ARCHIFRAME

ArchiFrame instructions 27th March 2019

Pdf-version here

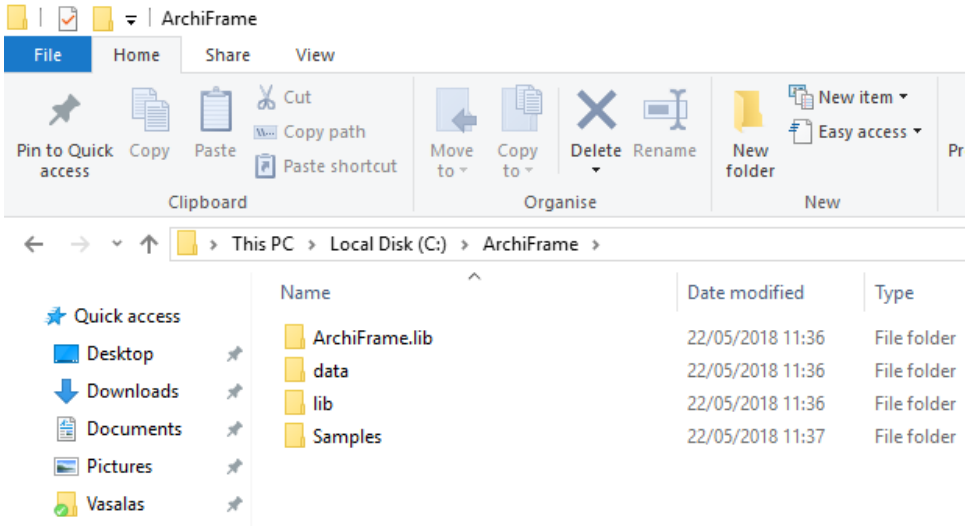
1 Windows Installation

Context sensitive help has been tested with Acrobat Reader and Foxit PDF Reader.

The installer installs the add-on to all supported and user-selected ArchiCAD-versions. By default, the library is installed into the folder C:\ArchiFrame\lib.

Next to the *lib*-folder is the *data*-folder which contains main ArchiFrame settings including an Excel template and a cnc-file producer. This folder or its contents should not be edited. Instead, the data folder should be copied and renamed for editing. All ArchiCAD-versions share the same library and *data*-folder.

If you would like to check some sample projects, this can be found in C:\ArchiFrame\Samples, another folder which will be installed together with the lib and *data*-folder. In that, you can find two samples: Model ArchiFrameDemo.pla, which contains examples of individual planks and ArchiFrameElemDemo.pla, which provides examples of elements and related topics.



The installer does not include an uninstaller. ArchiFrame needs to be removed manually by deleting:

- Library folder, default C:\ArchiFrame.
- Add-on inside ArchiCAD installation folders, for example C:\Program Files (x86)\Graphisoft\ArchiCAD 21 INT\Add-Ons\ArchiFrame.

After installation ArchiFrame can be found in the Archicad *Options*-menu. Inside ArchiLogs the tools are in menu *ArchiLogs / Joints*.

ArchiFrame and ArchiLogs cannot co-exist. If you are an ArchiLogs user, a license update will add Archiframe functionality to ArchiLogs.

2 Mac Installation

Context sensitive help works with Safari and requires Acrobat Reader to be installed.

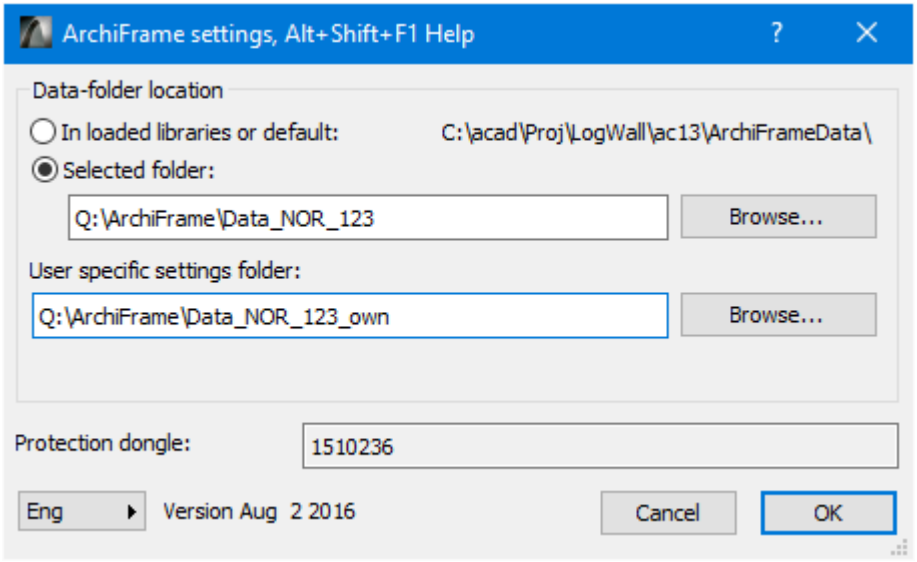
The zip-package has the following structure:

- ArchiFrameXxAddOn, the add-on for the specific ArchiCAD-version.
- ArchiFrame:
    - Data, all configuration files (described in the following section);
    - ArchiFrame.LibEng, English library;
    - ArchiFrame.LibFin, Finnish library;
    - Samples, sample projects;

The add-on is copied into the corresponding ArchiCAD Add-Ons folder (for example, Programs/Graphisoft/ArchiCAD 21/Add-Ons).

The recommended location for the *ArchiFrame*-folder is on the desktop. This means that the ArchiFrame-library folder (ArchiFrame.LibEng) must be added to every project where it is needed. It is also possible to place the library inside ArchiCAD-libraries, so it is always loaded.

ArchiFrame will find the *data*-folder from the default location (Desktop/ArchiFrame/Data). If using a different location, the folder must be selected from the settings:



Please read the following topic *Getting started: entering main settings and setting up the material list* about the data folders.

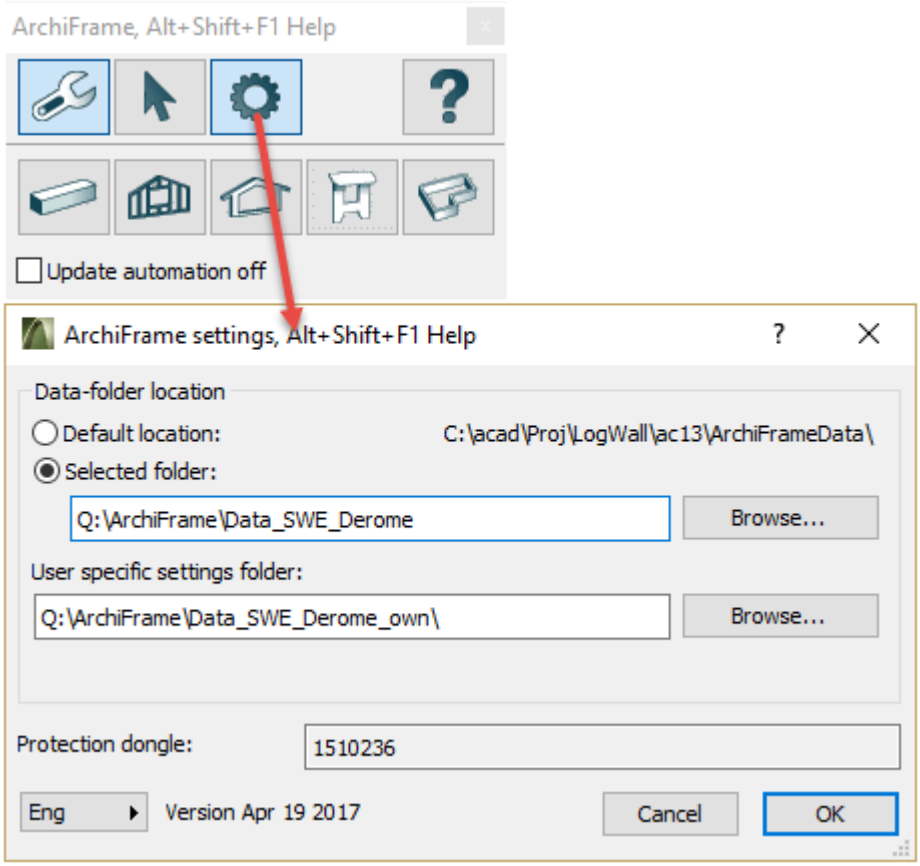
The samples contain PC-library references which lead to missing libraries in Mac – please edit the library list so this way the ArchiFrame-library will be properly loaded.

ArchiFrame and ArchiLogs cannot co-exist. If you are an ArchiLogs user, a license update will add Archiframe functionality to ArchiLogs.

3 Getting started: entering main settings and setting up the material list

Before starting your own project, make a copy of ArchiFrame *data*-folder, place it in C:\ArchiFrame next to the original folder and rename it, for example, C:\ArchiFrame\data_own. The ArchiFrame *data*-folder contains ArchiFrame's main settings by default. Also, create another folder, this time an empty one, to be used in the *user specific settings folder* and set it in the settings dialog. With this setup folder C:\ArchiFrame contains all settings and can be shared to other users or other computers or just backed up easily.

Using your own *data*-folder allows installation of updates without overwriting settings which the user has edited. You must also set the folder new locations in ArchiFrame settings:



The settings of different tool palettes are saved into the file ArchiFrameMain.cfg, which is in a *user-specific*-folder. In Windows 7, this folder is C:\Users\[username]\AppData\Local, for example, C:\Users\petteri\AppData\Local. In Mac, the folder is [user]/Library/Preferences.

Settings files in xml format are edited with a text editor, rather than Word or any other word processor. Notepad++ is recommended instead of plain Notepad, although plain Notepad also works (http://notepad-plus-plus.org/). The benefit of Notepad++ is that it gives warnings about xml-markup errors and formats the file, making it easier to read.

The main *data*-folder contains the following files:

- ArchiFrameBlocks.xml, material list. The list of available materials is entered here.
- ArchiFrameCnc.lua, cnc-producer. Built in script supports Hundegger bvn-files. There may be many files like ArchiFrameCncBtl.lua and ArchiFrameCncWup.lua to write different cnc-formats.

- ArchiFrameCoveringBoards.xml, weatherboard settings.
- ArchiFrameElements.xml, element related settings.
- ArchiFrameFoundation.xml, foundation related settings.
- ArchiFrameFoundationFin.xml, foundation related settings in Finnish.
- ArchiFrameListing.lua, listings producer. Default script produces listings either in Excel (Windows only) or text format.
- ArchiFrameTrusses.xml, truss settings.
- FrameListingEng.xls, Excel-template in English.
- FrameListingFin.xls, Excel-template in Finnish.
- FrameListingElemEng.xls, Excel-template for element listing.
- FrameListingSummaryEng.xls, Excel-template for summary listing.

All measurements in xml-setting files are given in meters and the decimal separator used is a full stop, for example, 57 mm is 0.057.

User specific settings folder is by default the same as folder where the settings (file ArchiFrameMain.cfg) are saved. To easily share these settings, the folder can and *should* be set here to use company specific settings. Usually all updates contain new versions for ArchiFrame library, add-on and the *data*-folder. To keep own settings over updates, some of the standard data-files listed above can be edited or replaced with version from user specific settings folder. *User specific settings*-folder contains files (not all have to be present):

- ArchiFrameOwnElems.xml, own element types defined in the main element tools. Please see section Own element types/Custom element tools.
- ArchiFrameOffsetsEdge.xml, edge offset settings from the main element tool palette. Please see section Custom layer edge/opening offsets.
- ArchiFrameOffsetsOpening.xml, opening offset settings from the main element tool palette. Please see section Custom layer edge/opening offsets.
- ArchiFrameCorners.xml, custom corner types from element tool palette. Please see here.
- ArchiFrameLic.txt, the license file may be located here or in the *data*-folder. This is the recommended place since it allows seamless operation even if the *data*-folder is updated with program update.
- ArchiFrameBlocks.xml, to use this file instead of the file in the *data*-folder. There may be also ArchiFrameBlocksChanges.xml file to change parts of the base *data*-folder's ArchiFrameBlocks.xml -file.
- ArchiFrameElements.xml/ArchiFrameElementsChanges.xml, ArchiFrameCoveringBoards.xml/.
- ArchiFrameCoveringBoardsChanges.xml, ArchiFrameTrusses.xml/.
- ArchiFrameTrussesChanges.xml: Handled as ArchiFrameBlocks.xml/.
- ArchiFrameBlocksChanges.xml.

The protection dongle information is used to make ArchiFrame license. When asked, please copy the license information here to be pasted into an e-mail to your vendor.

3.1.1 The ArchiFrame*Changes.xml files

It is possible to add, remove or replace information from base *data*-folder's xml-file for easier maintenance. That allows new features to come from the base *data*-folder and still have own customizations in certain parts of the file. ArchiFrame reads first the file from base *data*-folder and then applies operations from user specific *Changes.xml-file. For example, to add some materials

before default materials there is file ArchiFrameBlocksChanges.xml in *user specific*-folder with following content:

```
<?xml version="1.0" encoding="utf-8"?>
<archiframe>
  <materials merge="addpre" mergechildren="1">
    <material id="25x38" name="25x38" thickness="0.025" height="0.038" shape="block">
    </material>

    <material id="45x75" name="45x75" thickness="0.045" height="0.075" shape="block">
    </material>
  </materials>

  <materials>
    <material merge="addpost" mergepath="[@id='45x170']" id="45x170 C24" name="45x170 C24"
thickness="0.045" height="0.170" shape="block" maxlen="5.1">
    </material>

    <material merge="replace" mergepath="[@id='45x45']" id="45x45" name="45x45"
thickness="0.045" height="0.045" shape="block" maxlen="999">
    </material>

    <material merge="remove" mergepath="[@id='45x220']">
    </material>
  </materials>
</archiframe>
```

ArchiFrame processes only nodes having attribute merge = “xxx”. It may have values:

- addpre, add current node before existing children.
- addpost, add current node after existing children.
- remove, remove current node with all its children.
- replace, replace base data xml with current node.

If attribute mergechildren is given with value 1 like in the example, current node’s children are processed instead of current node. It is also possible to give attribute mergepath to specify the target with attributes using the xpath-like syntax like in the example.

To debug the xml-settings ArchiFrame copies the resulting text to clipboard if *Shift*-key is pressed, for example, when closing the settings dialog with OK.

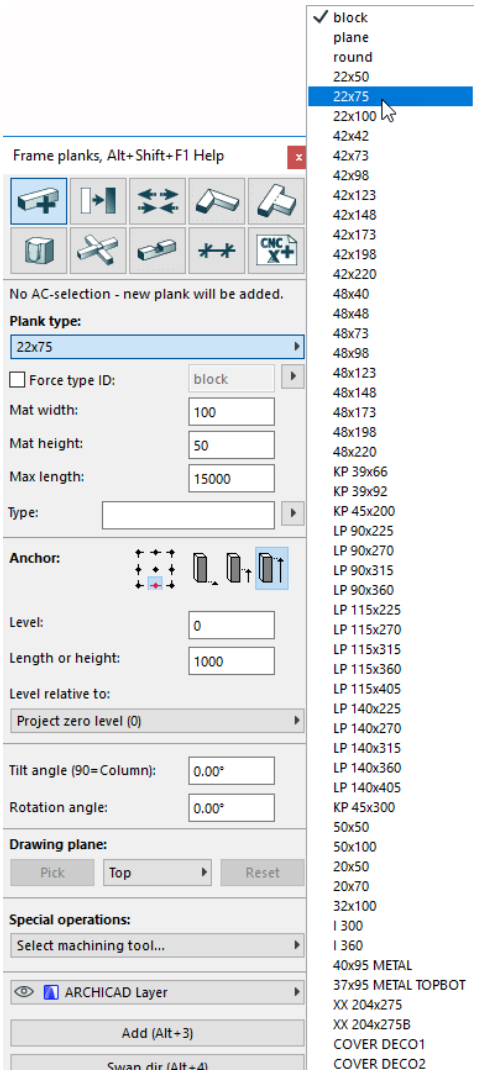
Another option is to make the changes in the related file in your own *data*-folder. Using the same example of the materials, select the ArchiFrameBlocks.xml and search for the materials. It will be like this:

```
  <materials>
    <!--shape must be specified block/plane/round even if custom profile is given m3factor is
the coefficient factor to get m3-quantity from the material. If you omit it, default formula
will be used. -->
    <material id="block" name="Block" thickness="0.000" height="0.000" shape="block">
</material>
    <material id="plane" name="Plane" thickness="0.000" height="0.000" shape="plane">
</material>
    <material id="round" name="Round" thickness="0.000" height="0.000" shape="round">
</material>
    <material id="22x50" name="22x50" thickness="0.022" height="0.050" shape="block"
maxlen="99" m3factor="0.000"> </material>
    <material id="22x100" name="22x100" thickness="0.022" height="0.100" shape="block"
maxlen="99" m3factor="0.000"> </material>
    <material id="42x42" name="42x42" thickness="0.042" height="0.042" shape="block"
maxlen="99" m3factor="0.000"> </material>
```

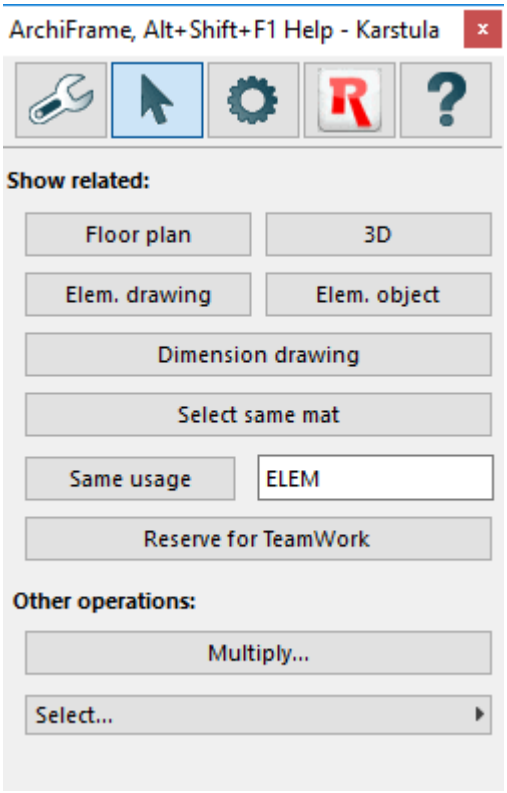
Making changes in this file, which it can be adding, removing or simply change some data of some material, it will be saved in your own *data*-folder and then you just need to reload the library in the ArchiFrame toolbar.

```
    <material id="22x50" name="22x50" thickness="0.022" height="0.050" shape="block"
maxlen="99" m3factor="0.000"> </material>
    <material id="22x75" name="22x75" thickness="0.022" height="0.075" shape="block"
maxlen="99" m3factor="0.000"> </material>
    <material id="22x100" name="22x100" thickness="0.022" height="0.100" shape="block"
maxlen="99" m3factor="0.000"> </material>
    <material id="42x42" name="42x42" thickness="0.042" height="0.042" shape="block"
maxlen="99" m3factor="0.000"> </material>
```

This way your settings will be available in your own *data*-folder.

