detailed description of the DESITE script functions can be found in the chapter: **DESITE API**.

In post-processing, additional API objects are available for accessing the clashing objects and their properties.

- **desiteThis:** The object representing the collision or inclusion.
- **desiteLeft:** The object from the left check set.
- **desiteRight:** The object from the right check set.
- The *ID()* property contains the 'cpID' of the object.
- The *Name()* property contains the 'cpName' of the object.

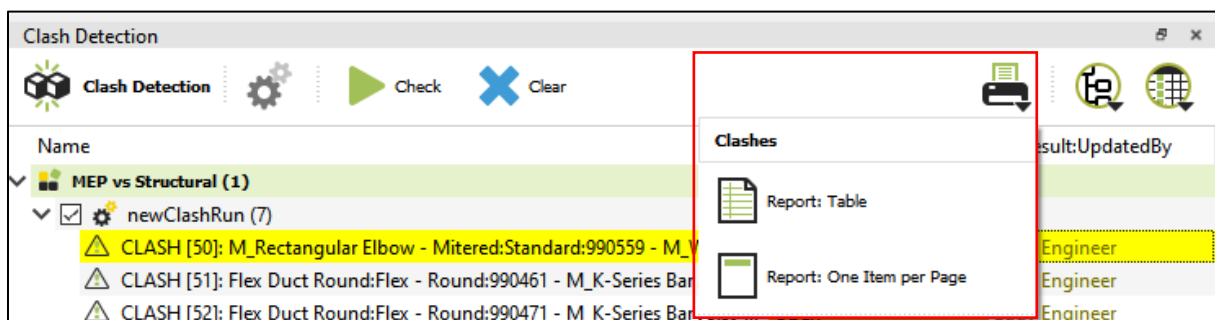
This example scripts writes the name of the current user (Windows user name) into the property 'Result:UpdatedBy' of the clash:



To test an input, an already found clash result must be selected (via context menu or with the short key SHIFT + ALT + #). Now it is possible to apply the script to the clash with the button 'Evaluate'.

#### 4.12.5 Printing clash reports

With the print options you can create a PDF report from the Clash Run.



The layout and content of the PDF report can be controlled by HTML templates. The templates have to be in the folder **[Project Name].templates/cd/** (if there are no templates in this directory, or the directory does not exist in the project, default templates are used).

There are templates for a print with one result per page:

- **cdPrintTemplateItemView.Item.html**
- **cdPrintTemplateItemView.Page.html**

And also templates with a list view:

- **cdPrintTemplateTableView.Item.html**

- **cdPrintTemplateTableView.Page.html**

A template consists of two files. The file *\*.Item.html* is a template for one clash result. The list of all clash results is embedded in the file *\*.page.html*. The item is specified with the variable `[[ITEMLIST]]`.

It's possible to use variables in the template. This can properties of a clash result and also a default variable to access the image of the clash: `[[IMAGE]]` (The variable will be replaced by the image, encoded as Base64).

The expression `[[PAGEBREAK]]` forces a page break. The variable `[[NR]]` is replaced by the current, consecutive number of the clash result.

Properties of the clashing 3D objects of a clash result can be accessed via the keywords LEFT and RIGHT. LEFT and RIGHT are prefixed in the variables to the attribute name and separated by ##. Examples:

- Name of the left object: `[[LEFT##cpName##xs:string]]`
- Name of the right object: `[[RIGHT##cpName##xs:string]]`

Attributes of the test run can be accessed via the keyword *CLASHRUN*. *CLASHRUN* are prefixed in the variables with the attribute name and separated by ##. Examples:

- Name of the clash run: `[[CLASHRUN##cpName##xs:string]]`

### Example for one result per page:

#### cdPrintTemplateItemView.**Page**.html

```
<h2>Print template for Clash Results: one item per page</h2>
<br>
[[ITEMLIST]]
```

#### cdPrintTemplateItemView.**Item**.html

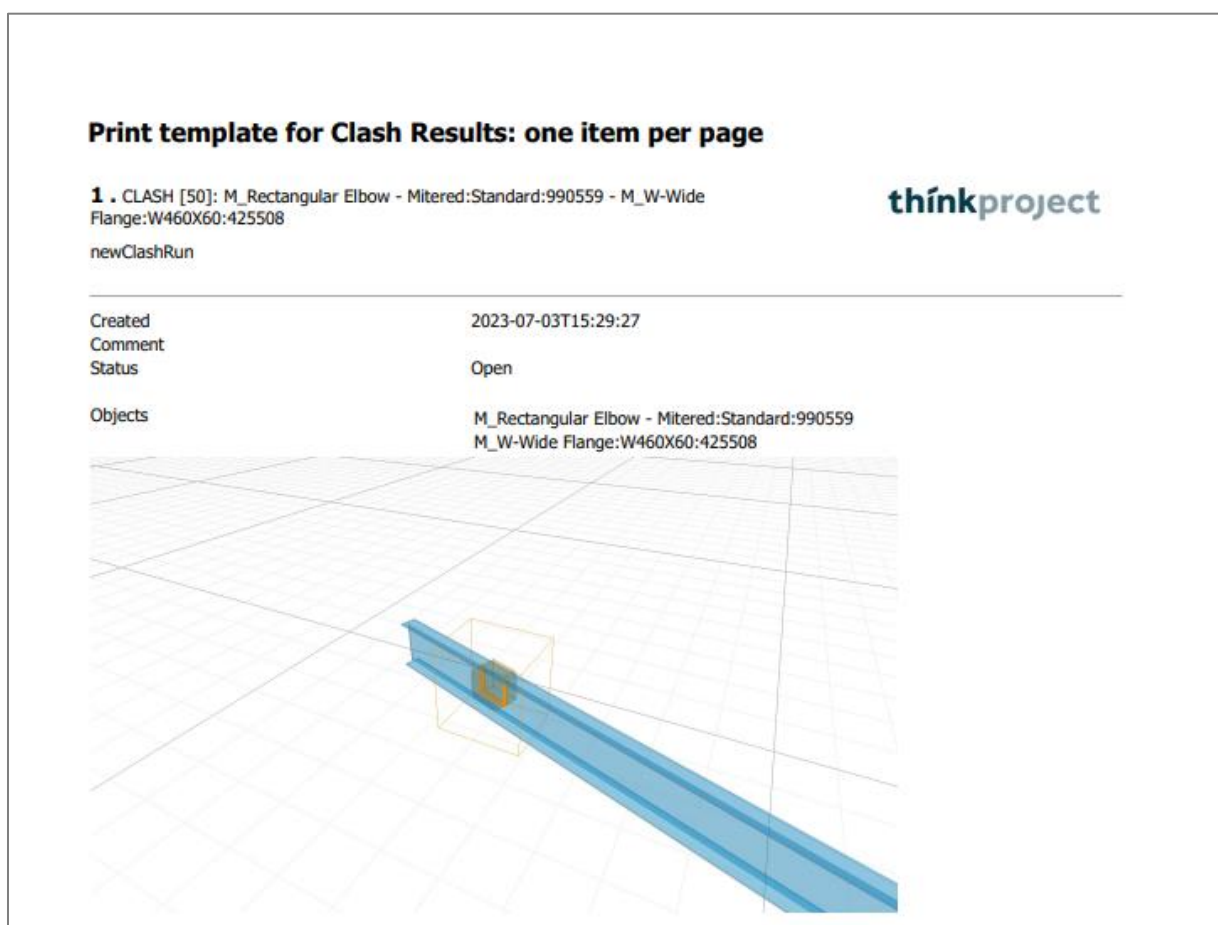
```
<table width='100%'>
<tr>
  <td colspan='2' valign='middle'><b><font size='+1'> [[NR]] </font></b> [[cpName##xs:string]]</td>
  <td align='right'><img height='50' src='tp_300x80.png'></img></td>
</tr>
<tr>
  <td colspan='1'> [[CLASHRUN##cpName##xs:string]] </td>
</tr>
<tr>
  <td colspan='3'><hr></td>
</tr>
<tr>
  <td>Created</td>
  <td colspan='2' valign='top'> [[Result:Created##xs:dateTime]] </td>
</tr>
<tr>
  <td>Comment</td>
  <td colspan='2' valign='top'> [[Comment##xs:string]] </td>
</tr>
<tr>
  <td>Status</td>
  <td colspan='2' valign='top'> [[Status##xs:string]] </td>
</tr>
<tr>
  <td>Objects</td>
  <td colspan='2'>
```

```

<table>
  <tr>
    <td valign='middle'> [[LEFT##cpName##xs:string]] </td>
  </tr>
  <tr>
    <td valign='middle'> [[RIGHT##cpName##xs:string]] </td>
  </tr>
</table>
</td>
</tr>
<tr>
  <td colspan='3'>
    <img width='600' src='data:image/png;base64, [[IMAGE]]' />
  </td>
</tr>
</table>
[[PAGEBREAK]]

```

Resulting PDF:



## 4.13 Model Check

With the Model Check module it is possible to validate the information and data quality of the imported models.

To organize the check, several check runs can be defined that verify different sets of objects with different options.

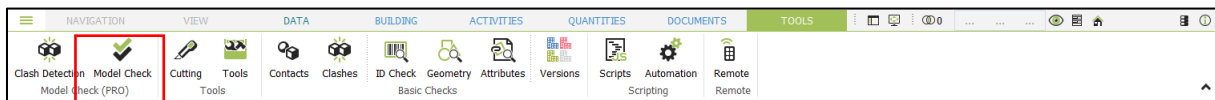


Furthermore, there is the possibility to define arbitrary functions in a post-processing with a script that sets property values for a check result based on user-defined criteria.

### Hint:

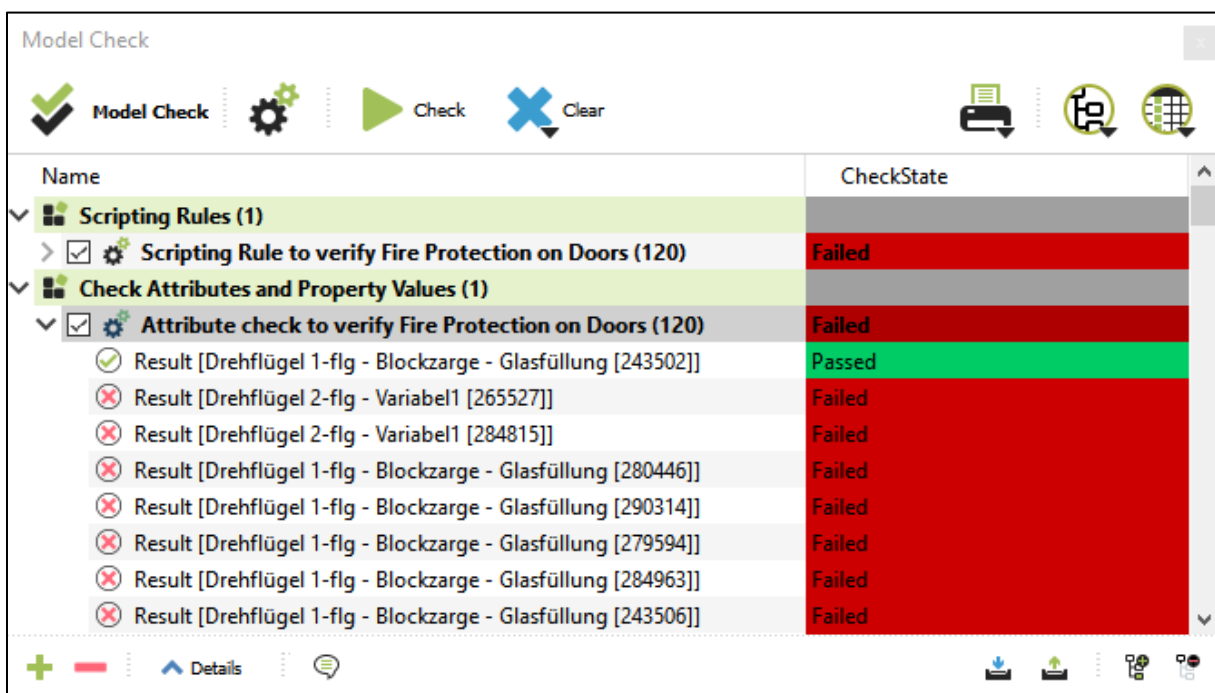
The module Model Check is only available for DESITE MD PRO

Open the Model Check dialogue from the Tools Ribbon tab:



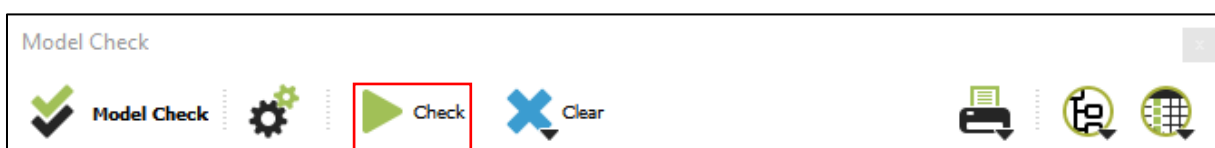
Similar to the other domains, Model Check is organized in a hierarchical structure containing:

- Check Models
  - Check Runs
    - Check Results



You can add new Check Models and Check Runs with the '+' button.

To perform a Model Check, click on 'Check':



Now all currently active Check Runs will be validated based on their current settings. Progress and status of the calculation is visible in the 'Protocol' area:



**Model Check**

Model Check    Check    Clear

Name	CheckState
Scripting Rules (1)	
Scripting Rule to verify Fire Protection on Doors (120)	Failed
Check Attributes and Property Values (1)	
Attribute check to verify Fire Protection on Doors (120)	Failed
Result [Drehflügel 1-flg - Blockzarge - Glasfüllung [243502]]	Passed
Result [Drehflügel 2-flg - Variabel1 [265527]]	Failed
Result [Drehflügel 2-flg - Variabel1 [284815]]	Failed

**Protocol**

Create Results with WARNINGS: yes  
 Create PASSED Results: yes  
 Create IGNORED Results: no  
 Create UNDEFINED Results: no  
 Started on: **2023-07-04T16:55:36**

Resolve References ...  
 No check of script rules.  
 Verifying 1 Rule(s) for 1858 Object(s) ...  
 No check of unique property values.  
 Calculation time: 1,464 sec  
**Overall check state: Failed**

Calculation finished.  
 Total time: 1,668 sec

CANCEL

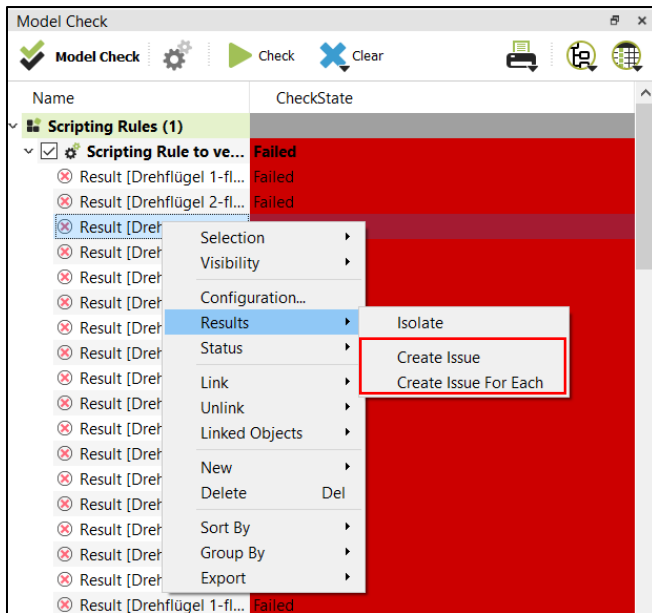
The results of the calculation are added as sub items in the tree hierarchy.

The 'CheckState' property is set automatically while the property 'Status' can also be set manually in the context menu. Newly generated results automatically have the status 'new'. If you run the check a second time, the status changes to 'open'. You can manually set the status to: 'assigned' / 'critical' / 'fixed' / 'ignored'.

The overall status of a Check Run is set to the authoritative status of the contained results. The following valence applies:

- Failed → Warning → Passed → Ignored → Undefined.

If all tests are passed, the overall status is also passed. If only one check has failed, the overall status is failed.



You can also create 'Issues' from the results to manage them in the Issues and Viewpoints domain.

Within the options in the context menu, you can also start active Check Runs, show objects of a result, group results etc.

### Dialog options



Opens **Configure Check Runs** dialog



Starts calculation of all active Check Runs (active Check Runs are marked with a Checkmark in the tree view)



Removes results from all Check Runs, based on the different options in the drop down menu.



Print, PDF export



Add / remove Models and Check Runs



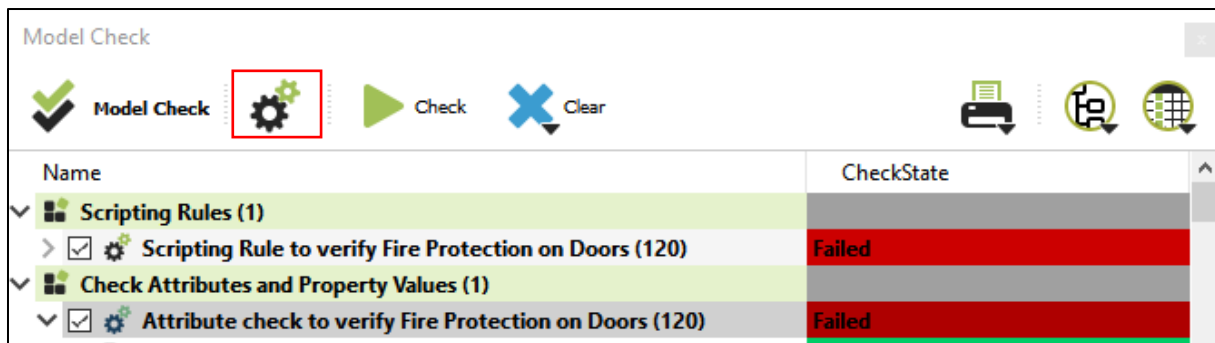
Show protocol



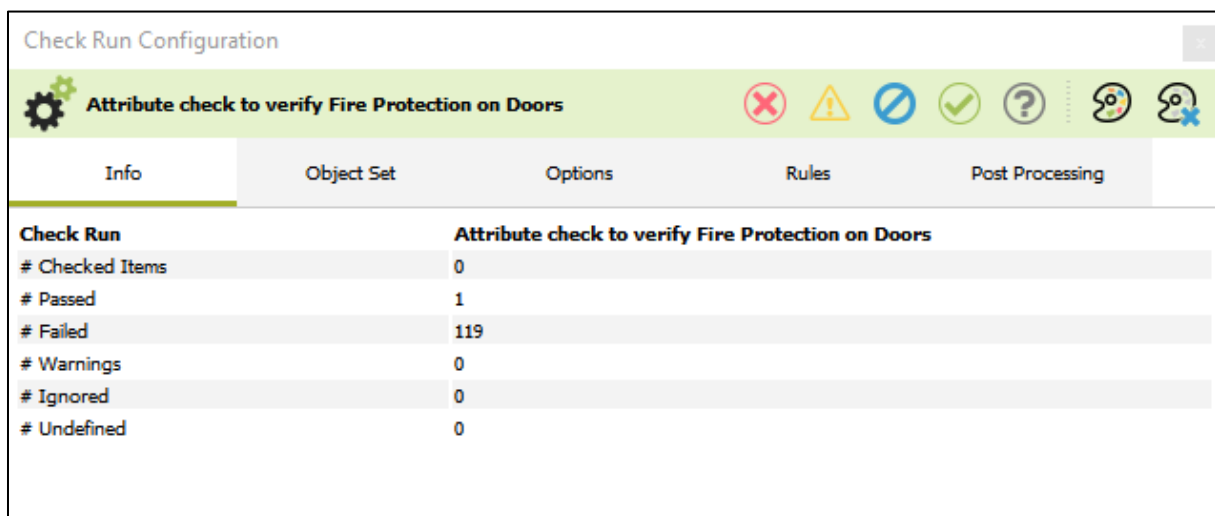
Import / export Check Runs.

#### 4.13.1 Configure Check Runs

To define the settings for the Check Run, select the dedicated entry in the hierarchy and click on 'Configure Check Run':



The Configuration dialogue opens:



### Info Tab

The information tab shows details about the Check Run itself like the number of passed or failed items.

### Object Set Tab

In this area, the objects for which the test is to be performed are defined. For a detailed description how to define the sets, see [Defining objects for Check Sets](#).

### Options Tab

In this area the options for the Check Run are defined. For a detailed description about the options, see [Defining objects for Check Sets](#).

### Rules Tab

In this area the rules for the Check Run are defined. For a detailed description, see [Defining Check Run Rules](#).

### Post Processing

Optionally, it is possible to write a post-processing in JavaScript for the check results. For a detailed description see [idPost Processing of Check Run results](#).

### 4.13.2 Check Run Options

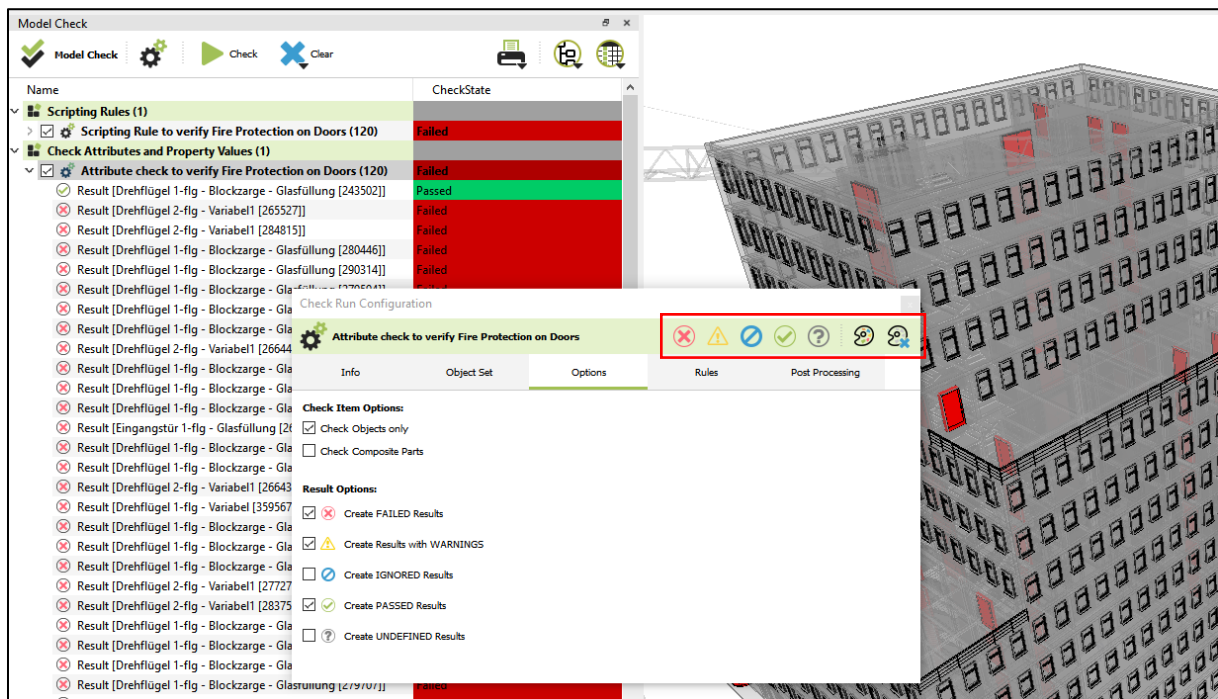
#### Check Item Options

- **Check Objects only:** only geometric objects will be checked, including Composites. Containers as part of the hierarchical project structure will be skipped.
- **Check Composite Parts:** The individual parts of a Composite will be checked, instead of the Composite object (the parent object in the hierarchy) itself.

#### Result Options:

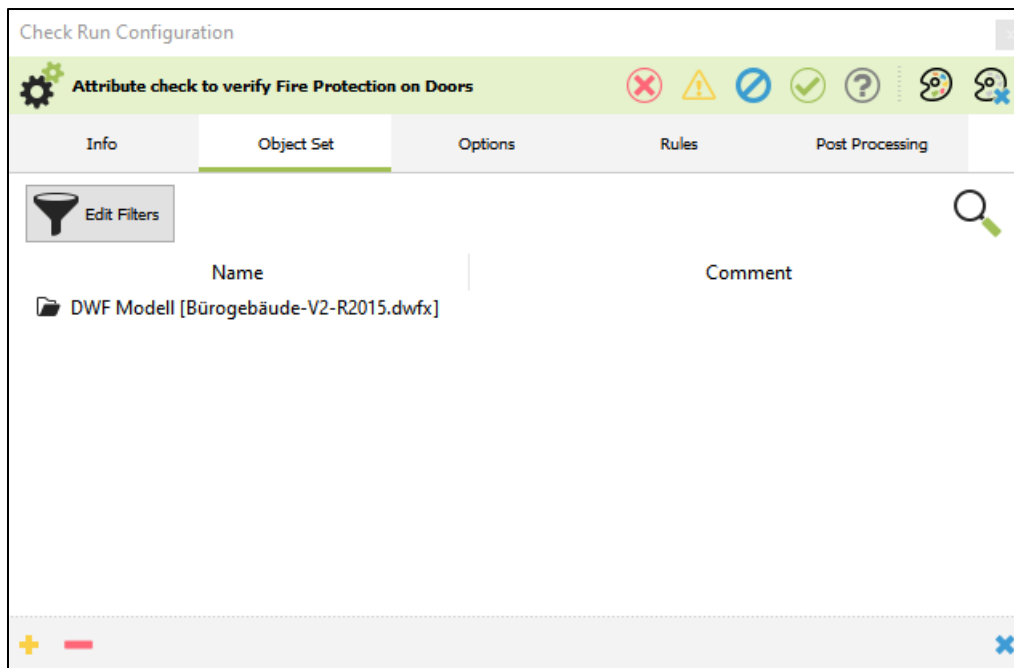
Define which type of results of the check should be created as items in the tree view.

With the icons at the top of the dialogue, you can also filter the objects in the 3d view based on the results and also color code the related objects:








### 4.13.3 Defining objects for Check Sets

You can specify which objects should be included in the check either by added them individually to the Object Set or by defining filters.



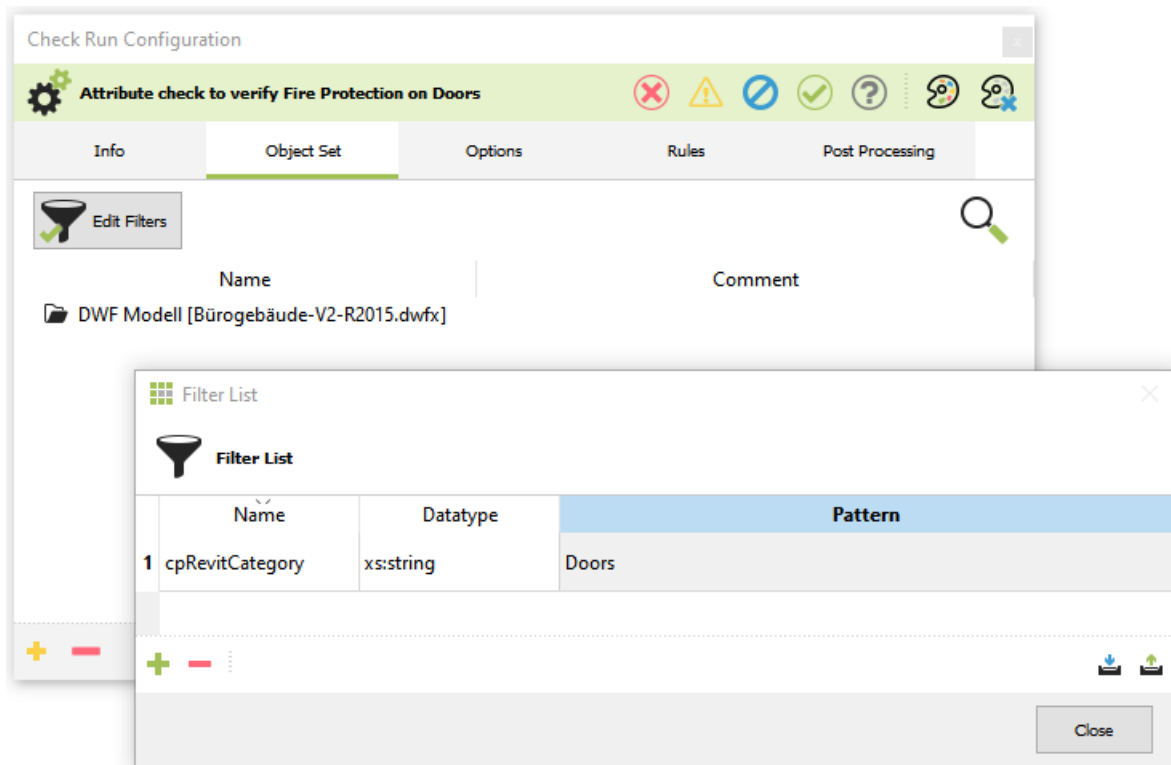
### Hint:

If no specific objects are added to the list and no filter is defined, the check will be performed on all 3d objects in the project.

-  The desired objects can be added to the check set list either by drag & drop from a tree view (for example, from **Selection Sets** or the **Geometry Domain / Project Structure** or **Building Structure**) or by manual selection from the 3D view and confirmation with the yellow '+' button at the bottom left. If a folder is inserted, the check sets are updated before each test run.
-  Remove selected objects from the list
-  Remove all objects from the list
-  Isolates all objects from the respective list in the 3d view
-  With the filter option, you can further reduce the number of objects for the check. The filter is applied to all folders and objects that are currently part of the list. If the list does not contain any objects or folders, the filter is applied to all geometry objects in the project.  
If filters are defined, the icon shows a green check mark on the bottom left.

### Example for using filters:

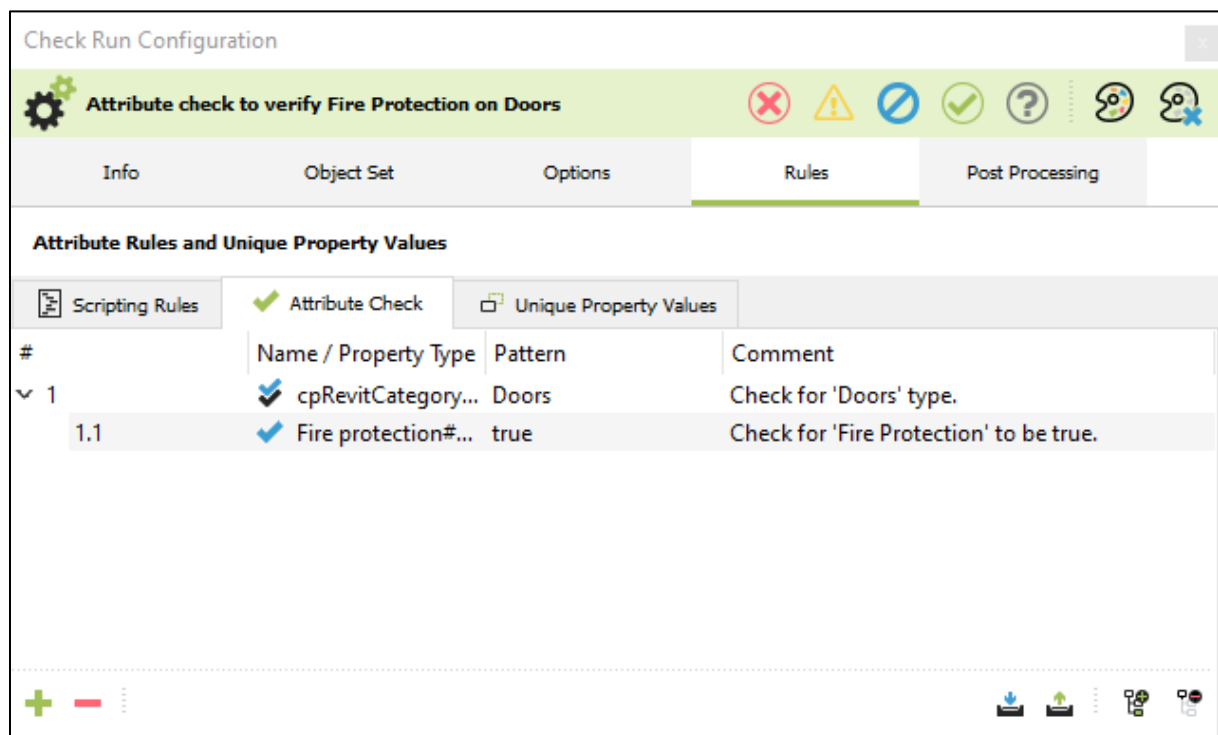
For the object set the model 'DWF Modell [Bürogebäude-V2-R2015.dwf]' was added with drag & drop to the object set. In order to reduce the number of objects for the validation, an additional filter was applied, which filters for the property 'cpRevitCategory' with value 'Doors':



To check if the filters work correctly, click on the 'magnifier' icon in order to isolate all objects of the current list in the 3d view.

#### 4.13.4 Defining Check Run Rules

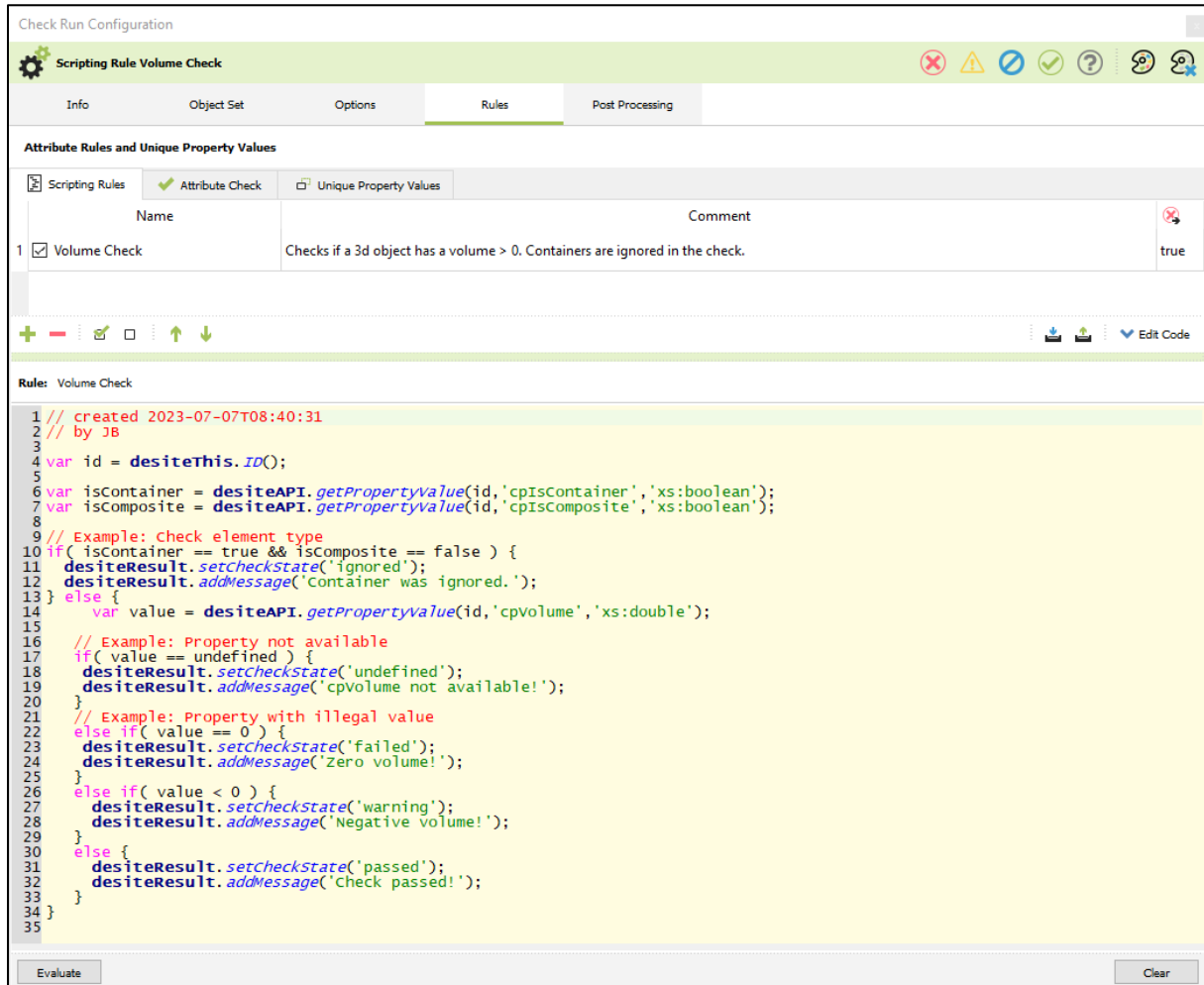
Rules for the Model Check can be defined in three different ways.