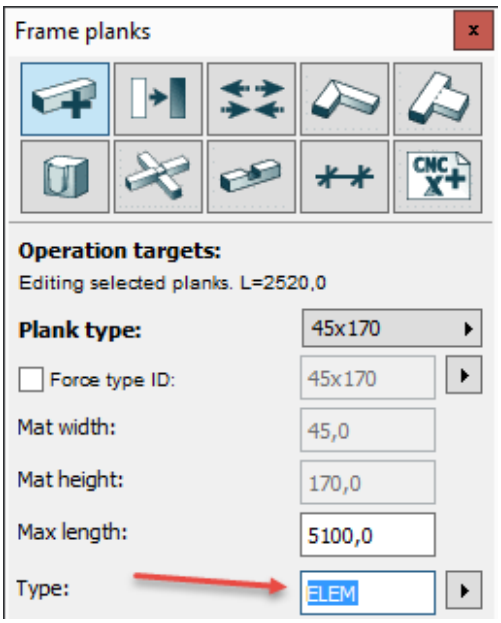


4 Selection tool



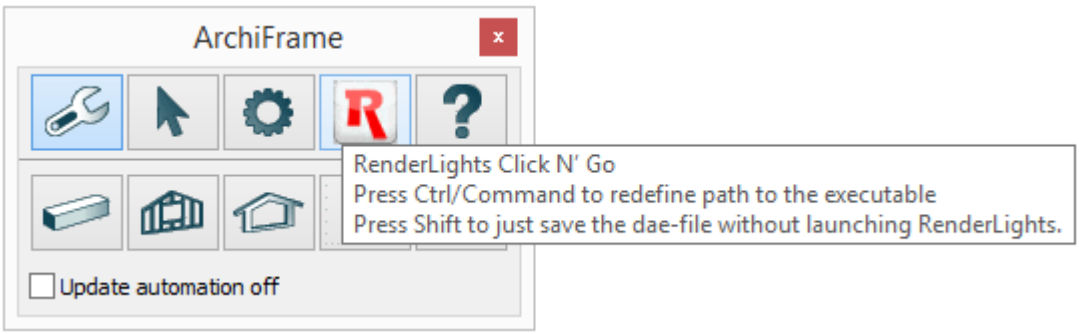
The selection tool makes it easy to switch between Floor plan, 3D and element drawing. By clicking one of the *Show related*-buttons it will open a view showing currently selected elements in the selected view. For example, if you have a plank selected in floor plan and you click on *Elem. Drawing,* it will open an element drawing with the originally selected planks zoomed into view. These buttons will work with any ArchiCAD element type:

- *Floor plan* opens floor plan, showing the selected elements.
- *3D,* the same in the 3D window.
- *Element drawing* opens element elevation and selects ArchiFrame planks/boards from all projections. Please note that projection pieces are special 2D objects maintained by ArchiFrame.
- *Element object* selects the ArchiFrame element object owning current selection. If the selection contains just one or more *ArchiFrameElement*-objects, the button is changed to *AC element*. Then it will show the ArchiCAD elements where the geometry and *door & window* weights come.
- *Dimension drawing* opens the dimension drawing made with *Create and update dimension drawings*-tool.
- *Select same mat* selects planks with same material ID as selected on.
- *Same usage* selects all planks having given type/usage text given here:



- *Reserve for TeamWork* takes the selection and extends it to contain everything in the element including other layers, elevation drawings, floor plan pieces, dimension drawings and related ArchiFrameElement-objects. Then it reserves these elements for editing. After the editing is done, all changes are sent to the TeamServer with standard *Send & Receive* and finally released with *Release All*-button.
- *Multiply* opens ArchiFrame's multiply dialog that has some extra features compared to ArchiCAD stock one.

5 RenderLights Click N' Go



ArchiFrame contains integration to RenderLights visualization and virtual reality software. Please see http://www.renderlights.com/ for more information. RenderLights must be purchased separately and there is also a free trial version available.

ArchiFrame integration does following:

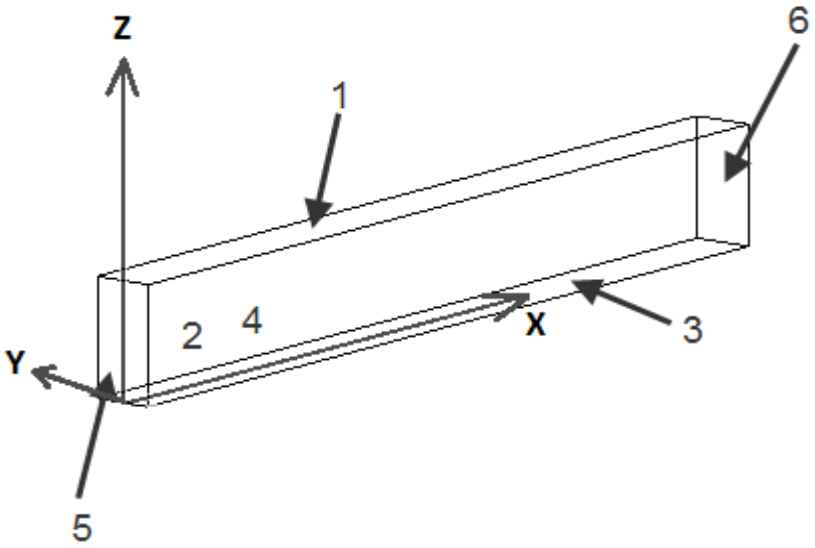
- When *RenderLights*-icon is clicked in the 3D window the current view is saved with current ArchiCAD file name extension changed to .dae. The textures are saved in a separate folder next to the dae-file. Then ArchiFrame launches RenderLights to continue working with the 3D-model.
- If the user presses *Control*-key or *Command*-key in Mac while clicking *RenderLights*-icon, it may point to where the RenderLights application is installed (RLPower.exe in Windows and RL.app in Mac).
- If the user presses *Shift*-key just the dae-file is saved, RenderLights is not launched.

6 Individual planks

6.1 About the planks

Planks can be copied with standard *ArchiCAD*-tools which makes creating intermediate floor structures, for example, easier from scratch. ArchiFrame observes changes in planks and removes mirroring planks maintaining the geometry intact. Because of this, planks should not be mirrored or edited if ArchiFrame is not installed.

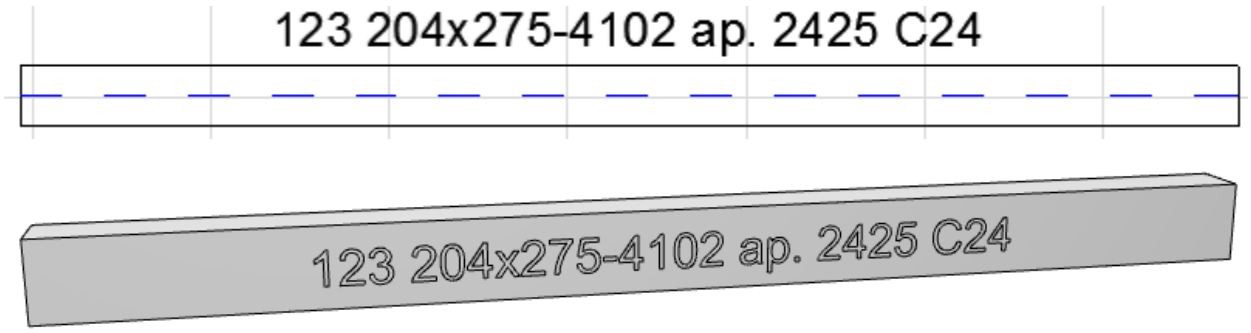
Plank reference line is at the bottom middle and the surfaces are numbered as:



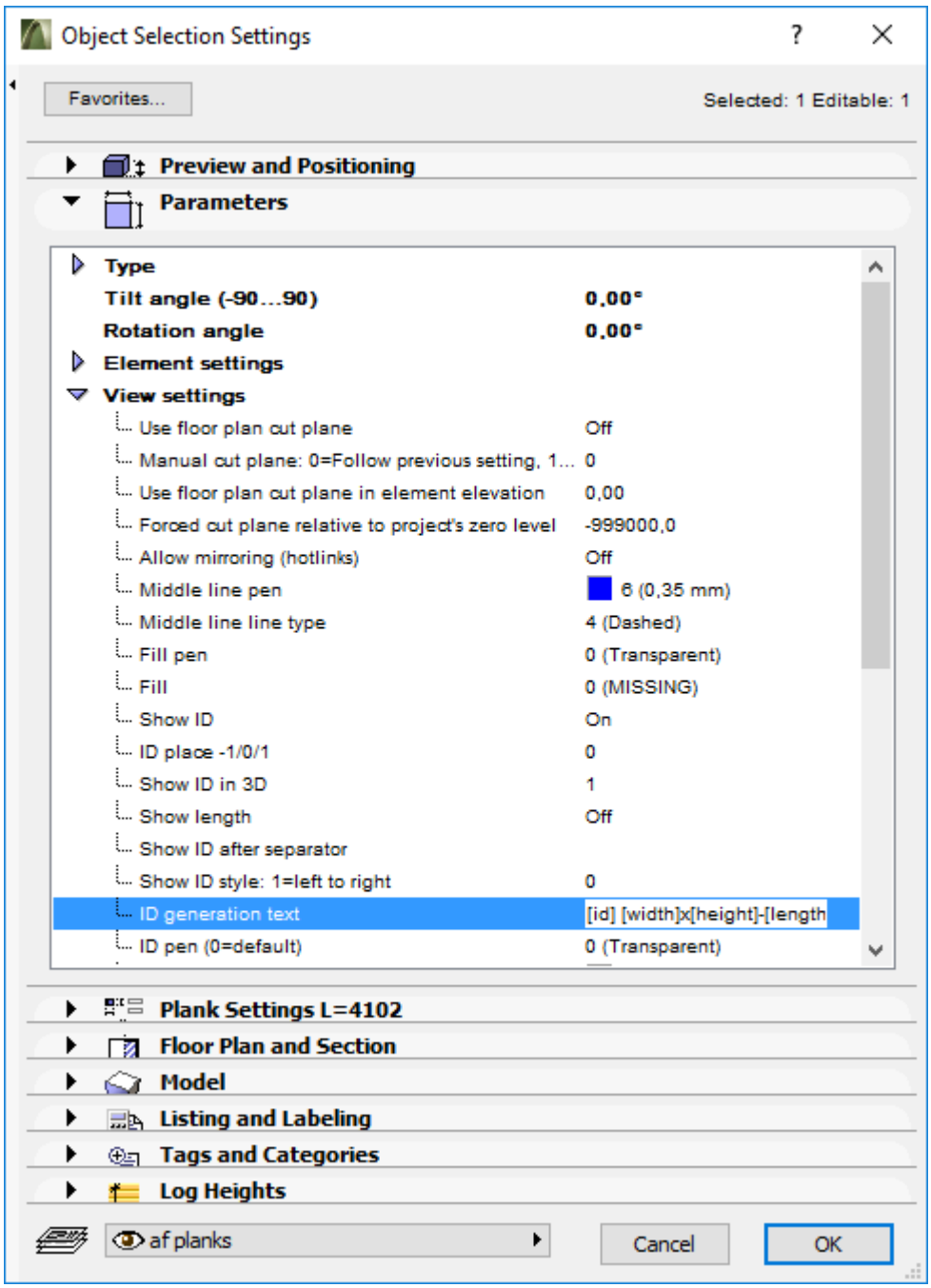
1. Top.
2. Front.
3. Bottom.
4. Back.
5. Begin.
6. End.

The reference line is the *X*-axis shown in the image above. The surface coordinates start at the front and back sides of the lower left corner from the direction of view. Bottom, back and end side coordinates (3, 4 and 6) are similar to the corresponding front surfaces. In other words, those surfaces are viewed through the plank.

Plank can show additional information instead of just ID:



Special IDs are generated according to plank object's ID generation text parameter (example value is: [id] [width]x[height]-[length] ap. [level] [quality]):



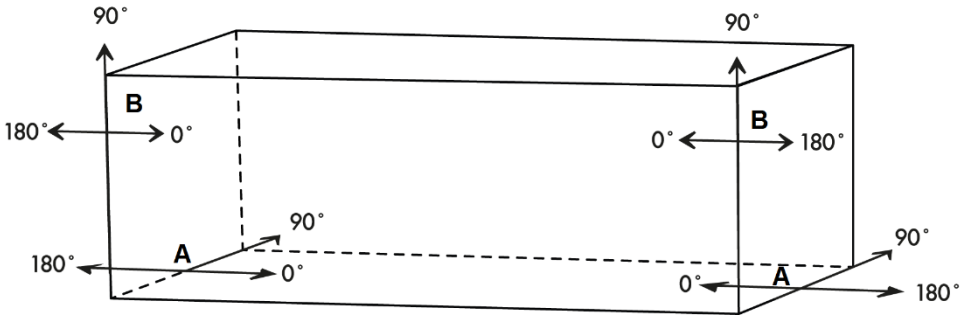
It can contain following tags:

- [id], plank's id.
- [matid], material id.
- [width].
- [height].
- [length].
- [level], global level at reference line.
- [quality], value of parameter Type/Quality class.
- [module], value of parameter cnc-Module (iElemModule).

6.2 Angled endings

The angled endings are specified with values:

- Angle viewed from top A.
- Bevel angle B.
- The plane is anchored with point defined by three coordinates:
    - X, distance from plank end, positive inside the plank and negative outside the plank.
    - Y, offset in plank's Y-axis.
    - Z, offset in plank's Z-axis.



For example, cut with 20 degrees bevel angle starting from half of the plank's height is defined with values:

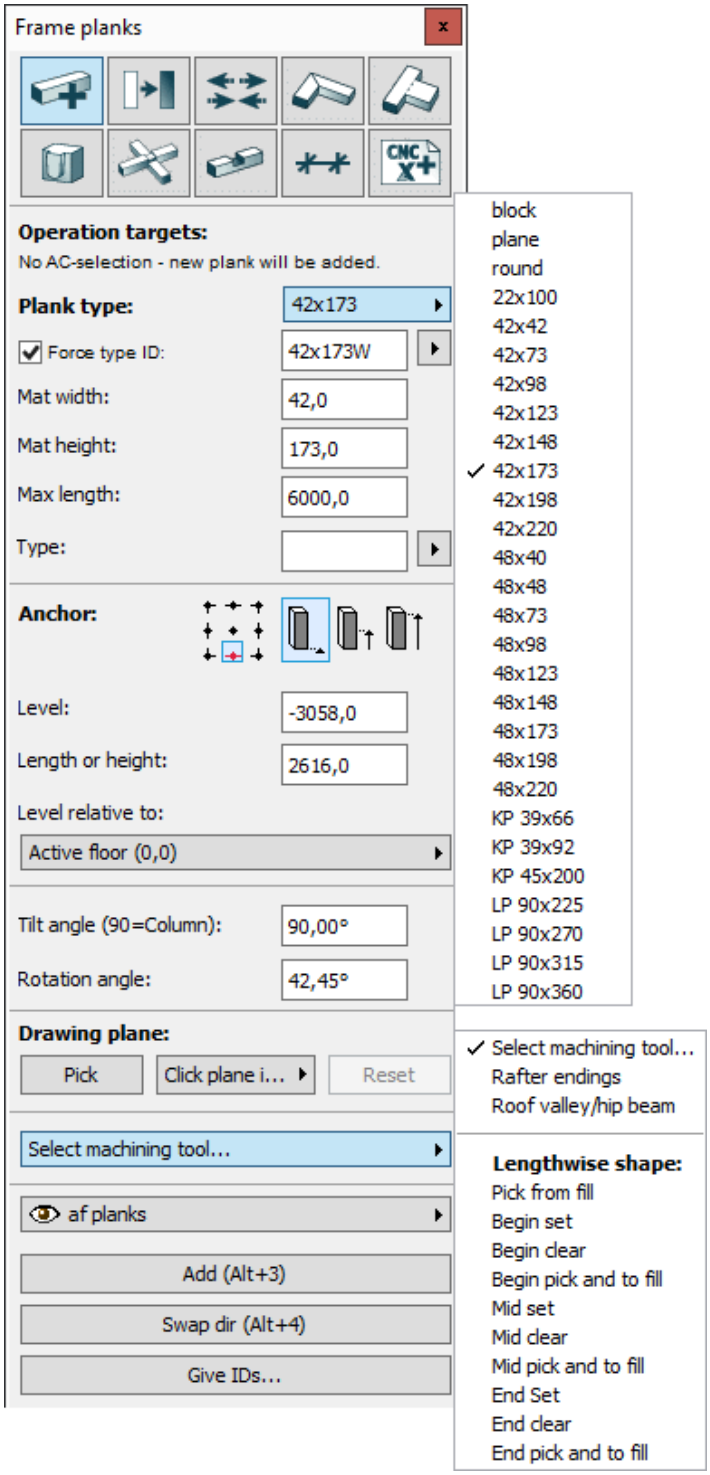
- Angle 90 (perpendicular).
- Bevel angle 160.
- X = 0, Y = 0, Z = height/2.

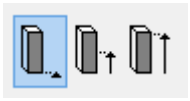
Plank may be cut with multiple angled cuts. ArchiFrame calculates the resulting plank length and adjusts the cut values as necessary.

7 Plank tools

7.1 Add and edit planks

First, the settings in this palette and ArchiCAD object settings are adjusted. If something is selected, then changes are saved to the selection. If nothing is selected, then a new plank is created for use at the chosen location. If placing a new plank, it will be selected by pressing *Alt*-key when adding it.



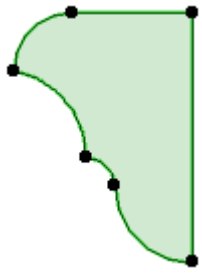
- Plank type can be either the full type or the base type if *Force type ID* is checked.
- *Force type ID* can be used if the plank type is seldom used and not worth adding to ArchiFrameBlocks.xml. Also, can be used for example to distinguish weatherproofed materials from normal materials. Dropdown list next to the forced type ID contains last saved forced types to quickly edit the planks. The dropdown list saves the base plank type, material size, maximum length and type text.
- Type is the usage id. It is used to distinguish, for example, weatherboards from other planks.
-  Anchor point defines the anchor in plank's cross section. This anchor affects adding the plank, changing its size (anchor point remains fixed) and the level.
-  Define anchor for the level in plank's direction: begin, middle and end.
- The tilt angle is lengthwise tilted from plank begin to endpoint. Positive tilt makes the plank rise and negative lower.
- The rotation angle is plank rotation counter clockwise viewed at the beginning of the plank.
- Drawing plane is useful for example to draw roof structures. After picking the drawing plane, the plank positions in floor plan (XY-coordinates) are kept and the level is adjusted to remain relative to the drawing plane when adding and editing the planks. It can be picked from:
    - Frame-plank.
    - Roof.
    - Beam.
    - Slab.
    - Tilted wall.
- *Swap dir* swaps the direction of the plank or board. All machinings remain in place. For cladding boards it swaps the front/back sides of the target.
- *Give IDs*, re-numbers the planks. The operation can be started with or without selection. More information here.
- Rafter endings, more information here.
- Roof valley/hip beam, more information here.

7.1.1 Lengthwise shape

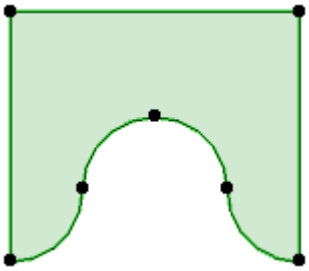
These tools allow to make for example a piece like below (note that using a lot of arcs will make 3D window slow):

The shape is picked from an ArchiCAD fill and then injected to target plank's begin/middle/end. In the middle the fill is automatically repeated from begin to end excluding possible begin and end shapes. End shape is drawn as begin shape – it is mirrored when used at the end automatically. The fill is automatically adjusted to the plank height if either top or bottom side is straight.

An example of the fill used at the ends:



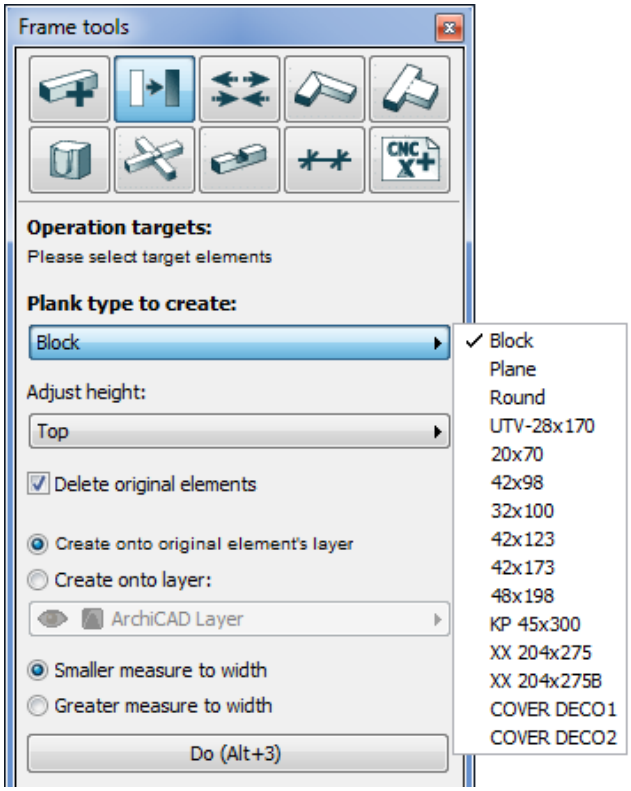
And the repeating middle part:



It is also possible to pick the shape from an existing plank with the commands Begin/Mid/end pick and to fill. In this command Esc can be pressed to avoid placing the fill into floor plan.

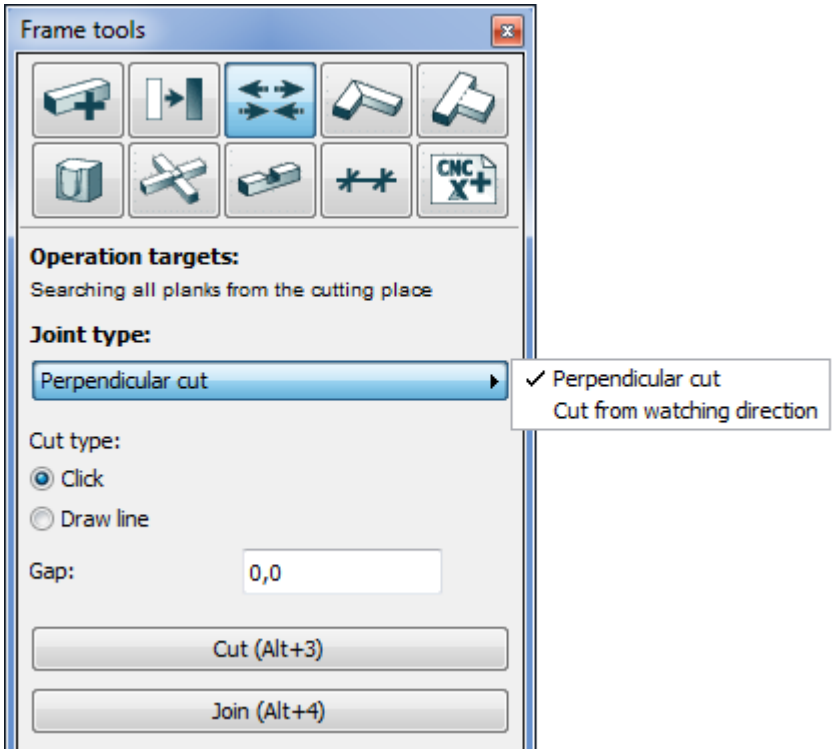
7.2 Convert to planks

Converts select ArchiCAD walls, beams or columns into ArchiFrame planks. First, the target elements are selected and the conversion settings adjusted. The first three plank types are generic and will result in exactly the same size as the original element. For example, a glued laminated beam with any size is converted to generic block since there is no matching size stock material.



Conversion groups elements into single group. Text (Alt+3) refers to the keyboard shortcut for the command.

7.3 Cut and join



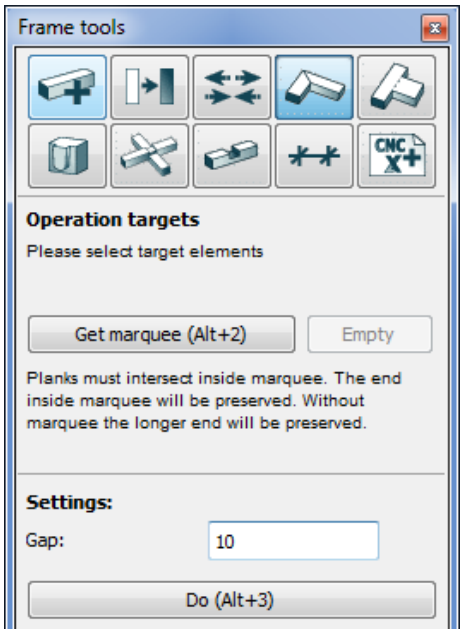
Target planks can be selected with ArchiCAD selection tool before cutting. ArchiFrame cuts the clicked plank. Alternatively, you can draw a line that will cut all planks intersecting that line. Joint type setting *Cut from watching direction* cuts planks vertically in 2D and at element watching direction if cut from element elevation.

To join planks the target planks are first selected and then these will combine planks at the same line, combining all machinings.

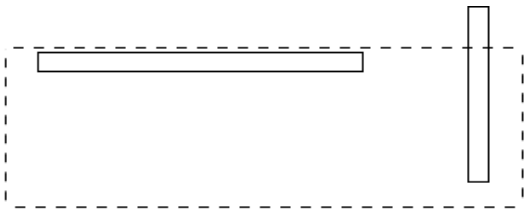
Press shift if you want to keep small pieces left from the operation.

7.4 Adjust endings

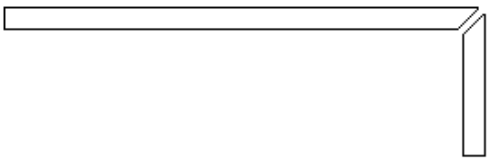
This tool combines planks from ends halving the angle and leaving a given gap between the planks.



Limit the working area by clicking on *Get marquee*-command, so that right hand side plank's lower end will be preserved:



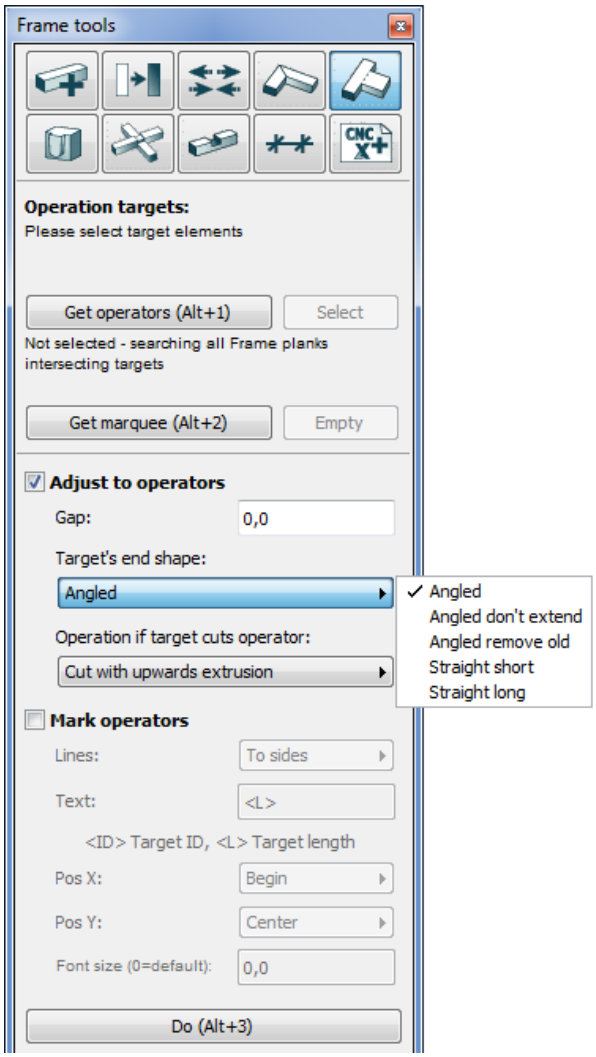
Select the planks and click *Do*. The result is here:



Without picked marquee ArchiFrame preserves the furthest end of the plank. In other words, the unchanged end is selected to produce the longest possible plank.

7.5 Adjust to operators and markings

This tool joins target planks to operators, or trims the targets to a roof for example. In addition, the tool allows the addition of markings to joining places.

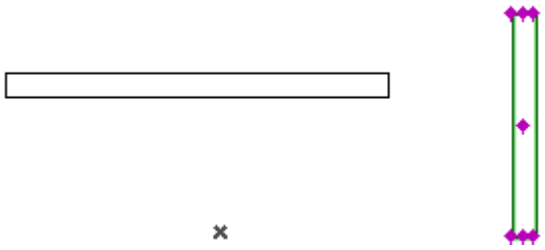


Target's end shapes are:

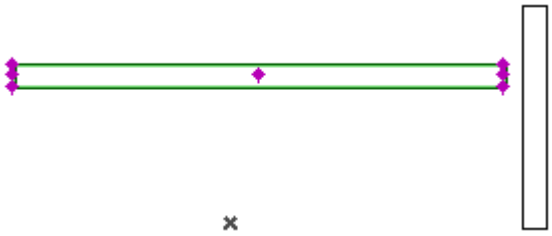
- Angled, extend and join to operator.
- Angled don't extend, cut with operator.
- Angled remove old, like angled but all existing angled cuts are removed.
- Straight short, straight ending to first intersection position.
- Long straight, straight ending to furthest intersection position.

For example, joining a horizontal plank to a vertical plank (in floor plan) with 100 mm gap.

1. Select vertical plank as operator (if operators are not set, ArchiFrame searches the closest intersecting plank for target plank).

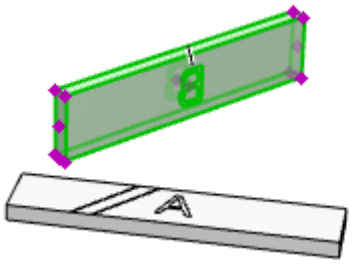


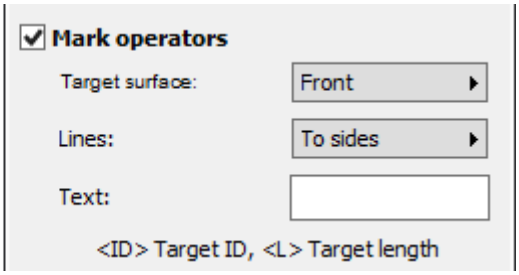
2. Select horizontal plank and click *Do*:



Press shift if you want to keep small pieces left from the operation.