## Scripting Rules

Scripting Rules using the JavaScript code to define a check.

**Hint:** For a detailed description of the DESITE script functions can be found in the chapter: **DESITE API**.



With the '+' button you can add a new Scripting Rule. With the checkmark, the rules are set to active / non-active. The option 'true / false' on the right determines whether in case of an error (status 'Failed') the rule evaluation should be continued.

Open the script area with 'Edit Code'. A new script rule contains already an example code.

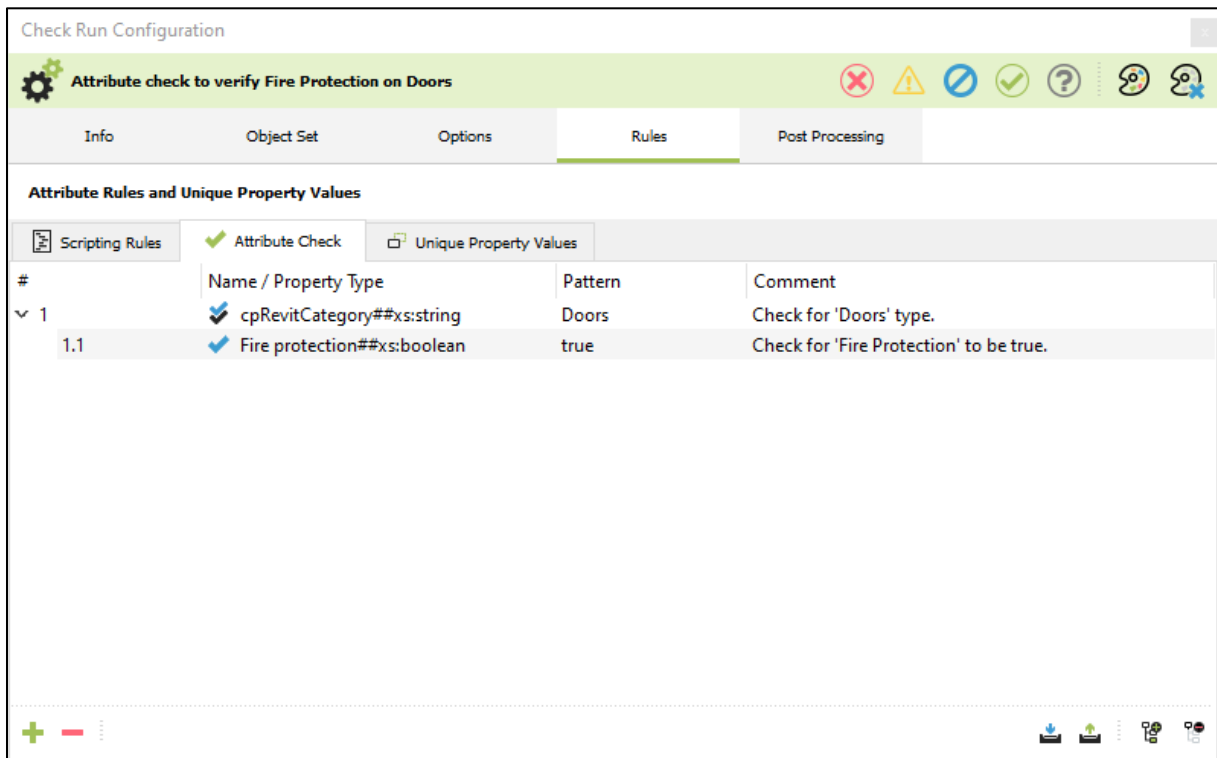
'Evaluate' allows you to test the script for selected objects.

The API Object 'desiteResult' allows you to set the status of the check and also to send a message:

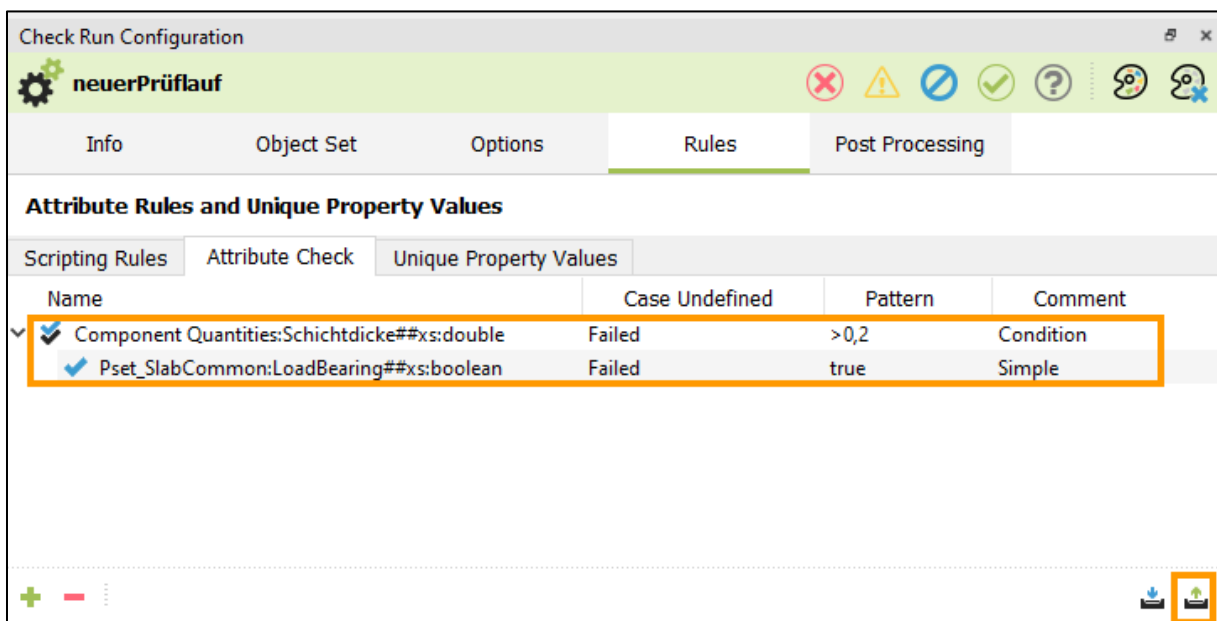
- **desiteResult.setCheckState(<status>):** sets the property 'CheckState' with *status = {undefined, ignored, passed, warning, failed}*
- **desiteResult.addMessage('Text'):** sets (or adds) a text for the property 'Result:Message'

## Attribute Check

Attribute check allows you to define checks without coding.



There are two types of rules in the Attribute Check: Conditional Rule and Simple Rule. The Conditional Rule is used when a prerequisite should be added to a Simple Rule. E.g. you want to check if all ceilings which are thicker as 20 cm have property 'IsLoadBearing = true', you can set a conditional rule like this:



It is also possible to enter various values in the pattern, so that the attribute value should only equals one of them. You can either type them in easily with space between, such as



'F3o F6o F9o'. Or if the value includes space itself, add quotes to each value and divide them with space, such as "John Doe" "Ann Ade" "Thomas Tip".

With using the export/import button you can get/put in your own rules in editable version, such as .csv or .xml file. The CSV file looks like this:

	A	B	C	D	E	F	G	H	I	J	K
1	#	Exported 2021-11-15T16:22:34 by DESITE MD					<div>Export Information</div> <div>Header for rules same as in Desite</div> <div>Concrete information about rules</div>				
2	H	Property Name	Data Type	Rule	Comment						
3	C	Component Quantities:Schichtdicke	xs:double	>0,2							
4	R	Pset_SlabCommon:LoadBearing	xs:boolean	true							
5											
6	H: Headers										
7	C: Conditional Rule										
8	R: Simple Rule										
9											

## Unique Property Values

These rules allows you to check for the unique occurrence of property values. This example checks, if the window numbers (property: [[GHH:FensterNummer##xs:string]]) only appear exactly one time per object (in the Object Set a filter was set to get all objects with IfcType 'IfcWindow'):

Check Run Configuration

Fensternummern

Info
Object Set
Options
Rules
Post Processing

Attribute Rules and Unique Property Values

Scripting Rules
Attribute Check
Unique Property Values

Pattern

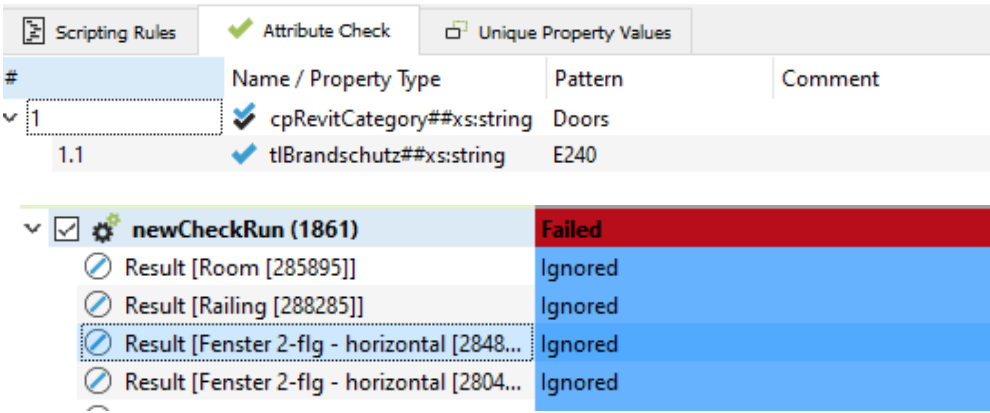
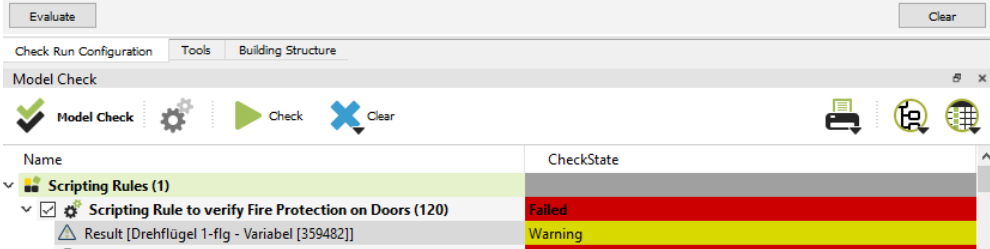
Comment

1
[[GHH:FensterNummer##xs:string]]
Check unique Window Number

+
-

### 4.13.5 Meaning of Check Run Status

Status	Meaning												
Passed	When all requirements in test rules are fulfilled.												
Failed	<div>If an attribute does not exist or its value does not fulfill the operation.</div> <table><thead><tr><th>No.</th><th>Rule</th><th>Check State</th><th>Text</th></tr></thead><tbody><tr><td>E.g. 1</td><td>DoorMaterial##xs:string</td><td>Failed</td><td>Property [DoorMaterial##xs:string] does not exist</td></tr><tr><td>or 6</td><td>DoorMaterial##xs:string</td><td>Failed</td><td>[DoorMaterial]: Wood not Glas</td></tr></tbody></table>	No.	Rule	Check State	Text	E.g. 1	DoorMaterial##xs:string	Failed	Property [DoorMaterial##xs:string] does not exist	or 6	DoorMaterial##xs:string	Failed	[DoorMaterial]: Wood not Glas
No.	Rule	Check State	Text										
E.g. 1	DoorMaterial##xs:string	Failed	Property [DoorMaterial##xs:string] does not exist										
or 6	DoorMaterial##xs:string	Failed	[DoorMaterial]: Wood not Glas										

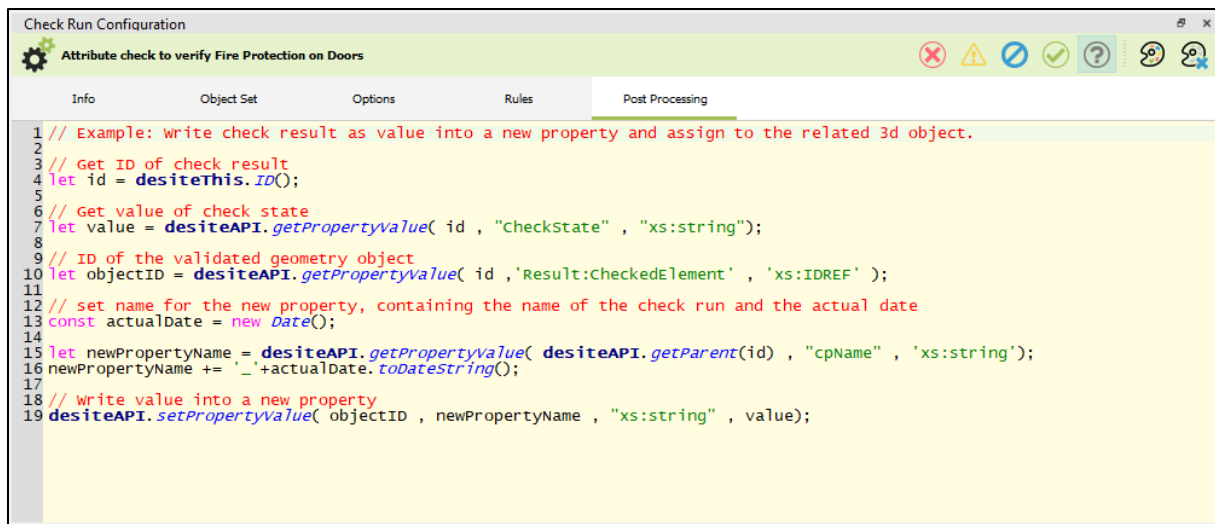
Ignored	<p>If the objects in the selection do not fulfill the specified rule. For instance, if "Doors" are defined in the conditional rule, all other objects are ignored, such as rooms, windows, columns...</p>  <p>The screenshot shows the 'Scripting Rules' window with the 'Attribute Check' tab selected. A rule is defined with the name 'cpRevitCategory##xs:string' and pattern 'Doors'. Below the rule, a 'newCheckRun (1861)' is shown with a 'Failed' status. The results table lists four items: 'Result [Room [285895]]', 'Result [Railing [288285]]', 'Result [Fenster 2-flg - horizontal [2848...]]', and 'Result [Fenster 2-flg - horizontal [2804...]]', all with a status of 'Ignored'.</p>
Undefined	<p>Test runs are not executed at all. Or the test results have been reset.</p>
Warning	<p>If you write the check rule in the script, you can define the "warning" yourself. Here is an example: If an object does not have the "Fire protection" attribute, it receives the status "Warning".</p> <p>Rule: GetDoorsWithoutValue</p> <pre> 1 var id = desiteThis.ID(); 2 var fireProtection = desiteAPI.getPropertyValue(id, 'Fire protection', 'xs:boolean'); 3 if (fireProtection != undefined) 4 { 5     if (fireProtection) 6     { 7         desiteResult.setCheckState('passed'); 8         desiteResult.addMessage('check passed!'); 9     } 10    else 11    { 12        desiteResult.setCheckState('failed'); 13    } 14 } 15 else 16 { 17     desiteResult.setCheckState('warning'); 18 } </pre>  <p>The screenshot shows the 'Check Run Configuration' window with the 'Tools' tab selected. A 'Model Check' is shown with a 'Warning' status. The results table lists one item: 'Result [Drehflügel 1-flg - Variabel [359482]]' with a status of 'Warning'.</p>

#### 4.13.6 idPost Processing of Check Run results

It is possible to define a post-processing step for a check result in JavaScript. The variable 'desiteThis.ID()' represents the ID of the current result.

**Hint:** For a detailed description of the DESITE script functions can be found in the chapter: **DESITE API**.

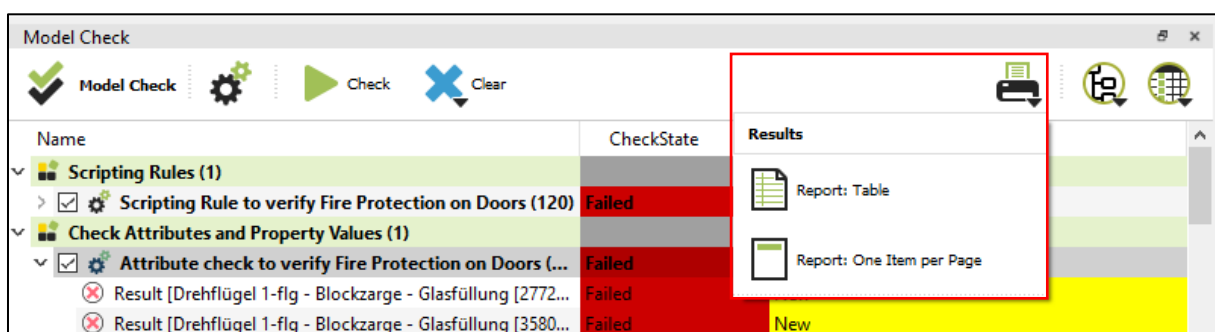
The following script writes a check state into a new property of the related 3d object. The new property name is created from the name of the check run and the actual date:



To test an input, an check result must be selected (via context menu or with the short key SHIFT + ALT + #). Now it is possible to apply the script to the check result with the button 'Evaluate'.

#### 4.13.7 Printing check result reports

With the print options you can create a PDF report from the Check Run.



The layout and content of the PDF report can be controlled by HTML templates. The templates have to be in the folder **[Project Name].templates/qa/** (if there are no templates in this directory, or the directory does not exist in the project, default templates are used).

There are templates for a print with one result per page:

- **PrintTemplateItemView.Item.html**
- **PrintTemplateItemView.Page.html**

And also templates with a list view:

- **PrintTemplateTableView.Item.html**
- **PrintTemplateTableView.Page.html**

A template consists of two files. The file *\*.Item.html* is a template for one check result. The list of all check results is embedded in the file *\*.page.html*. The item is specified with the variable `[[ITEMLIST]]`.

It's possible to use variables in the template. This can properties of a check result and also a default variable to access the image of the result: `[[IMAGE]]` (The variable will be replaced by the image, encoded as Base64).

The expression `[[PAGEBREAK]]` forces a page break. The variable `[[NR]]` is replaced by the current, consecutive number of the clash result.

Properties of the checked 3D object can be accessed via the keywords `ELEMENT`:  
Example:

- Name of the object of a check result: `[[ELEMENT##cpName##xs:string]]`

Attributes of the Check Run can be accessed via the keyword `CHECKRUN`. Example:

- Name of the Check Run: `[[CHECKRUN##cpName##xs:string]]`

### Example for one result per page:

PrintTemplateTableView.**Page**.html

```
<table width='100%'>
<tr>
  <td colspan='2' valign='middle'><b><font size='+3'> Model Check </font></b>
  <td align='right'><img height='50' src='tp_300x80.png'></img></td>
</tr>
<tr>
  <td>Project Name</td>
  <td colspan='2'>[[GLOBALPROJECT##ProjectName##xs:string]]</td>
</tr>
<tr>
  <td>Project Number</td>
  <td colspan='2'>[[GLOBALPROJECT##NumberCode##xs:string]]</td>
</tr>
<tr>
  <td>Short Description</td>
  <td colspan='2'>[[GLOBALPROJECT##ShortDescription##xs:string]]</td>
</tr>
</table>

<table width='100%' cellpadding='2' cellspacing='4'>
[[ITEMLIST]]
</table>
```

PrintTemplateTableView.**Item**.html

```
<tr>
  <td colspan='3' valign='middle'><b><font size='+1'> [[NR]]. </font></b> [[cpName##xs:string]]</td>
</tr>
<tr>
  <td rowspan='5'><img height='150' src='data:image/png;base64,[[IMAGE]]'></td>
  <td>Status</td>
  <td>[[Status##xs:string]]</td>
</tr>
<tr>
  <td>Check State</td>
  <td>[[CheckState##xs:string]]</td>
</tr>
<tr>
  <td>Created on</td>
  <td>[[Result:Created##xs:dateTime]]</td>
</tr>
<tr>
  <td>Updated on</td>
  <td>[[Result:Updated##xs:dateTime]]</td>
</tr>
<tr>
  <td>Comment</td>
  <td>[[Comment##xs:string]]</td>
```

</tr>

Resulting PDF:


### Model Check

Project Name  
Project Number  
Short Description

Waldstrasse  
A001  
Waldstrasse Sample

thinkproject


1. Result [Drehflügel 1-flg - Blockzarge - Glasfüllung [277283]]



Status  
Check State  
Created on  
Updated on  
Comment

New  
Failed  
2023-07-10T08:08:42


2. Result [Drehflügel 1-flg - Blockzarge - Glasfüllung [358087]]



Status  
Check State  
Created on  
Updated on  
Comment

New  
Failed  
2023-07-10T08:08:42

3. Result [Drehflügel 2-flg - Variabel1 [279590]]

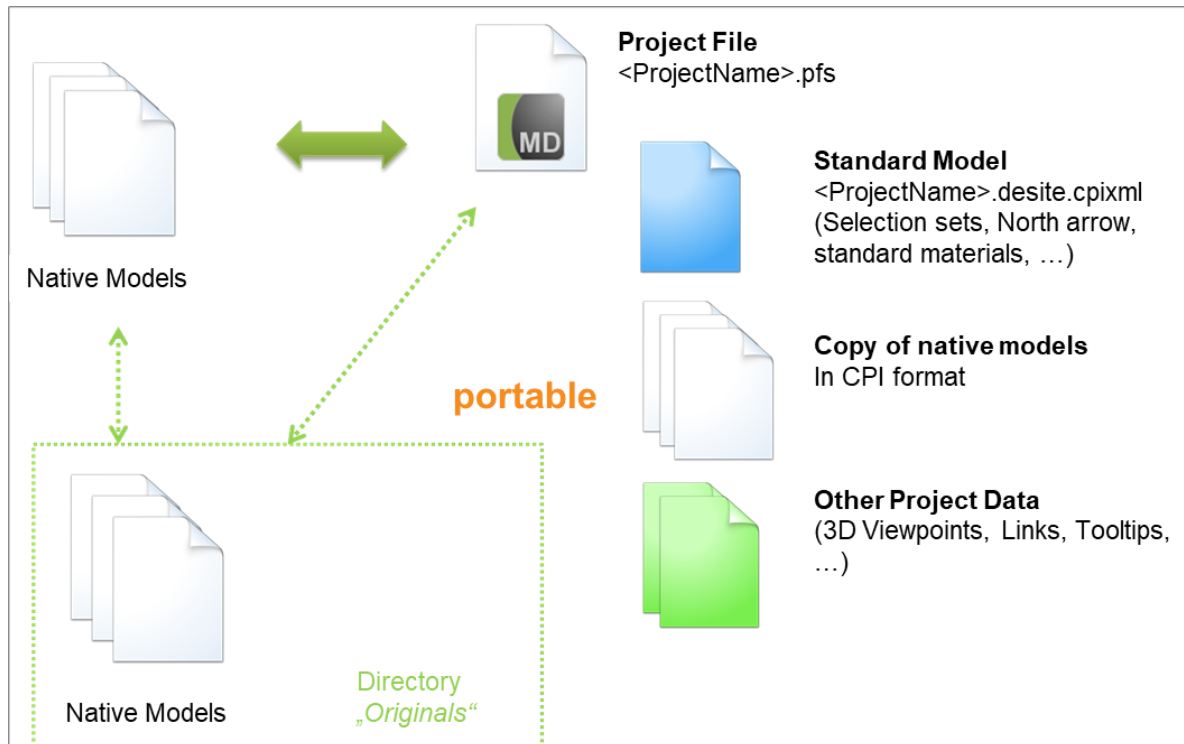


Status  
Check State  
Created on  
Updated on  
Comment

New  
Failed  
2023-07-10T08:08:42

## 5. Project File Structure

Current project can be saved in native pfs format in the application menu. A new folder should be created for this purpose since individual files are created directly in the specified directory.



Project can also be opened by executing the project file.

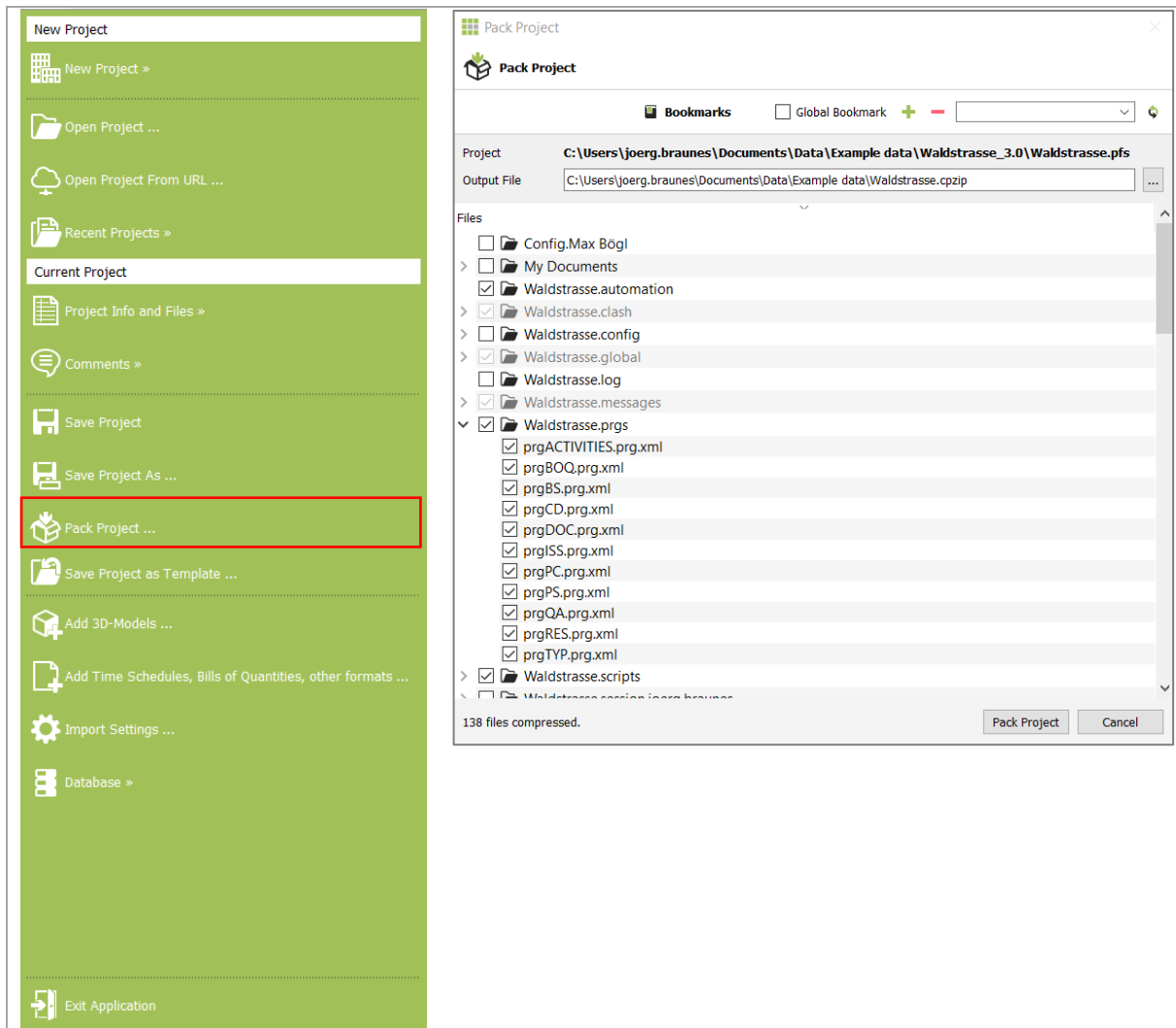
Contents of a DESITE Project

Folder	
desite.modules	Standard DESITE folder for additional Web Forms
<ProjectName>.automation	Automated scripts (Menu: Tools/Automation)
<ProjectName>.clash	Results of the clash runs
<ProjectName>.cloud	
<ProjectName>.global	Global bookmarks (available to everyone project-wide)
<ProjectName>.messages	Session events
<ProjectName>.prgs	Property scripts (Menu: Data/Scripts)
<ProjectName>.scripts	Global scripts (Menu: Tools/Scripts)
<ProjectName>.session.dh	Personal bookmarks

<ProjectName>.templates	Templates, e.g. QuickInfo.html
<ProjectName>.textures	
<ProjectName>.viewpoints	Thumbnails of 3D viewpoints
<b>Files</b>	
<ProjectName>.pfs	Project file, contains a directory of all project files
<ProjectName>.db	Standard project database (SQLite)
<ProjectName>.lock	Lock file, shows that a user has opened the project
<ProjectName>.prjACT.xml	Domain projects for the data domains (in this order): Activities, Bill of Quantities, Building Structure, Clash Detection, Documents, Geometry, Process Components, Properties, Quality Assurance, Resources, Types
<ProjectName>.prjBOQ.xml	
<ProjectName>.prjBS.xml	
<ProjectName>.prjCD.xml	
<ProjectName>.prjDOC.xml	
<ProjectName>.prjGEO.xml	
<ProjectName>.prjPC.xml	
<ProjectName>.prjProp.xml	
<ProjectName>.prjQA.xml	Contains a Repository, Links (Relations) and also Linking Rules
<ProjectName>.prjRES.xml	
<ProjectName>.prjTYP.xml	
10816.modBOQ.xml	Various domain models For performance reasons, geometric models are compressed and saved in native .cp2 format by default
18822.modTYP.xml	
25178.modBOQ.xml	
27827.modDOC.xml	
28728.modTYP.xml	
49614.modPC.xml	
53384.modGEO.cp2	
63650.modBS.xml	
<ImportedGeometryFile1>.desite.cp2	Cache of an imported geometry file
<ImportedGeometryFile2>.desite.cp2	Cache of a second imported geometry file
<ImportedSchedule1>.tsmap	Cache of an imported time schedule (mapping/link activity to geometric object and time schedule itself)
<ImportedSchedule1>.tsxml	
<ProjectName>.cs.xml	Color scheme s
<ProjectName>.cut.xml	Cutting positions for split objects into parts
<ProjectName>.desite.cpixmap	North arrow, standard colors
<ProjectName>.lck.xml	Locked objects
<ProjectName>.location.xml	Geocoordinates of the project
<ProjectName>.objMap.xml	Links between objects
DbInputForm.html	Front page for Forms
<ProjectName>.ptype.xml	Property types
<ProjectName>.qa.xml	Model checking rules (is no longer used)
<ProjectName>.setGEO.xml	Selection sets
<ProjectName>.tip.xml	List of objects for permanent tooltips
<ProjectName>.tv.xml	Top views
<ProjectName>.vis.xml	Visualisation styles for 4D simulation

<ProjectName>.vp.xml	Saved 3D Viewpoints
<ProjectName>.wms.xml	Wire mode scheme

If you would like to forward a project to another stakeholder, you can pack it in the native .cpzip format. When you open a .cpzip file, all the files contained within the project are unpacked and placed in the same directory.

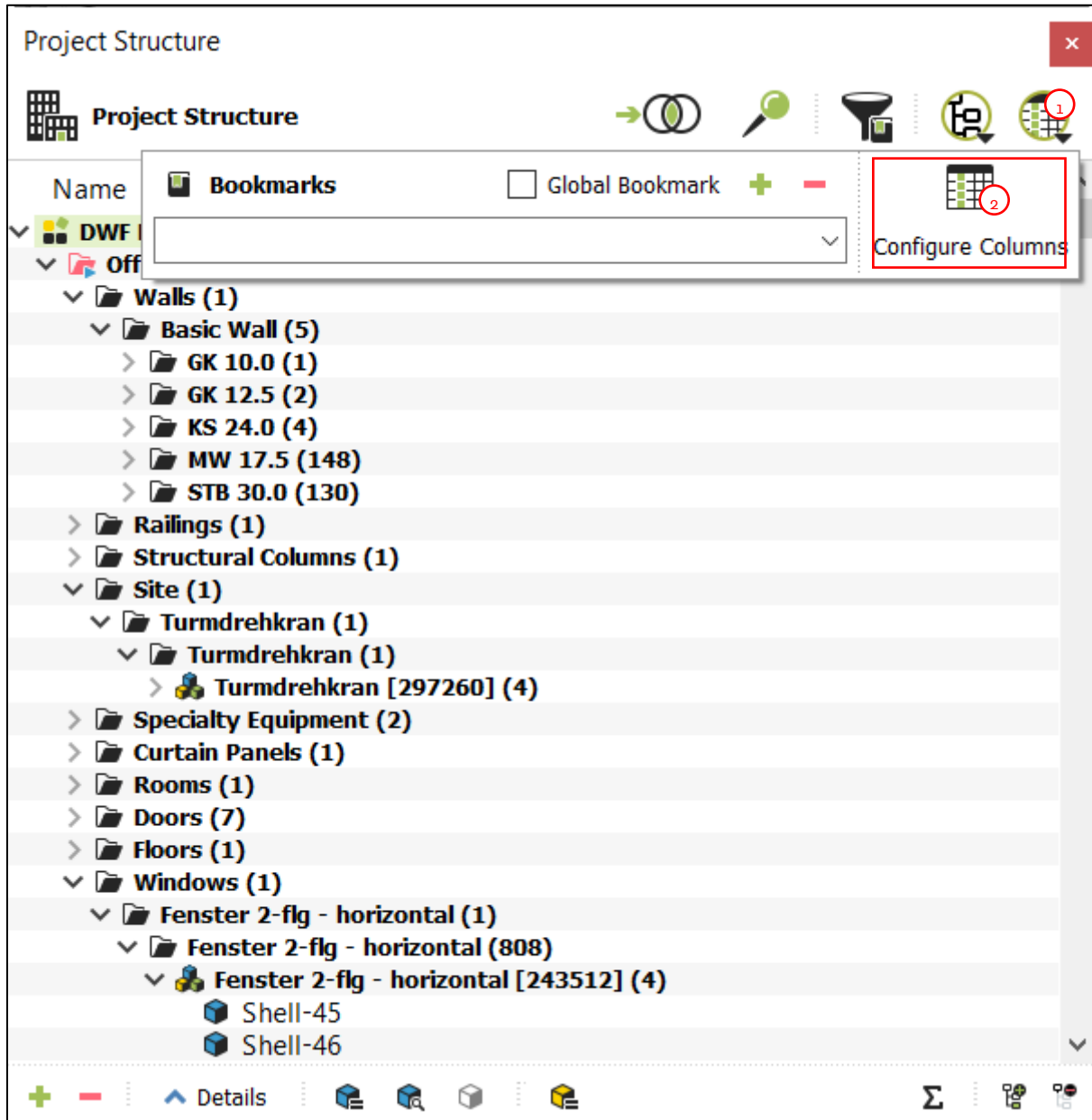




## 6. Working with Properties

### 6.1 Properties in the domain tree structure

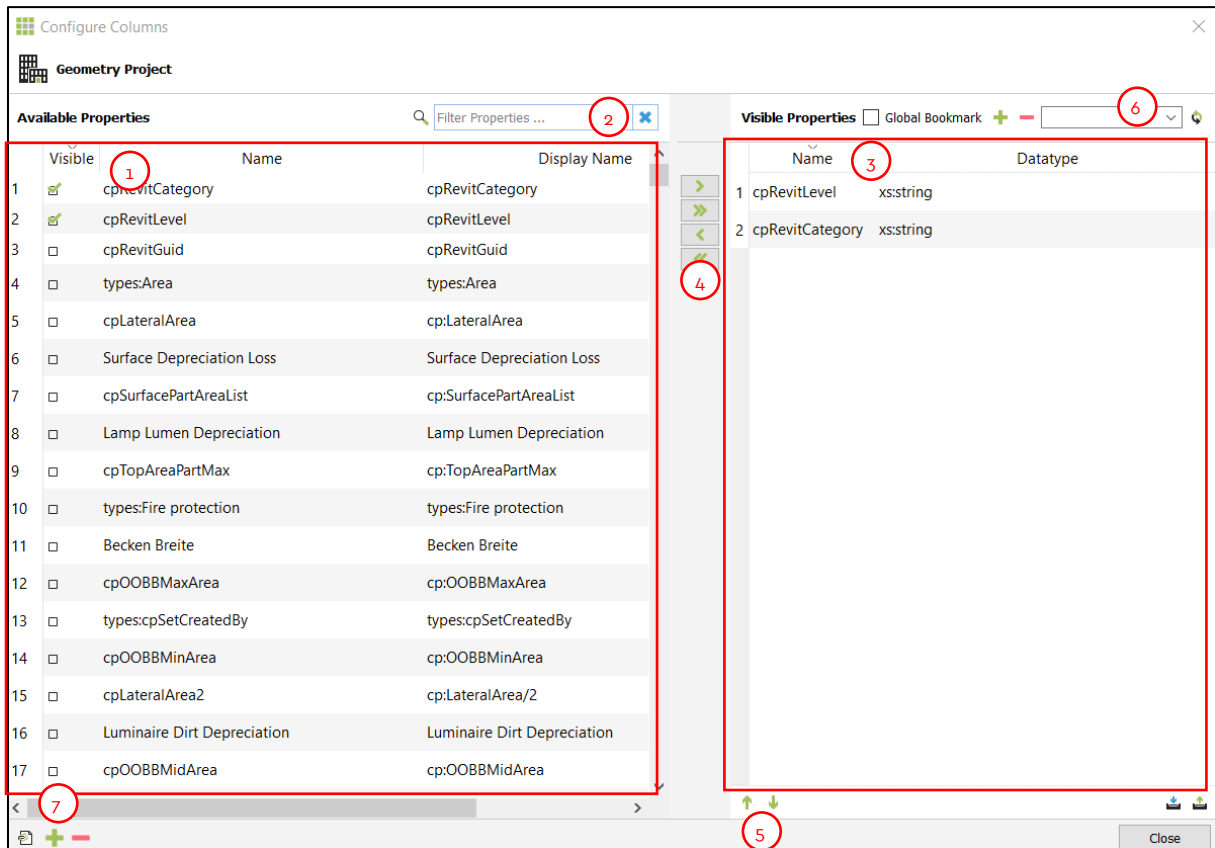
Properties of objects and domain elements can be displayed in the domain tree structure in table format. You can set the object properties to be displayed in the structure table.