Select target surface to mark planks even if there is no intersection:

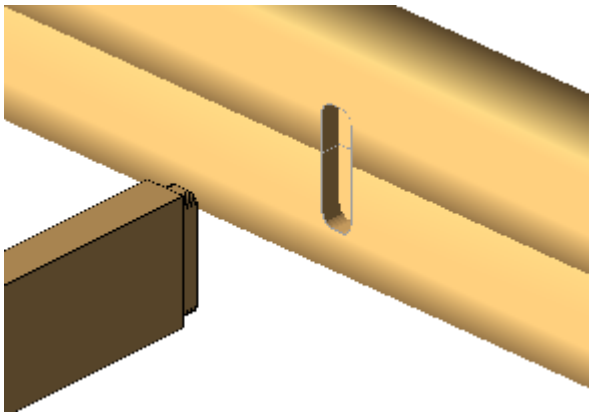




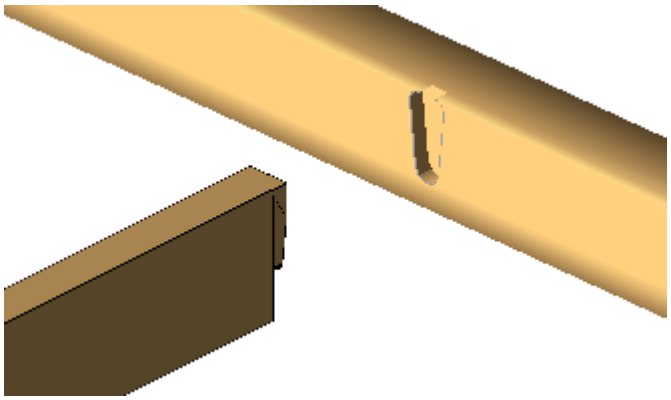
7.6 Dovetail, tenon and beam shoe joints

This tool is used to make mortise & tenon, dovetail and beam shoe joints. All joint types are done in the same way. Hidden shoe does not make any female machining and balk wedge does not make any male machining.

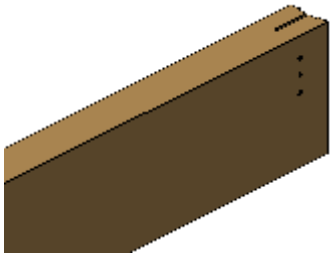
Mortise and tenon:



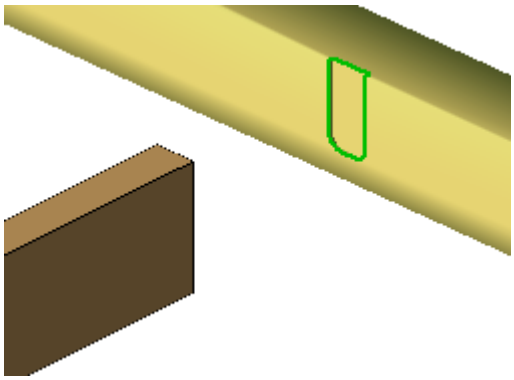
Dovetail:



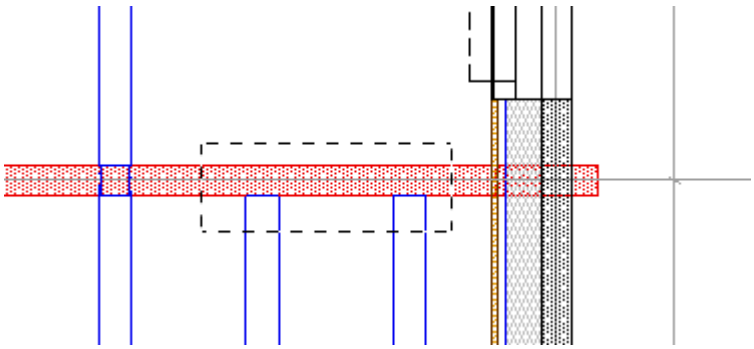
Hidden shoe, the accessory is placed next to the plank (operator) and the beam (target) is attached with dowels to the accessory:



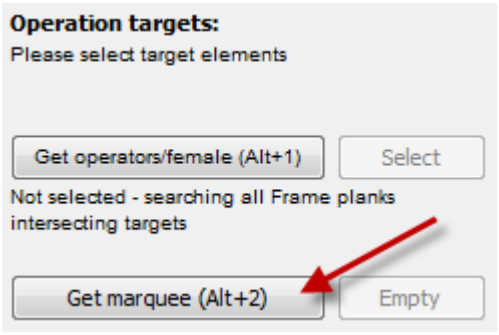
Balk wedge, U-shaped pocked to the operator and join accessory to the end of the beam:



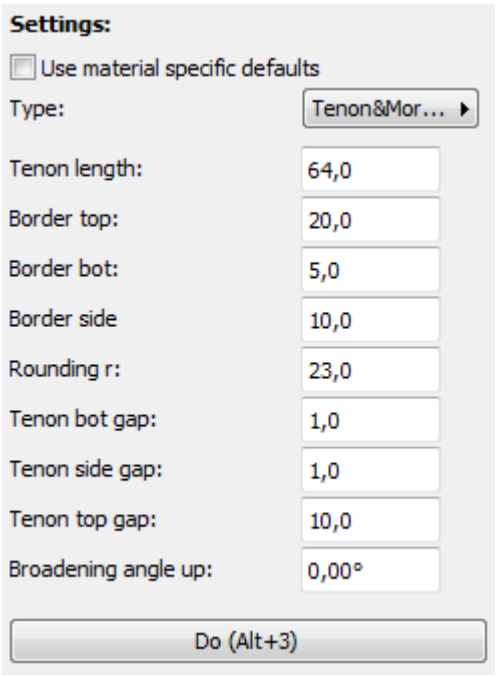
For example, mortise and tenon joint to two intermediate floor beams connected to ArchiLogs log objects:



Pick the marquee as example above to limit the operation only to the upper ends of the beams. Then select two target beams. The female parts can be picked with *Get operators/female* if needed. If the operators are not picked, ArchiFrame will search for every plank (or log) intersecting with targets that are inside the picked marquee, if there are any.



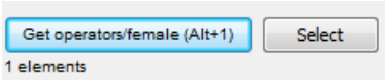
The default tenon length is half of the plank thickness. This value can be edited to match the female part. In this case the mortise will be placed at 128 mm thick log and the mortise depth will be 64 mm + 1 mm bottom gap. Rounding radius has two different values that affect the tenon: Zero makes rectangular tenon and any other value, a rounded tenon. With rounded shapes it is best to use the value that is close to the cnc-machine's cutter tool radius, for example 23 mm.



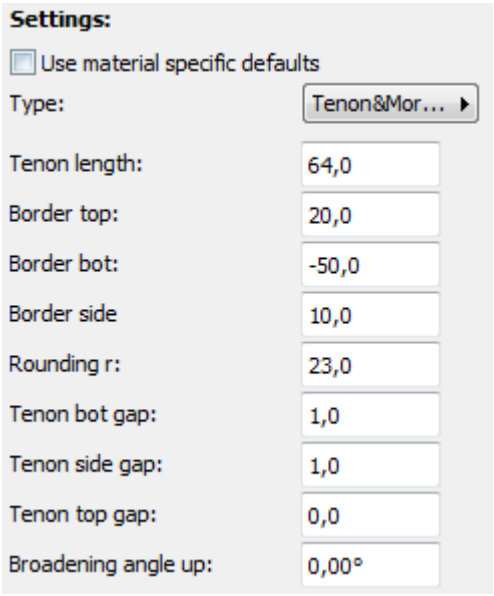
The result is like this in the female parts:

This is not satisfactory. Instead make tenons that are rectangular from bottom and that do not extend to the lower log. To do that, first manually remove the mortises from the logs.

Then select the upper log as operator/female part:



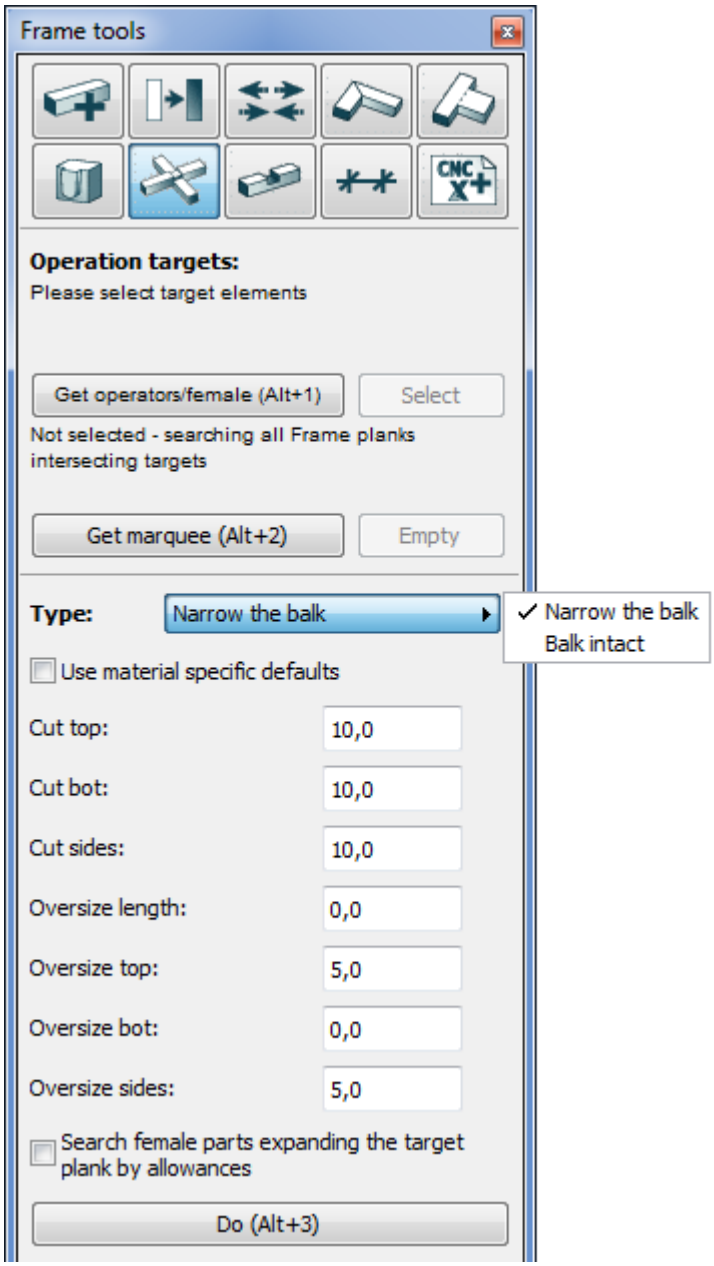
Measure the level difference between beam bottom and female part bottom. In this case, it is 10 mm. Set negative value to *Border bottom* to be sure that cutter goes far enough to make rectangular bottom:

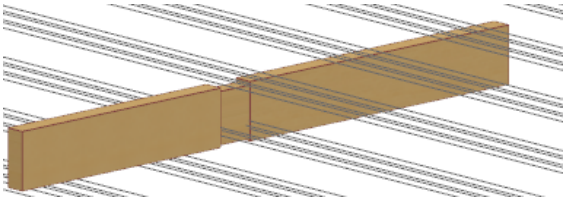


Then select the two beams and click *Do*. Note that the negative value in the bottom border causes a square shape at bottom of the joint.

7.7 Balk joints

This tool is used to make balk joints which have the balk either intact or narrowed. It also allows setting oversize for the female parts.

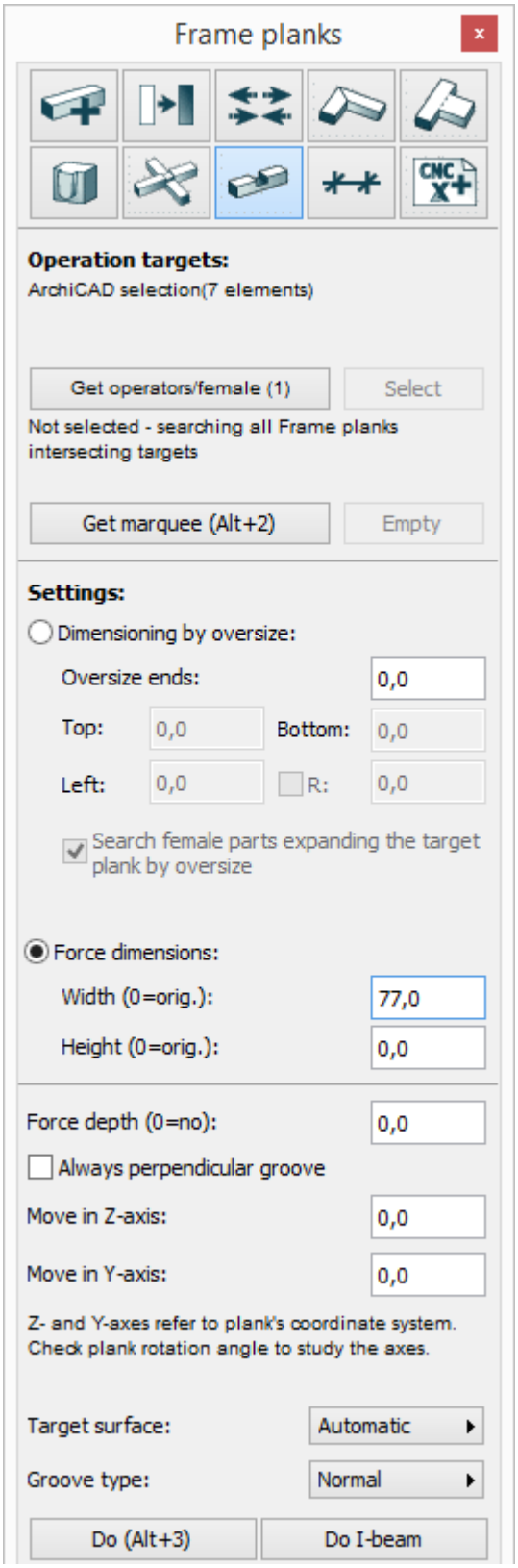




Narrowing values must be carefully adjusted for each situation. Oversize top can be used to set settlement space. Oversize length defines how much longer the narrowed part is compared to the female parts.

7.8 Grooves

This tool is used to make solid subtraction-like operations with selected elements (male parts). Male parts can be Frame-plank, log object, ArchiCAD wall or beam. Roof is also acceptable, if its slope is rectangular and parallel to the reference line. Female parts must be either ArchiFramePlanks or ArchiLogs' log objects.



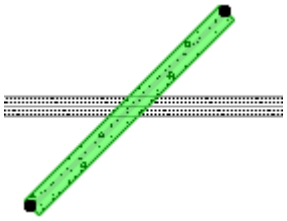
The tool resembles with the balk tool except for the following differences:

- Selected male elements are not changed – those are used to make grooves to female parts.
- More adjustments to create grooves.

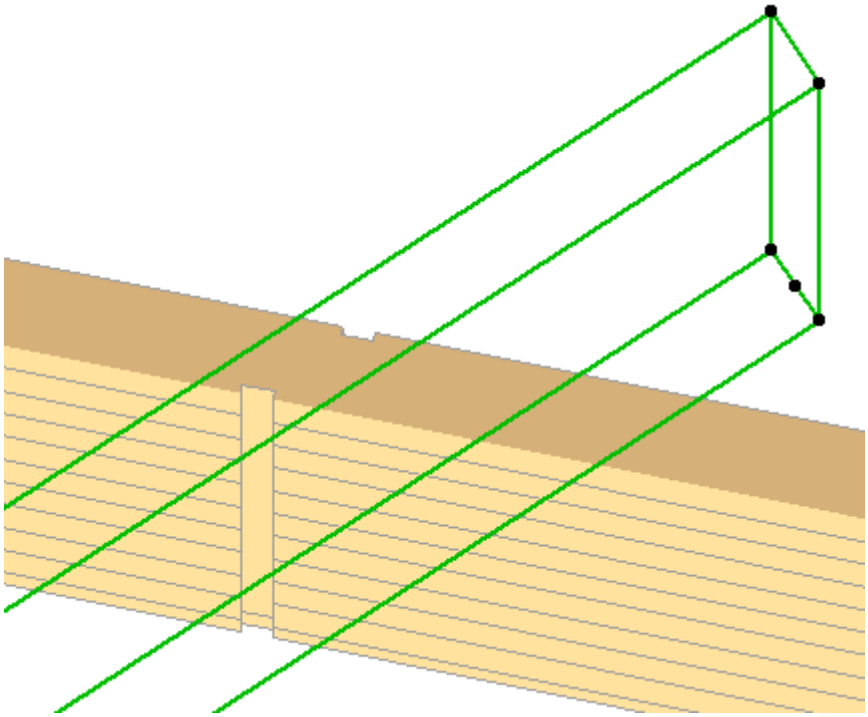
When working with *Dimensioning by oversize,* given values are added to male part sides. If option *Search female parts expanding the target plank by oversize* is not checked, ArchiFrame searches the intersecting female parts with original male part size. This can be used to ensure the female part will be cut, leaving a piece above or below untouched.

Force dimensions option changes male parts to a given size, keeping the reference line unmoved at the bottom middle. In other words, the bottom side and middle line remain at the same place and top side and sides are moved according to forced values.

Force depth makes the groove a given depth. If the male part intersects the female part, the female part remains intact but with forced depth grooves to both sides. Forcing depth works only for the male parts X-axis. For example, let's make 8 mm deep and 77 mm wide grooves in following situation to a log wall:



This is the result the male part on wireframe layer:



See also I-beam grooves.