1. Click on Column configuration / Bookmarks button and then on
2. Configure columns.

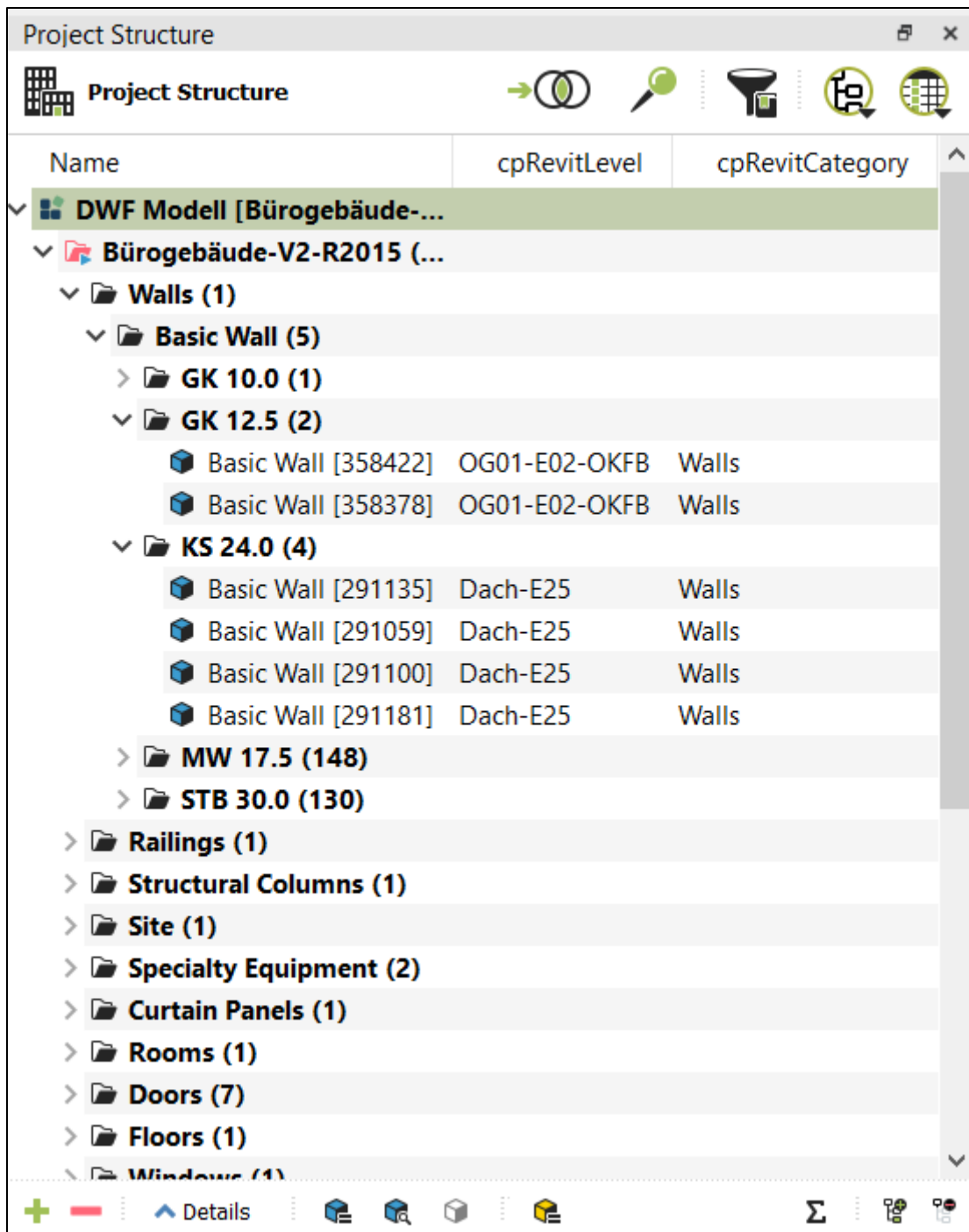
Configure columns window will open. From a list of all available object properties, you can select those to be displayed as columns in the project structure table.

**Tip:** A customized column configuration is useful for filtering the model elements and creating selection sets in the geometry domain.



1. List of all available properties for the active Domain
2. Filter the list of available properties by typing in any name. Use the check box under Visible or click on the desired property and use > button to add it to the visible properties.
3. List of visible properties to be displayed as columns in the project structure table
4. Use the arrow button to add / remove one or multiple properties from the list of visible properties
5. Use the arrow button to move a selected property up or down
6. You can save your configuration of properties if you insert a name in the bookmark list and press '+' button at the upper right corner to create a bookmark.
7. Use the '+' button to create new properties and the - button to delete a property permanently from the project

**Result:** two new columns are displayed now in the project structure table:



### 6.1.1 Custom Properties Types

You can create own Properties Types with the '+' button on the 'Configure Columns' dialog, in order to extend your project with additional data:

**Create Property Type**

.....

Name

Data type

Unit

Inheritable ☒

.....

Create in ☒ Memory / Domain Project

☐ Database

.....

The name and data types defines the uniqueness of a Property Type per domain. The option 'Inheritable' defines, if a property value will automatically assigned to all child elements of an object. Inherited values are marked in red:

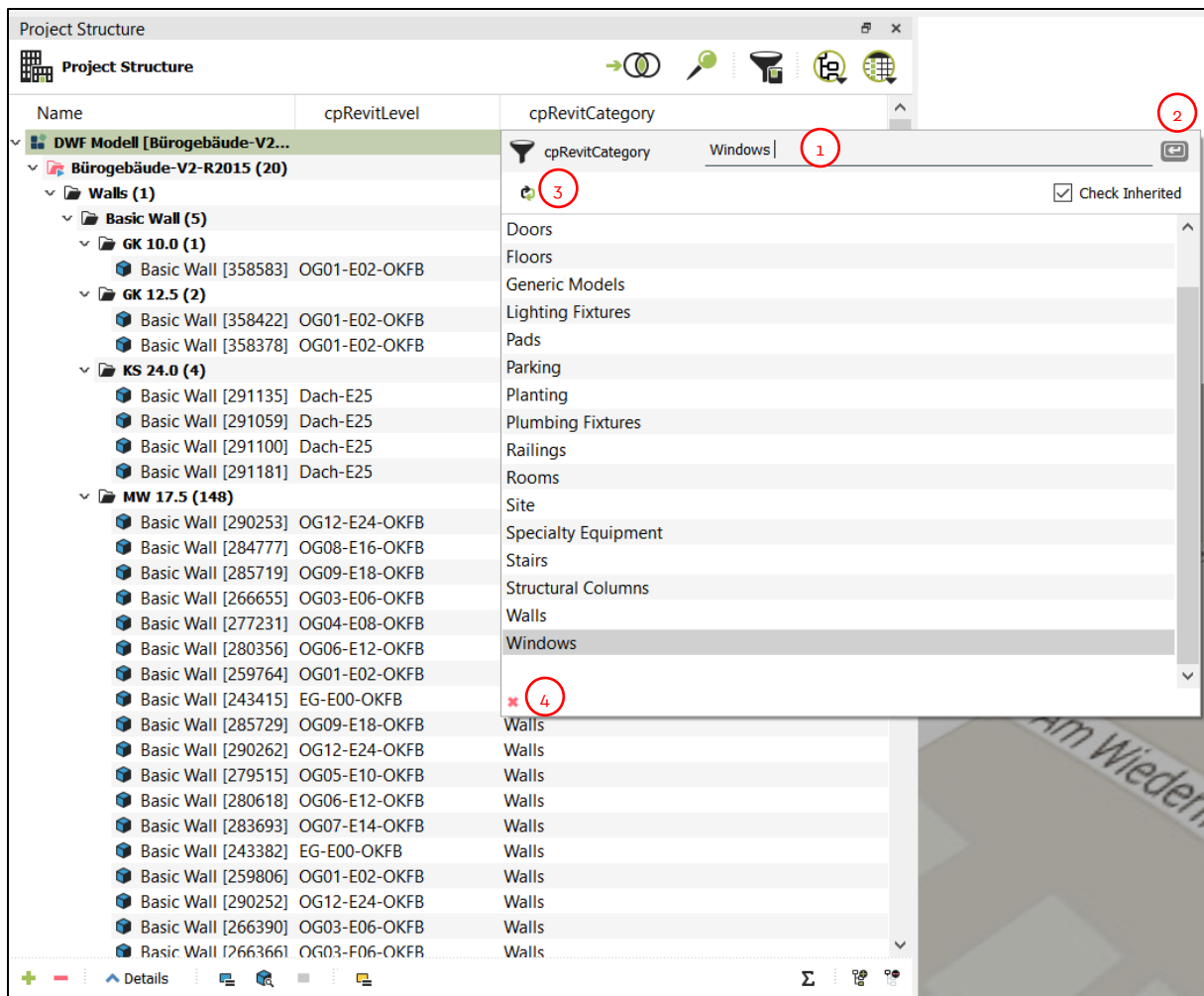
Project Structure	
Project Structure	
Name	CustomProperty [m3]
✓ IFC Model [Cube_Architectural Model.ifc] (1)	
✓ 0001 (1)	
✓ Default (1)	
✓ Building (2)	
✓ Level 1 (14)	3,0000
Basic Wall:Generic - 300mm:299076	3,0000
> Basic Wall:Generic - 300mm:299077 (1)	3,0000

The Property Values can either be stored within the project (domain xml files) or in a database, if any is assigned to the project (see [Connecting to a Database](#)).

### 6.1.2 Filters

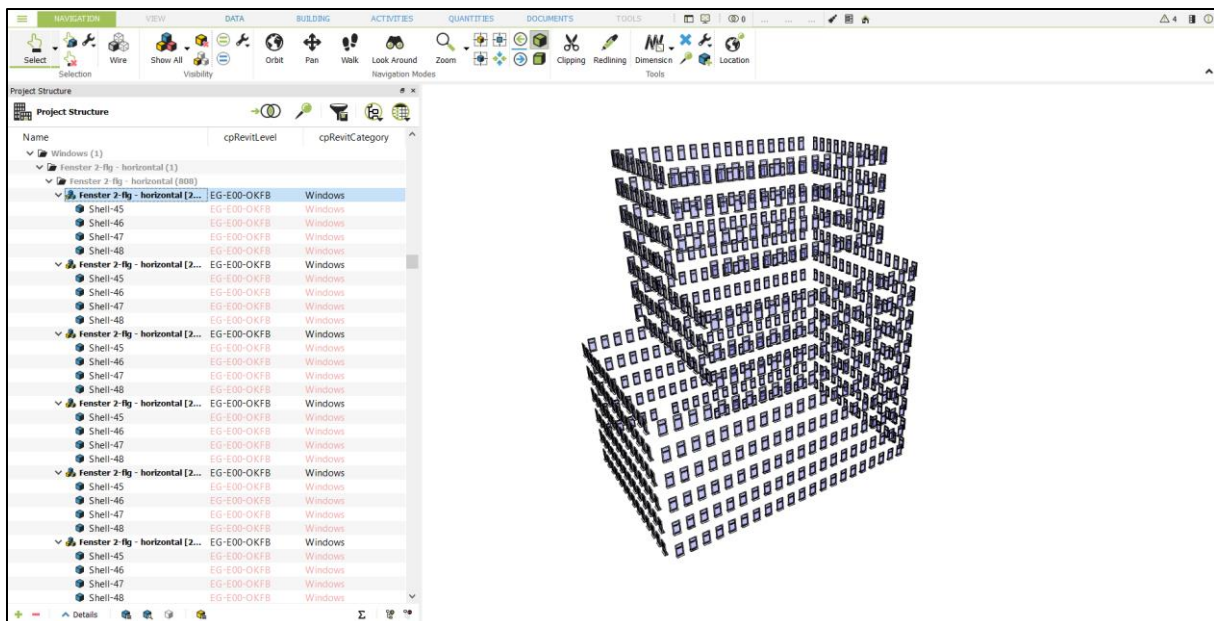
Model objects in the tree structure can be filtered according to any desired criteria. Filtered object will be drawn in grey color in the tree structure. If you filter 3d geometry models, this will also hide the model in the 3d view.

Click on the column header (e.g. 'cpRevitCategory' in this example) in order to open the filter menu:



1. Enter field for filter expression
2. Apply filter (you can also press 'Enter')
3. Update preview (note: not all available values are visible in the preview list. This depends on the amount of different values)
4. Remove current filter

If you want to filter model objects by value 'Windows' for the attribute 'cpRevitCategory', for example, choose 'Windows' from the list. Once you press Enter key, only those objects which contain property value 'Windows' will be displayed in 3D view.



Wildcards and operators can also be used to apply filters. For example '!' symbol negates the filter and all objects will be found which do not contain the value in a property.

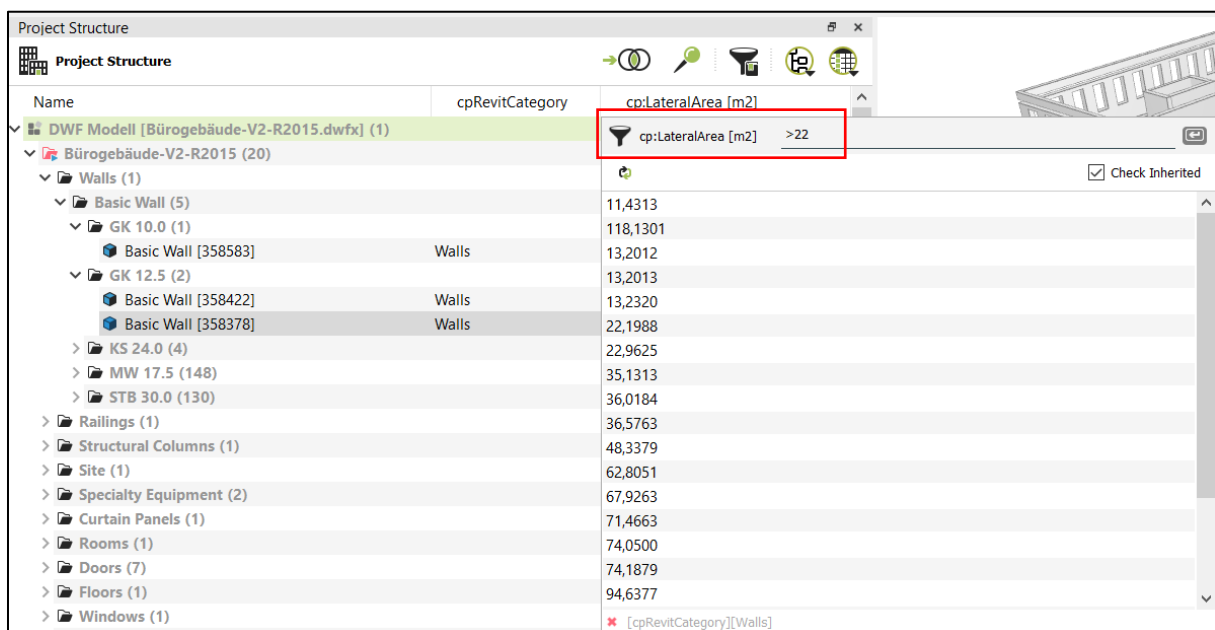
The following wildcards and operators are supported:

Search term	Description
<b>Wa*</b>	Text that begins with 'Wa', such as 'Wall'
<b>*wall*12.*</b>	Text that contains the elements 'Wall' and '12.', such as 'BasicWall_12.5'
<b>Concrete Steel Masonry</b>	Text is either 'Concrete' or 'Steel' or 'Masonry'
<b>'Basic Wall XY'</b>	Text equals 'Basic Wall XY' (without quotation marks, this would mean 'Basic Wall' or 'XY')
<b>!Wa*</b>	Text that does not begin with 'Wa'
<b>Wa* Wind*</b> (space in between)	Text that begins with 'Wa' or 'Wind'
<b>&gt; C</b>	All text that begins with D, E, F etc.
<b>&lt; 0</b>	All negative numbers
<b>!*</b>	Property is not present/defined
<b># Wa*</b>	Remove all applied filters (i.e. no AND relation) and then show all objects which the property value begins with 'Wa'.

## Notes:

- Filter is always applied to the currently visible partial models.
- Placing a '#' at the beginning of search term means that filter will not only be applied to the visible elements, but also to all available elements in the project structure.
- Space in the expression means an OR relation.
- Multiple filters include an AND relation.
- Search term is not case sensitive.
- '\*' is a placeholder for any desired term.

It is also possible to combine multiple filters. In the following example, the objects are already filter by the value 'Walls' in the property 'cpRevitCategory'. Now an additional filter is entered to show only walls with a lateral area bigger than 22:



It is possible to filter with values and also with operators:

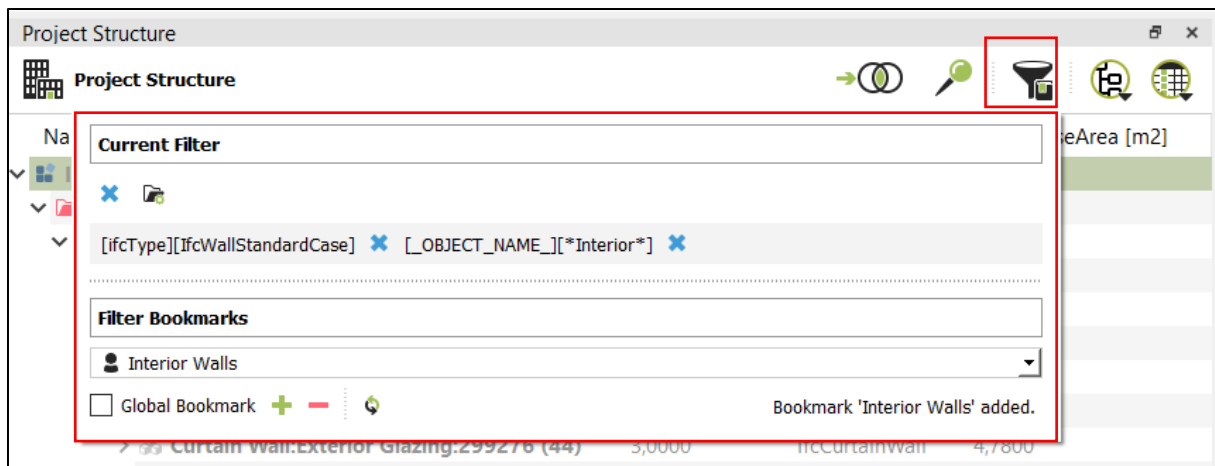
## Values:

- Single values or wildcards
- Enumerations: a1 a2 a3

## Operators:

- Equal: [without special character]
- Unequal/ not equal: !
- Greater/ greater than equal: >, >=
- Smaller/ less equal: <, <=

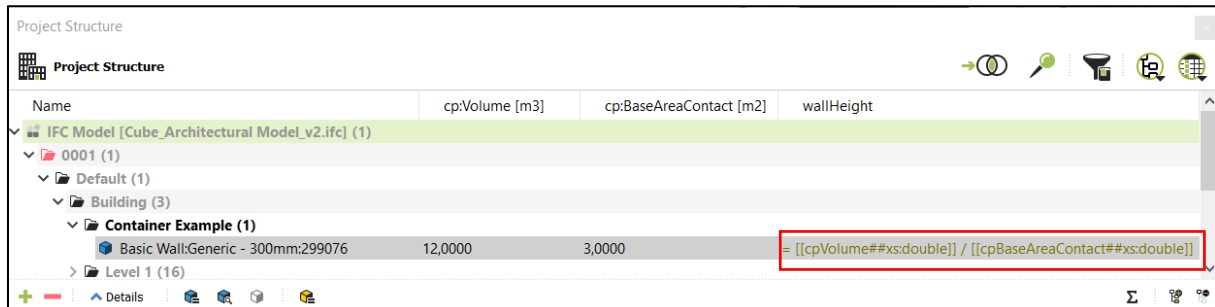
With the 'Filter Bookmarks' tool, you can edit all currently assigned filters, save them as bookmarks and apply already available filter bookmarks:



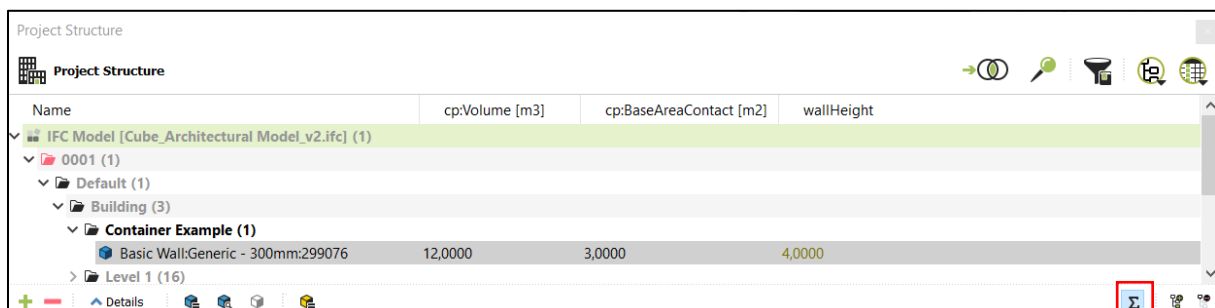
### 6.1.3 Formulas and Inheritance

You can use formulas in property value fields to create simple calculations based on other property values of the element.

The following example shows a simple calculation of a wall height from the geometric properties 'cpVolume' and 'cpBaseAreaContact' (for more information about derived geometric properties, see **Properties for Geometry domain**) in the property 'wall height':



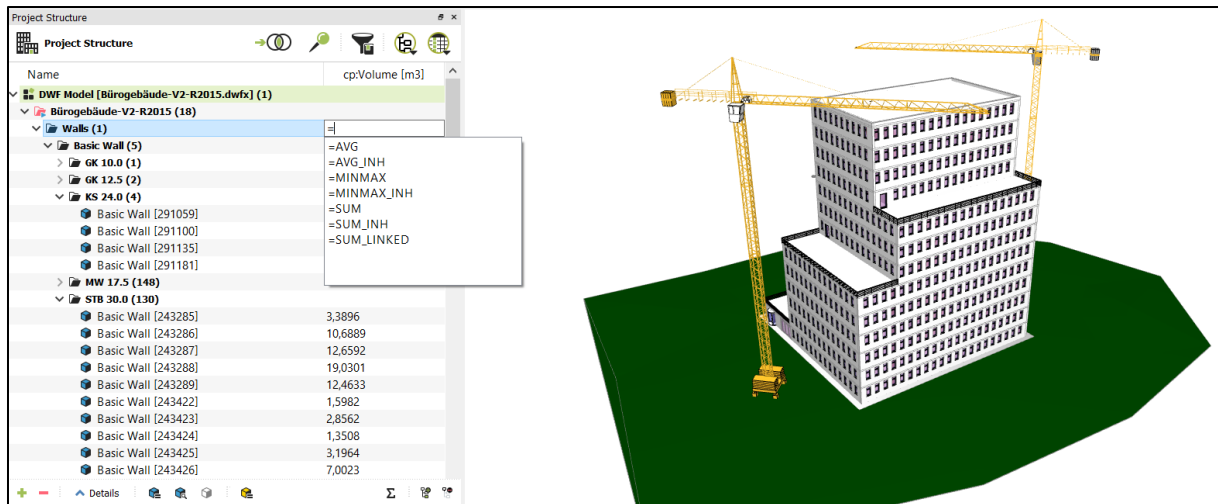
To see the result, turn on the calculation with the 'Sum' icon:



### Aggregate functions

Aggregate functions can be used at any point in domain structure, i.e. to calculate total volume of all walls.



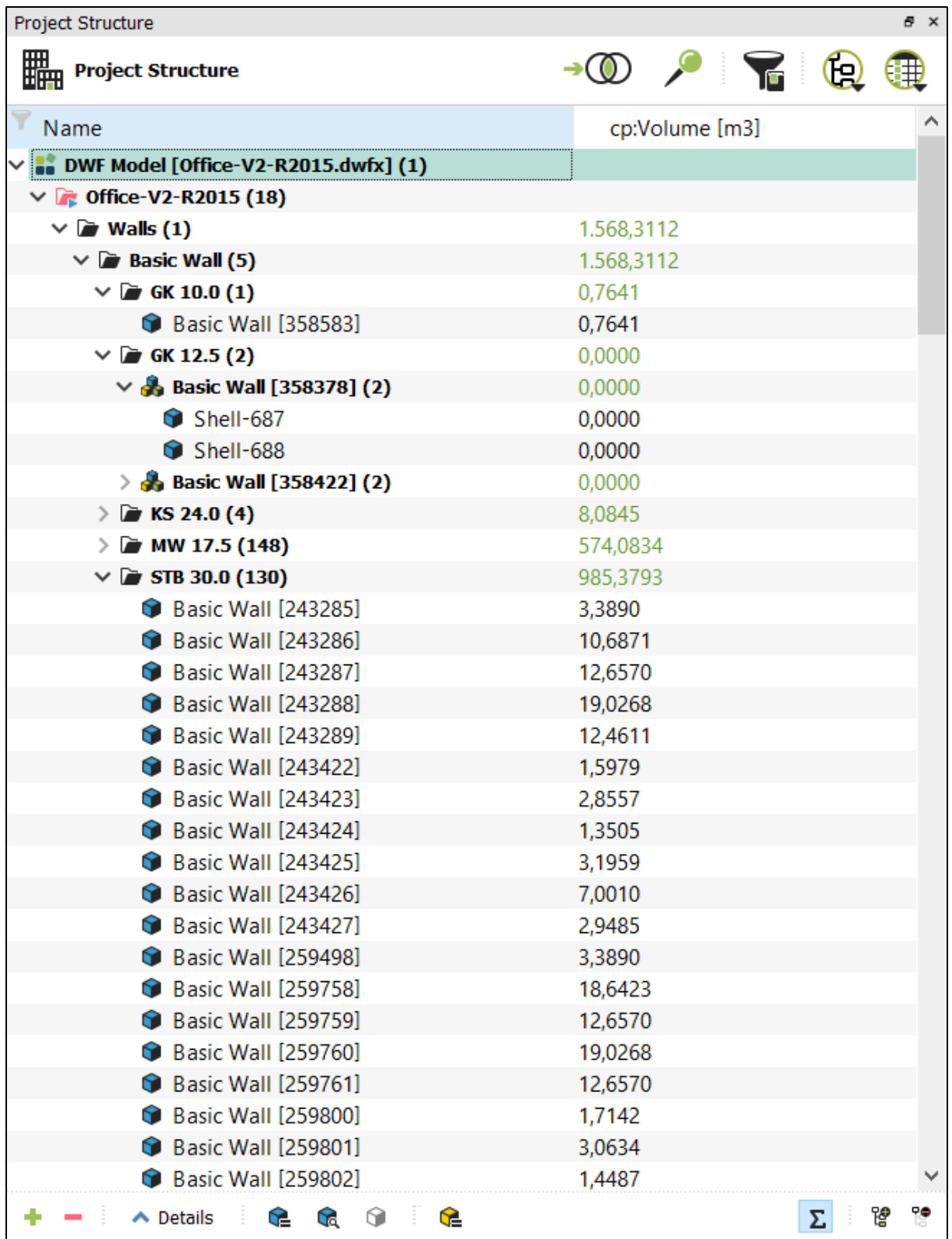


Since property values are inherited from parent elements to their children by default, you often need to insert aggregate function at the highest hierarchical level.

1. Inherited values are displayed in red.

Project Structure	
Name	cp:Volume [m3]
▼ <b>DWF Model [Office-V2-R2015.dwf] (1)</b>	
▼ <b>Office-V2-R2015 (18)</b>	
▼ <b>Walls (1)</b>	=SUM
▼ <b>Basic Wall (5)</b>	=SUM
▼ <b>GK 10.0 (1)</b>	=SUM
Basic Wall [358583]	0,7641
▼ <b>GK 12.5 (2)</b>	=SUM
▼ <b>Basic Wall [358378] (2)</b>	=SUM
Shell-687	0,0000
Shell-688	0,0000
> <b>Basic Wall [358422] (2)</b>	=SUM
> <b>KS 24.0 (4)</b>	=SUM
> <b>MW 17.5 (148)</b>	=SUM
▼ <b>STB 30.0 (130)</b>	=SUM
Basic Wall [243285]	3,3890
Basic Wall [243286]	10,6871
Basic Wall [243287]	12,6570
Basic Wall [243288]	19,0268
Basic Wall [243289]	12,4611
Basic Wall [243422]	1,5979
Basic Wall [243423]	2,8557
Basic Wall [243424]	1,3505
Basic Wall [243425]	3,1959
Basic Wall [243426]	7,0010
Basic Wall [243427]	2,9485
Basic Wall [259498]	3,3890
Basic Wall [259758]	18,6423
Basic Wall [259759]	12,6570
Basic Wall [259760]	19,0268
Basic Wall [259761]	12,6570
Basic Wall [259800]	1,7142
Basic Wall [259801]	3,0634
Basic Wall [259802]	1,4487

2. Calculated values are displayed in green,



The screenshot shows the 'Project Structure' window with a tree view of project elements. The 'Name' column lists the hierarchy, and the 'cp:Volume [m3]' column shows the volume for each element. The tree is expanded to show the 'Office-V2-R2015' folder, which contains 'Walls', 'Basic Wall', 'GK 10.0', 'GK 12.5', 'Basic Wall [358378]', 'Shell-687', 'Shell-688', 'Basic Wall [358422]', 'KS 24.0', 'MW 17.5', and 'STB 30.0'. The 'STB 30.0' folder is further expanded to show a list of 'Basic Wall' elements with their individual volumes.

Name	cp:Volume [m3]
DWF Model [Office-V2-R2015.dwfx] (1)	
Office-V2-R2015 (18)	
Walls (1)	1.568,3112
Basic Wall (5)	1.568,3112
GK 10.0 (1)	0,7641
Basic Wall [358583]	0,7641
GK 12.5 (2)	0,0000
Basic Wall [358378] (2)	0,0000
Shell-687	0,0000
Shell-688	0,0000
Basic Wall [358422] (2)	0,0000
KS 24.0 (4)	8,0845
MW 17.5 (148)	574,0834
STB 30.0 (130)	985,3793
Basic Wall [243285]	3,3890
Basic Wall [243286]	10,6871
Basic Wall [243287]	12,6570
Basic Wall [243288]	19,0268
Basic Wall [243289]	12,4611
Basic Wall [243422]	1,5979
Basic Wall [243423]	2,8557
Basic Wall [243424]	1,3505
Basic Wall [243425]	3,1959
Basic Wall [243426]	7,0010
Basic Wall [243427]	2,9485
Basic Wall [259498]	3,3890
Basic Wall [259758]	18,6423
Basic Wall [259759]	12,6570
Basic Wall [259760]	19,0268
Basic Wall [259761]	12,6570
Basic Wall [259800]	1,7142
Basic Wall [259801]	3,0634
Basic Wall [259802]	1,4487

3. Results of formulas can be displayed by clicking on  $\Sigma$  sign

Following aggregate functions are available:

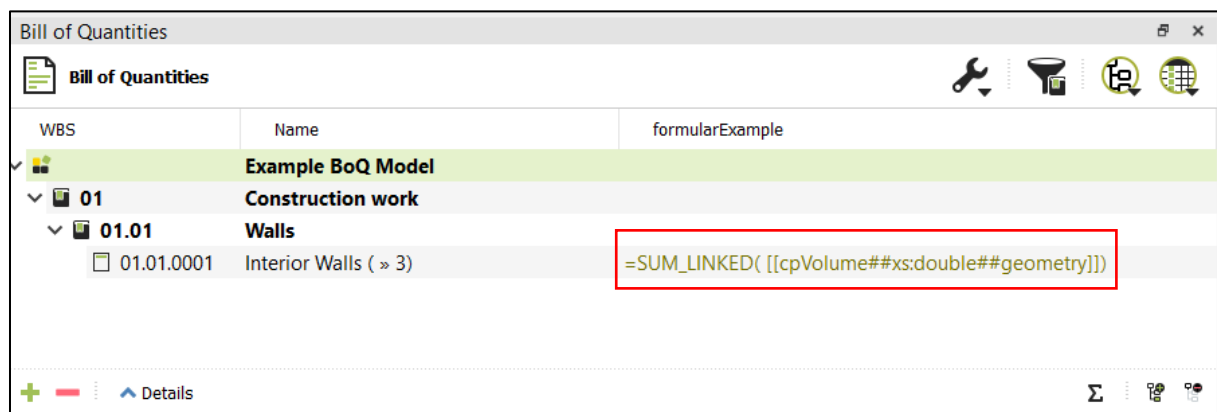
<b>AVG</b>	Average value
<b>AVG_INH</b>	Average value including inherited values

<b>MINMAX</b>	Minimum and maximum values
<b>MINMAX_INH</b>	Minimum and maximum values including inherited values
<b>SUM</b>	Sum
<b>SUM_INH</b>	Sum including inherited values
<b>SUM_LINKED(variable)</b>	Sum of expression in brackets (e.g. a property type) of all linked objects

### Aggregation function with linked objects

The function 'SUM\_LINKED' allows you to do a calculation based on property values of the linked geometry objects.

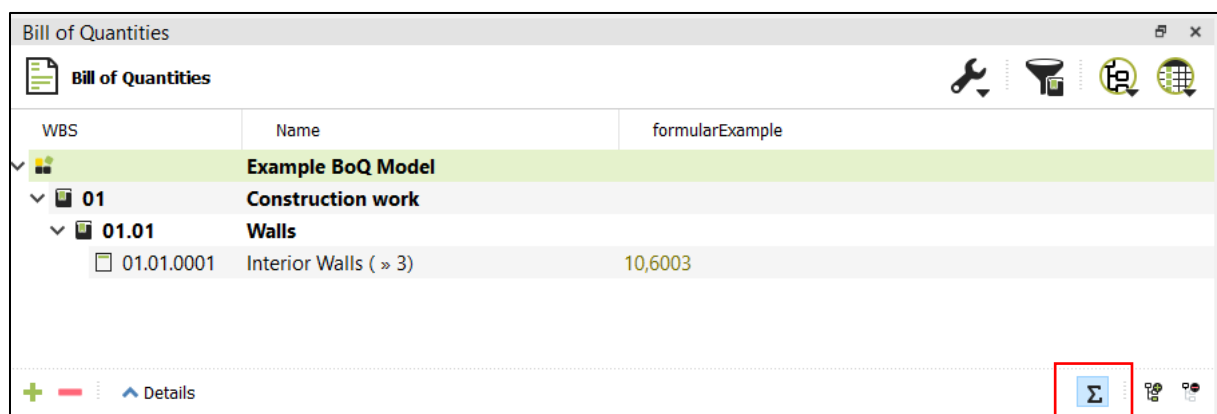
The following example shows a work item in the **Bill of Quantities** domain. A custom property type 'formularExample' (see Custom Properties Types) with the data type 'double' was created. The formula calculates the sum of the property 'cpVolume' off all linked geometry objects:



The screenshot shows the 'Bill of Quantities' window with a tree view on the left and a table on the right. The tree view shows a hierarchy: '01 Construction work' > '01.01 Walls' > '01.01.0001 Interior Walls ( » 3)'. The table has columns 'WBS', 'Name', and 'formularExample'. The formula '=SUM\_LINKED( [[cpVolume##xs:double##geometry]])' is entered in the 'formularExample' column for the '01.01.0001 Interior Walls ( » 3)' item. A red box highlights the formula.

WBS	Name	formularExample
✓	Example BoQ Model	
✓	01 Construction work	
✓	01.01 Walls	
01.01.0001	Interior Walls ( » 3)	=SUM_LINKED( [[cpVolume##xs:double##geometry]])

To see the result, turn on the calculation with the 'Sum' icon:



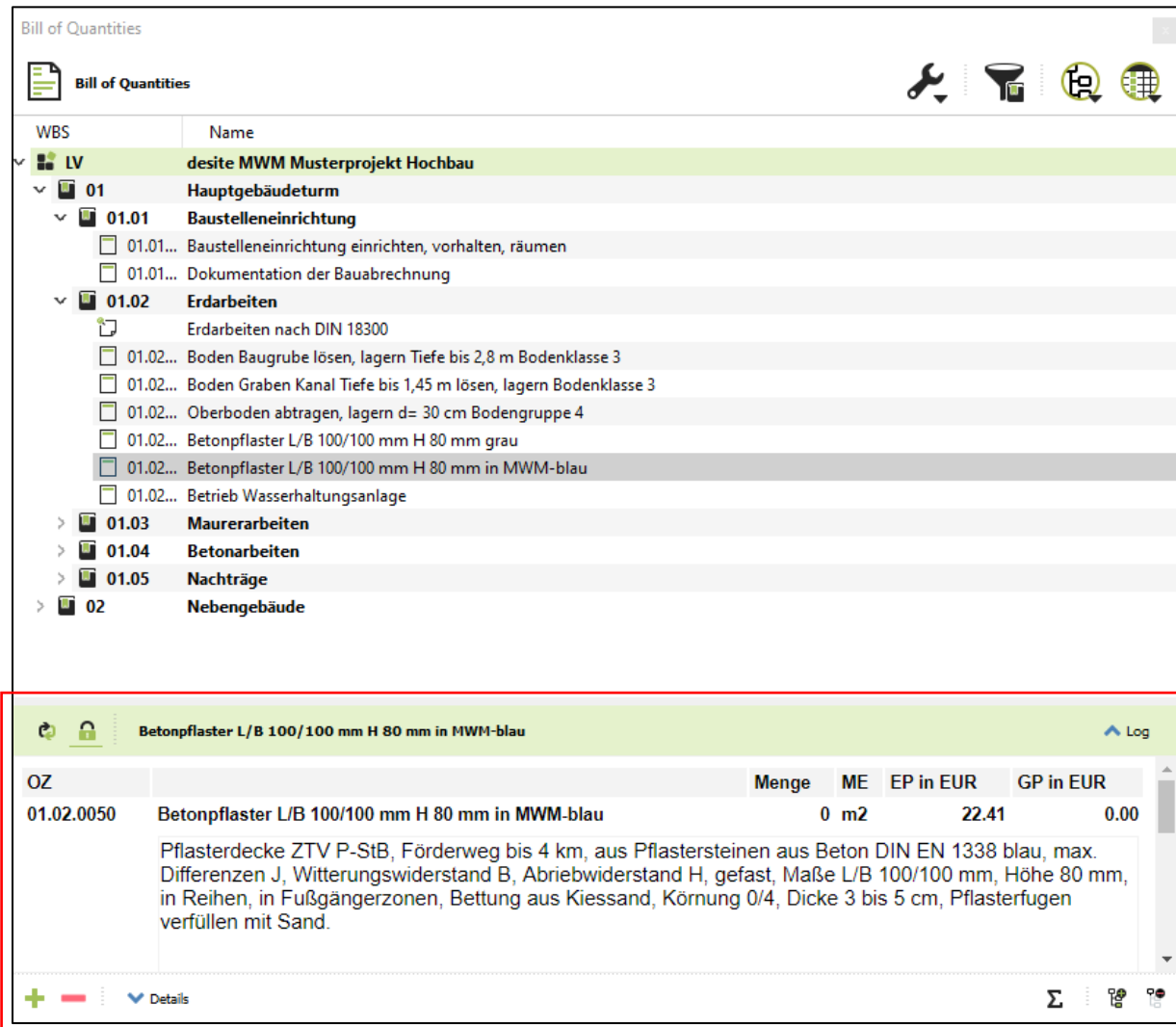
The screenshot shows the same 'Bill of Quantities' window as before, but now the 'Sum' icon (Σ) in the bottom right corner is highlighted with a red box. The value '10,6003' is displayed in the 'formularExample' column for the '01.01.0001 Interior Walls ( » 3)' item.

WBS	Name	formularExample
✓	Example BoQ Model	
✓	01 Construction work	
✓	01.01 Walls	
01.01.0001	Interior Walls ( » 3)	10,6003

### 6.1.4 Details View

In the domains, a detail view can be expanded, in which details of the current element (selected row) are displayed with an HTML form.

The detail view can be configured with a template. A standard template is delivered with DESITE and is deposited in the installation of the software in the program directory.



Bill of Quantities

Bill of Quantities

WBS Name

- LV desite MWM Musterprojekt Hochbau
  - 01 Hauptgebäudeturm
    - 01.01 Baustelleneinrichtung
      - 01.01... Baustelleneinrichtung einrichten, vorhalten, räumen
      - 01.01... Dokumentation der Bauabrechnung
    - 01.02 Erdarbeiten
      - 01.02... Erdarbeiten nach DIN 18300
      - 01.02... Boden Baugrube lösen, lagern Tiefe bis 2,8 m Bodenklasse 3
      - 01.02... Boden Graben Kanal Tiefe bis 1,45 m lösen, lagern Bodenklasse 3
      - 01.02... Oberboden abtragen, lagern d= 30 cm Bodengruppe 4
      - 01.02... Betonpflaster L/B 100/100 mm H 80 mm grau
      - 01.02... Betonpflaster L/B 100/100 mm H 80 mm in MWM-blau
      - 01.02... Betrieb Wasserhaltungsanlage
    - 01.03 Maurerarbeiten
    - 01.04 Betonarbeiten
    - 01.05 Nachträge
  - 02 Nebengebäude

Betonpflaster L/B 100/100 mm H 80 mm in MWM-blau

OZ	Menge	ME	EP in EUR	GP in EUR
01.02.0050	Betonpflaster L/B 100/100 mm H 80 mm in MWM-blau	0 m2	22.41	0.00

Pflasterdecke ZTV P-StB, Förderweg bis 4 km, aus Pflastersteinen aus Beton DIN EN 1338 blau, max. Differenzen J, Witterungswiderstand B, Abriebwiderstand H, gefast, Maße L/B 100/100 mm, Höhe 80 mm, in Reihen, in Fußgängerzonen, Bettung aus Kiessand, Körnung 0/4, Dicke 3 bis 5 cm, Pflasterfugen verfüllen mit Sand.

Details

**Tip:** If you select a new entry in the domain table, you have to click on the button Update details. When the padlock symbol is closed, details are updated automatically for selected entry.

### Configuration

The templates for the details are reloaded at runtime. The order of the directories where templates are searched is:

1. [PRJNAME].session/gui/ user
2. [PRJNAME].templates/gui/ project
3. [APPDIR]/gui/ Global

The following files are searched in these folders according to the domain.

Example: Bill of Quantity:

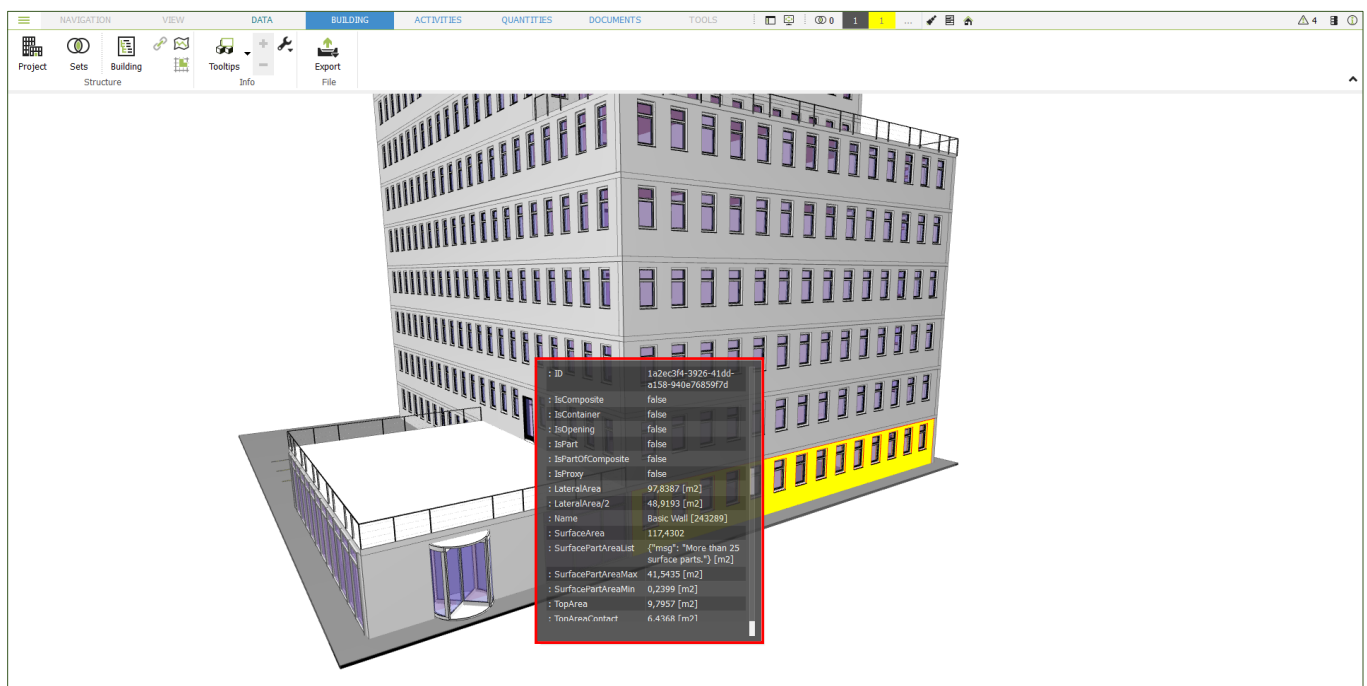
- <...>/gui/domainWorkScope/Item.html
- <...>/gui/domainWorkScope/Model.html

## 6.2 Display Property Information

Properties of Geometry Objects can be displayed in several ways.

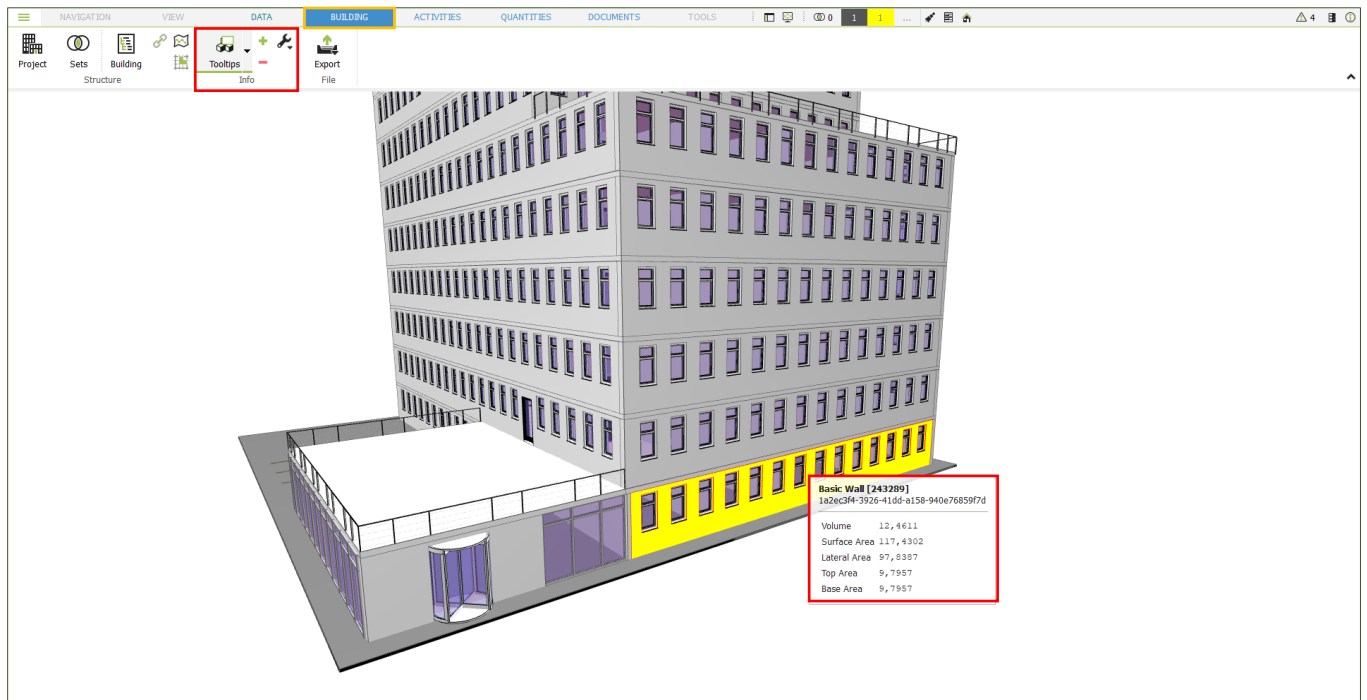
### 6.2.1 Quickinfo & Tooltips

Use I key to access QuickInfo. A list of properties of the selected objects are listed in the QuickInfo window by default.



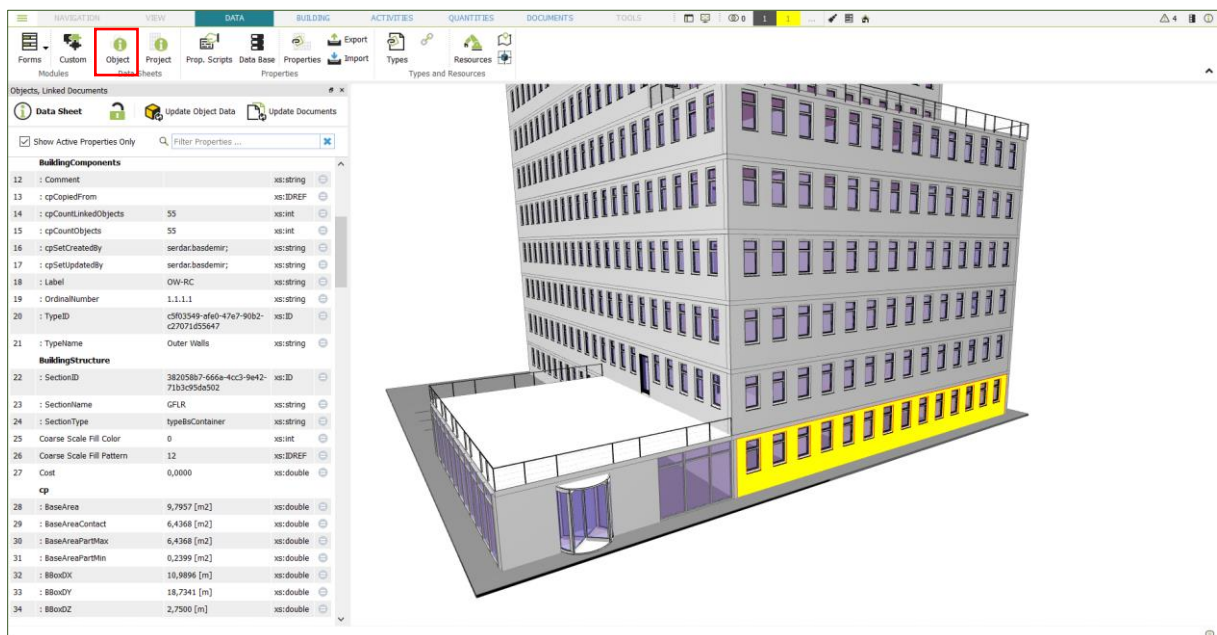
**Tip:** Properties listed in QuickInfo can also be displayed in the Data Sheet under Data menu tab.

Use 'T' key to access Tooltips. Name, ID and geometric properties of the selected object(s) are displayed in the tooltip window. Alternatively, you can open tooltips by clicking on the Tooltips button in the Building toolbar.



## 6.2.2 Data Sheet

You can open Data Sheet by clicking on Object under Data menu tab. All available information and properties of the model objects are displayed in data sheet.



Close the lock to get an immediate update of the object data after selecting another object



If the lock is not closed, you can update the data sheet for the current object selection

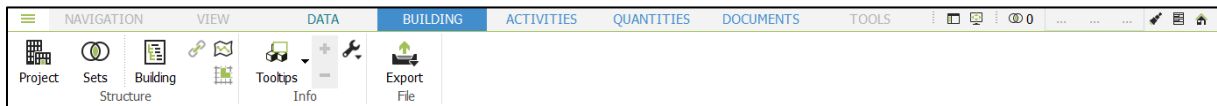


Shows a list of all linked Documents (see **Documents Domain**) for the current selected object.

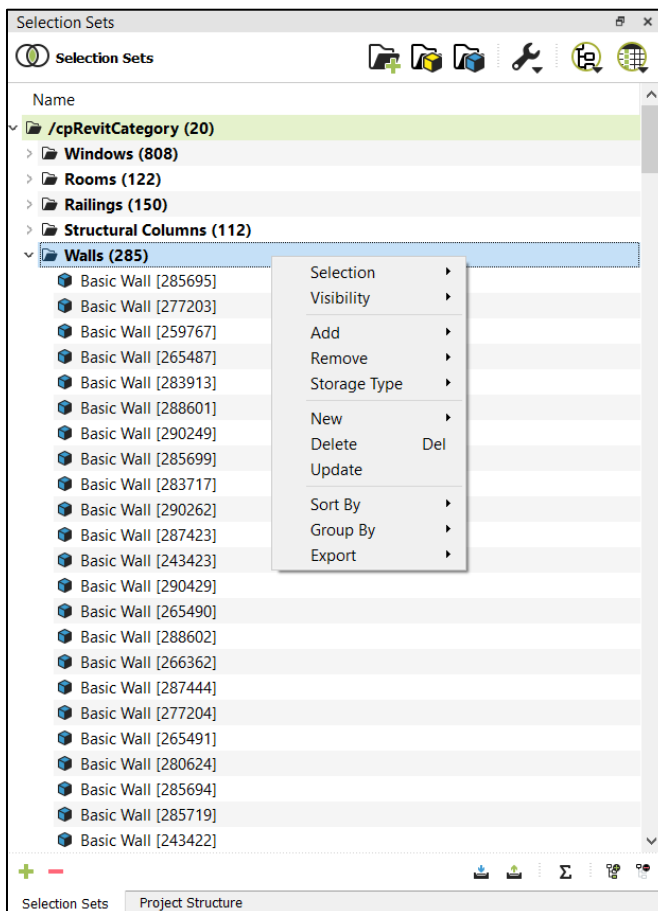


## 6.3 Selection Sets

Selection sets are a way of combining certain partial models and defining your own structure of model data. You can open selection sets window by clicking on Sets button in the Building toolbar.

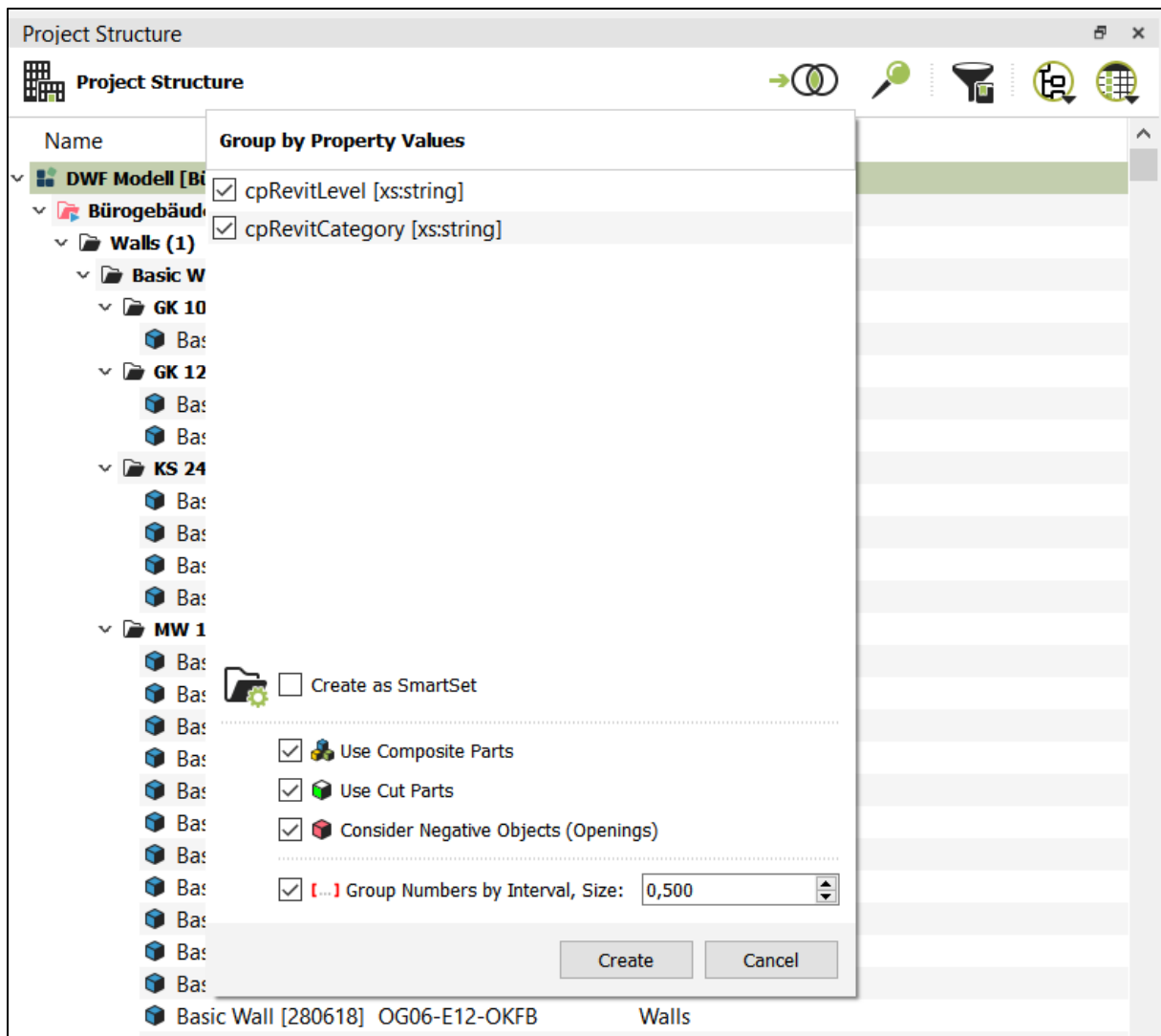


Object properties can also be displayed in this window, as in the project structure. You can also show and hide, select and deselect objects in the selected sets.



Selection sets can automatically be created in Project Structure. Here visible objects, which are hierarchically structured according to the selected properties in the dialogue, are combined into new data sets.

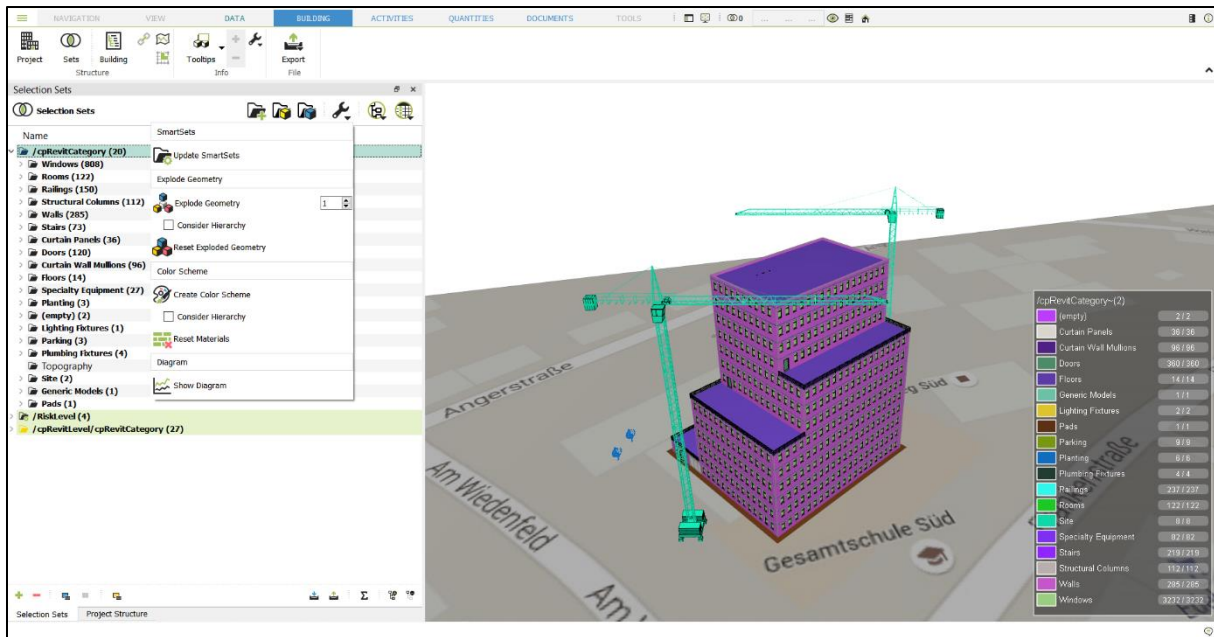




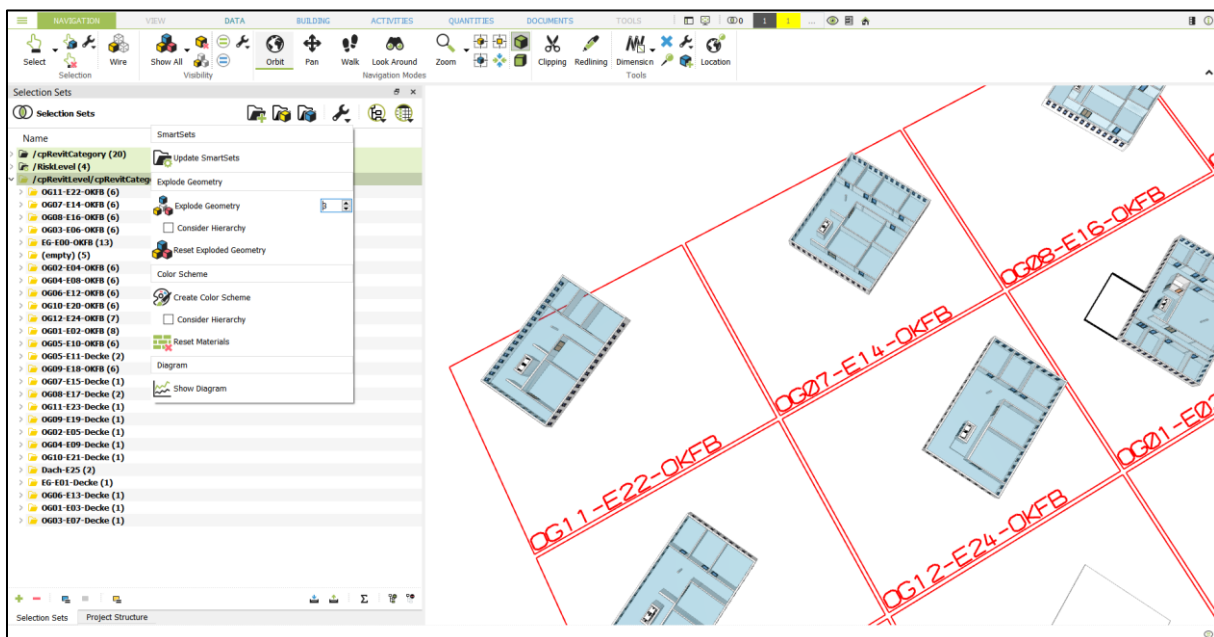
### Tips:

- When creating selection sets, it is possible to specify whether sub-parts and Composites are included in the set or their sub-parts. Opening elements can also be taken into consideration in the same way.
- Smart sets store the rules for creating selection set, in addition to the structure and the objects. Use smart sets if you want to update the selected set with the new objects after updating the model.
- Both in project structure and in selection sets, you can export entire model or only partial models in cpixml format.

With selection sets, it is possible to visualise elements in color. First click on the spanner icon, and then select **Color schemes**. Elements that have the same property values will be assigned the same color.

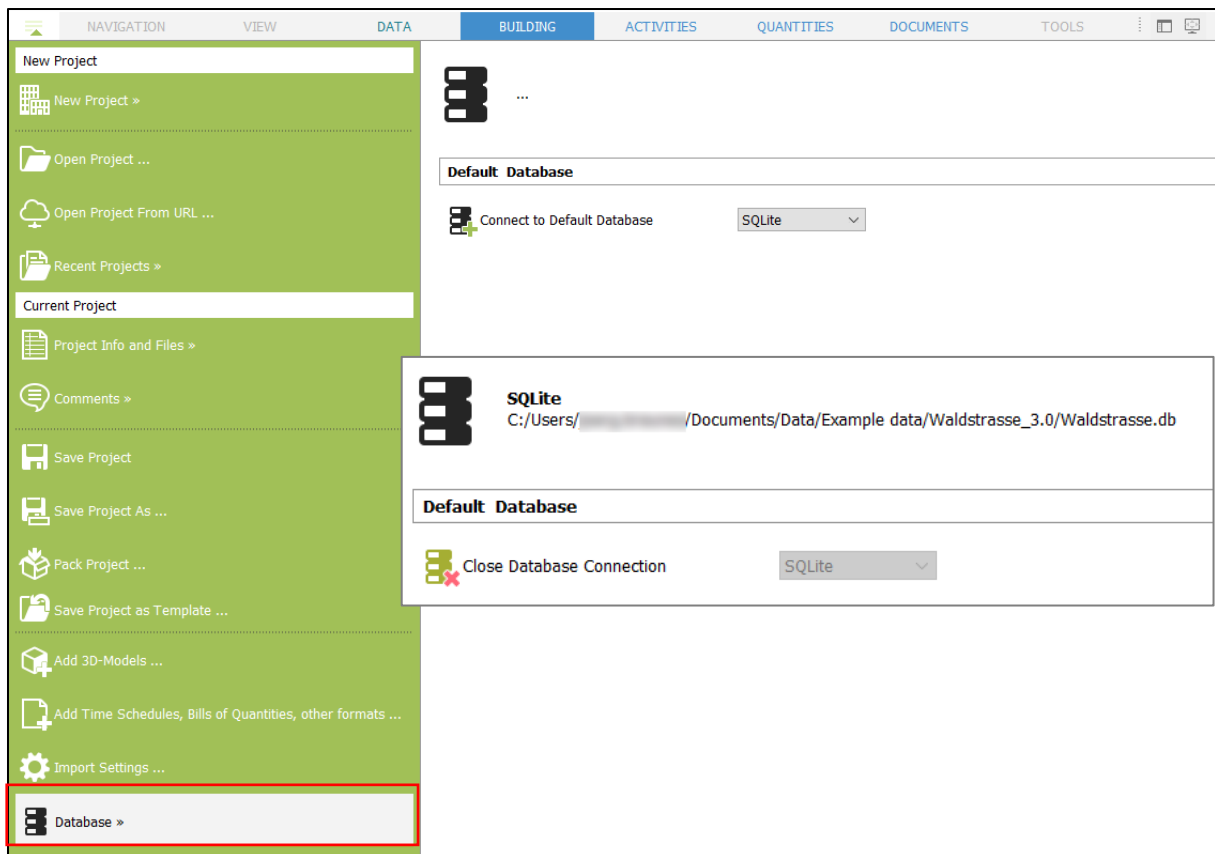


Similarly, selection sets can also be used to generate an exploded view. First click on the spanner icon, and then select Explode Geometry.

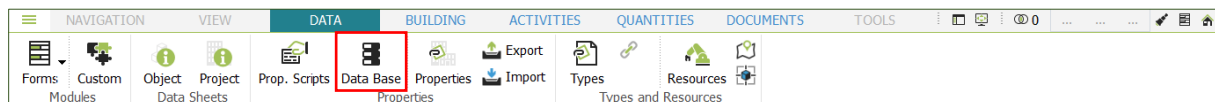


## 6.4 Connecting to a Database

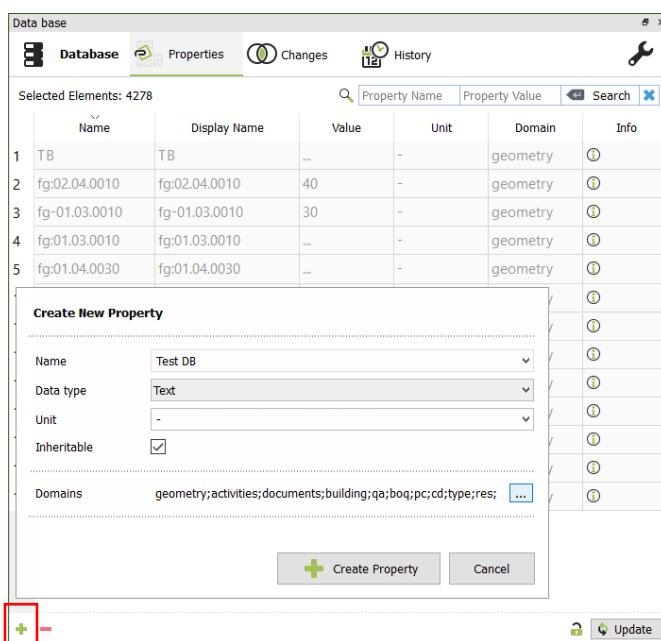
Once the project has been saved, standard project database can be integrated via the application menu. If the building model is linked to a database, you can add further information and data to your model.