7.9 Dimension tools

This palette contains two sub-parts: *Create and update dimension drawings* and *Dimensioning tool.*

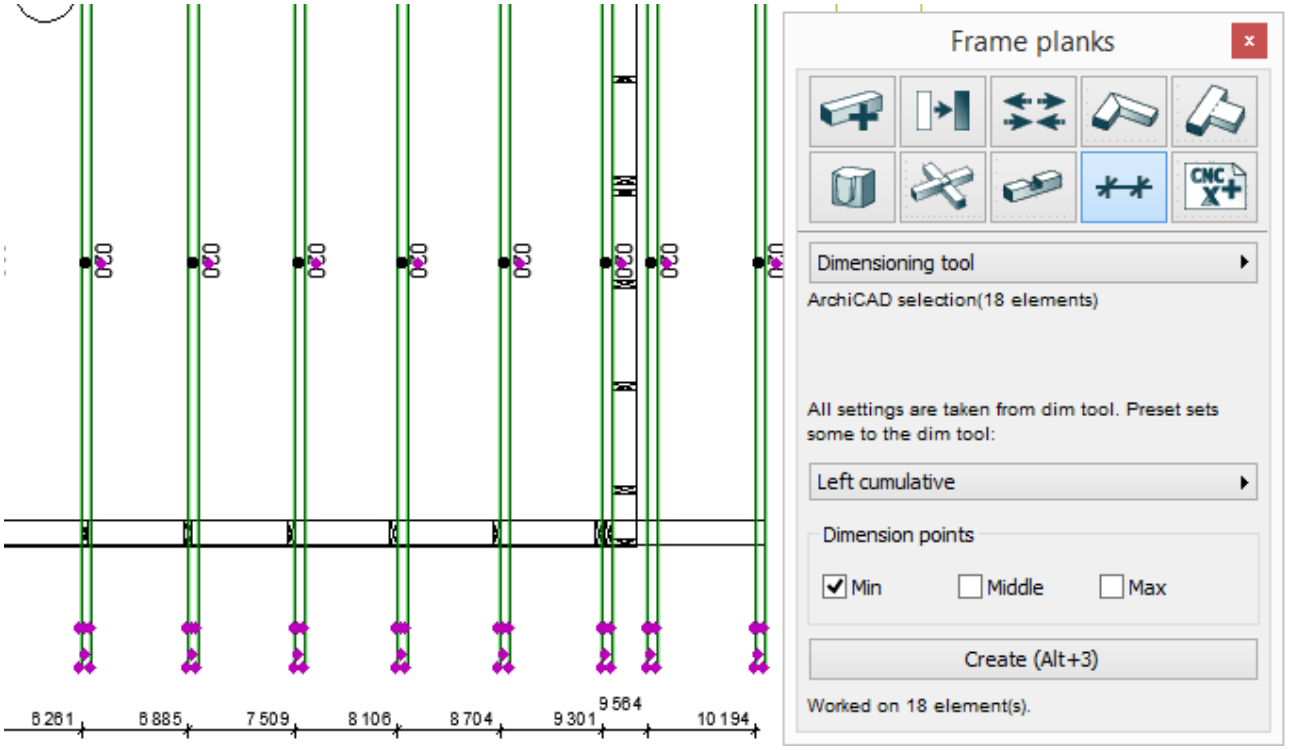
7.9.1 Create and update dimension drawings

This tool is used to make detailed dimension drawings for selected planks. If the operation is started without any ArchiCAD selection, ArchiFrame will update all existing changed dimension drawings. To force updating all dimension drawings, the *Ctrl*-key needs to be pressed while clicking *Update*.

Dimension drawings are not updated automatically and changes to dimension drawings won't be reflected in the original planks.

7.9.2 Dimensioning tool

This tool is used to make a dimension line for selected elements. Pre-settings are loaded from current *data*-folder's file ArchiFrameBlocks.xml. For example, cumulative dimension line anchored to the left sides of the rafters:



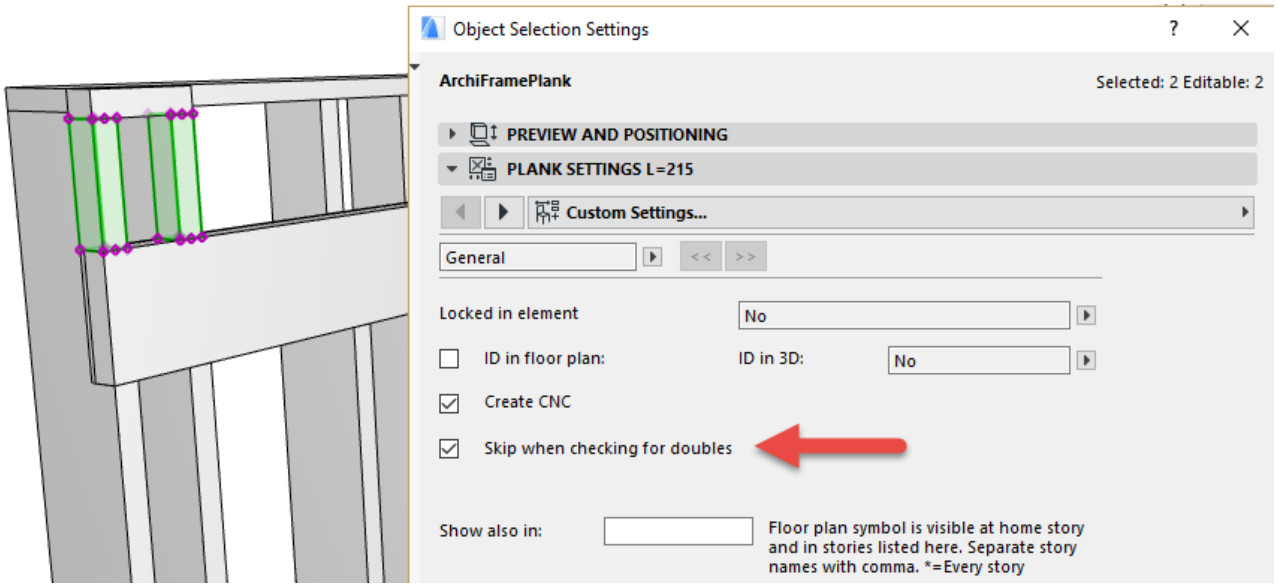
Dimension points define which places of the elements are added to the dimension line in dimension line's direction. Due to technical limitations, it is possible to add automatically updated dimensions only for objects. For other types, the dimension line will be static even if not selected from the dimension tool. This tool can be used with following ArchiCAD element types:

- Walls.
- Columns.
- Beams.
- Roofs.
- Objects.

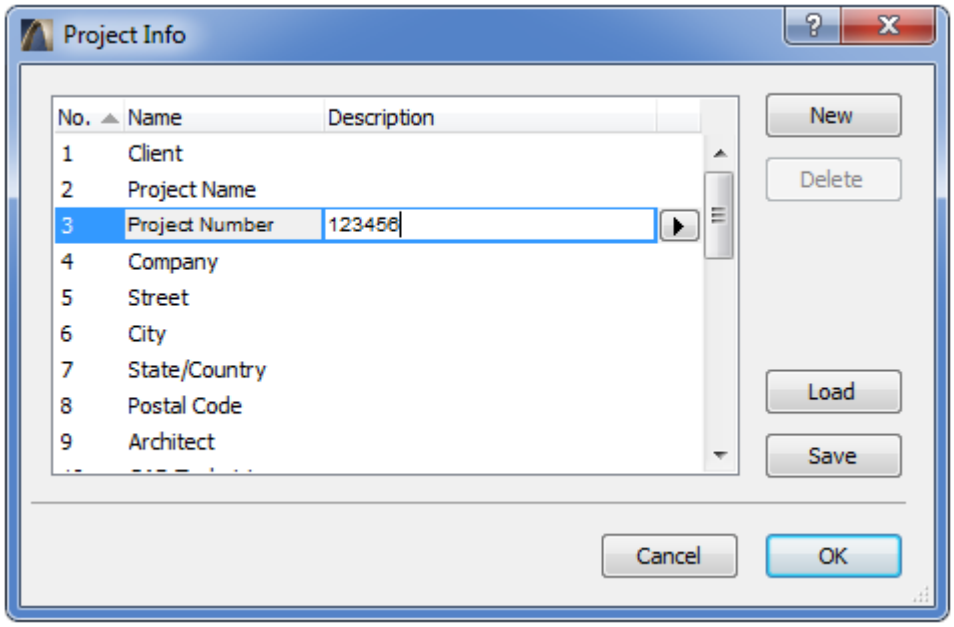
7.10 Xls- and cnc-files

These tools are used to produce cnc-file and quantity takeoffs either in txt or Excel format. Saving as an Excel file only works in Windows if there is Microsoft Office 2003 or newer installed.

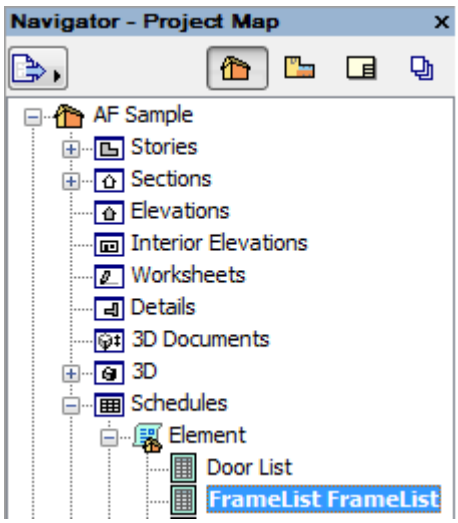
Check for doubles will scan either active storey or the whole model based on the selection. It will select all found doubles. This check is automatically made when writing any listing or cnc-file. ArchiFrame does just full plank collision checking skipping possible cuts and grooves. It is possible to skip false reported planks from plank's settings:



Excel´s default file name is the same as the current pln-file name with _excel.xls added. Cnc-file default name is taken from *File / Info / Project info / Project Number*:



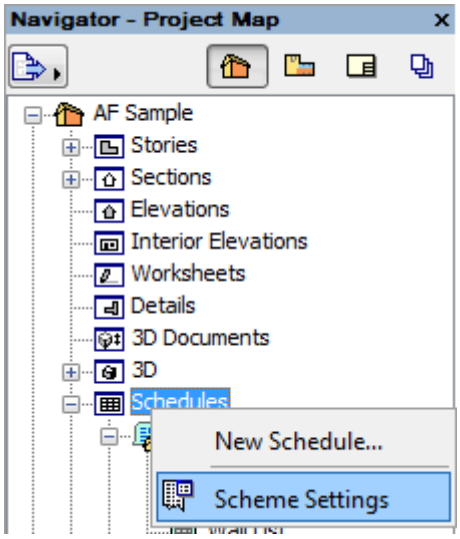
It is possible to use ArchiCAD *schedule*-tool also. Please note that grouping similar planks must be left to ArchiFrame's *Give IDs* feature (or automatic for elements). With ArchiCAD schedule it is impossible to distinguish similar planks with different machinings.



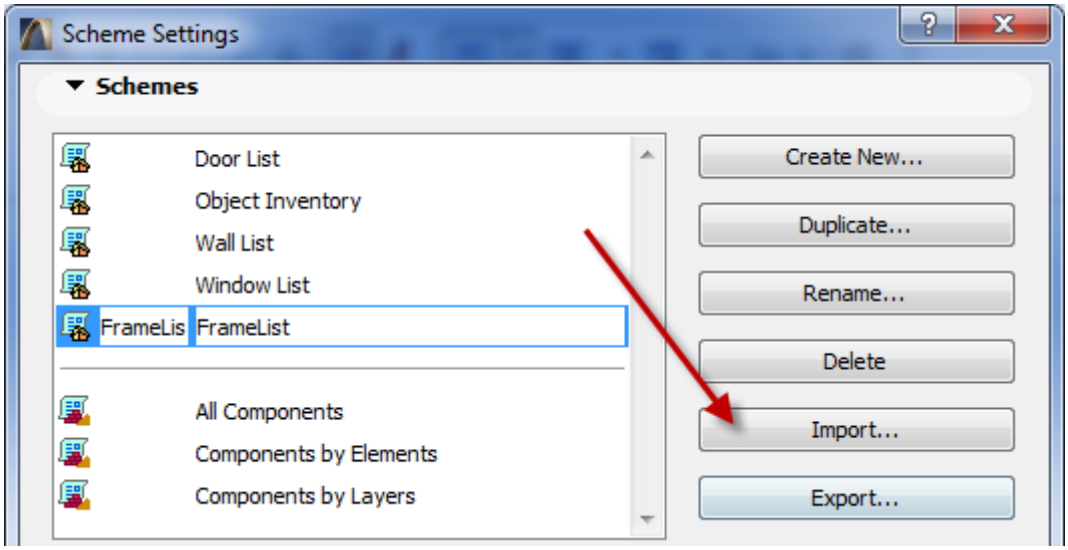
FrameList					
ID	Mat ID	Width	Height	Length	3D Front Axonometry
	I 300	47	300	4 744	
A	block	50	200	2 050	
B-001	45x45	45	45	2 509	
B-002	45x45	45	45	2 415	
B-003	45x45	45	45	6 077	
B-004	45x45	45	45	1 210	

The sample schedule is in *samples*-folder with name FrameList.xml. It can be imported to the actual model with these steps:

- Right-click Schedules in Navigator and select *Scheme settings*:



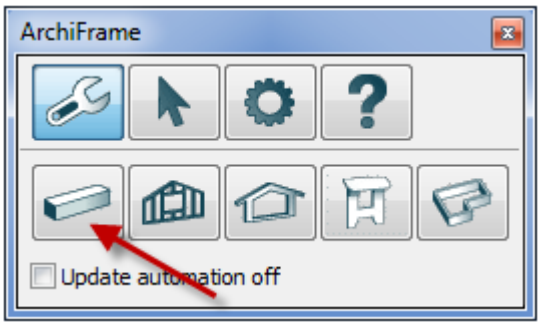
- Import FrameList:



8 Extended tool palette

8.1 General

Extended tool palette opens by holding *Ctrl*-key down when clicking the plank tools icon (in Mac press *Option*-key):

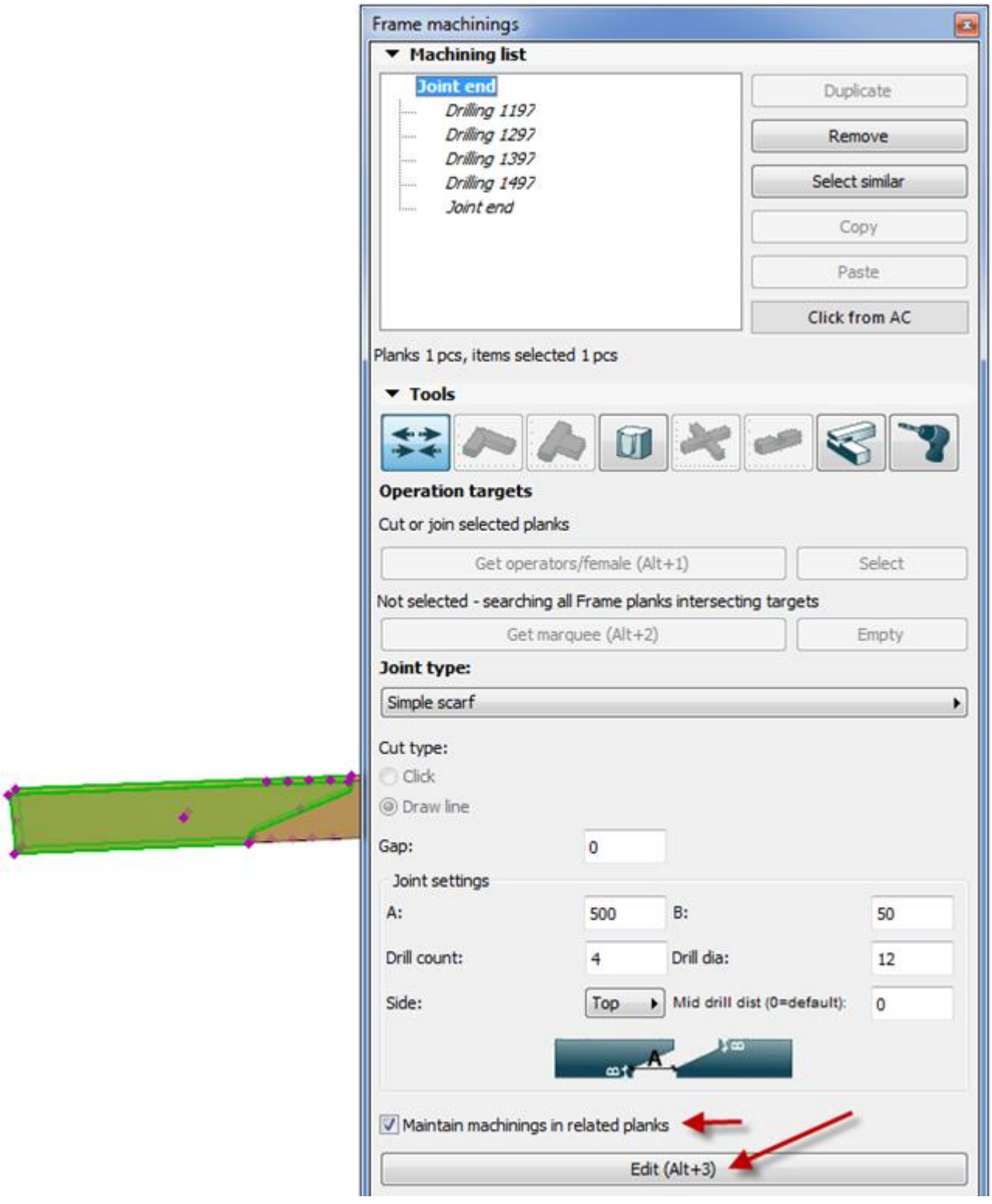


Extended tools have the option to update the machinings automatically. For example, in mortise and tenon joints moving the tenon part moves automatically the related mortise. Using the cut tool moving plank's end moves automatically also the related end.