Once your model is linked to a database, click on the Database button under Data menu tab.

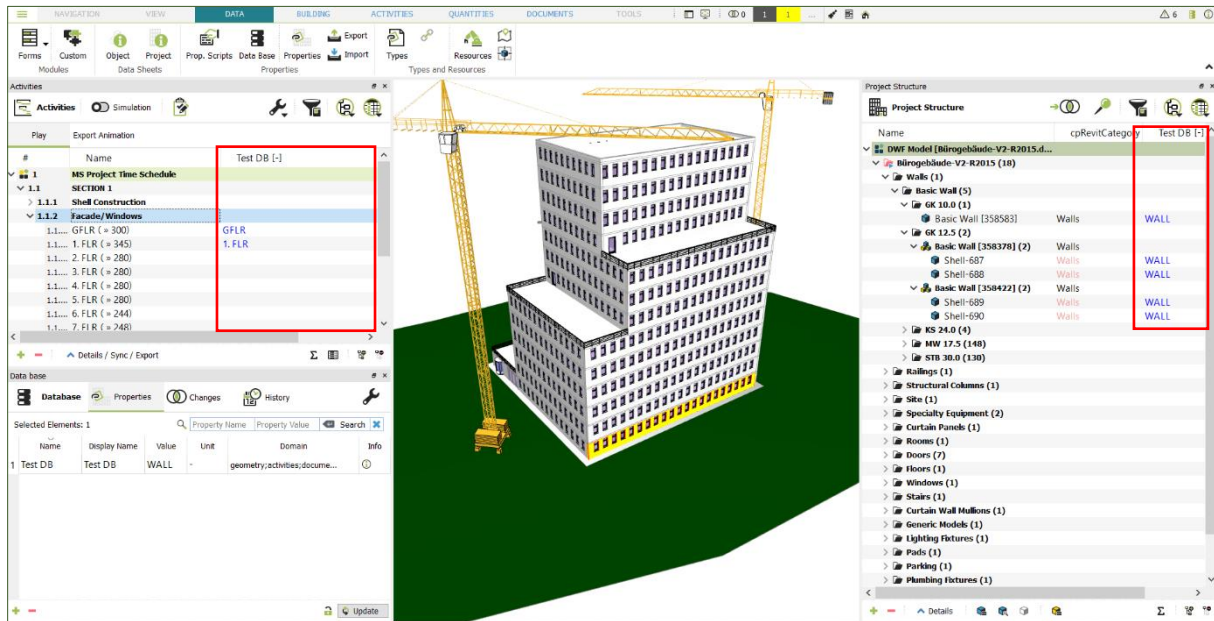


Select at least one object and click on '+' button to create a new property. Select name, data type, unit and domain in which the property is to be available.



**Note:** New attributes are saved only in the database and they are not present in the model. They can only be accessed by connecting the project to the database.

In the following example, a new property 'Test DB' is created and assigned to each domain. The attribute can be displayed in the tree view. Values that come from the database are shown in blue in the project.



## 6.5 DESITE cp-Properties

Properties of a 3D object or also of a container, which can be determined from the geometry of the object, are automatically available in DESITE. The names of these properties start with the abbreviation 'cp'. In the following these properties are described.

### 6.5.1 General properties for all domains

Property	Data Type	Description
<i>Identification</i>		
cpID	xs:ID	Unique Object/Element ID, assigned by DESITE
cpExternalID	xs:ID	Object/Element ID provided by the imported 3d model file (e.g. IFC GUID)
cpLookupString	xs:string	String to identify objects by their external ID and parent hierarchy. Used with redirects to match same objects.
cpName	xs:string	Object/Element name
<i>Linking Rules</i>		
cpLinkRule	xs:string	Name of the Linking Rule

pcLinkRuleValue	xs:string	Value of the Linking Rule in evaluated form
<i>Counter / structure</i>		
cpCount	xs:long	Helper property to count objects/elements. The value is always '1'
cpCountLinkedObjects	xs:int	Number of linked geometric objects
cplsContainer	xs:boolean	'true' if the element is a container

## 6.5.2 Properties for Geometry domain

Property	Data Type	Description
<i>General Properties</i>		
cplsPart	xs:boolean	'true' if the object is part of another object, e.g. it's the result of <b>Cutting</b>
cplsOpening	xs:boolean	'true' if the object is an opening (negative object) and child of another object
cpNegRefID	xs:IDREF	If the object is an opening, the ID of the object that contains this opening.
cplsComposite	xs:boolean	'true' if the object is a Composite
cplsPartOfComposite	xs:boolean	'true' if the object is a part of a Composite
cpCompositeID	xs:ID	ID of the Composite object (parent) in which the object (child) is included
cpCompositeIndex	xs:ID	Index assigned by DESITE for a Composite. All parts of a composite have the same index.
cpCompositePartIndex	xs:int	Index number assigned by DESITE for all parts of a Composite starting with 'o'
cpRefID	xs:IDREF	ID of the source object, if the object is a Proxy
<i>Geometry counter</i>		
cpCountTriangles	xs:long	Number of triangles of a geometric object

cpCountVertices	xs:long	Number of vertices of a geometric object
cpCountLines	xs:long	Number of lines of a geometric object
<i>Linking counter</i>		
cpCountLinkedBuildingStructureElements	xs:int	Number of Building structure elements to which the object is linked
cpCountLinkedDocuments	xs:int	Number of Documents to which the object is linked
cpCountLinkedProcessComponents	xs:int	Number of Process Components to which the object is linked
cpCountLinkedResources	xs:int	Number of Resources to which the object is linked
cpCountLinkedTasks	xs:int	Number of Tasks to which the object is linked
cpCountLinkedWorkScopeItems	xs:int	Number of elements in the BoQ domain to which the object is linked
<i>Geometric properties</i>		
cpCOGx	xs:double	Coordinates of the center of gravity of a 3d body
cpCOGy	xs:double	
cpCOGz	xs:double	
cpVolume	xs:double	The volume of a 3D body. If the volume cannot be calculated (e.g. the shell of the object body is not closed), the value is taken from a possibly existing attribute 'Volume' (which can be part of the import)
cpVolumeUnchecked	xs:double	The unchecked volume of a 3D body. Calculated regardless of whether the 3D body is topologically correctly closed, and may be wrong in this case.
cpVolumeInterval	xs:string	The volume of a 3D body as string for the intervals '0-5', '5-10' etc. This can be used to filter objects by it's size. If the body is topologically not correctly closed, the value is 'not closed'
cpLinesLength	xs:double	Length of line elements. If several lines are contained in the object, the sum of the lengths of the contained lines is calculated.
cpHasCurvedSurfacePart	xs:boolean	'True' if there is at least one curved subarea in the object. The subarea is curved if at least

		one normal vector of a triangle deviates more than 2° from the average normal vector of the subarea.
cpCircumference	xs:double	Perimeter of a surface. Edges of holes in the surface are also counted.
cpSurfaceArea	xs:double	Total surface. For composites, the sum of the surfaces of the child objects.
cpSurfacePartAreaMax	xs:double	Largest connected part of the surface.
cpSurfacePartAreaMin	xs:double	Smallest connected part of the surface.
cpSurfacePartList	xs:double	List with the size of connected subareas in JSON format (unordered)
cpTopArea	xs:double	Surface area whose normal vector has a deviation of less than 45° from the positive z-axis. For composites the sum of the corresponding surfaces of the parts (child objects).
cpTopAreaPartMax	xs:double	Largest partial surface area in positive z-direction.
cpTopAreaPartMin	xs:double	Smallest partial surface area in positive z-direction
cpTopAreaContact	xs:double	Surface area in the direction of the z-axis that has a distance from the largest z-coordinate of the object of less than 1 cm. E.g. for a wall with openings, the upper surface without the parts of the window openings.
cpBaseArea	xs:double	Surface area whose normal vector has a deviation of less than 45° from the negative z-axis. For composites the sum of the corresponding surfaces of the parts (child objects).
cpBaseAreaPartMax	xs:double	Largest partial surface area in negative z-direction.
cpBaseAreaPartMin	xs:double	Smallest partial surface area in negative z-direction
cpBaseAreaContact	xs:double	Surface area in the direction of the z-axis that has a distance from the smallest z-coordinate of the object of less than 1 cm. E.g.



		for a wall with openings, the footprint without the parts of the window openings.
cpLateralArea	xs:double	Lateral surface area. For composites, the sum of the lateral areas of the child objects.
cpLateralArea2	xs:double	Half lateral surface area. For composites, the sum of the half lateral areas of the child objects.
<i>Material properties (see also <a href="#">Materials</a>)</i>		
cpMaterialID	xs:IDREF	ID of the material
cpMaterialName	xs:string	Material name
cpMaterialTransparency	xs:double	Material transparency between 0 (completely transparent) and 1 (completely opaque)
cpMaterialAmbient	xs:string	Ambient color as string
cpMaterialAmbientRed	xs:double	Red portion of the ambient color
cpMaterialAmbientGreen	xs:double	Green portion of the ambient color
cpMaterialAmbientBlue	xs:double	Blue portion of the ambient color
cpMaterialDiffuse	xs:string	Diffuse color as string
cpMaterialDiffuseRed	xs:double	Red portion of the diffuse color
cpMaterialDiffuseGreen	xs:double	Green portion of the diffuse color
cpMaterialDiffuseBlue	xs:double	Blue portion of the diffuse color
cpMaterialHasTexture	xs:boolean	Material has a texture yes/no
<i>Bounding Box</i>		
cpBBoxDX	xs:double	Dimensions and largest/smallest values of the bounding box aligned with the coordinate axes. For composites and containers, the common bounding box of all child objects.
cpBBoxDY	xs:double	
cpBBoxDZ	xs:double	
cpBBoxMinX	xs:double	
cpBBoxMinY	xs:double	
cpBBoxMinZ	xs:double	
cpBBoxMaxX	xs:double	
cpBBoxMaxY	xs:double	
cpBBoxMaxZ	xs:double	
<i>Optimized oriented bounding box (OOBB)</i> <i>(For composites and containers, the common optimized oriented bounding box of all child objects; see also <a href="#">Optimized Oriented Bounding Box</a>)</i>		
cpHasOOBB	xs:boolean	,True' if the object already has an optimized, oriented bounding box.



cpHasOOBBxy	xs:boolean	,True' if the object already has an optimized, oriented bounding box in xy-plane.
cpOOBBxyArea	xs:double	Surface area of the xy-plane of the OOBB in xy-plane
cpOOBBxyLength	xs:double	Length of the OOBB in xy-plane
cpOOBBxyWidth	xs:double	Width of the OOBB in xy-plane
cpOOBBHeight	xs:double	Height of the OOBB. The height is based on the axis that has the smallest deviation from the z-axis and thus points upwards.
cpOOBBLength	xs:double	Length of the OOBB. The length is calculated from the longer of the two remaining axes.
cpOOBBWidth	xs:double	Width of the OOBB. The width is calculated from the shorter of the two remaining axes.
cpOOBBMaxLength	xs:double	Maximum length of the OOBB
cpOOBBMidLength	xs:double	Medium length of the OOBB
cpOOBBMinLength	xs:double	Minimal length of the OOBB
cpOOBBVolume	xs:double	Volume of the OOBB.
cpOOBBArea-LH	xs:double	Surface area of the OOBB, calculated from length and height values
cpOOBBArea-LW	xs:double	Surface area of the OOBB, calculated from length and width values
cpOOBBArea-HW	xs:double	Surface area of the OOBB, calculated from height and width values
cpOOBBMaxArea	xs:double	Maximum surface area of the OOBB
cpOOBBMidArea	xs:double	Medium surface area of the OOBB
cpOOBBMinArea	xs:double	Minimal surface area of the OOBB
cpSurfaceArea_OOBB_LH_Max	xs:double	Surface parts of a 3D object that point in the corresponding direction of the side surfaces of the OOBB (deviation of the surface normal less than 45°). As long as the face portions are parallel to the sides of the OOBB, there is no difference between min and max.
cpSurfaceArea_OOBB_LH_Min	xs:double	
cpSurfaceArea_OOBB_HW_Max	xs:double	
cpSurfaceArea_OOBB_HW_Min	xs:double	
cpSurfaceArea_OOBB_LW_Max	xs:double	
cpSurfaceArea_OOBB_LW_Min	xs:double	

		These attributes only provide a value if the OOB is present for the 3D object and are not calculated for containers and composites.
cpSurfacePartAreaOOBB_LH_Max	xs:double	Largest/smallest parts of the surfaces of a 3D object pointing in the corresponding direction of the side faces of the OOB. If a surface is curved, only the portions of the surface with a deviation from the normal of +/- 5° are taken into account.
cpSurfacePartAreaOOBB_LH_Min	xs:double	
cpSurfacePartAreaOOBB_HW_Max	xs:double	
cpSurfacePartAreaOOBB_HW_Min	xs:double	
cpSurfacePartAreaOOBB_LW_Max	xs:double	
cpSurfacePartAreaOOBB_LW_Min	xs:double	
		These attributes only provide a value if the OOB is available for the 3D object and are not calculated for containers and composites.
cpSurfaceContactAreaOOBB_LH_Max	xs:double	Parts of the surfaces of a 3D object that are in contact with the corresponding side faces of the OOB within a tolerance range of 1 cm.
cpSurfaceContactAreaOOBB_LH_Min	xs:double	
cpSurfaceContactAreaOOBB_HW_Max	xs:double	
cpSurfaceContactAreaOOBB_HW_Min	xs:double	
cpSurfaceContactAreaOOBB_LW_Max	xs:double	
cpSurfaceContactAreaOOBB_LW_Min	xs:double	
		These attributes only provide a value if the OOB is available for the 3D object and are also calculated for composites but not for containers.

## 7. DESITE API

The DESITE BIM product line provides an API (Application Programming Interface) to enrich and to manipulate data and actions in the application. DESITE md and md pro offer different possibilities to take advantage from the API, from property and makro scripts up to WebForms that can run in DESITE custom and touch as well.

Moreover, they serve as an automation mechanism in a BIM-Project for property manipulation, camera control, objects query and more.

The supported programming language is JavaScript. The API is organized in 4 levels:

<b>Level 1: CoreAPI</b>	User Defined Properties	can handle property values only.
<b>Level 2: AutomationAPI</b>	Automation	provides function to change and augment models and their objects.
<b>Level 3: ProjectAPI</b>	Scripts to automate actions	can also get and set selection/visibility of objects, call viewpoints and material mappings. In this level navigation modes can be set and objects can be filtered.
<b>Level 4: NavigatorProject API and WebForm API</b>	Scripts embedded in the web forms	also provides signals to notify linked objects in a form about changes in selection and visibility objects in the 3D-model.

The individual levels have an inheritance relationship to each other. Thus, a higher level API has all the functions of all lower level APIs.

CoreAPI	AutomationAPI	ProjectAPI	NavigatorProjectAPI
CoreAPI	CoreAPI	CoreAPI	CoreAPI
	AutomationAPI	AutomationAPI	AutomationAPI
		ProjectAPI	ProjectAPI
			NavigatorProjectAPI

The following links<sup>2</sup> can serve as an introduction to the topic of JavaScript (JS) and HTML and can ease the handling of scripts and WebForms in DESITE:

- The website <https://www.w3schools.com/js/default.asp> offers up-to-date methods and approaches with JavaScript.
- Good examples of HTML, DOM, CSS and HTML with JS can be found on the website <https://www.w3schools.com>.

A detailed description of all available DESITE API functions can be found here:

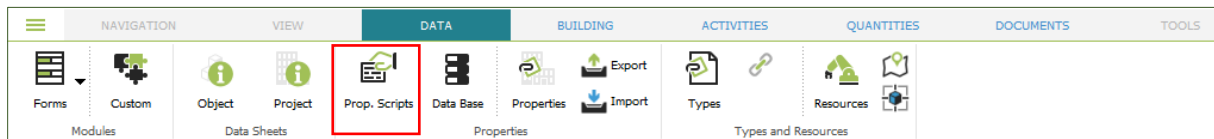
<https://bimdocs.thinkproject.com/desiteAPI/3.4/index.html>

The API documentation should match your DESITE release. You can find the latest API documentation in our Thinkproject Support Portal.

## 7.1 Property Scripts

Property Scripts allow you to define object properties based on a script that calculates the property value. The calculation is updated automatically, if the value of dependent properties has changed.

To create new Property Scripts, open the related dialog from the DATA ribbon:



<sup>2</sup> Disclaimer:

The mentioned links partly lead to external websites of third parties, on whose contents we have no influence. Therefore, we cannot assume any liability for these external contents. For the contents of the respective provider or operator of the pages is always responsible for the content of the linked pages. The linked pages were checked for possible legal violations at the time of linking. Illegal contents were not recognizable at the time of linking. However, a permanent control of the contents of the linked pages is not reasonable without concrete evidence of a violation of the law. In the case of infringements become known, we will remove such links immediately.

Property Scripts

Global Project Properties

Objects

Objects 1

Search in Properties ...

Name	Display Name	Unit	Datatype
1 Topology Floor 3	Topology Floor		xs:string

2

5

4

Topology Floor

Value

Foreground

Background

```

1 // created 2021-06-11T17:57:11
2 // by ek
3
4
5 var id = desiteThis.ID();
6 var n = desiteAPI.getPropertyvalue( id, 'cpRevitLevel', 'xs:string');
7 n.split('-')[0];
8

```

Evaluate ... ☒ Is Cached Clear Cache Calculate Cache ... ☒ Is Inheritable Apply

Activities

Documents

Scope of Work / Bill of Quantities

Building Structure

Process Components

Types / Property Sets

Resources

Issues / Viewpoints

Model Check

Clash Detection

Property Scripts are available for all Domains and also as Global Project Properties. To create a new Property Script:

1. Click on the related Domain Tab, e.g. 'Objects'
2. Click on the '+' button: A new row for the Property Script appears
3. Set a name for the Property Script and a data type. Display name and unit are optional.
4. Click on the row with the newly created Property Script and open the code area
5. With the 'Import' and 'Export' buttons, you can save your scripts and re-use them in other projects.

Now you can enter your JavaScript code that calculates the property values.

### Example 1:

The two properties 'Text1' and 'Text2' of type 'xs:string' should be combined according to this pattern:

The screenshot shows the 'Property Scripts' window with the 'Global Project Properties' tab. The 'Objects' table lists three properties: 'Topology Floor' (xs:string), 'PaintWallArea' (xs:double), and 'ConnectedText' (xs:string). The 'ConnectedText' script is selected, showing the following code:

```
1 // created 2023-07-11T14:42:21
2 // by JB
3
4 let id = desiteThis.ID();
5 let t1 = desiteAPI.getPropertyValue( id, 'Text1', 'xs:string' );
6 let t2 = desiteAPI.getPropertyValue( id, 'Text2', 'xs:string' );
7 t1 + '-' + t2; // The expression is the value of the property!
8
9
```

### Example 2:

This script calculates the wall area that needs to be painted. The objects for the wall surface were created with the **Contacts** tool and already contain the property 'cpLateralArea' for the area and a 'xs:boolean' value that indicates if this object has to be painted. The Property Script calculates the area, if the property 'paint' is set to 'true':

The screenshot shows the 'Property Scripts' window with the 'Global Project Properties' tab. The 'Objects' table lists four properties: 'PaintWallArea' (xs:double), 'test' (xs:double), 'Topology Floor' (xs:string), and 'cpLateralArea' (xs:double). The 'PaintWallArea' script is selected, showing the following code:

```
1 // created 2023-07-11T10:27:19
2 // by JB
3
4 let id = desiteThis.ID();
5 let area = desiteAPI.getPropertyValue( id, 'cpLateralArea', 'xs:double' );
6 let paint = desiteAPI.getPropertyValue( id, 'Paint', 'xs:boolean' );
7
8 if ( paint == true )
9 {
10     area;
11 }
12 else
13 {
14     0;
15 }
16
```

In addition to the value, the font color of the property and its background can also be set. In this case, the script must output a string at the end, which can be evaluated as a color based on HTML color codes in the form:

`'#xxxxxx'`

### Example:

'#ffffff'	white
'#000000'	black
'#ff0000'	red
'#00ff00'	green
'#0000ff'	blue
'#ffa500'	orange

This script defines foreground text color for the example 2 above:

The screenshot shows the 'Property Scripts' window for 'Global Project Properties'. It lists three objects: 'ConnectedText', 'PaintWallArea', and 'Topology Floor'. The 'PaintWallArea' object is selected, and its script is displayed in the 'Foreground' tab. The script is as follows:

```

1 // created 2023-07-11T10:27:19
2 // by JB
3
4 let id = desiteThis.ID();
5 let paint = desiteAPI.getPropertyvalue( id, 'Paint', 'xs:boolean' );
6
7 if( paint == true )
8 {
9   "#0000ff";
10 }
11 else
12 {
13   "#ff0000";
14 }

```

At the bottom of the window, there are buttons for 'Evaluate', 'Is Cached', 'Clear Cache', 'Calculate Cache', 'Is Inheritable' (checked), and 'Apply'.

The result looks like this:

Project Structure			
Name	cp:LateralArea [m2]	Paint	Wall area to be paint [m <sup>2</sup> ]
CP2 [map_01.desite.cp2] (1)			0,0000
CONTACTS (1)			0,0000
Contacts 2023-07-11T11...			223,5222
Contact Faces A-B (644)			0,0000
Contact [Basic Wall [... 46,9556		true	46,9556
Contact [Basic Wall [... 16,6889		true	16,6889
Contact [Basic Wall [... 9,4014		false	0,0000
Contact [Basic Wall [... 9,6764		true	9,6764
Contact [Basic Wall [... 9,7452		true	9,7452

## 7.2 Scripts

Scripts provide an easy way to perform routine tasks on the push of a button.

Open the Script dialog in the Tools ribbon bar:

Scripting

Scripts

Search in Scripts ...

Scripts

Risk Assignment

Floor Wiremode

Wallpaper Costs

Create Facility Management Selection Set

Run Clash Check

Wall paint

11 objects selected  
undefined

```

1 // created 2023-07-11T16:16:47
2 // by JB
3
4 let selection = "";
5 desiteAPI.clearSelection(false);
6
7
8 let elements = desiteAPI.getAllElements("geometry");
9 let selNumber = 0;
10
11 for (let i = 0 ; i < elements.length; i++)
12 {
13   let element = elements[i] ;
14   let sel = desiteAPI.getPropertyValue(element , "Paint", "xs:boolean");
15   if(sel)
16   {
17     selection += elements[i];
18     selection += ";";
19     selNumber++;
20   }
21 }
22
23
24
25 desiteAPI.selectElements(selection, true);
26 desiteAPI.zoomToSelected();
27 desiteAPI.showElementsOnly(selection);
28
29 console.log(selNumber + " objects selected");
30

```

Run Script

1. Show / hide list of scripts
2. Create a new script by clicking the '+' button. Delete a selected script with '-'. Re-order the scripts with the 'top' and 'down' arrows



3. Enter the JavaScript code for your script
4. The 'Console' shows outputs with *console.log()* as well as errors thrown by the JavaScript engine
5. Run a selected script with the 'run' button

Scripts are saved as *\*.js* files in the project directory under *[project name].scripts*.

### Example:

The Example 2 in **Property Scripts** shows how to calculate the wall area that needs to be painted based on a boolean property. The following script should now select all objects with this property set to 'true', isolate them and zoom to the selection:

```
// created 2023-07-11T16:16:47
// by JB

let selection = "";

//Clear the current selection of 3d objects:
desiteAPI.clearSelection(false);

//Get all objects for the Geometry Domain
let elements = desiteAPI.getAllElements("geometry");
let selNumber = 0;

//Iterate over all items in 'elements' and retrieve the value of the property
'Paint'. Please note that for the correct query of the value, the data type
must always be specified in addition to the name of the property.
for (let i = 0 ; i < elements.length; i++)
{
    let element = elements[i] ;
    let sel = desiteAPI.getPropertyValue(element , "Paint", "xs:boolean");

    //If the value of 'Paint' is true, the ID of the object is stored in
    'selection'.
    if(sel)
    {
        selection += elements[i];
        selection += ";";
        selNumber++;
    }
}

//Now select all objects in 'selection' and zoom to all selected objects.
desiteAPI.selectElements(selection, true);
desiteAPI.zoomToSelected();

//Isolate all objects in 'selection'
desiteAPI.showElementsOnly(selection);

//Write number of selected objects to console
console.log(selNumber + " objects selected");
```

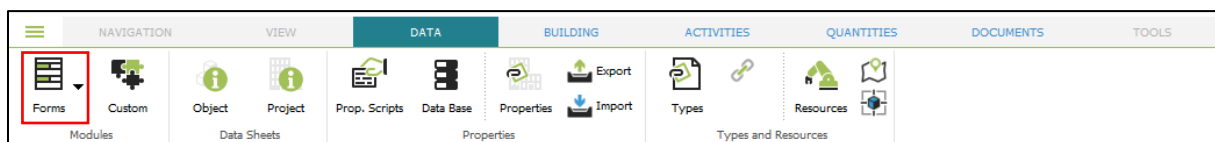
## 7.3 WebForms

WebForms provide the most powerful customization functionality of DESITE using the DESITE API. WebForms are custom UI elements (dialogs) created with HTML and programmed with JavaScript.

To work with WebForms in DESITE md, a HTML file named *DbInputForm.html* is loaded from the project folder. The project folder is the directory in which the project file with the file extension \*.pfs is located (see also **Project File Structure**).

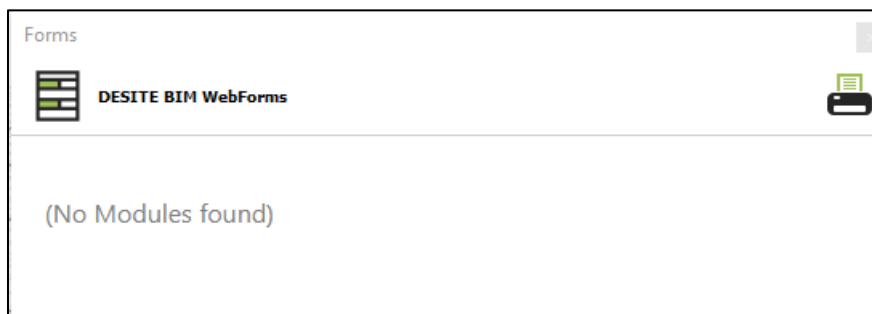
When starting a new project, the project (even if it is still empty) must first be saved so that a folder with the appropriate directories and a prepared *DbInputForm.html* is created.

Open the WebForm menu from the Data ribbon bar:



The *DbInputForm.html* represents the start page. If several WebForms are needed, it is possible to link from the start page to further pages.

The WebForm dialog appears and loads the *DbInputForm.html* start page. The WebForms dialog embeds a Chromium browser to display HTML files:

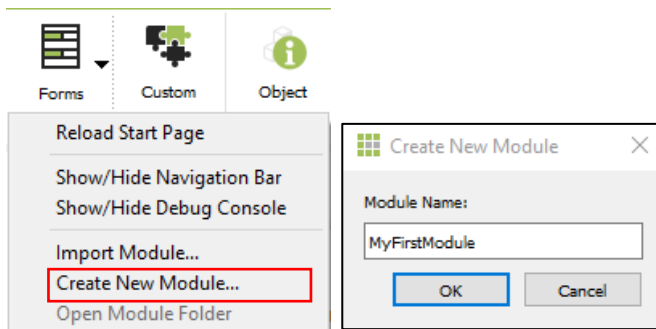


### Tip:

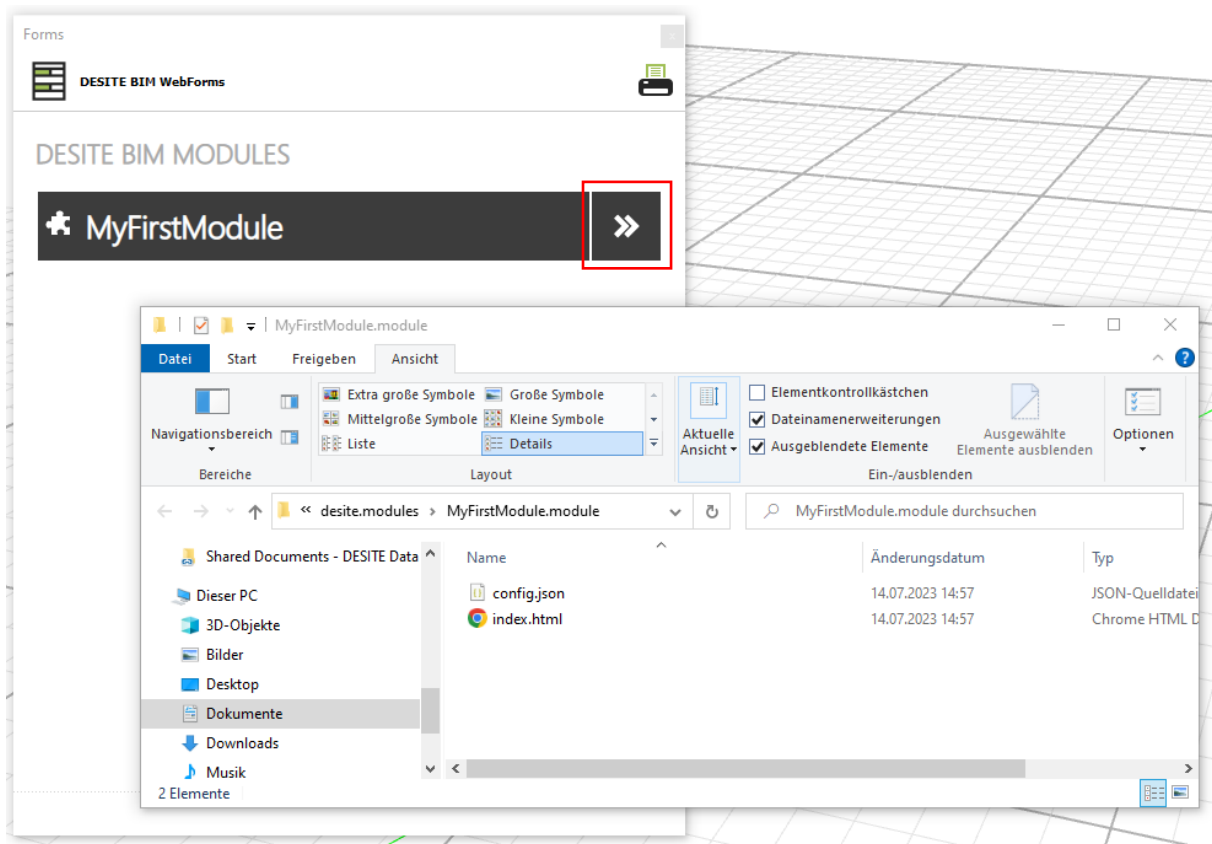
It's not strictly required that the WebForm entry page is named *DbInputForm.html*. This is just the default name, if new project is saved. You can also define other files or URLs as the default entry page and even structure according to your needs.

### 7.3.1 Creating own WebForm Modules

If you start from a fresh project, the start page is empty and shows no Module. To create a new Module, open the Forms drop down, select the appropriate function and enter a name for the new Module:

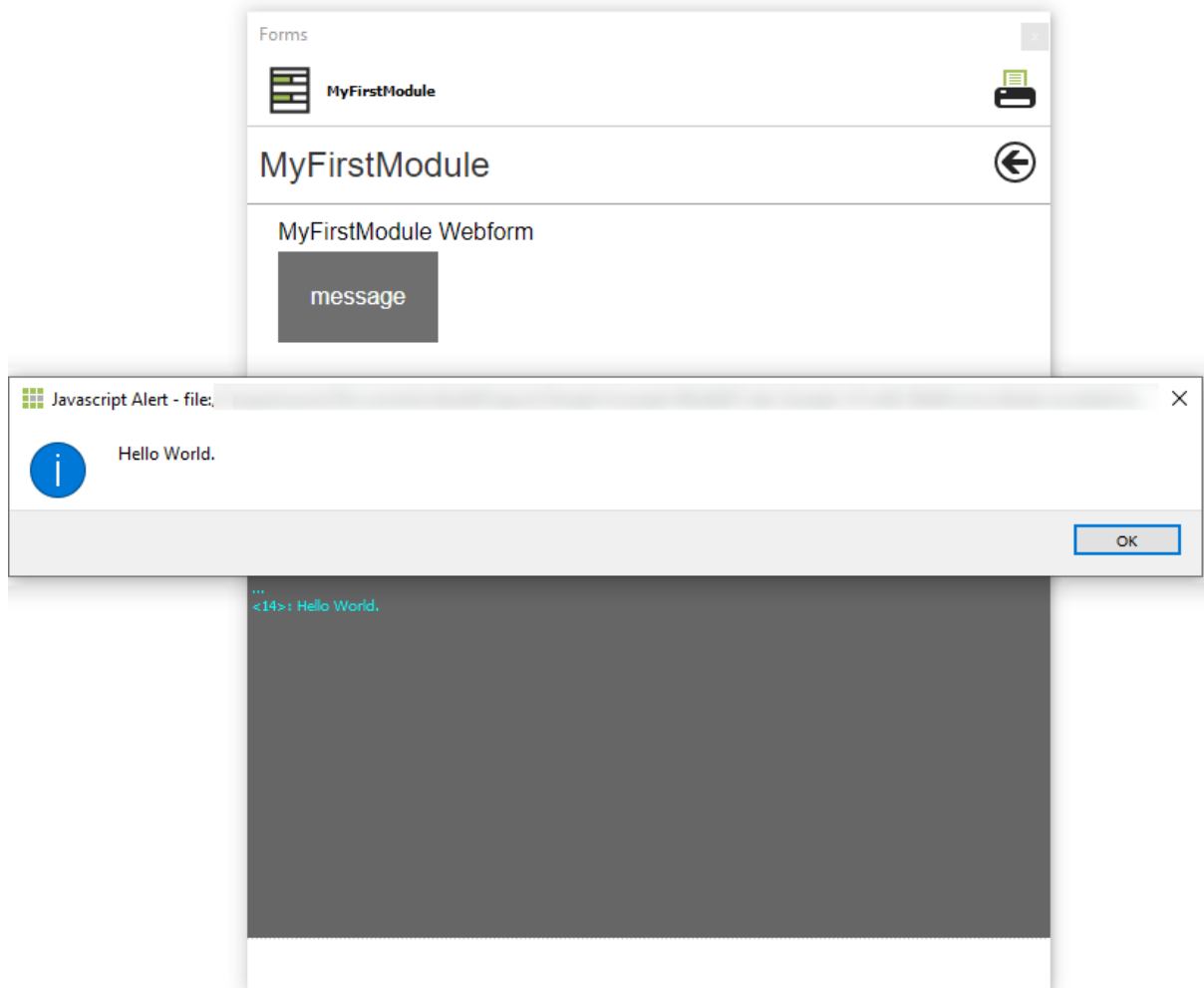


DESITE creates now a sub-folder for the new Module, including the *Index.html* file and the *config.js* file. The related Module folder is automatically opened in the Windows Explorer and the new Module is loaded into the start page:



To start the Module, click on the double arrow on the right.

Congratulations, you created your first DESITE WebForm:



The *desite.modules* folder contains also two subfolders with general resources for own WebForms:

- **css/desiteStyleLime.css:** This is the default stylesheet for new DESITE modules. You can edit this file to your own needs
- **img/[\*.svg]:** This folder includes a few images in the svg format, that can be used in the WebForms.

To edit the Module, open the *Index.html* in a text editor and implement your own functionality.

Let's take a look into this first module:

```

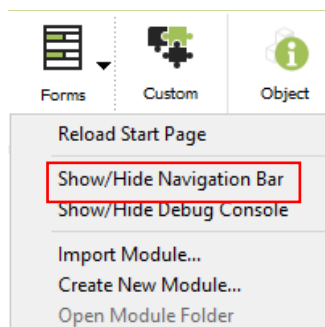
<? index.html > ...
1  <!DOCTYPE html>
2  <html Lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="content-type" content="text/html; charset=UTF-8"/>
6      <link href="../css/desiteStyleLime.css" rel="stylesheet" type="text/css"/>
7      <title>MyFirstModule</title>
8
9      <script type="text/javascript">
10
11          function debugAndAlert(text)
12          {
13              desiteMD.showConsole(true);
14              console.log(text);
15              alert(text);
16          }
17
18      </script>
19
20  </head>
21  <body style="margin-top: 50px">
22      <table class="tableRibbon">
23          <tr style="color: #373636;">
24              <td style="font-size: 24px;">MyFirstModule</td>
25              <td style="text-align:right" width="36">
26                  <a href="javascript:desiteMD.reloadPage()"></a>
27              </td>
28          </tr>
29      </table>
30      <table>
31          <tr>
32              <td>MyFirstModule WebForm</td>
33          </tr>
34          <tr>
35              <td>
36                  <button class="button" onClick="debugAndAlert('Hello World.')">message</button>
37              </td>
38          </tr>
39      </table>
40  </body>
41  </html>

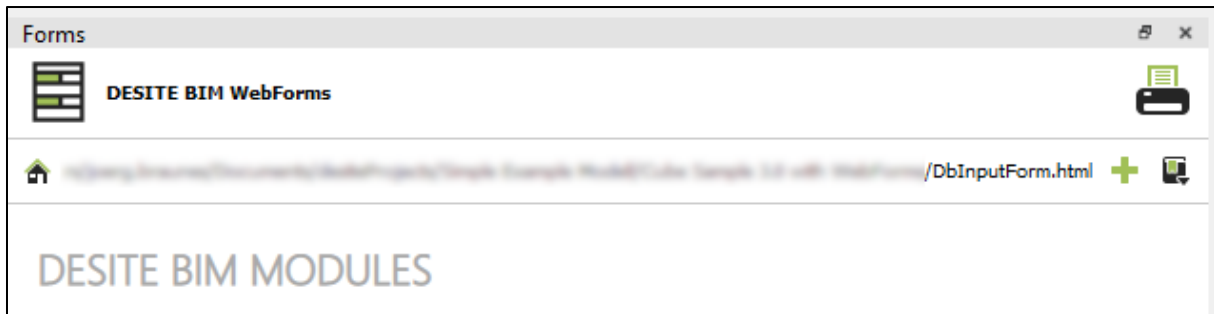
```

1. The header defines the title of the Module and refers to the generic DESITE stylesheet.
2. The embedded script includes the function *debugAndAlert()* that already uses the DESITE API to activate the Console area of the WebForm dialog and it shows an 'alert' message box with the value of the attribute 'text'.
3. The body of the page includes two tables. The first table presents the title of the module and a button to navigate back to the start page with the function *desiteMD.reloadPage()*. The second table included a button that calls the function *debugAndAlert()* with the passed text 'Hello World.'.

### 7.3.2 Navigation Bar

You can show/hide the navigation bar of the embedded chromium browser from the 'Forms' drop down menu:





Home button. This loads the DpInputForm.html per default (this can be changed in the bookmarks list)



Saves the currently loaded URL as a bookmark



List of all saved bookmarks. With a right click on a bookmark, you can rename, delete or save the bookmark as the start page

#### Tip:

It is also possible to store WebForms online as a web application. This requires an active internet connection. This is especially useful, if the WebForm is used across projects.

### 7.3.3 Entry points

The 'load' event of the HTML body does not automatically initialize the DESITE API. On successful initialization a custom 'desiteload' event will be sent that can be handled with an event listener.

```
addEventListener('desiteload', function(event)
{
    /* do your initialization here */
});
```

#### Example: Responding to selection changes

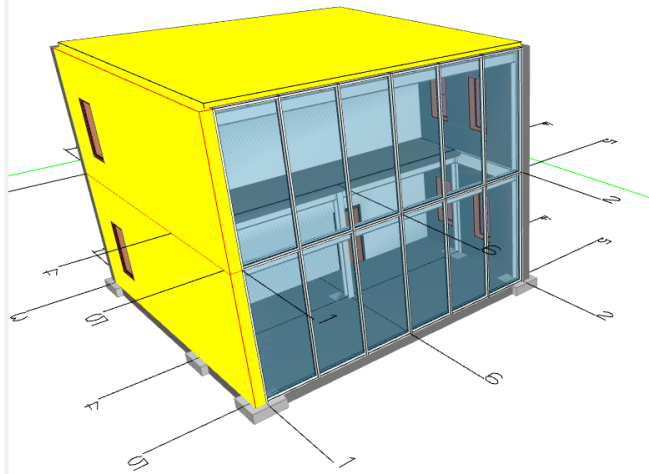
The following example uses the *selectionChanged* signal to return the actual number of selected 3d objects and list the names and the volume in a dynamic table:

Forms

SelectionTableAndPrint

SelectionTableAndPrint

Element name	Element volume	Unit
Basic Roof EPDM Membrane on Rigid Insul on Metal Deck 429437	11.744545493476616	m³
Basic Wall Generic - 300mm:299077	11.407657819553961	m³
Basic Wall Generic - 300mm:299519	11.40765781955398	m³
Basic Roof EPDM Membrane on Rigid Insul on Metal Deck 429437	11.744545493476616	m³



```

< index.html > ...
1  <!DOCTYPE html>
2  <html Lang="en">
3    <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="content-type" content="text/html; charset=UTF-8"/>
6      <link href="../css/desiteStyleLime.css" rel="stylesheet" type="text/css"/>
7      <title>SelectionTableAndPrint</title>
8
9      <script type="text/javascript">
10
11        var selectedElements = [];
12
13        //add the 'desiteLoad' event to the event listener:
14        addEventListener('desiteLoad', async function (event) {
15          await desiteAPI.selectionChanged.connect(onSelectionChanged);
16        });
17
18        //this function is called if the selection changes in the project
19        async function onSelectionChanged()
20        {
21          selectedElements = await desiteAPI.getSelectedElements('geometry');
22          await setupTableCurrentSelection(selectedElements);
23        }
24

```

1. The 'desiteLoad' event is added to the event listener with the function *desite.API.selectionChanged.connect()*, to react in any case the user selects or deselects an element in DESITE.
2. The function *onSelectionChanged()* is called from the related event. With *desiteAPI.getSelectedElements('geometry')* the IDs of all currently selected 3d objects can be retrieved and passed over to the function *setupTableCurrentSelection()*.

```

25 //add the information of the selected object into a table
26 async function setupTableCurrentSelection(idList)
27 {
28     let table = document.getElementById("myTable");
29
30     //with every selection change, let's clear the table first (except the header row)
31     for(i=1; i < table.rows.length; i++)
32     {
33         table.deleteRow(i);
34     }
35
36     //now iterate over all selected objects and write the name and volume to new table rows
37     for (i=0; i < idList.length; i++)
38     {
39
40         let name = await desiteAPI.getPropertyValue(idList[i], 'cpName', 'xs:string');
41         let volume = await desiteAPI.getPropertyValue(idList[i], 'cpVolume', 'xs:double');
42
43         let row = table.insertRow();
44         row.insertCell(0).innerHTML = name;
45         row.insertCell(1).innerHTML = volume;
46         row.insertCell(2).innerHTML = "m³";
47     }
48 }
49
50 </script>
51
52 </head>
53 <body style="margin-top: 50px">
54     <table class="tableRibbon">
55         <tr style="color: ■ #373636;">
56             <td style="font-size: 24px;">SelectionTableAndPrint</td>
57             <td style="text-align:right" width="36">
58                 <a href="javascript:desiteMD.reloadPage()"></a>
59             </td>
60         </tr>
61     </table>
62
63     <table id="myTable" width="100%" cellpadding="2" cellspacing="4" border="1">
64         <tr align="left">
65             <th scope="col">Element name</th>
66             <th scope="col">Element volume</th>
67             <th scope="col">Unit</th>
68         </tr>
69     </table>
70 </body>
71 </html>

```

3. The function *setupTableCurrentSelection()* iterates over all selected elements and retrieves the values for 'cpName' and 'cpVolume' and write them into the cells of the table with the id 'myTable'
4. The body of the HTML file contains the table 'myTable' that is filled by the JavaScript function *setupTableCurrentSelection()*.

### 7.3.4 Asynchronous API calls

Since version 3.0 of DESITE, API calls in WebForms are no longer handled synchronously, but instead are handled asynchronously. In JavaScript a function must be defined with the keyword 'async' in front. An asynchronous function will be executed in parallel with the calling function, so it is no longer guaranteed that all operations are handled in consecutive order. The following segment will prepare the user for handling asynchronous functions and their return parameters.

#### Completion time is not relevant:

The simplest case is when the time of completion is not relevant for the calling function, in that case the asynchronous call can be handled just like before. It must be considered that e.g., an asynchronous call to 'showAll' with a follow-up call to 'getVisibleElements' where the first would manipulate the state of the second could lead to unexpected behavior.



```
function run()
{
    desiteAPI.showAll(true, 'geometry');
    desiteAPI.showAll(true, 'documents');
}
```

### Wait for one/multiple promises:

It is possible to handle asynchronous function calls in a synchronous way. JavaScript introduces the so-called promise object. It is returned when an asynchronous function is called and replaces the regular return value. A promise represents a running operation, in conjunction with the *await*-operator, the user can now wait for a function to complete. The usage of the *await*-operator itself is only possible in an asynchronous function, otherwise the execution would stall the running process. That's why the calling function must be declared with the *'async'* keyword.

```
async function waitForOnePromise()
{
    ids = await desiteAPI.getVisibleElements();
    for(var i = 0; i < ids.length; i++)
    {
        var obj = await desiteAPI.getAsJSON(ids[i]);
        /* or explicit */
        var promise = desiteAPI.getAsJSON(ids[i]);
        var obj = await promise;
    }
}
```

Besides waiting for a single promise, it is also possible to wait for multiple promises at once. This is beneficial and will reduce the overall time of the operation. The user has to collect all promises in an array and afterwards pass it on to the *await*-operator. The result of that operation will be a new array with the original function return values.

```
async function waitForAllPromises()
{
    ids = await desiteAPI.getVisibleElements();
    var promises = [];
    for(var i = 0; i < ids.length; i++)
    {
        promises.push(desiteAPI.getAsJSON(ids[i]));
    }
    var objs = await Promise.all(promises);
}
```

### Callbacks:

Another option to handle asynchronous calls are callback functions. The callback function will be passed into an asynchronous function as an additional parameter and will be called on function completion. The callback functions will pass the original return value through its parameter list. The calling function doesn't have to wait for the asynchronous call and can continue with its work, the 'async' keyword is not necessary.

```
function runWithCallback()
{
    desiteAPI.getVisibleElements(function(ids)
    {
        for(var i = 0; i < ids.length; i++)
        {
            desiteAPI.getAsJSON(ids[i], function(obj)
            {
                /* do something with object */
            });
        }
    });
}
```

In summary, it is always advantageous to **NOT** wait for single API calls and use callback functions instead. Alternatively, if an operation depends on former results and the user wants to handle it in a synchronous way, waiting for as many operations as possible at once is preferred and will reduce the overall time of the process.

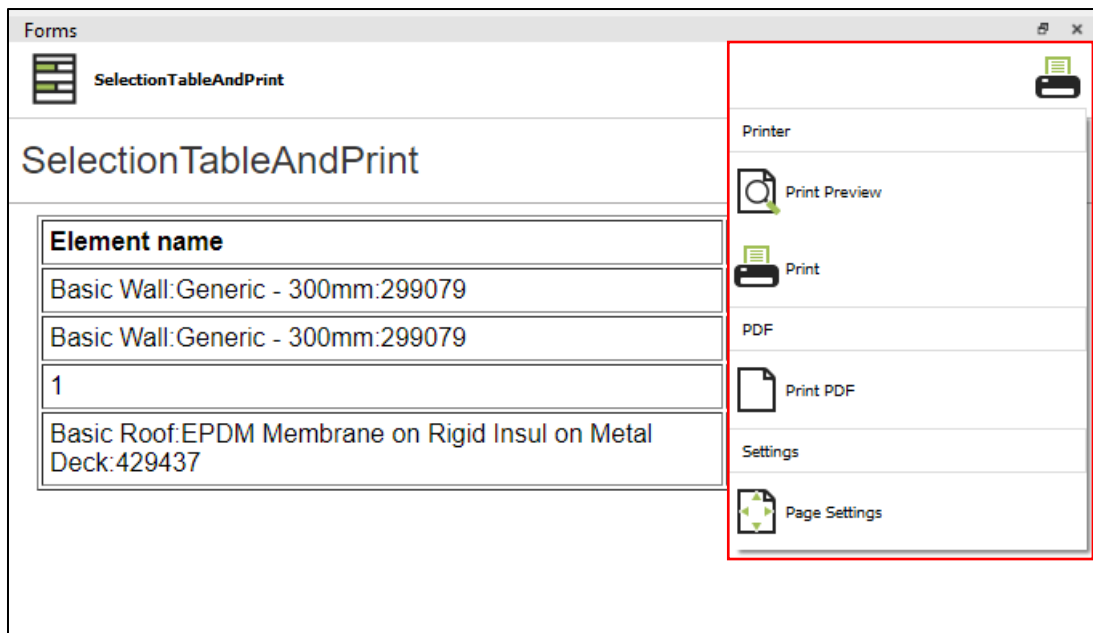
### API functions with an internal state

Extra precautions have to be taken when using API functions with an internal state.

For example, when you open a file via *openFile()*, write multiple chunks of data via *writeToFile()*, and then close the file via *closeFile()*. Since the calls must be made in exactly this sequence, WebForm implementers have to make sure that they are not executed asynchronously, and therefore out of order.

### 7.3.5 Printing of WebForms

The WebForm dialog itself provides a print function to print the current content of the WebForm:



Beside this standard functionality, it is also possible to define an own print to PDF function in the WebForm by using the function `desiteMD.createPdfByTemplate()`. This function requires a template as html file that defines the style of the PDF.

### Example: Printing the selection table from the chapter **Entry points**

Print template `'SelectionTable.printTemplate.html'`:

```

1  <html>
2  <head>
3      <style>
4          table, th, td {
5              width: 100%;
6              text-align: left;
7              color: darkslateblue;
8              border-collapse: collapse;
9              border: 1px solid #b1aeae;
10             padding: 12px;
11         }
12     </style>
13 </head>
14 <body>
15 <h1>Selected Elements</h1><br>
16
17 <p><b>The following paragraph is created from Text Blocks:</b></p>
18 <br><br>
19
20 [[printTextBlock##TableSelection]] <!-- This attribute is exchanged by the definition in the printTable() function -->
21
22 <br><br>
23 <p>This is the end of the text block definition</p>
24
25 </body>
26 </html>

```

1. Define the style of the elements on the html with css definitions
2. The attribute `[[printTextBlock##TableSelection]]` will be later replaces with the contend from the WebForm. You can define as much as attributes as you need.

To use this template, the `'Index.html'` of your WebForms requires an appropriate 'print' function and a button to call this function:

```

50 //Function to print the Selection Table
51 async function printTable()
52 {
53     desiteMD.setPageMargins(10,10,10,10);
54     desiteMD.setOrientation(0) ;
55     desiteMD.setPaperSize( 0 );
56
57     var printBlock = {TableSelection: '<table>'}
58
59     printBlock.TableSelection += document.getElementById("myTable").innerHTML;
60
61     printBlock.TableSelection += "</table>";
62
63     var fn = await desiteMD.createPdfByTemplate('SelectionTable.printTemplate.html','', printBlock ,true);
64
65     if( fn.includes('Error') )
66     {
67         console.log('could not print to PDF');
68     } else
69     {
70         console.log('Printed!');
71     }
72 }
73
74

```

1. These functions define the paper size, orientation and margins for the resulting PDF
2. In the object '*printBlock*' the actual content is defined, which will later be inserted into the attributes in the html template. In this example, the string value for the attribute '*TableSelection*' is filled with the html description for a table. The content for each row is read from the table with id '*myTable*' that is already defined in the body area of the '*index.html*'.
3. The prepared object with the content is now passed into function *desiteMD.createPdfByTemplate()* together with the prepared template '*SelectionTable.printTemplate.html*' to create the final PDF.

```

79 <body style="margin-top: 50px">
80 <table class="tableRibbon">
81 <tr style="color: #373636;">
82 <td style="font-size: 24px;">SelectionTableAndPrint</td>
83 <td style="text-align:right" width="36" >
84 
85 </td>
86 <td style="text-align:right" width="36">
87 <a href="javascript:desiteMD.reloadPage()"></a>
88 </td>
89 </tr>
90 </table>
91 <table id="myTable" width="100%" cellpadding="4" border="1">
92 <tr align="left">
93 <th scope="col">Element name</th>
94 <th scope="col">Element volume</th>
95 <th scope="col">Unit</th>
96 </tr>
97 </table>
98
99 </body>



```

4. The button to start the print is added to the WebForm with the '*onclick*' event that calls our function '*printTable()*'

Forms

SelectionTableAndPrint

SelectionTableAndPrint

Element name	Element volume	Unit
Basic Wall:Generic - 300mm:299079	10.260659161480497	m <sup>3</sup>
Basic Wall:Generic - 300mm:299079	10.260659161480497	m <sup>3</sup>
1	402.1426223628416	m <sup>3</sup>
Basic Roof:EPDM Membrane on Rigid Insul on Metal Deck:429437	11.744545493476616	m <sup>3</sup>

And this is the resulting PDF following the style defined in the html template:

## Selected Elements

The following paragraph is created from Text Blocks:

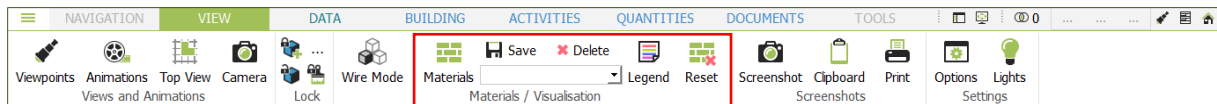
Element name	Element volume	Unit
Basic Wall:Generic - 300mm:299079	10.260659161480497	m <sup>3</sup>
Basic Wall:Generic - 300mm:299079	10.260659161480497	m <sup>3</sup>
1	402.1426223628416	m <sup>3</sup>
Basic Roof:EPDM Membrane on Rigid Insul on Metal Deck:429437	11.744545493476616	m <sup>3</sup>

This is the end of the text block definition

## 8. Materials and Colors

### 8.1 Materials

Material information for model objects can be managed and expanded via Materials tool under View menu tab.



Open the Material dialogue

*Drop  
Down list*

Select the active Color scheme



Save current Materials in a new Color scheme



Delete the active Color scheme



Turn on/off the legend for the current Color scheme in 3D View



Resets the displayed Materials in the 3D View to the Materials stored in the imported model files

In the 'Default' setting, 16 materials with associated colors are generated in DESITE md. In addition, materials that are provided in the uploaded partial models are also imported into the project.

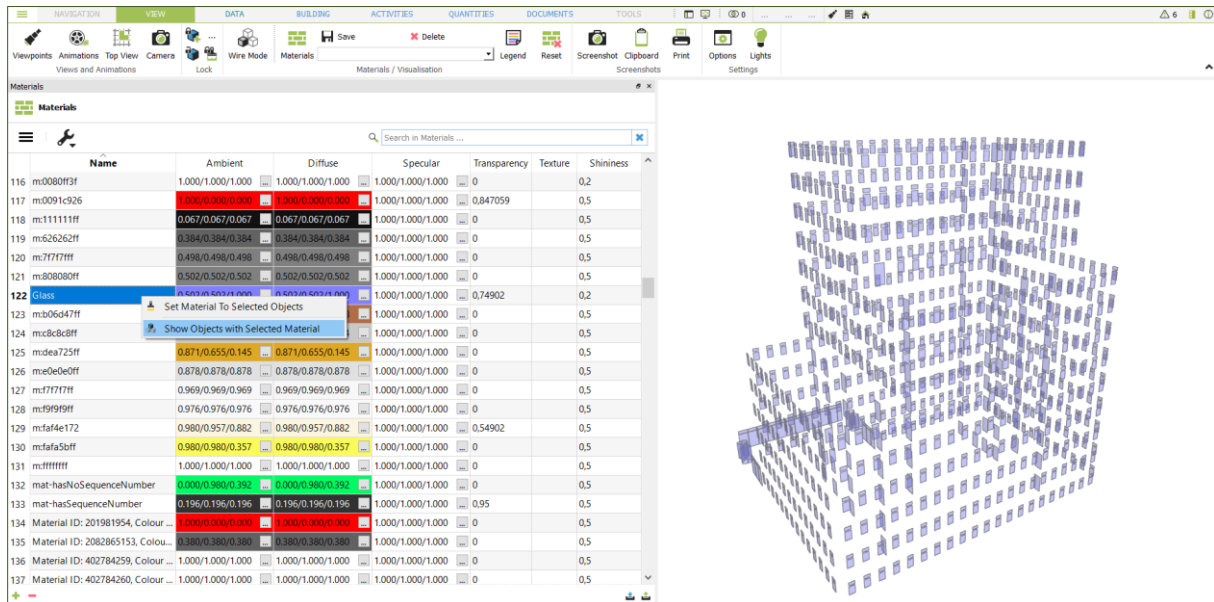
Materials							
Materials							
	Name	Ambient	Diffuse	Specular	Transparency	Texture	Shininess
1	default-mat-01	0.353/0.353/0.549	0.353/0.353/0.549	1.000/1.000/1.000	0		0.5
2	default-mat-02	Set Material To Selected Objects		1.000/1.000/1.000	0		0.5
3	default-mat-03	Show Objects with Selected Material		1.000/1.000/1.000	0		0.5
4	default-mat-04	0.824/0.624/0.373	0.824/0.624/0.373	1.000/1.000/1.000	0		0.5
5	default-mat-05	0.000/0.502/0.753	0.000/0.502/0.753	1.000/1.000/1.000	0		0.5
6	default-mat-06	0.722/0.698/0.494	0.722/0.698/0.494	1.000/1.000/1.000	0		0.5
7	default-mat-07	0.094/0.698/0.886	0.094/0.698/0.886	1.000/1.000/1.000	0		0.5
8	default-mat-08	0.878/0.306/1.000	0.878/0.306/1.000	1.000/1.000/1.000	0		0.5
9	default-mat-09	0.392/0.392/0.494	0.392/0.392/0.494	1.000/1.000/1.000	0		0.5
10	default-mat-10	0.047/0.392/0.494	0.047/0.392/0.494	1.000/1.000/1.000	0		0.5
11	default-mat-11	0.549/0.588/0.157	0.549/0.588/0.157	1.000/1.000/1.000	0		0.5
12	default-mat-12	0.118/0.549/0.392	0.118/0.549/0.392	1.000/1.000/1.000	0		0.5
13	default-mat-13	0.996/0.306/0.494	0.996/0.306/0.494	1.000/1.000/1.000	0		0.5
14	default-mat-14	0.878/0.698/1.000	0.878/0.698/1.000	1.000/1.000/1.000	0		0.5
15	default-mat-15	0.878/0.306/0.494	0.878/0.306/0.494	1.000/1.000/1.000	0		0.5
16	default-mat-16	0.486/0.502/0.494	0.486/0.502/0.494	1.000/1.000/1.000	0		0.5

You can customize material properties such as name, color or transparency.

Using '+' button at the bottom left corner, you can create your own, user-defined materials. By right-click on a material in the list allows you to:

- Set the material to selected objects
- To show all objects with the material assigned to by filtering in 3D View.

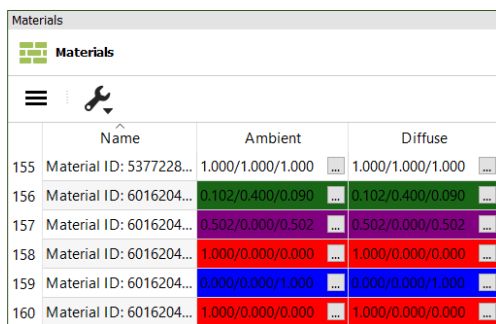
In the following example, only elements assigned to material property Glass are displayed in 3D view.



## 8.2 Color schemes

You can organize multiple materials for one object with Color scheme s. Depending on the selected Scheme, the material assigned to this Scheme is shown in the 3D View.

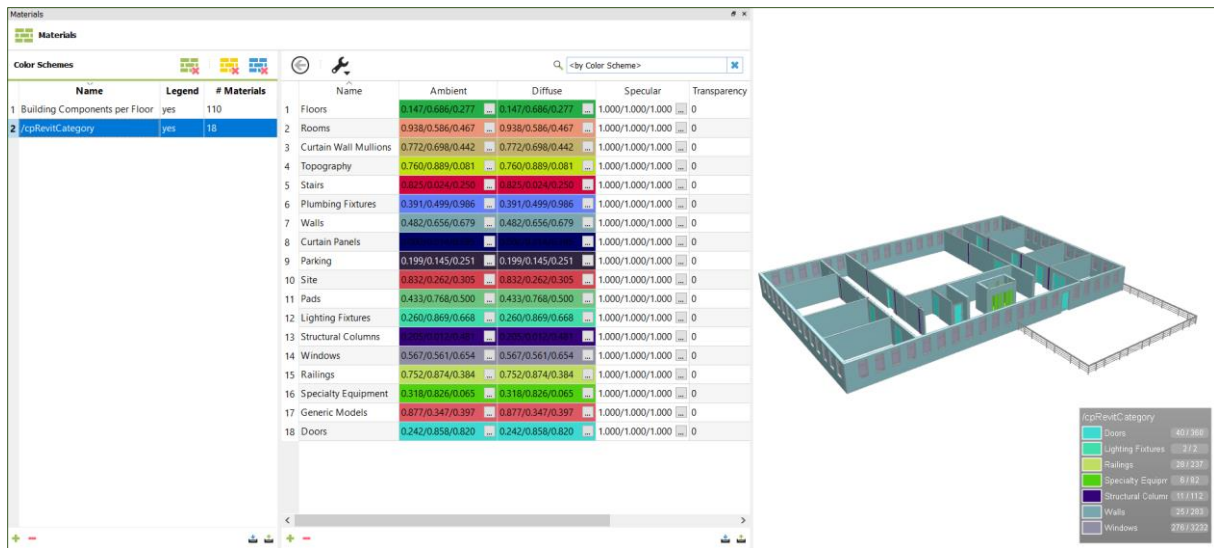
Click on Show/Hide Color scheme list icon in order to create a Color scheme .



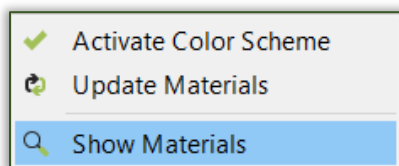
To create a new Color scheme :

1. Select the desired objects in 3D view
2. Assign a material to the selected objects (right-click on the material)
3. Repeat steps 1 and 2 to assign other materials to other objects
4. Create a new scheme using '+' button at the bottom left corner in Color scheme s window

The newly created Color scheme contains now all the materials that were assigned to the objects in the 3D View.



In order to customize a Color scheme, right-click on a Color scheme and select Activate Color scheme. Equip the objects with new materials in the 3D view and then update the Color scheme with right-click and the 'Update Materials' option.



With the option 'Show Materials', you can filter the Materials list to display only materials that belong to selected Color scheme.

While working with the Color scheme dialogue you have following additional options:



Resets the Materials in the 3D View



Resets the Materials of the selected objects in the 3D View

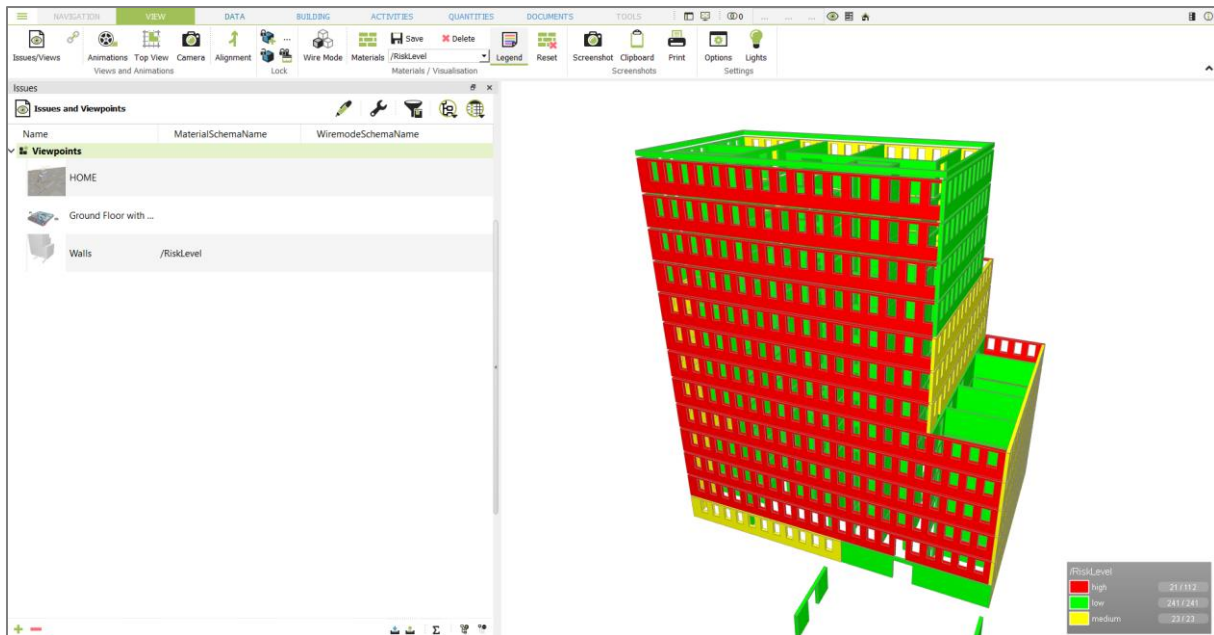


Resets the Materials of the visible objects in the 3D View

**Tip:** As an alternative to creating a color scheme manually, it is also possible to create based on **Selection Sets**.

In order to associate a 3D Viewpoint with a Color scheme, add the name of the corresponding Color scheme into the 'MaterialSchemeName' property in the **Issues and Viewpoints** domain:

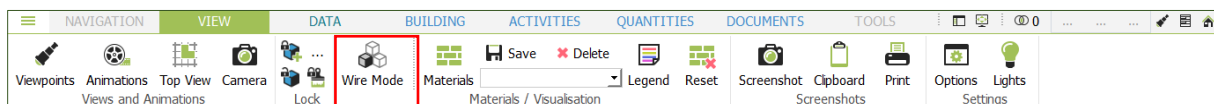




When the 3D viewpoint is applied, the associated color scheme will also be applied. It will remain active until you deactivate it again.

### 8.3 Wire Mode Schemes

In a similar way to color scheme, a scheme for wireframe representation of model elements can also be created with Wire Mode function under View menu tab.



With wire mode scheme selected elements will be displayed in transparent only with edges, while rest of model objects are displayed in standard colors.

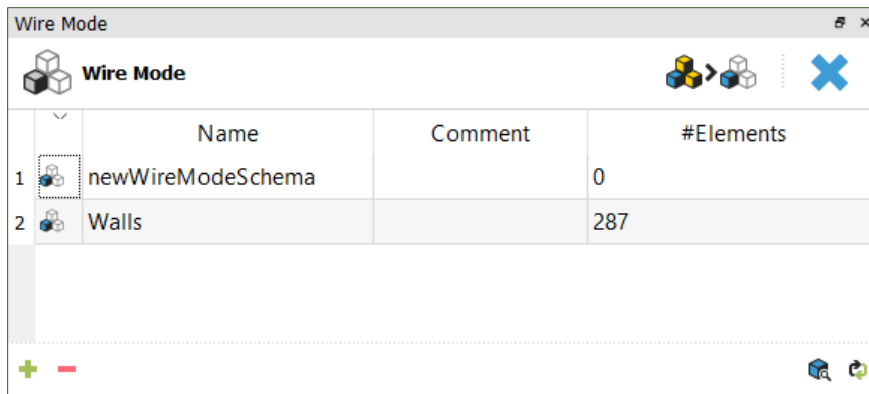


Using Show Selected Objects in Wire Mode. Only selected elements will be displayed with wireframes without working with a specific scheme.



Reset Wire mode.

To Create a new wire mode schema, click on the '+' button at the bottom left corner of the wire mode dialogue.

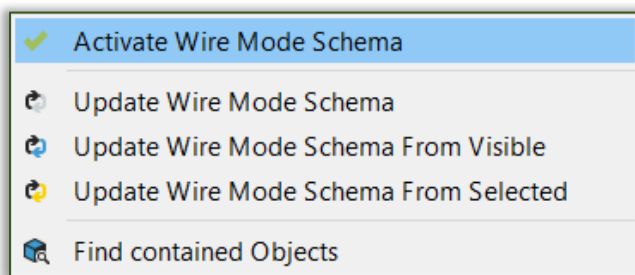


The 'Wire Mode' window displays a table with the following data:

	Name	Comment	#Elements
1	newWireModeSchema		0
2	Walls		287

At the bottom of the window, there are icons for adding (+) and removing (-) schemas, and a search icon.

Similar to color scheme, you can activate or update a wire mode scheme by right-click.



**Tip:** These wire mode schemes can also be associated to a viewpoint. For that, type the name of the corresponding wire mode scheme into the 'WiremodeSchemeName' property in the **Issues and Viewpoints** domain.

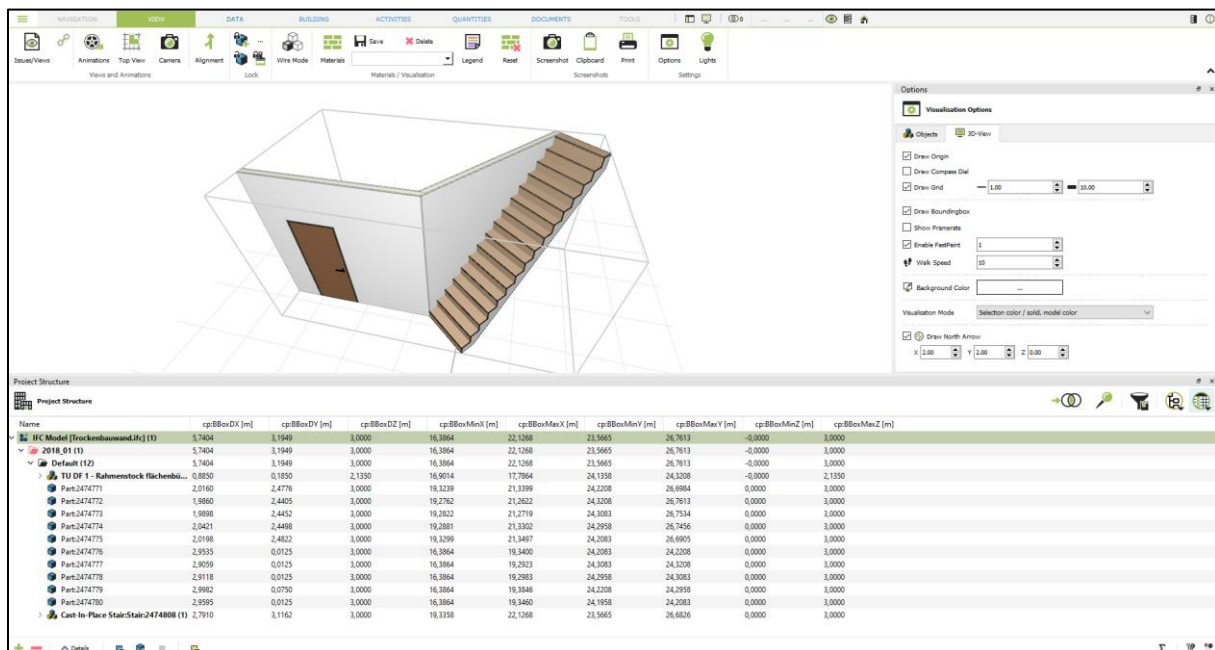
## 9. Tools

### 9.1 Tool Functions

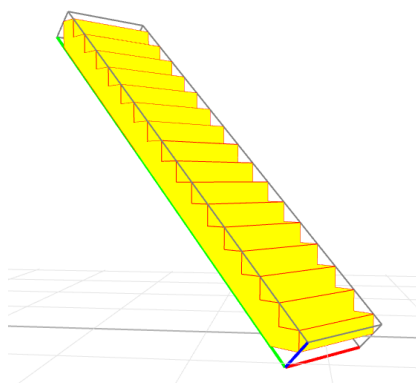
#### 9.1.1 Optimized Oriented Bounding Box

A bounding box is the minimal or smallest cube volume that encloses a 3d object completely aligned to the x, y and z-axis of the model coordinate system. In DESITE a bounding box is used to calculate different geometric properties (see also **Properties for Geometry domain**).