When copying planks having automatic machinings, these rules are valid:

- If both parts of the machining are copied, the operation is copied to the new planks. For example, if a pair of lap jointed rafter is copied, the lap joint is also copied.
- If only other of the parts is copied, the operation is copied only if it is valid for the new plank. For example, copying groove female part will have the groove if new part intersects with other part.

Existing machinings are edited by first selecting the machining to be edited, adjusting the values and finally clicking *Edit*:



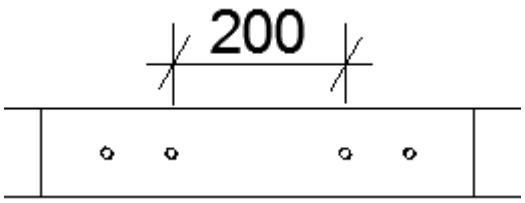
Greyed tools will be implemented later. Buttons related to machining list are:

- *Duplicate*, duplicates selected machining. The machining is edited at the bottom of the palette and finally added with *Add*-button.
- *Remove* removes selected machining(s).
- *Select similar* allows user to quickly expand the selection for example to all grooves.

- *Copy* saves selected machining(s) to be later pasted into other parts.
- *Click from AC* allows user to click the machining's hotspot in floor plan or ArchiFrame element elevation and then selects the clicked machining from the list. Works in AC 22 and newer versions.

8.2 Cut and join

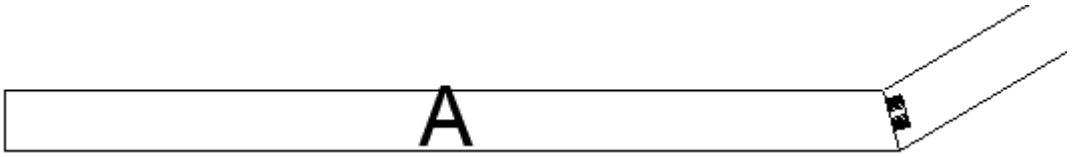
Please see Cut and join. In the simple scarf it is possible to give middle drill distance (for example with 200 mm):



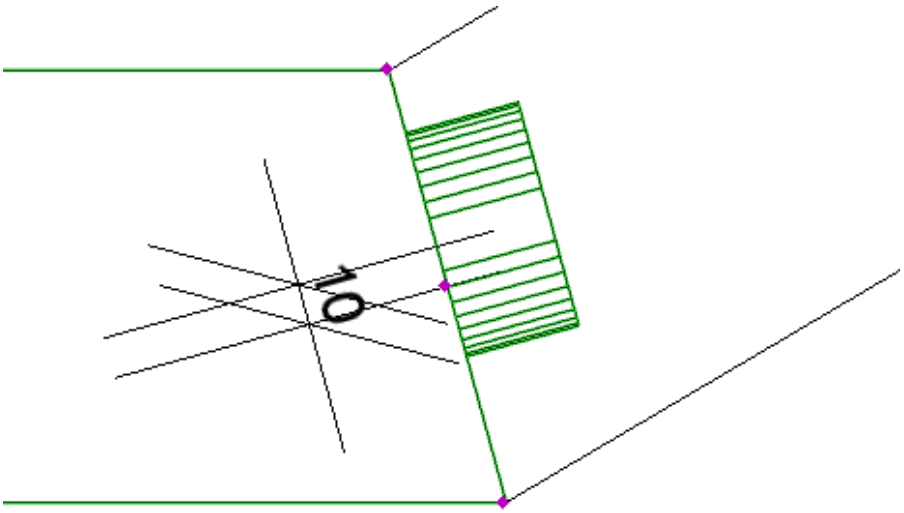
8.3 Dovetail, tenon and beam shoe joints

Settings:

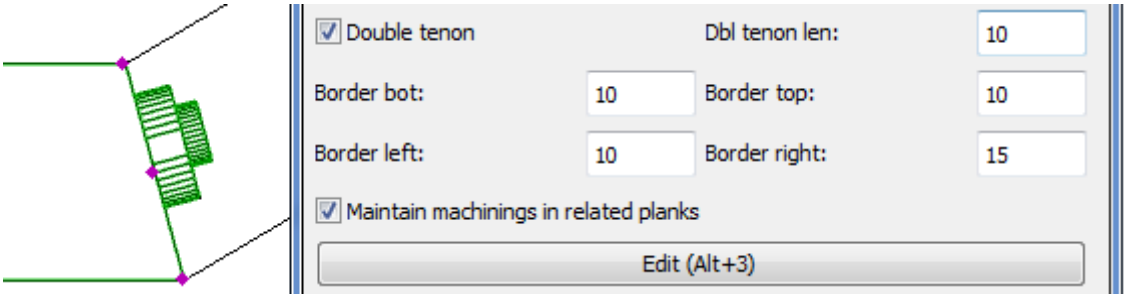
- Join to Side: Makes the joint always to operators' sides.
- End or side: Makes the joint to the end, if male plank's middle line intersects with operator's end.
- End and join: Joins the pieces together from ends and adds selected joint:



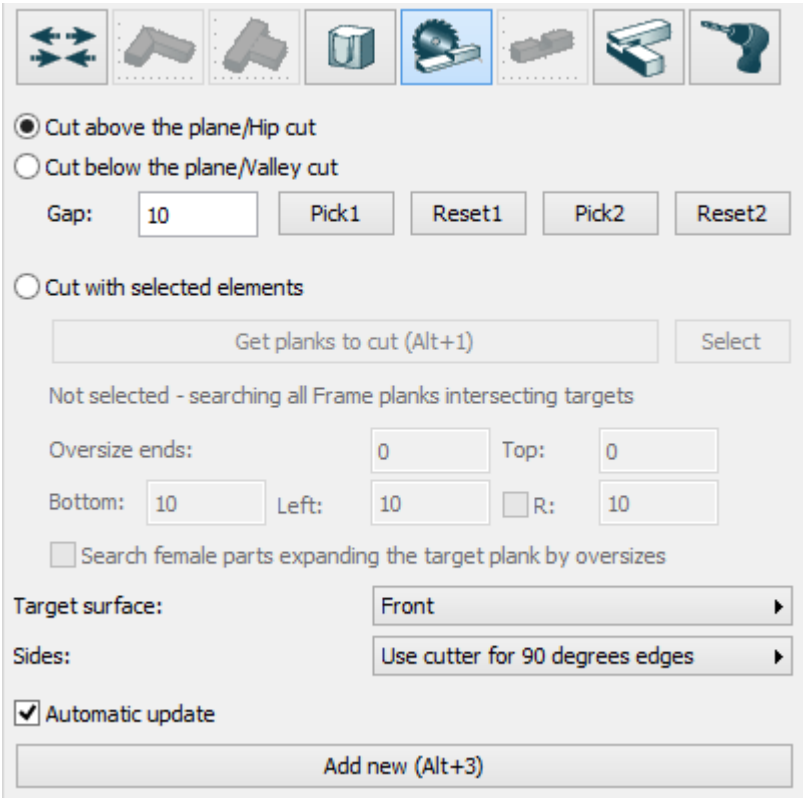
- Tenon length: Tenon total length also if double tenon is checked.
- Border bottom: Border for the tenon anchor side.
- Border top or Tenon height: Border for the opposite side to the anchor side or tenon height directly.
- Border sides or tenon width: Tenon width defined by border or by direct value.
- Rounding r: Zero makes rectangular tenon.
- Tenon bot gap: How much deeper mortise is compared to tenon.
- Tenon side gap: On both sides.
- Tenon top gap: Gap at side opposite to the anchor side.
- Broadening angle up: Mostly used with dovetail.
- Move tenon right: Offset from middle to right, negative value moves the tenon to left:



- Tenon anchor/dir: For example, upwards makes dovetail to widen upwards. For special cases the bottom side can be anchored to given plank side.
- Tenon rot angle: Counter clockwise angle looking in tenon direction.
- Double tenon: The borders for the end tenon. Mortise is bottom depth deeper than the tenon also in both the base and end tenon. Watching direction is the tenon direction also in this case:



8.4 Lengthwise cut

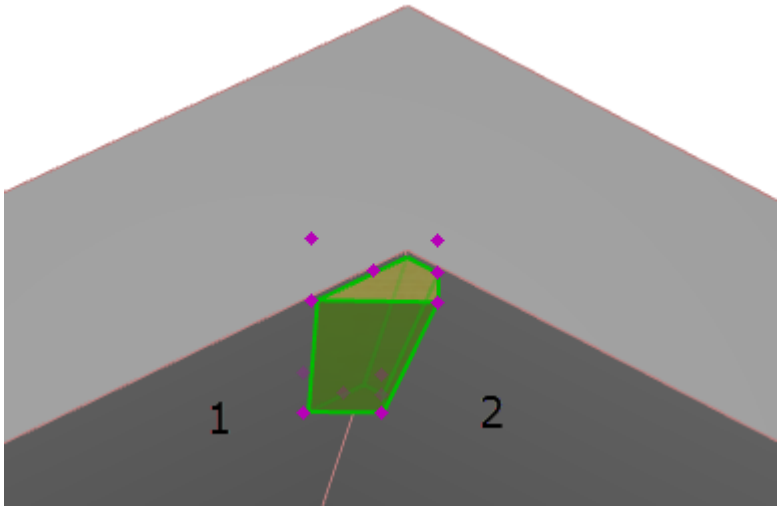


This tool is meant to produce lengthwise cuts made with a saw in resulting cnc-file. The cut must be parallel to the plank and it may be single or double cut. Use *groove*-tool to make free form cuts (made with a cutter in resulting cnc-file).

The tool can be used in two ways:

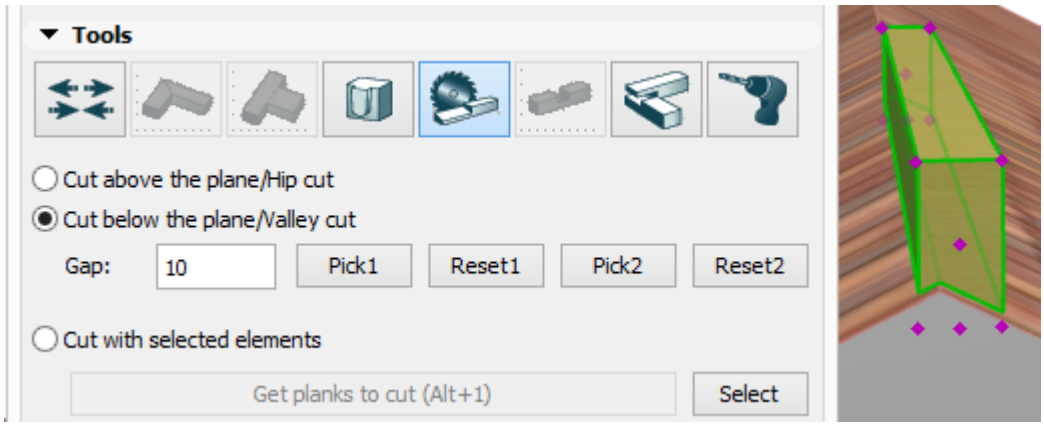
1. Pick static cutting plane or two planes from existing elements to make single or double cut. In this case the cut can be automatically updated if target plank is moved.
2. Cut target plank with another plank or ArchiCAD wall, roof, slab, column or beam. If cutting piece is a plank, the operation can be automatically updated.

To produce cutting like this:

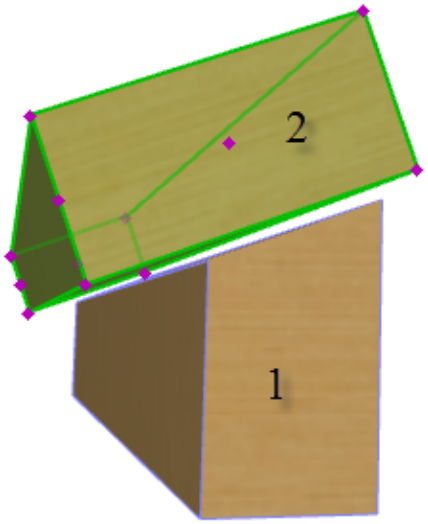


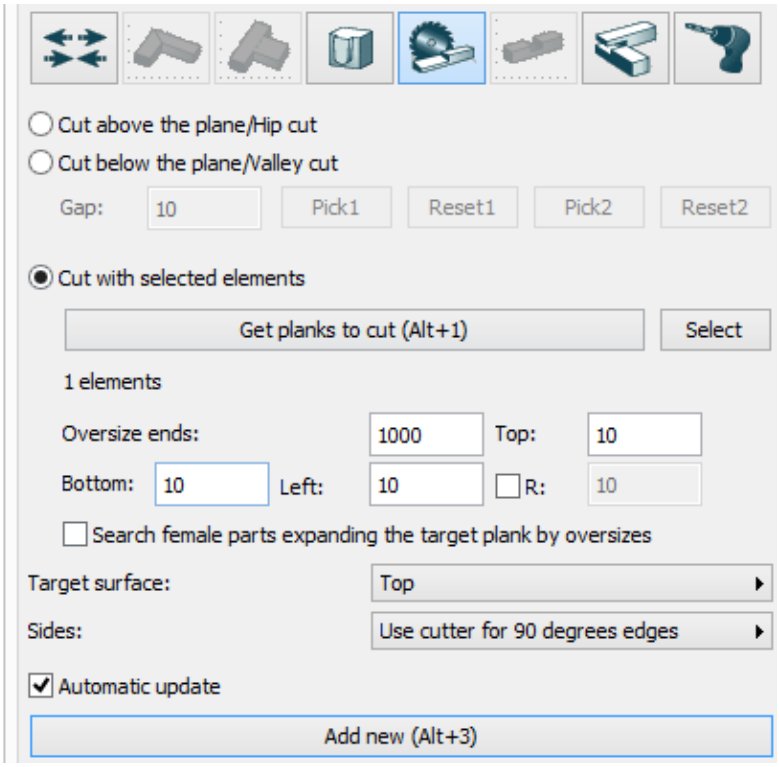
- Use settings above.
- Click on the button *Pick1* to pick cutting plane 1 (left hand side roof lower surface).
- Click on the button *Pick2* to pick cutting plane 2 (left hand side roof lower surface).
- Select the target plank and set target surface to front in this case (plank has been rotated 90 degrees counter clockwise, the reference line has pink hotspot).
- Click *Add new.*

V-cut is made the same way picking the cutting planes from roof upper surface and changing cut type:



To cut with planks like below:

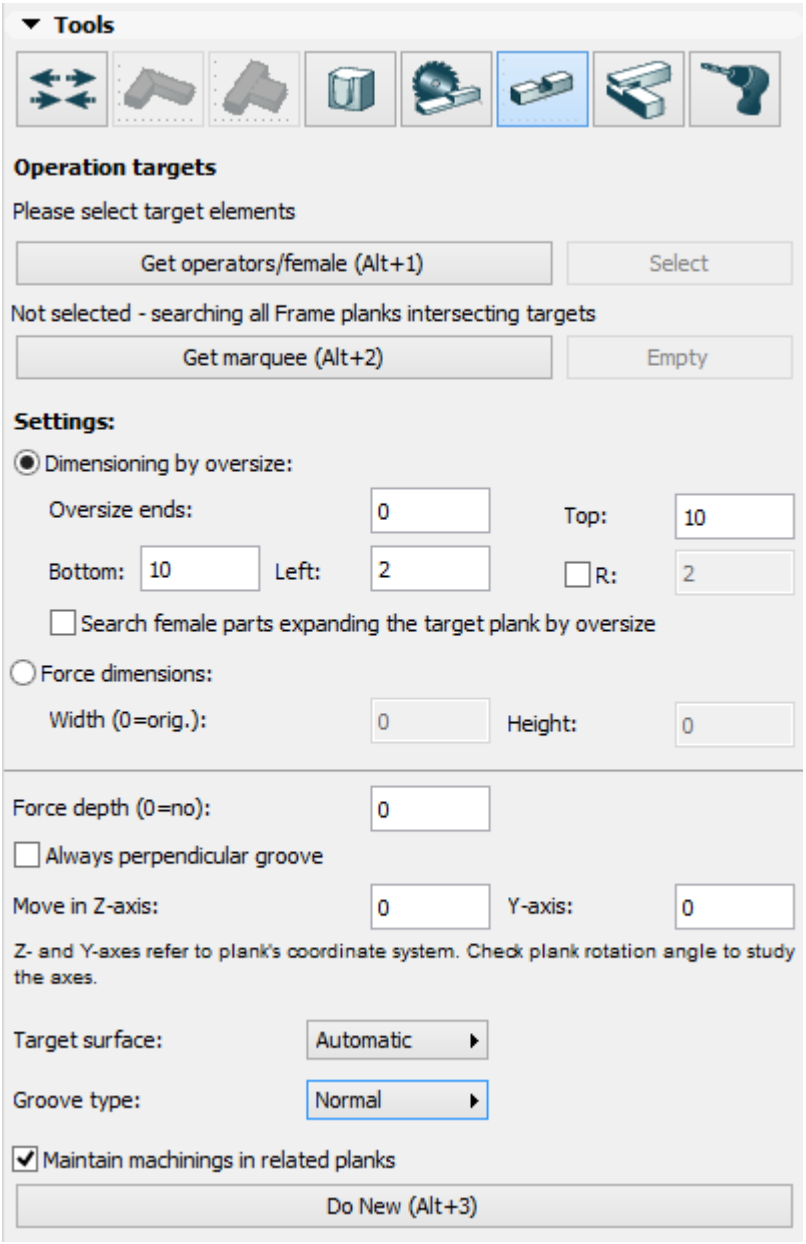




- Select plank 1 and click *Get planks to cut.*
- Adjust oversizes and other settings.
- Select cutting piece (2).
- Click *Add new.*

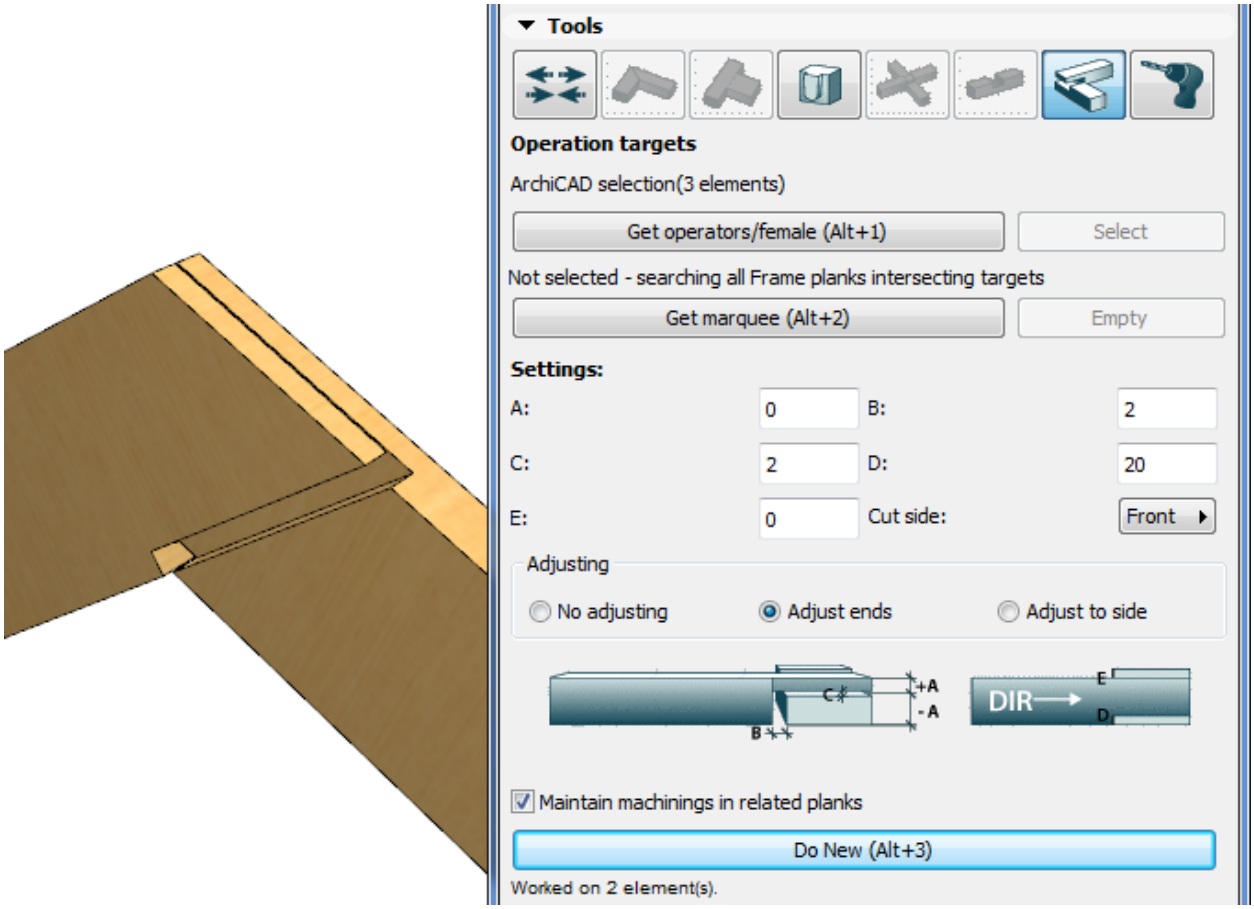
Sides-setting controls resulting cnc-: If the cut is not full plank's length, the cutter may be used to remove the piece and to make sharp corners. Final result depends on the actual cnc-writer.

8.5 Groove



Please see Grooves.

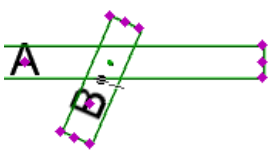
8.6 Lap joint

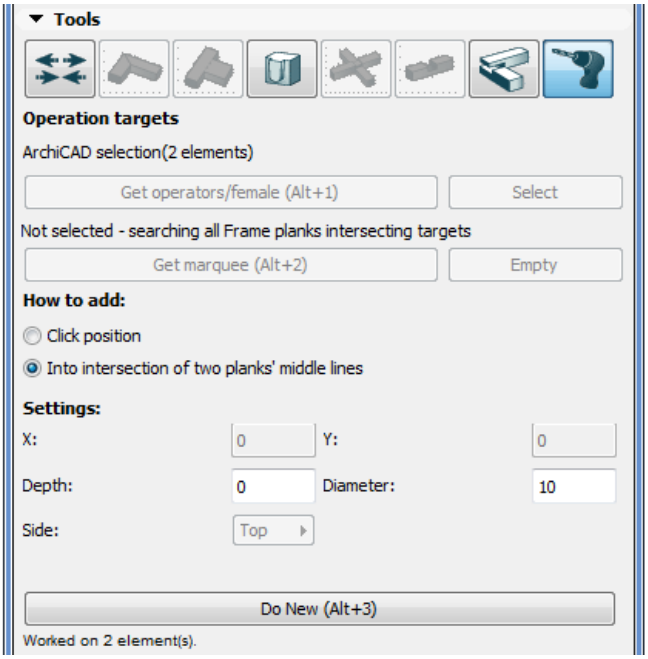


For the lap joint the plank pairs are selected (for example one or more rafter pairs), the settings are adjusted and readjusted if any specified. Then the joint is done with *Do New*.

8.7 Drillings

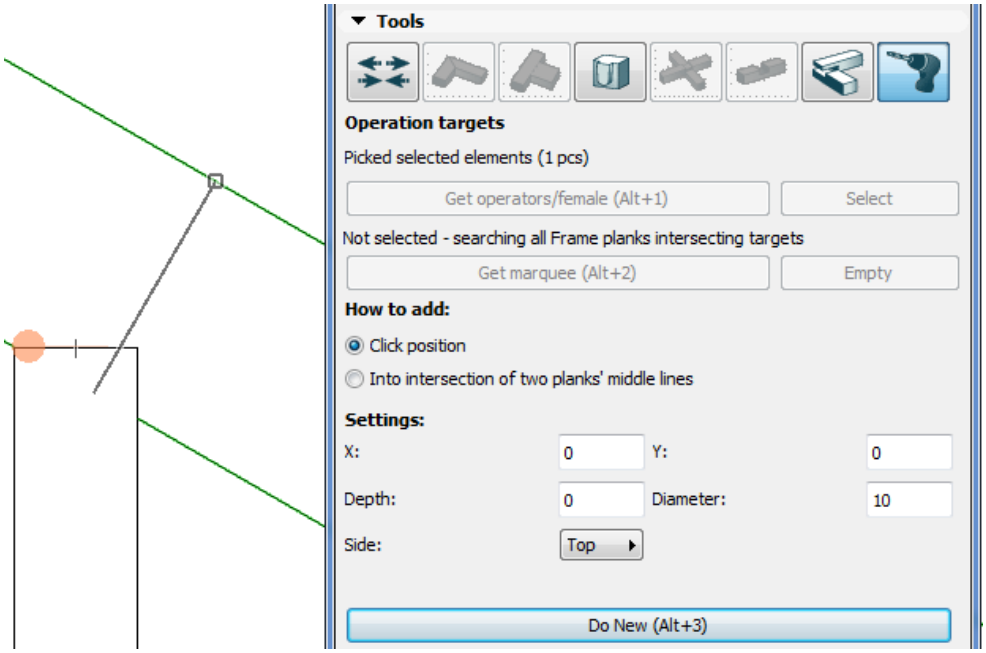
The simplest way to add a drilling is to place it to the intersection of middle plane of two pieces:





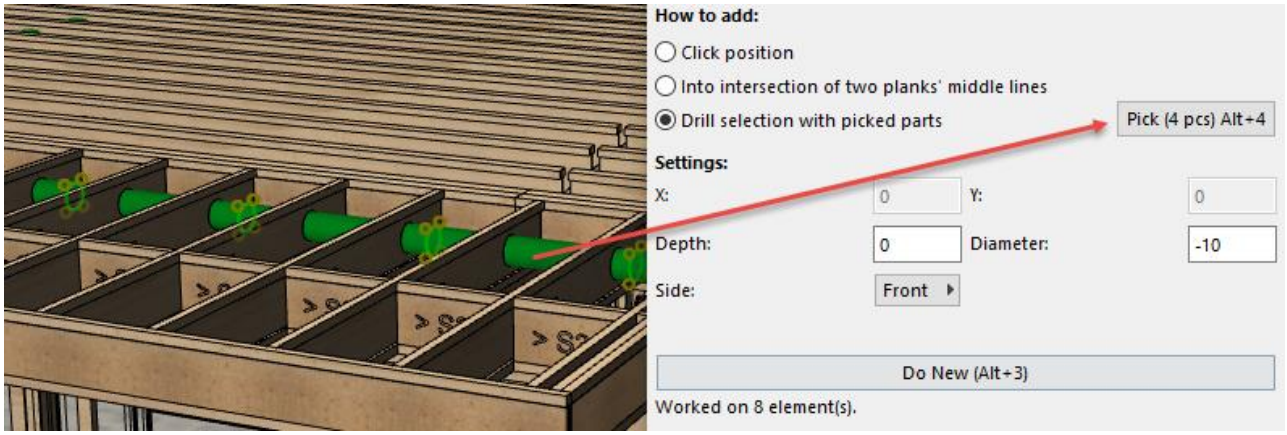
Zero depth means drill through.

Using *Click position* in elevation is the easiest way to add a drilling to a rafter based on supporting beam placement:



- X: Drilling position on target surface X-axis. With value zero the position is at the middle of the axis if the current view does not allow pointing the X-axis.
- Y: Similar for Y-axis.
- Depth: Value zero makes the drill through.
- Diameter: Drill diameter.
- Side: Target surface which must be given when using *Click position.*

Third option is to work in 3D and use any 3D element as drilling piece and drill the selected pieces. For example, here the green tubes are picked to make the drillings, then target pieces are selected and by clicking *Do New* it will add the drillings based on the 3D-information:



In this case negative value in *Diameter* defines the oversize used in the drill (diameter is 10 mm bigger than the green tube). 3D representation of round pieces is inexact, so it may be better to force the diameter. ArchiFrame takes the section of picked parts on the surface defined in Side-setting.

9 Elements

9.1 Element object

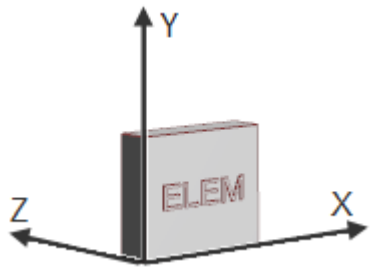
Element object represents the orientation and shape of a single structural area in a building. There is one element drawing per element object. The single plank can be visible in none or many element drawings. If the house is built on-site it is best to set the element object to contain the whole wall. If the elements are manufactured at the factory, then element object models one element. Element object IDs are set to planks that the element owns. The plank ID will have the form: [element id]-001 where 001 is number inside the single element. Similar planks will have a similar ID.

The process to make framed walls (and other structural elements) is:

1. Place element object and edit those to give correct corners and general shape.
2. Create planks to elements.
3. Attach required beams and supporting columns to individual elements.
4. Check and edit the element elevations. Lock the key planks and recreate planks if necessary.
5. If the element geometry changes, the planks must be recreated with *Create planks* command.
6. Use *Update* without any AC selection to recreate all elevations (recreates every dimension line).

Changes to the original AC-elements (walls, roofs, slabs) are not reflected in the element objects or from those to framed elements, so updating must be done manually.

Element coordinate system is:



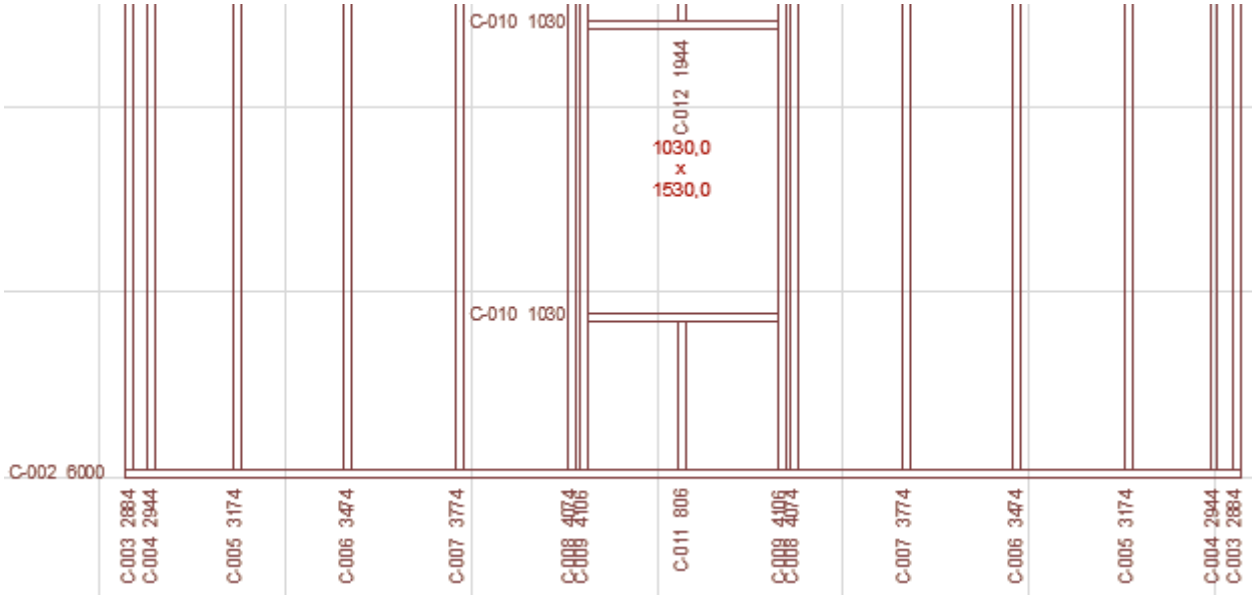
The main elevation viewing direction is the Z-axis. The element is always viewed from its right side.

9.2 Planks and element elevations

Elevations contain projections of the original planks. These original planks are called master planks. ArchiFrame keeps the master planks and projected planks synchronized. In the projections it is possible to move, drag plank ends and edit the planks. Also mirroring, mirroring a copy and dragging a copy are supported in projections. The master planks will move on the projection plane. Projection planks do not have 3D – it is not possible to select these planks and show those in 3D. Instead *ArchiFrame selection* tool must be used to select related master planks and to see those in 3D.