The following screenshot shows a model of two walls, a door and a stair. In the **Geometry Domain / Project Structure** the **DESITE cp-Properties** of the bounding box ('cpBB...') are shown and in the view settings (see **Selection and Visibility**) the option 'Draw Bounding Box' is enabled to show the model bounding box:

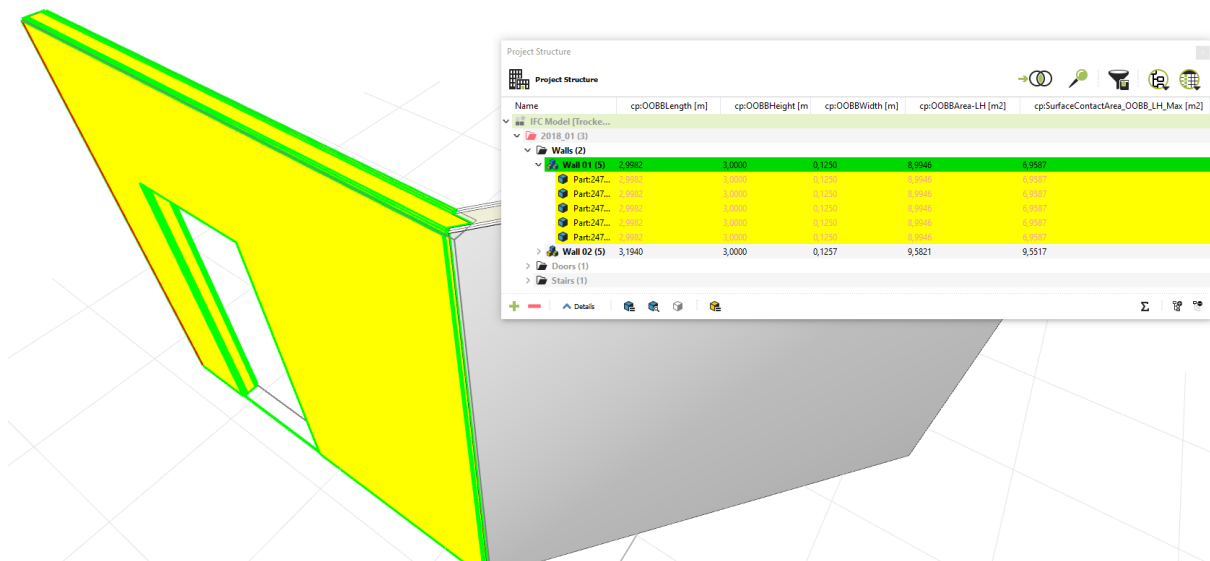


The Optimized Oriented Bounding Box (OOBB) is the smallest volume rotated in a way, so that it's encloses a 3d object in its minimum:



## Example for OOB properties

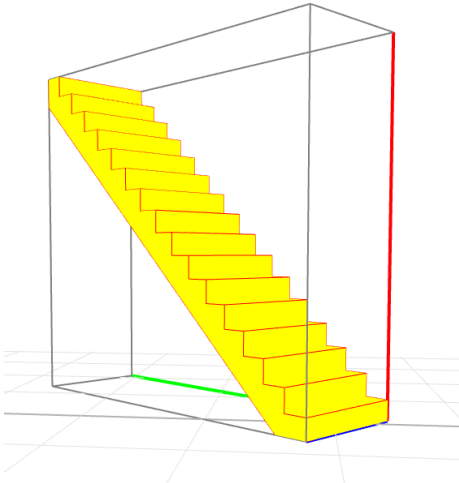
(for a complete overview see: [Properties for Geometry domain](#)):



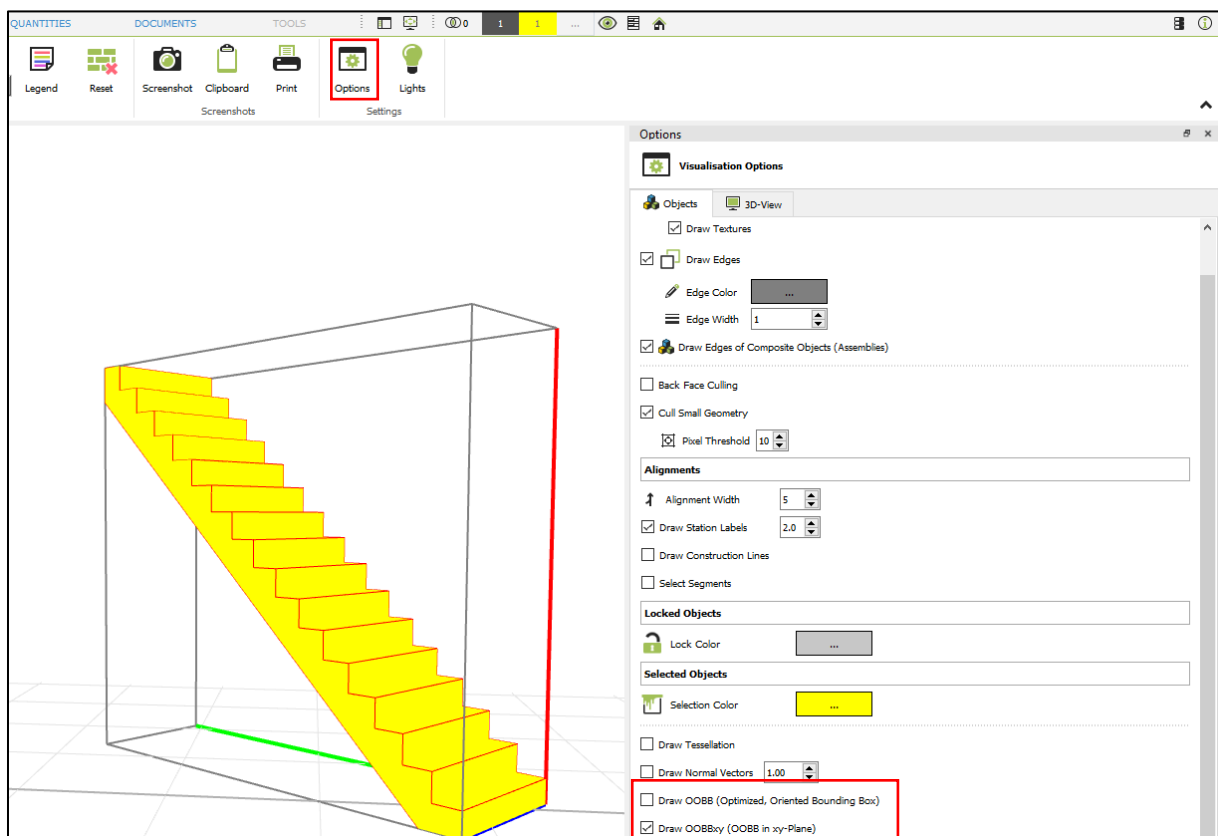
The example shows a wall that is modelled as a 'Composite' with the different layers as single 3d objects. The OOB was calculated for the 'Composite'.

- **cp:OOBHeight:** height of the OOB. This is always the vector closest to the z-axis of the model coordinate system.
- **cp:OOBLength:** length of the OOB. This is always the longer of the two remaining vectors.
- **cp:OOBWidth:** width of the OOB. This is always the shorter of the two remaining vectors.
- **cp:OOBArea-LH:** the area calculated from the length (L) and height (H) value
- **cp:SurfaceContactArea\_OOB\_LH\_Max:** Parts of the surfaces of a 3D object that are in contact with the corresponding side faces of the OOB within a tolerance range of 1 cm. In this case the maximal area calculated from length and height.

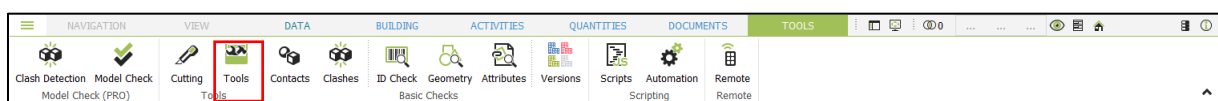
In some cases an OOB is needed that is parallel to the xy-plane of the model coordinate system, the OOBxy:

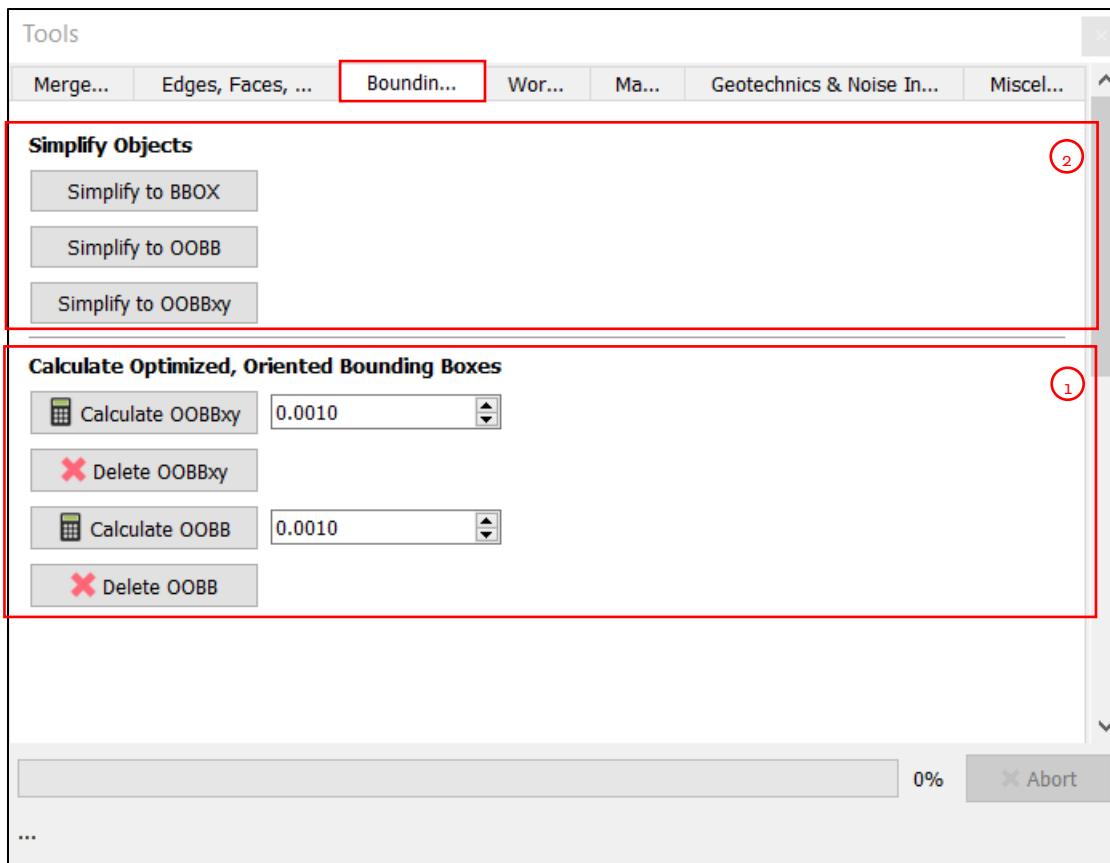


Per default, the OOB is not drawn for an object. You can enable this in the View Settings (see also **Selection and Visibility**) for selected objects:



The OOB and OOBxy are not calculated automatically. If the properties of the OOBs are needed, you can calculate them under Tools → Tools:





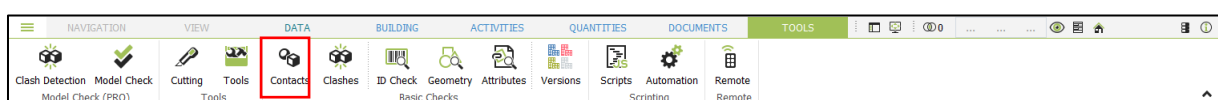
1. With these options the OOBb and OOBbxy can be calculates (and deleted) for the currently selected objects. The calculation is an interactive approach. The value indicates the accuracy up to which the calculation is to be performed. The smaller the value, the slower the calculation is.
2. With these options you can simplify the geometry of an object to it's bounding box. Attention, this steps changes the actual geometry of your object and cannot be reversed.

## 9.2 Contacts & Clashes

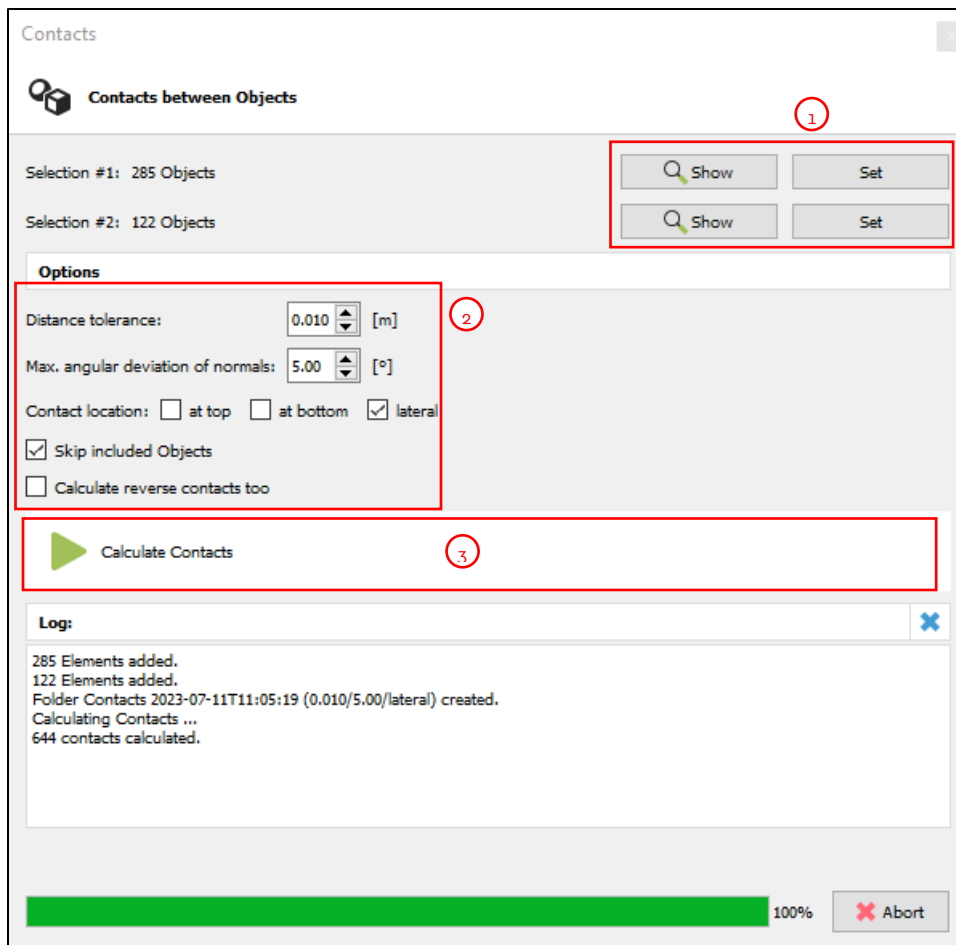
### 9.2.1 Contacts

The Contacts tool create new 3d objects at the contact faces of two other objects, e.g. to get all room side surfaces of a wall.

Start the 'Contacts' tool from the Tools Ribbon bar:

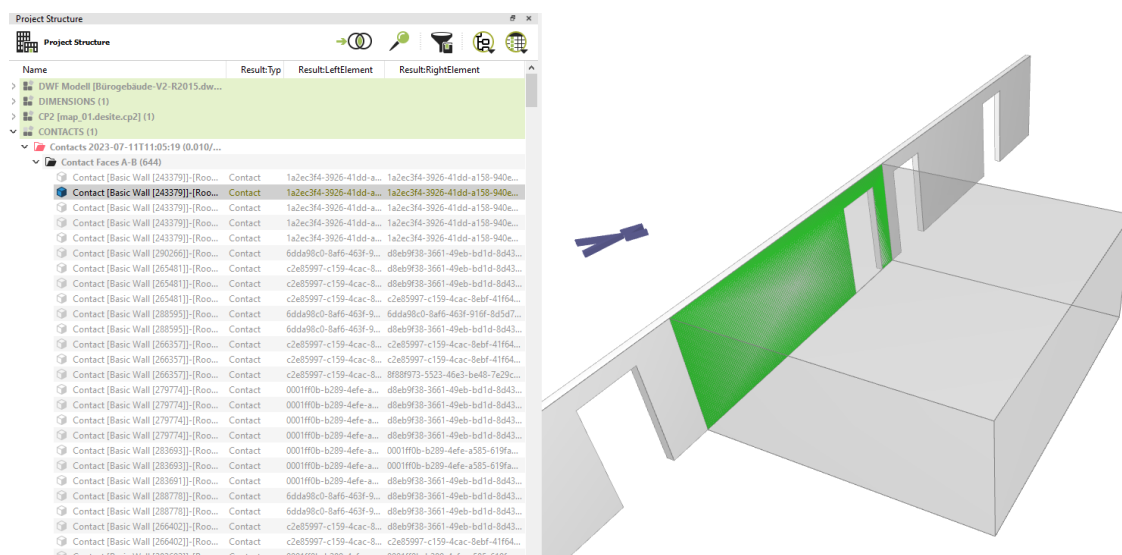


In the 'Contacts' dialog opens:



1. Select the objects for the calculation in the 3d view and click on 'Set'. The contacts will be calculated between two sets (e.g. between all 'Walls' and all 'Room objects' in the project). To validate the selected objects, click on the magnifier button to isolate the actual set of selected objects.
2. Define the options for the calculation
3. Start the calculation

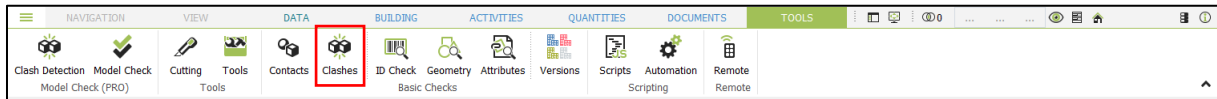
The resulting objects are generated under a new Model 'Contacts' in the **Geometry Domain / Project Structure**. The source objects are linked to the resulting contact:



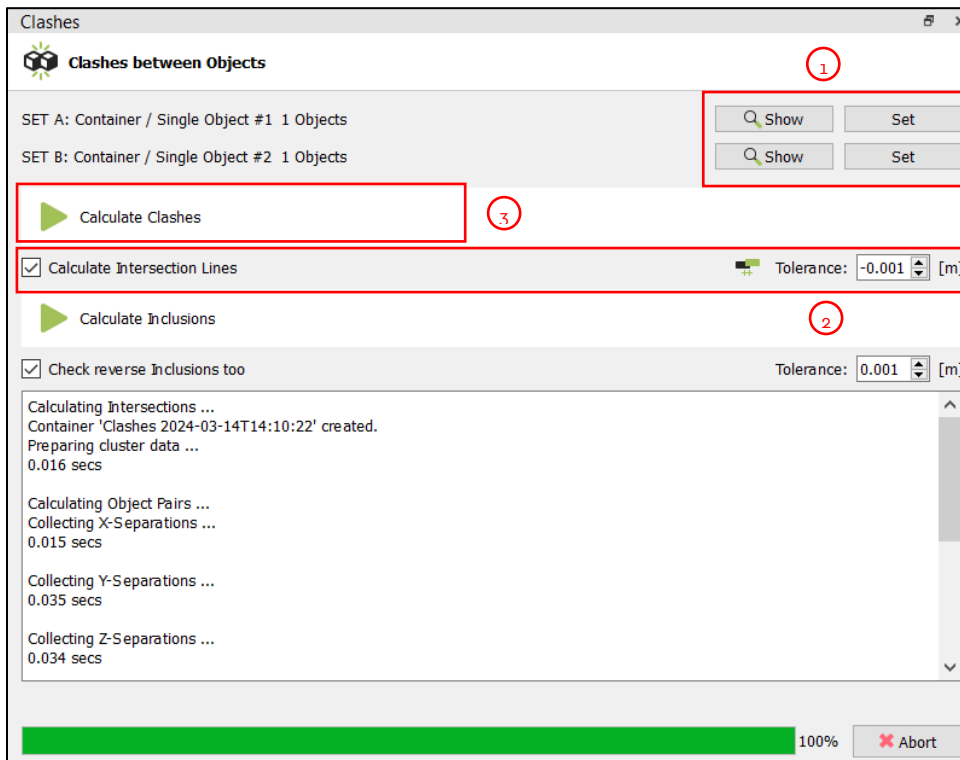
## 9.2.2 Clashes

The Clashes tool create new 3d objects at the collision part of two objects.

Start the 'Clashes' tool from the Tools Ribbon bar:



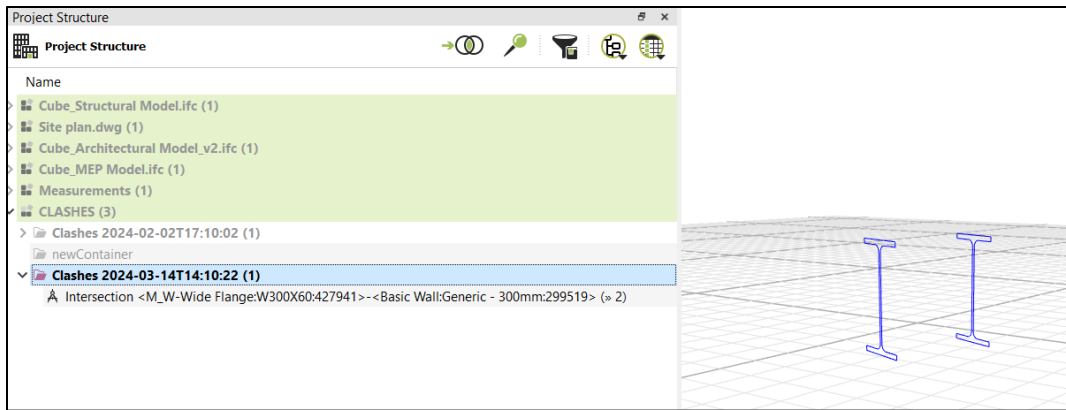
In the 'Clashes' dialog opens:



1. Select the objects for the calculation in the 3d view and click on 'Set'. The clashes/inclusions will be calculated between two sets (e.g. between all 'Walls' and all 'Room objects' in the project). To validate the selected objects, click on the magnifier button to isolate the actual set of selected objects.
2. Define the options for the calculation, e.g. intersection lines or tolerance.
3. Start the calculation

The resulting objects including intersection lines are generated under a new Model 'Clashes' in the **Geometry Domain / Project Structure**:





## 9.3 Cutting

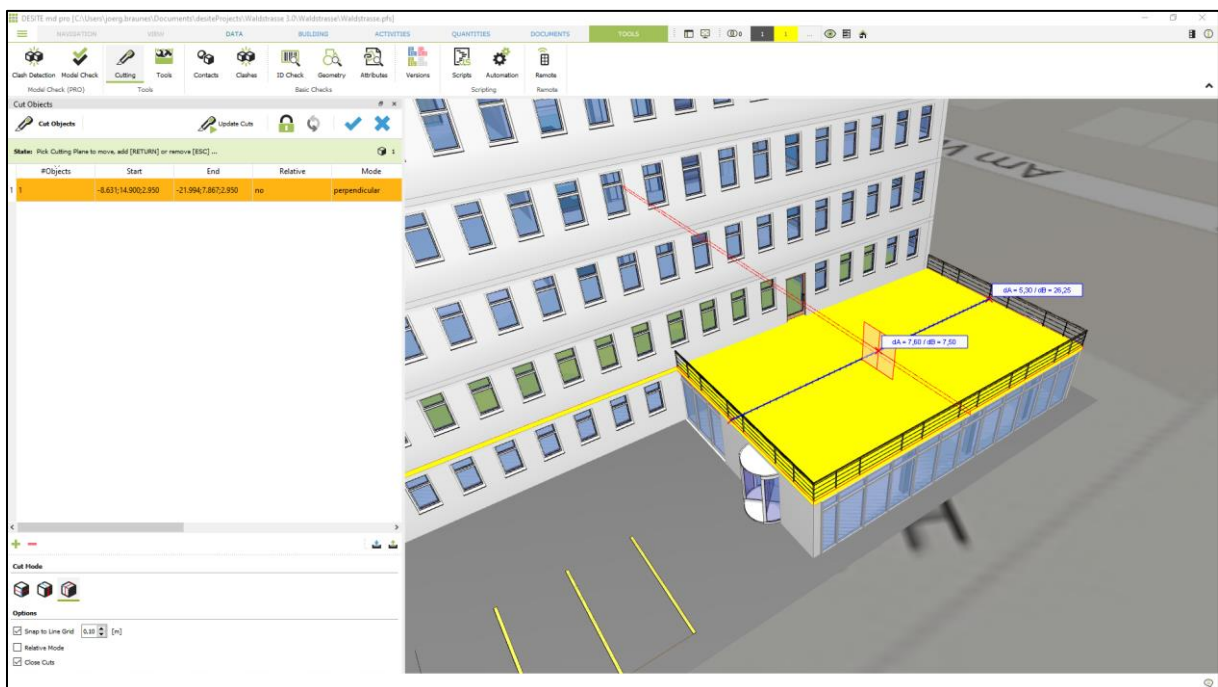
The Cutting tool allows you to cut a 3d object into parts. This could be useful, e.g. if a large object like a slab is produced in several steps and this should be mapped to multiple tasks in a schedule.

### 9.3.1 Using the Cutting Tool

Start the 'Cutting' tool from the Tools Ribbon bar:



The cursor turns into a 'slicing icon'. Now click on an object that you want to cut, this will be selected and marked with a red boundary. The cut itself is defined by a help line. Start drawing this help line by clicking a start and end point on any geometry:

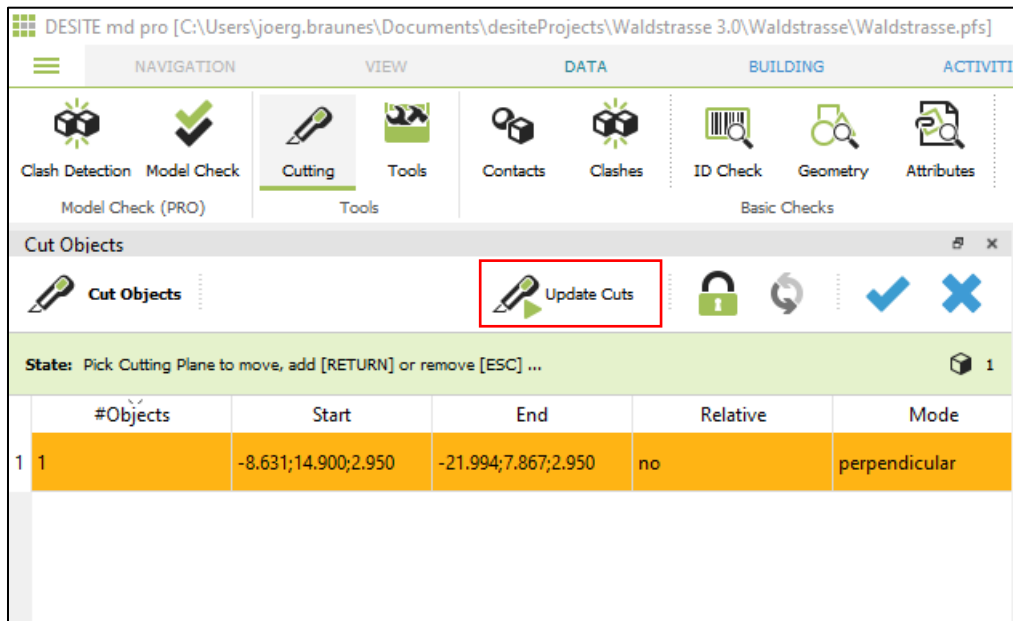


**Hint:**

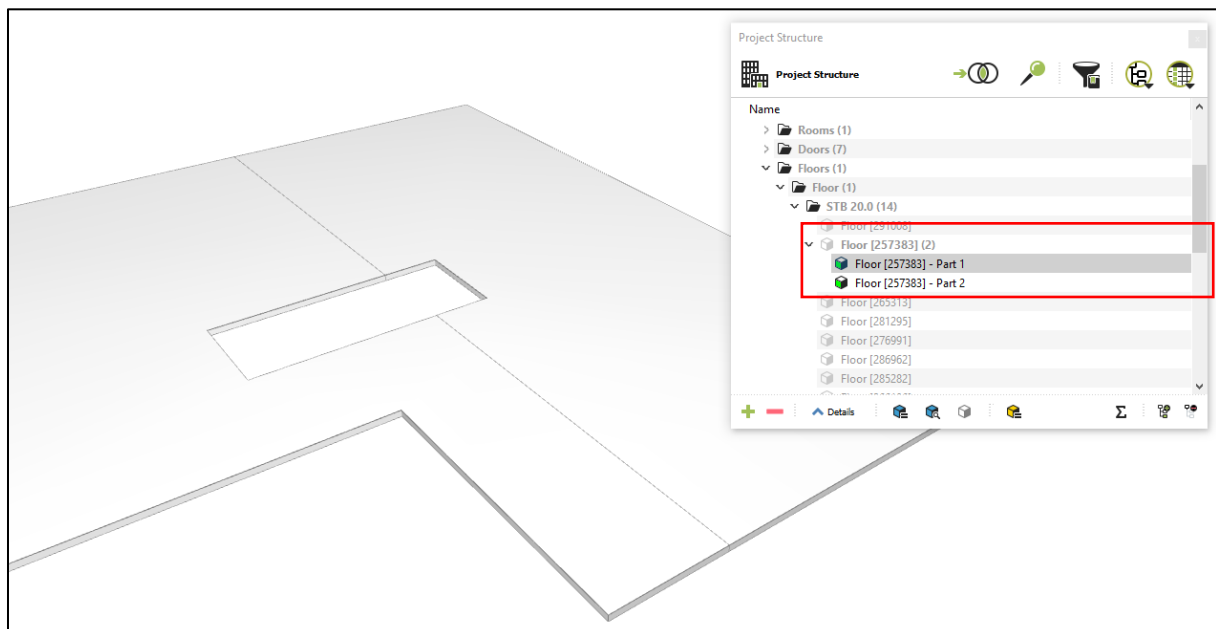
To pick the start and end point of the help line, DESITE uses the point snap settings that are defined for the measuring tool (see: **Measuring**).

As soon as the blue helper line is defined, the cut is shown as a preview in the 3d view and is also listed in the Cutting tool. In the 3d view, a red plane indicates the cutting plane and a red boundary the cut through the selected object. The cutting plane can be moved with drag & drop along the blue help line.

To perform the cut, click on 'Update Cuts' in the Cutting Tool:



The cut creates new 'Parts' of the selected object in the project structure:



### Hint:

Selection behavior of the 'Parts' can be defined in **Selection Settings**.

To perform another cut on the same object, you can either click again on the corresponding icon in the Ribbon bar, or on the '+' icon in the Cutting tool dialogue.

### 9.3.2 Cutting Tool options

The Cutting tool dialogue shows all cuts that were performed on the currently selected objects. Cuts are not permanent and can be deleted or edited at any time.

**Tip:**

If you select multiple objects before you start the Cutting tool, you can perform cuts on multiple objects at the same time. The cut lines are saved per object and can be edited individually afterwards.

Cut Objects

Update Cuts

State: ...

1

#Objects	Start	End	Relative	Mode	Distance	Created	CreatedBy	Comment
1 1	-8.631;14.900;2.950	-21.994;7.867;2.950	no	perpendicular	3,500	2023-02-08T08:52:02	joerg.braunes	

+

-

Cut Mode

Options

☒ Snap to Line Grid
 

0.10 [m]

☐ Relative Mode

☒ Close Cuts



Update Cuts. If a cut line was added, edited or deleted, click on this button to perform the changes on the selected object.



If the lock is closed, the list of cut lines will automatically updated, if the selection in the 3d view changes.



If the lock icon is not closed, you can update the list of cut lines



If a cut line is active (marked as red in the list), you can accept changes on the parameters.



Discard changes on the active cut line.



You can export and import cut lines. When exporting, a possible object reference is lost, therefore the cut lines are always absolutely defined.

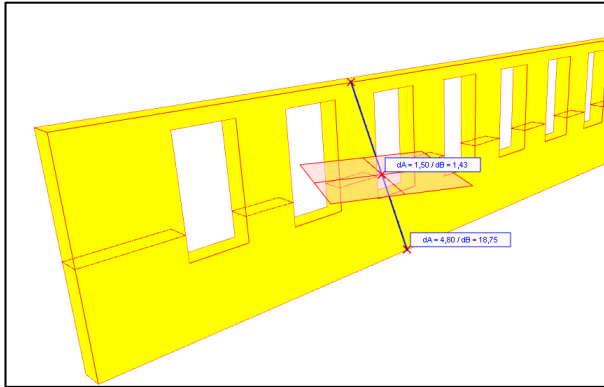
When importing a file with cut lines, they are assigned only to currently selected objects.

DESITE uses the file format \*.xml and provides each cut object with the extension \*cut.xml.

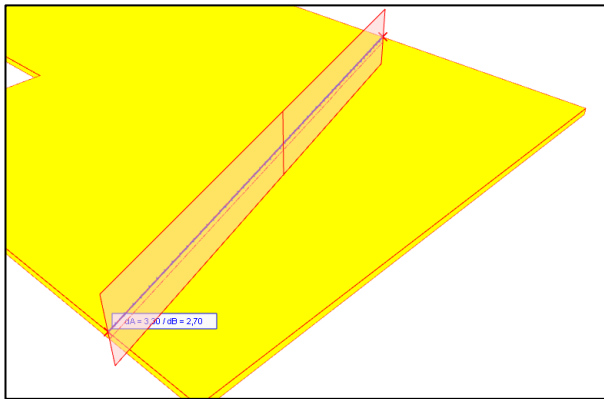
## Cut modes



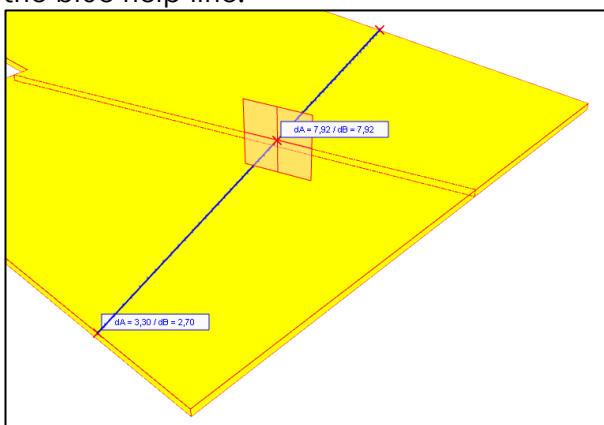
Cut mode horizontal. The selected object will be cut horizontally (parallel to X-Y plane).



Cut mode vertical. The selected object will be cut vertically (along the blue help line).



Cut mode perpendicular. The selected object will be cut perpendicular to the blue help line.



## Options

- **Snap to Line Grid:** Point snap on the help line is performed according to the defined value

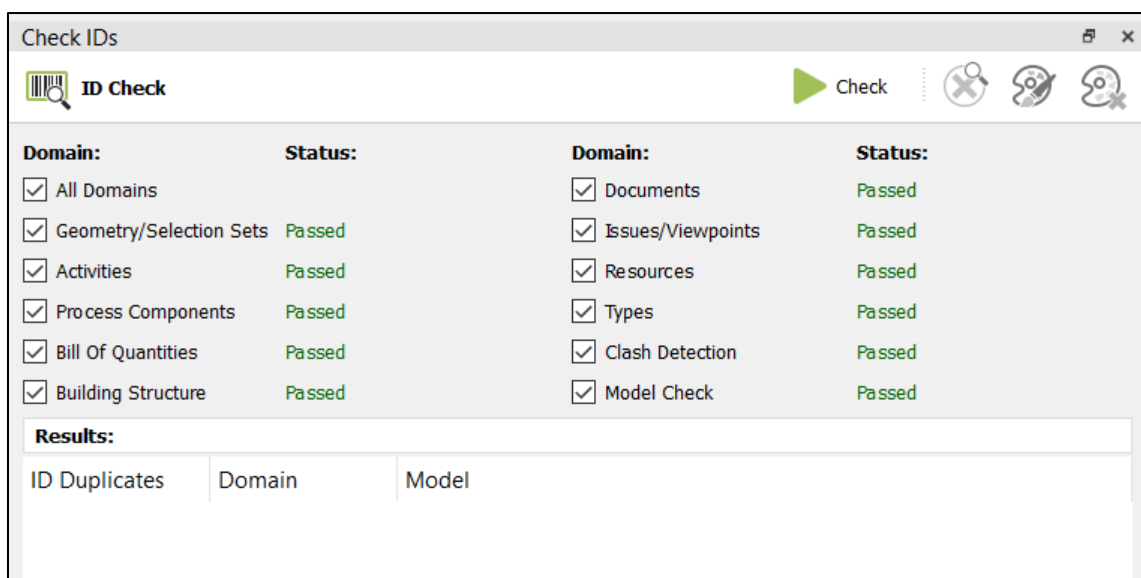
- **Relative Mode:** If this option is enabled, the cut planes are defined relative to the object, i.e. if the object is moved (e.g. after a model update), the cut plane moves with it, relative to the center of the object's bounding box.
- **Close cuts:** Closes the cut surface of the resulting parts

## 9.4 ID/Geometry/Attribute Check

The Cutting tool allows you to cut a 3d object into parts. This could be useful, e.g. if a large object like a slab is produced in several steps and this should be mapped to multiple tasks in a schedule.

### 9.4.1 ID Check

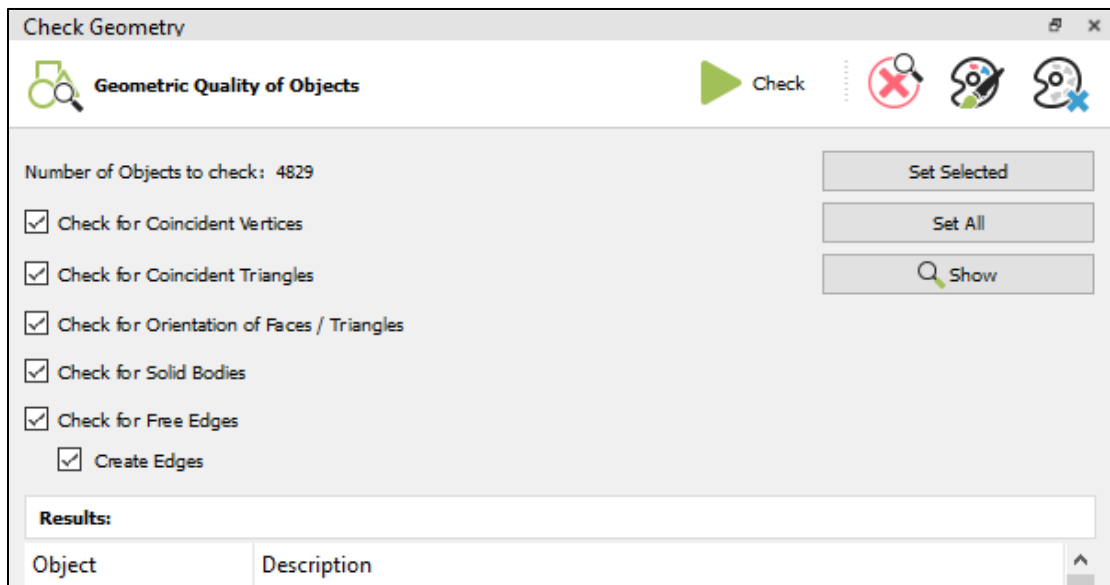
To check if there are duplicated IDs in one or different domains. In versions before 3.4, the unique ids that originate from external models or files are imported and applied to cp:IDs. When multiple models/files that contain the same part are imported into one project, there could be duplicated cpIDs, therefore there was an option during import to allow adding the model/file name as prefix to the cpIDs to avoid the duplication. However, when the prefix is not added, the duplication of cpIDs can cause fatal problems in DESITE. So it is recommended to use this ID check tool, especially when multiple versions of a same model are imported into one project. If a project is newly created with a version from 3.4, then you don't have to worry about the duplicate cpIDs, since all the cpIDs from version 3.4 are unique and those external IDs originate from models will be written to cpExternalIDs.



### 9.4.2 Geometry Check

In geometry check, it is possible to check vertices or triangles which lie in the same position (check for coincidency), or to check if faces or triangles of specified objects are in different direction, as well as to check if a body of object is geometrically closed and therefore has free edges. Before starting the check, please remember to define which

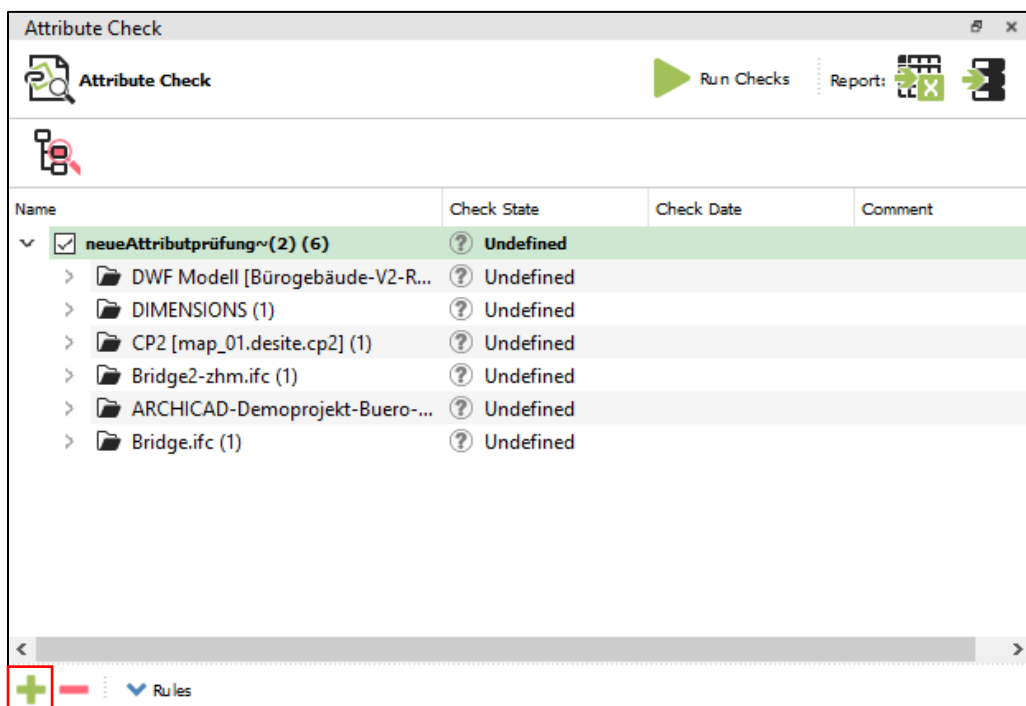
objects are to be checked. You can either set selected objects or set all objects in project to the check.



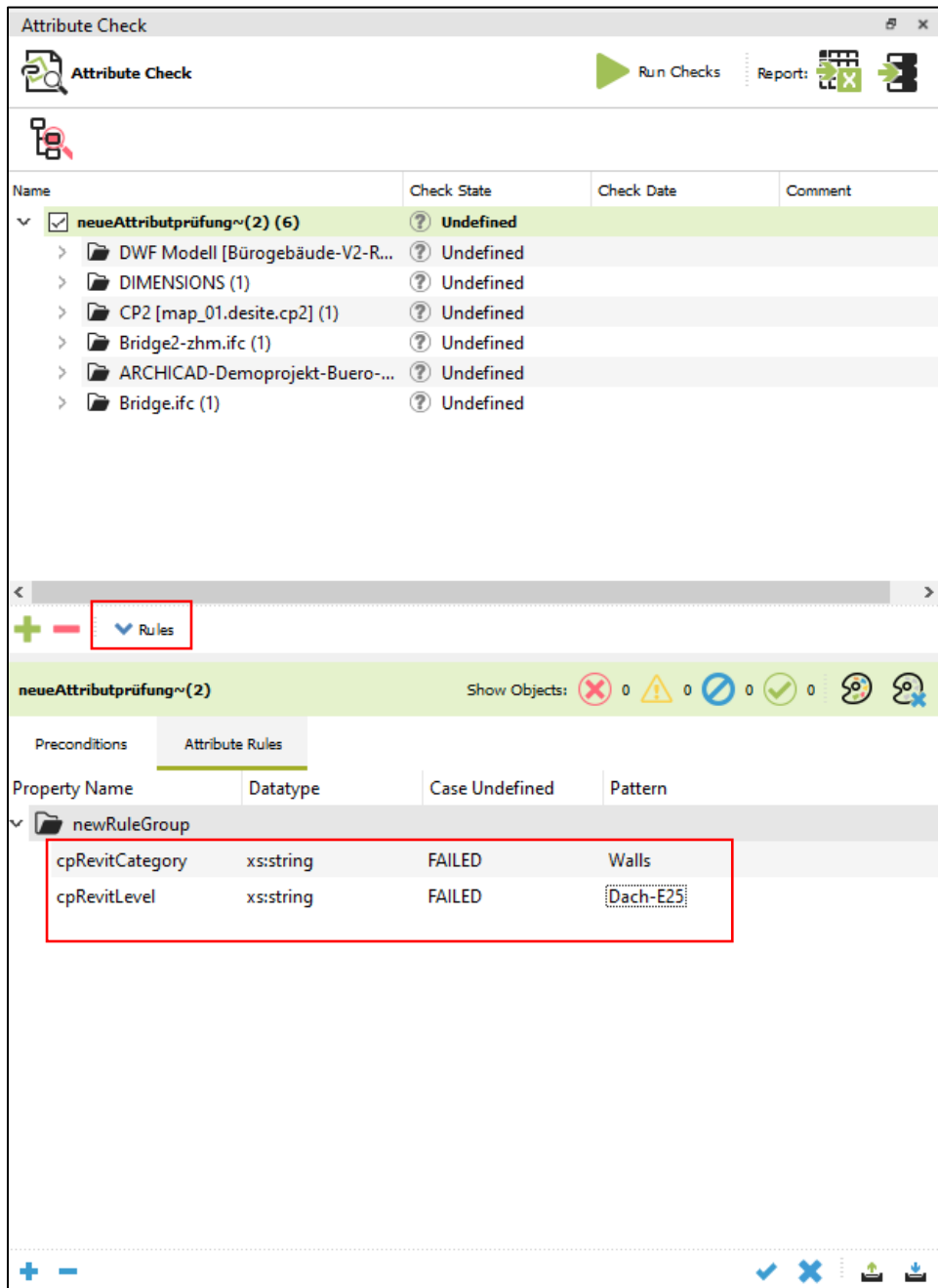
### 9.4.3 Attribute Check

To check if one or more attributes exist in the selected objects, and also to check if the value of this attribute fulfills the pattern.

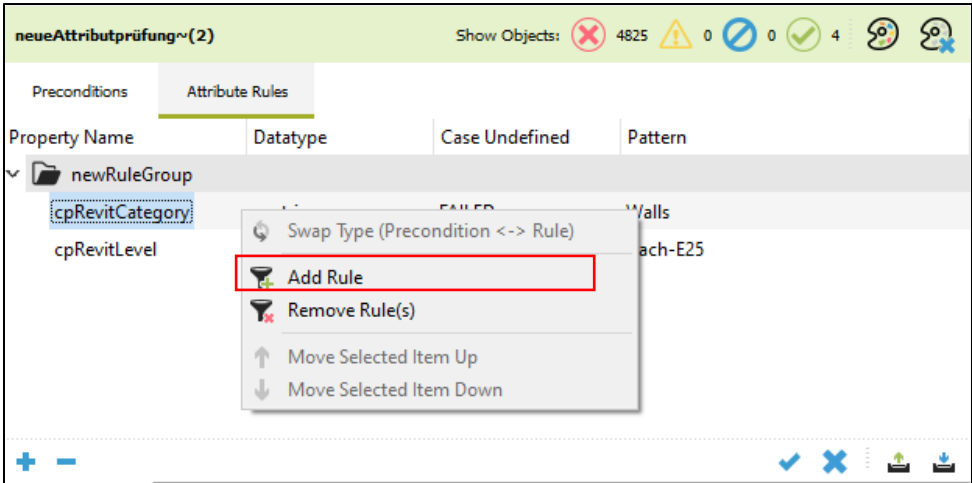
To set up a check run, you need to first display only the objects that you want to check in the 3D-view. Afterwards, click on the "+" button on the bottom-left side of the attribute check domain widget. Then you will see the whole project structure has been added there. But if you unfold the tree structure, you will see only those displayed objects have been really added in the container.



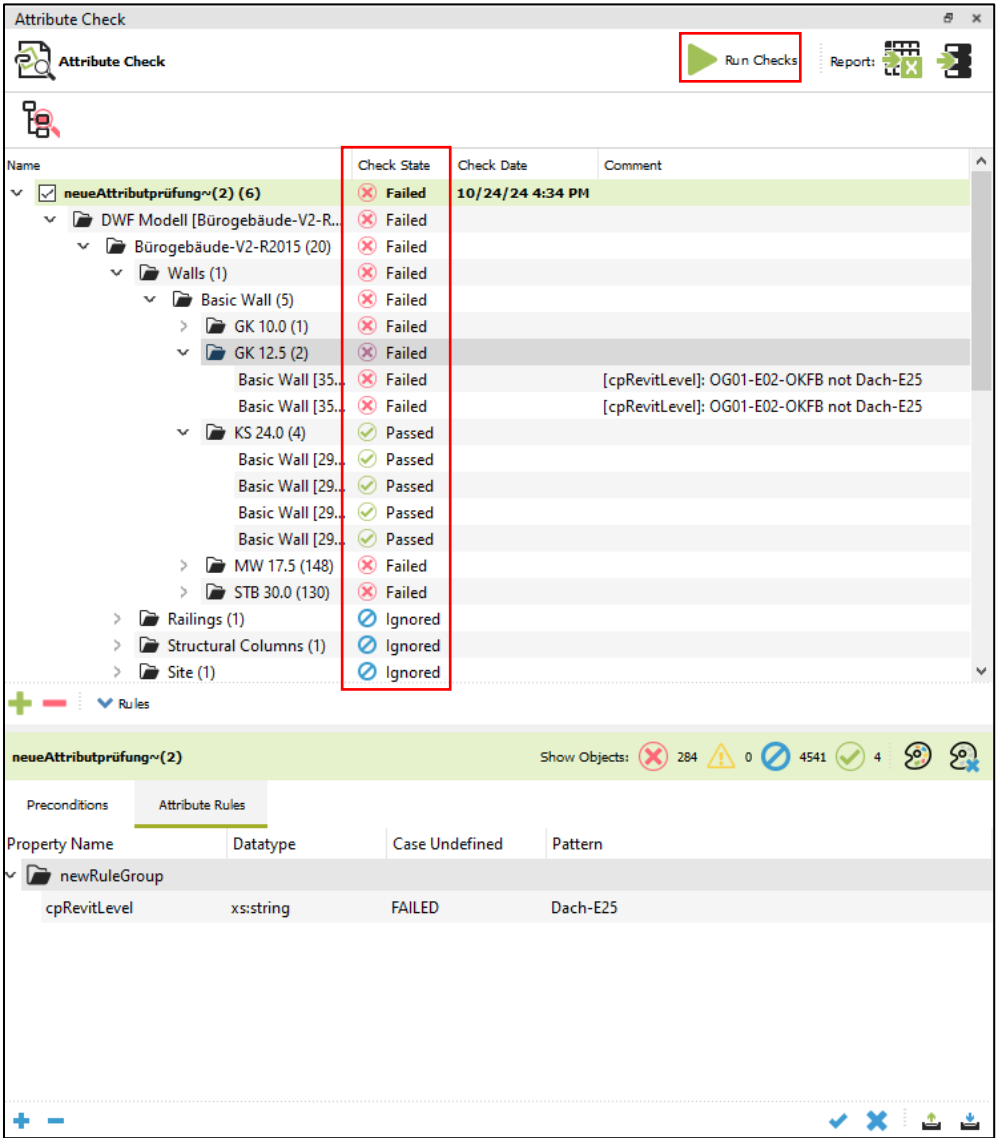
Once the objects have been added to the attribute check, you can define the rules by clicking on "Rules". In the Preconditions, you can define the rules for prerequisites of this check. Only those objects which fulfills this rules can be delivered to attribute check, otherwise they get the check state as "Ignored". And in attribute rules, you can define the actual checking rules by entering the property name and pattern.



Please use context menu to add multiple rules to the same rule group. The relationship between different rule groups is "or", only the relationship between rules within the same rule group is "and".



After all objects, preconditions and attribute rules are set up, you can click on "Run Checks". The results will be shown in the property "Check State".



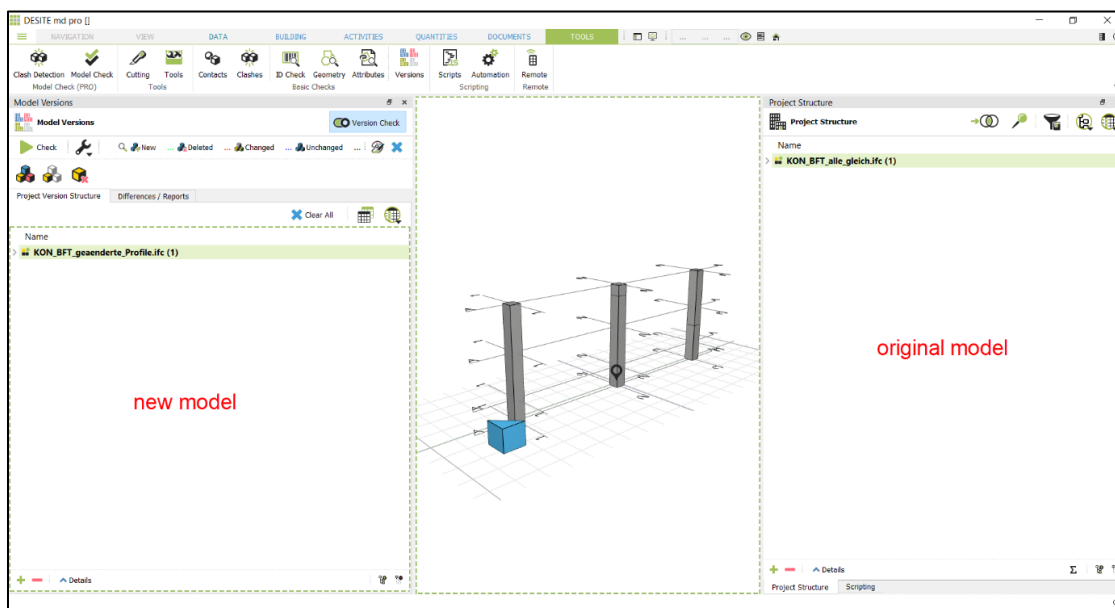


## 9.5 Model Versions

The module "Model Versions" is for comparing different versions of model files.

### 9.5.1 Activate Model Version Comparison

Make sure the original model has been load in project structure and load the new version under "Tools"- "Versions". You can first activate version check modus by switching it on, once the modus has been activated, you will see the green dotted borders around model comparison panel and the main view panel.



Version check switch. Switch on to view the results. If it is switched off, the function of version check is completely deactivated.



Execute version comparison. It should be pressed each time after loading a new model.



Configure version check. See the next screenshot for detailed description.



Show all objects in project version structure

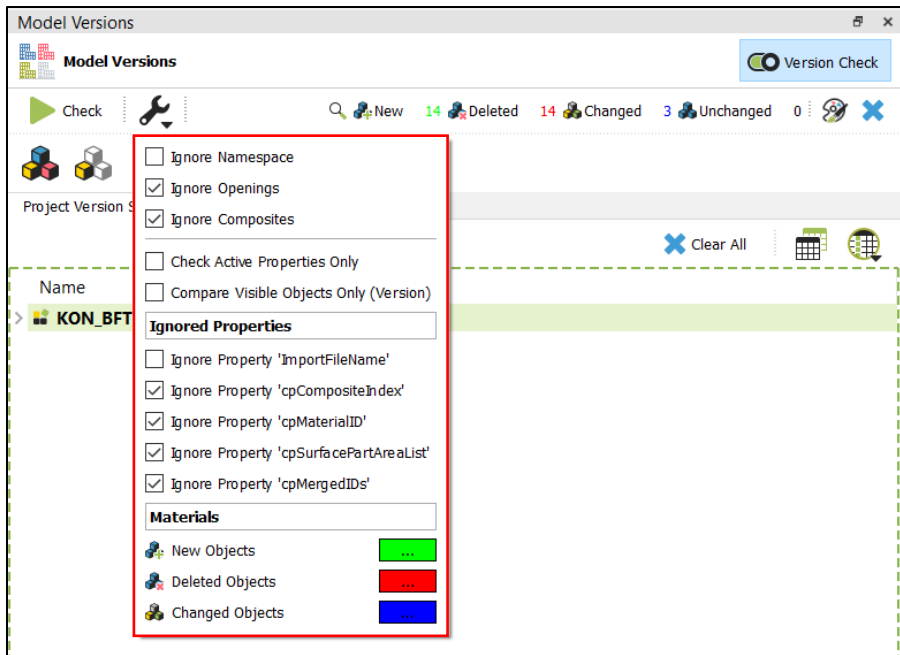


Only show the selected objects in project version structure. This can be combined with configuration option "compare visible objects only".



Hide the selected objects in project version structure

It is recommended to set the configuration for the check with pressing the wrench button. Because with this option one can decide for each check which objects or properties should be checked or ignored.



## Configurations

- **Ignore Namespace:** this should be chosen if namespace has been added to the cpID by importing models. Because only the objects from old and new models that are with the same cpIDs will be compared.
- **Ignore Openings:** this should be checked, if openings should not be considered in the geometric comparison.
- **Ignore Composites:** to prevent comparing subordinates of a object.
- **Check Active Properties Only:** to check only the active properties in the project, which can be configured under "Data-Properties".
- **Compare Visible Objects Only (Version):** check only the objects which are visible in project version structure.
- **Ignore Properties:** ignore the chosen properties for property comparison.
- **Materials:** set the color codes for displaying geometric changes.

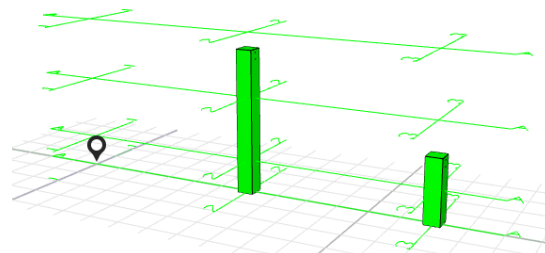
When all things are set up, you can start checking by clicking on the check button.

### 9.5.2 The Result of Comparison

After a check has been done, one can view the results of new, deleted, changed or unchanged objects by clicking on the corresponding icons.

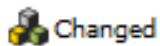
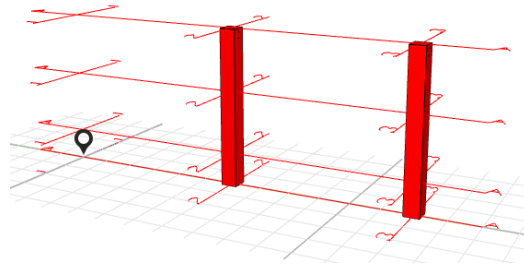


Only view the newly added objects. It shows besides the number of new objects.

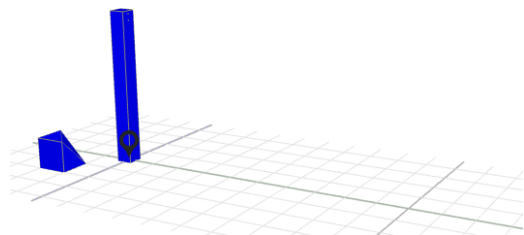




Only view the deleted objects. It shows besides the number of new objects.



Only view the objects that have been modified, either geometry or information. It shows besides the number of new objects.



Only view the objects that have not been modified, neither geometry nor information. It shows besides the number of new objects.



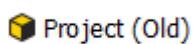
Turn on coloring modus with defined color mode (the colors can be defined in configuration)



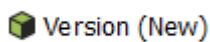
Turn off coloring modus



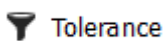
Find selected objects in tree view



View the original object of selected property change



View the new object of selected property change



Set tolerance for the changes



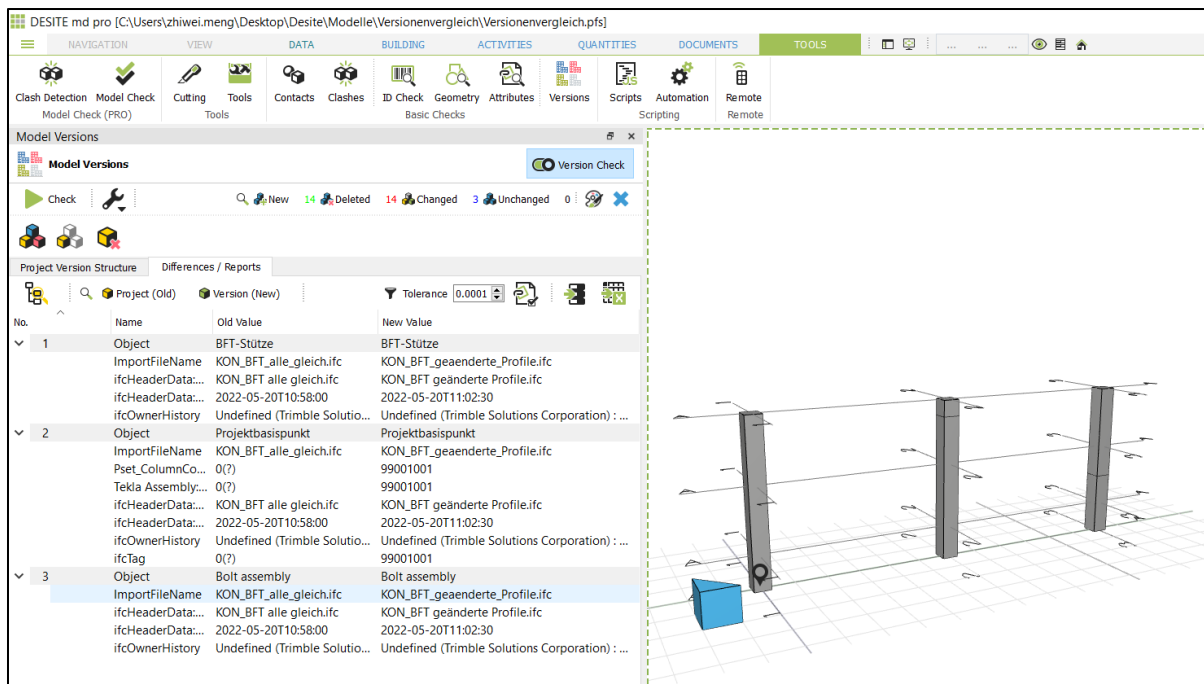
Configure shown properties in report



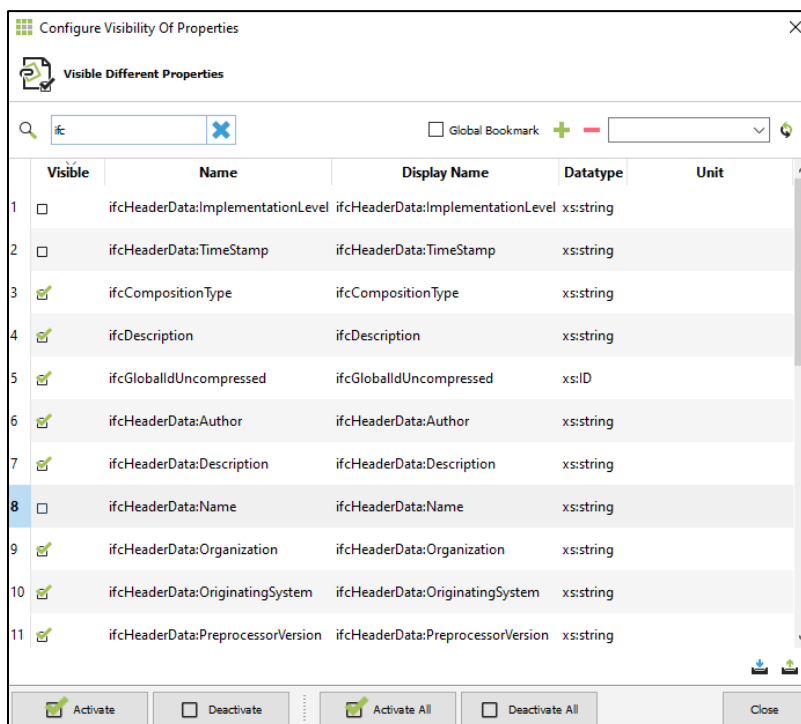
Export result to database



Export result to excel



The values of original and new objects for each changed properties are listed under "old value" and "new value". And in the result it is still possible to configure which properties should be shown or hidden (see the button for "Configure shown properties in report" in the table above), that would be considered in the export process.



After all useful information has been included, it is the time to export the result to a database or a excel file. And if you want to replace the existing model with the new version, you can use the function **Redirect Project Files**.

## 10. Command Line Switches

All desktop applications of the DESITE product family (md, md pro, custom, share, share pro) can be configured through command line switches. This means, that by starting the software some specific parameters could be passed to the executable file (.exe). This guide describes what the parameters are, what they do, how they are passed to the application, and what differences there are between DESITE applications to consider.

### 10.1 Passing Parameters

The parameters are typed with hyphen (-) after the program name (.exe), in order to be passed via command line.

Examples:

```
desitecustom.exe -lang "en"
desitecustom.exe -srvPort 45456
desitecustom.exe -unlockWebForms
```

The meaning of these parameters will be explained in the next chapter **Parameter Reference**.

Some parameters require a value (e.g. -lang and -srvPort), some work like simple switch without a value (e.g. -unlockWebForms). If a value is required, it is specified after the parameter with a single space as a separator. Be aware of the data types of the values. In the example, "en" is in quotes because it is a string, while 45456 is given without quotes because it is a port number.

The arguments can of course be combined:

```
desitecustom.exe -lang "en" -srvPort 45456 -unlockWebForms
```

Alternatively, the parameters can be written to a config file (JSON format) and passed to the application using -configFile:

```
desitecustom.exe -configFile "E:\somePath\desiteConfig.json"
```

The configuration file could be like this:

```
{
  "lang": "en",
  "srvPort": 45456,
  "unlockWebForms": true
}
```

This would set the same options as we did in the examples above. The JSON keys work the same as the command line arguments. However, there is one important difference with unlockWebForms: as a command line argument, no value is required for this switch, however in JSON file a value should be given.

Important: If -configFile is used, any other command line parameters that may be specified are ignored. This is to avoid ambiguities or conflicts.

## 10.2 Parameter Reference

### 10.2.1 file

Load a project or model with -file "filepath".

Example:

```
desitemd.exe -file "D:\Projects\SampleProject\mySampleProject.pfs"
```

Note: This does the same thing as passing the filename directly:

```
desitemd.exe "D:\Projects\SampleProject\mySampleProject.pfs"
```

### 10.2.2 srvPort, cltPort

Set the TCP ports for the remote interface with -srvPort Number and/or -cltPort Number.

Example:

```
desiteshare.exe -srvPort 45456 -cltPort 45455
```

DESITE applications can send and receive messages over local TCP sockets to communicate with other applications. By default, the server port (listener) is 45456 and the client port (sender) is 45455. Changing these values may be necessary if the default ports are not available in your network environment, or if multiple instances of DESITE applications are running simultaneously.

More details about the DESITE remote interface can be found in the corresponding documentation.

### 10.2.3 showWidget

Show a specific widget or window on application start with -showWidget

Example:

```
desitecustom.exe -showWidget WebForms
```

Use this switch with one of the following parameters to open the respective widget:

- WebForms
- Issues (alternatively: Viewpoints)
- ObjectInfo (alternatively: DataSheet)
- ProjectStructure

**Note:** This does not necessarily ensure that the window appears visibly in the foreground. Depending on the last saved interface configuration, it could also be in an inactive tab. This switch is the successor of `-showWebForms`. It will still work, but is deprecated. Use `-showWidget WebForms` instead.

#### 10.2.4 webFormLocations

Define form sources with `-webFormLocations "url1[url2]"`.

Example:

```
desitemd.exe -webFormLocations "https://www.thinkproject.com"
```

Since version 2.6 DESITE supports other sources as start page for the forms besides the traditional local `DbInputForm.html`. This can be especially useful if you need the same forms in many projects.

You can specify multiple URLs. When using the command line argument, set a semicolon as a separator. If a configuration file is used, use the JSON list notation (even with only one URL):

```
{
  "webFormLocations":["https://www.thinkproject.com"]
}
```

#### 10.2.5 unlockWebForms

Grant a general WebForms unlock with `-unlockWebForms`.

Example:

```
desitecustom.exe -unlockWebForms
```

Use this switch if you want to prevent the need to explicitly enable the use of WebForms and other browsers embedded in DESITE - a security feature introduced with version 2.6.1.

Note: For security reasons, we do not recommend routine use of this switch. It should only be used in WebForms development when the query would otherwise only interfere, or when you are really sure to open projects only from trusted sources.

#### 10.2.6 lang

Set your language with `-lang "langCode"`.

Example:

```
desitecustom.exe -lang "en"
```

The language setting set in this way overrides the corresponding setting in the application configuration, but does not overwrite it (override, not overwrite).

For example, if English is set as the language in the program options, and you start with the `-lang` parameter "de", the interface for this session will be localized in German. If you then however, start the application again without the parameter, the English setting is applied again.

Valid values (DESITE 3.0): en, de, fr.

### 10.2.7 scriptDebug

Enable JavaScript/WebForms debugging with `-scriptDebug [port]`.

Example:

```
desitemd.exe -scriptDebug
```

When script debugging is active, JavaScript errors in macros will trigger a debug window that allows you to inspect local variables, set breakpoints, etc.

Furthermore, you can remote-debug WebForms. To do that, open up a separate Chromium-based browser (like Google Chrome or Microsoft Edge) and navigate to the following address:

<http://localhost:1000>. Choose the WebForm you wish to inspect, and then you can analyse and debug the code with the usual Chromium developer tools. The default port for remote debugging is 1000. If you want another port for some reason, you can pass it as a parameter, e.g. `"-scriptDebug 1234"` and its corresponding address: <http://localhost:1234>.

Please note that `-scriptDebug` can have negative effects on performance. You should therefore use it only while developing or testing your code, not in production.

## 10.3 Availability in DESITE Applications

	md pro	md	custom	share pro	share
file	✓	✓	✓	✓	✓
srvPort, cltPort	✓	✓	✓	✓	✓
showWidget	✓	✓	✓	✓	✓
webFormLocations	✓	✓	✓	(no web forms)	(no web forms)
unlockWebForms	✓	✓	✓	(no web forms)	(no web forms)
lang	✓	✓	✓	✓	✓
scriptDebug	✓	✓	✗	✗	✗