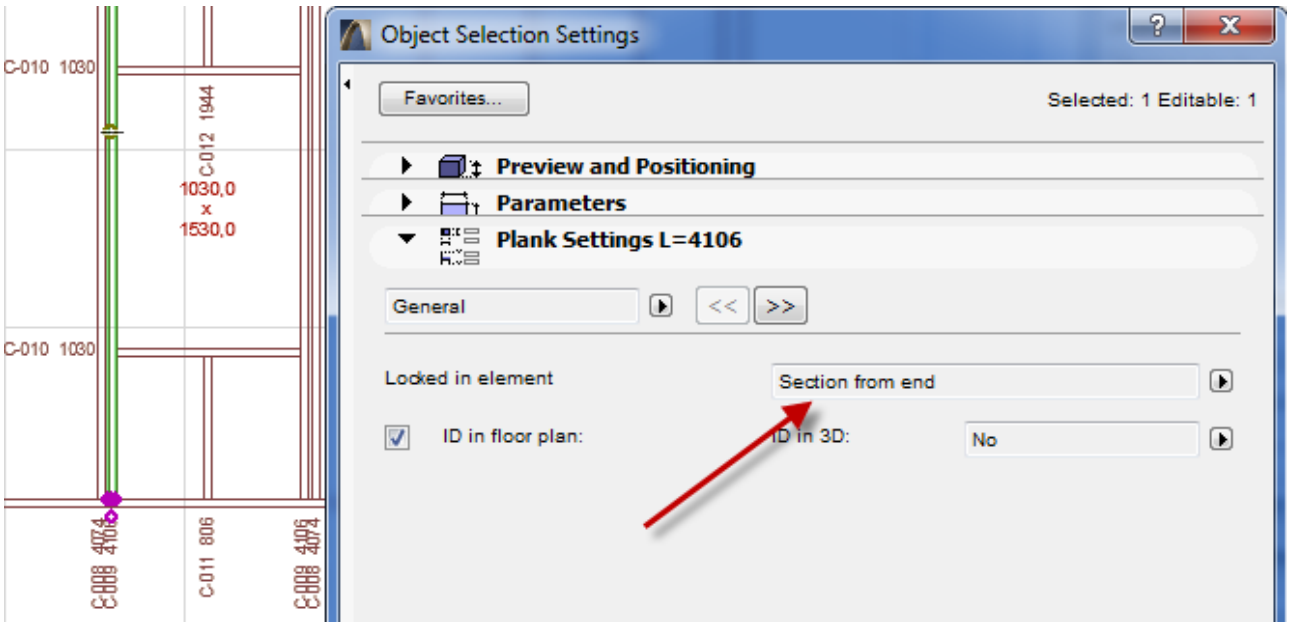
9.2.1 Locking planks in element elevations

Here is an example of an element created with default rules:

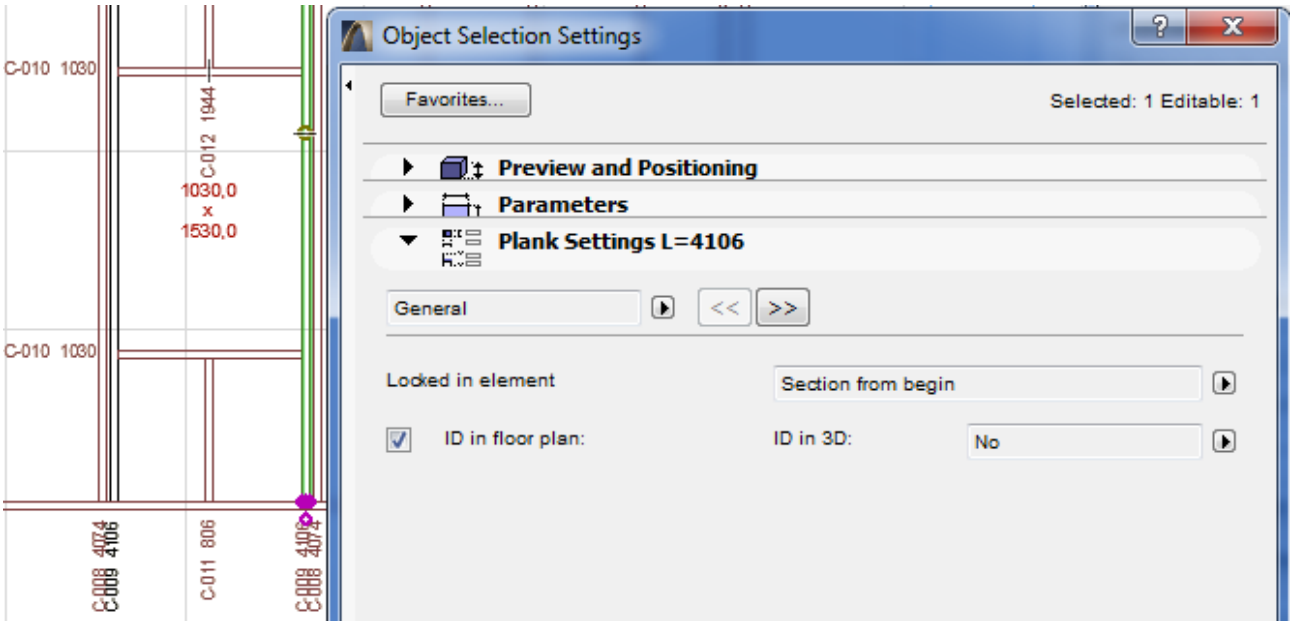


To force starting division from both sides of the opening we do these steps:

1. Set locking to *Section from end* (origin for the section is at the end of element, will continue to left side).

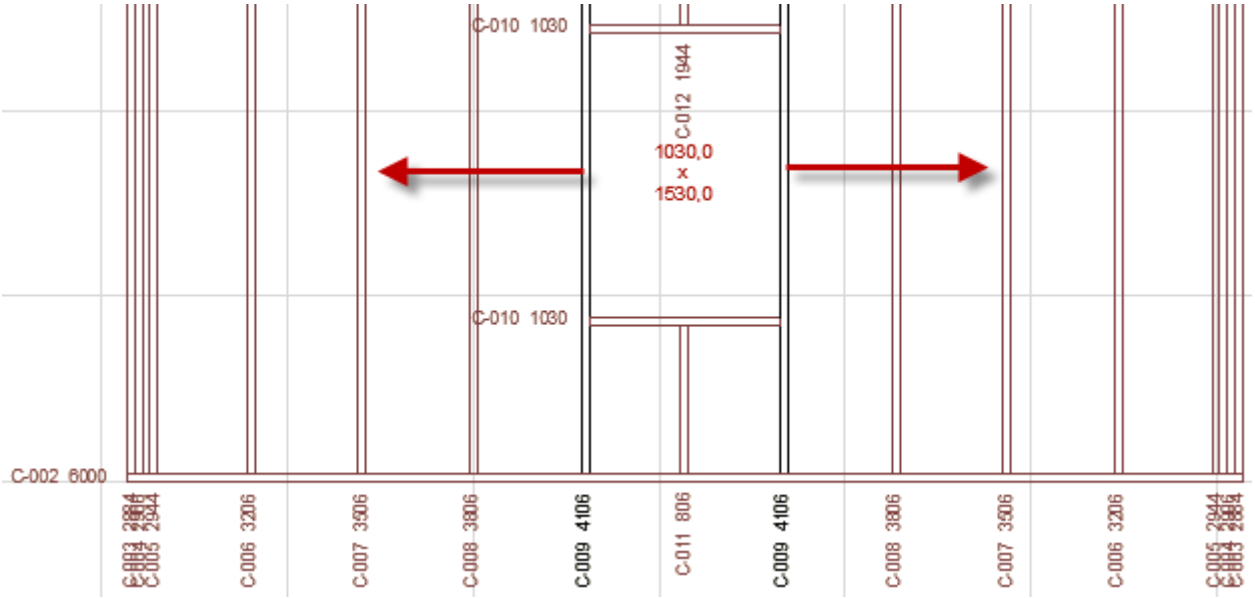


2. Do the same for other side setting *Section from begin*:



3. Recreate the element planks with element tool button *Create planks* having one plank selected (it defines the target element for the operation).

Result is like this (the arrows show the direction of guided stud placement):



It is possible to define multiple sections by having one or more pairs of locked studs.

9.3 Boards

Boards are like planks, but they have polygonal outline. In other words, the boards can be edited like planks and it is possible to apply machinings to the boards.

10 Add & Edit element tool

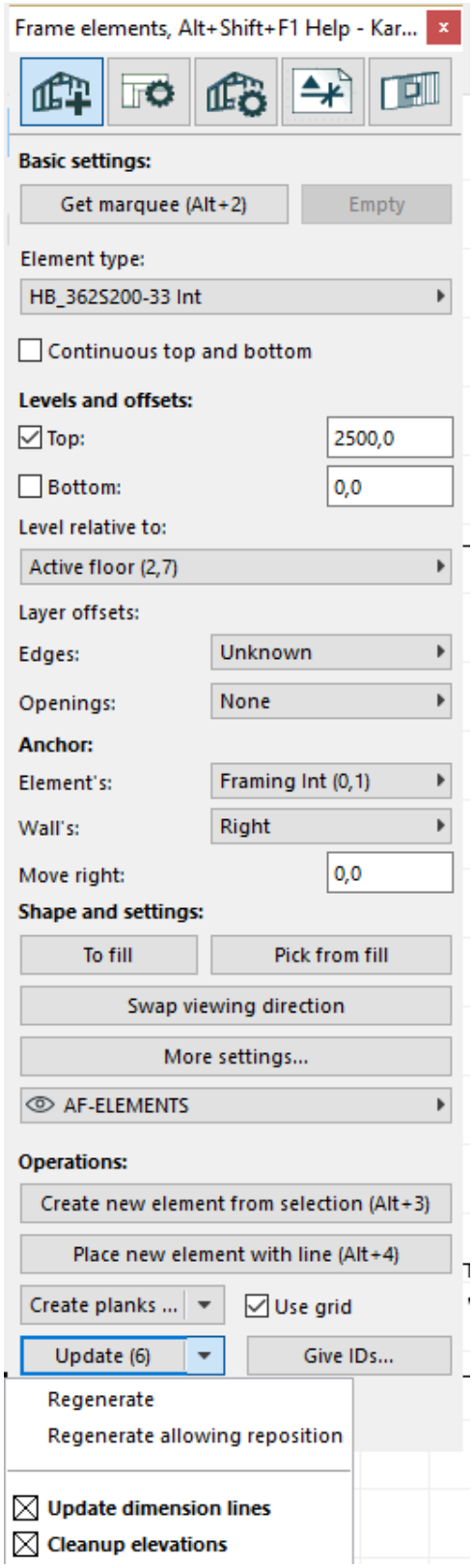
Video clips:

ArchiFrame elements: an overview, https://vimeo.com/178327076.
https://player.vimeo.com/video/178327076

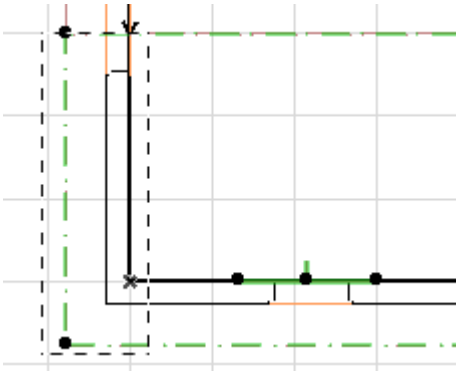
Placing rectangular wall elements, https://vimeo.com/180035580.
https://player.vimeo.com/video/180035580

Placing complex wall elements, https://vimeo.com/181255910.
https://player.vimeo.com/video/181255910

Adding a wall element with steel framing, https://vimeo.com/173754604.
https://player.vimeo.com/video/173754604



- *Get marquee* create an element with marquee boundaries instead of the original element, using ArchiCAD marquee. It can be also extend original ArchiCAD roof or slab.

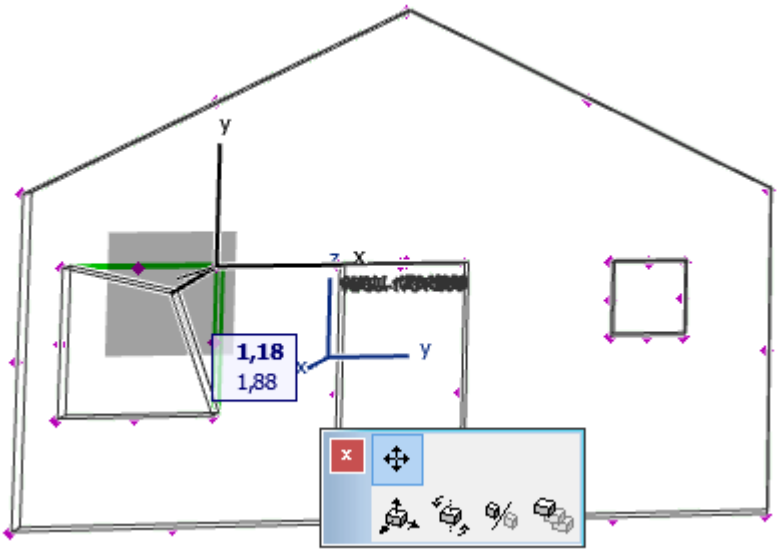


- Marquee can be picked from the AC marquee or any polygonal element (hatch, slab etc).
- The element types are defined in *data*-folder's file ArchiFrameElements.xml. It will set type for new elements or currently selected elements. Command *Apply element type's new definitions to selected target(s)* will apply changed settings from custom element type's composite settings to selected structure. This command is unavailable if there are no changes to apply. *Create planks* will apply rules defined in the layer level.
- *Continuous top and bottom* results as continuous top & bottom sills even if an opening goes all the way up or down, for example, in this case:

- *Force new/set old level* sets the element's bottom and top levels. For existing elements, first select the element object and then enter the value. Rules for searching the walls when placing the element with line:
    - If there are any selected walls when drawing the line, only those walls will define the element's geometry.
    - Without selection, all walls intersecting given bottom and top levels and drawn line will be taken. If no bottom and top level is forced, levels used for search are current storey's bottom and top levels. The wall placement storey does not have any effect.
- *Layer offsets edges/openings,* please see Custom layer edge/opening offsets. Please note that an area to work with openings can be limited by earlier picked marquee (picked with button *Pick marquee*). ArchiFrame will apply opening offsets to any opening that intersects with the picked marquee. The marquee can be set either on floor plan or on any projection.
- *Anchor* specifies the anchor point used for original AC-element and new element object. Selected points will be matched. For example for walls, it is useful to match the element's inner surface (plasterboard in for example) and wall's interior side.
- *Move right/up* is offset value for matching the anchors.
- *To fill / Pick from fill* allows you to edit the shape of ArchiFrame elements and boards with the help of fills.
    - *To fill* creates a 2D fill from the selected element. You can then edit the fill's shape with standard ArchiCAD tools.
    - With *Pick from fill,* you can inject the shape of the edited fill to the ArchiFrame element. If the fill origin is moved after placing it, the movement is reflected to original ArchiFrame object. To intentionally avoid this, make a duplicate of the fill,

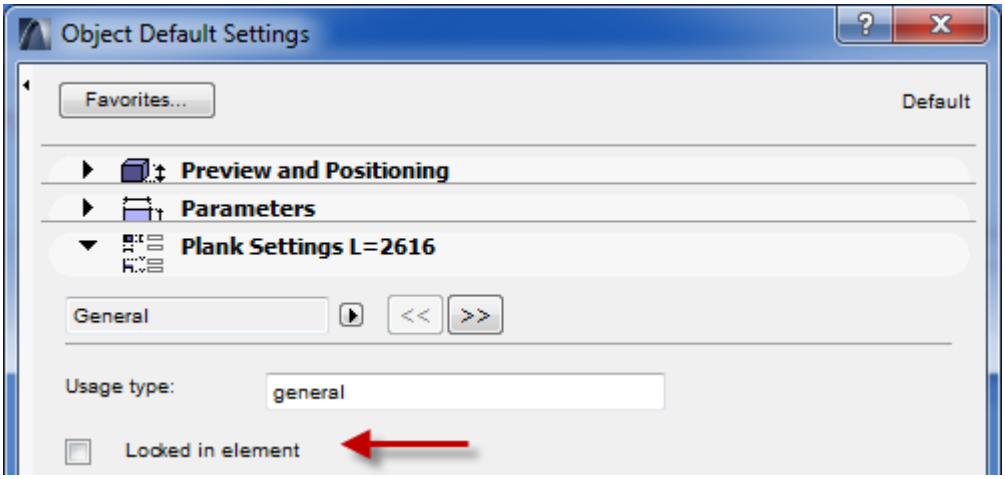
move it and pick the shape from the duplicate fill. In this case, you can have only a single fill since ArchiFrame has saved the target ArchiFrame-element when the fill was initially created with *To fill*-button.

- For multilayer elements *To fill* creates fills for each layer unless *Suspend element groups* is checked on. *To fill* can be used also to edit board shape in element drawing.
- Element and board shapes can be edited in 3D/sections with moving hotspots (converting to fill is still needed to add new points to the polygon):

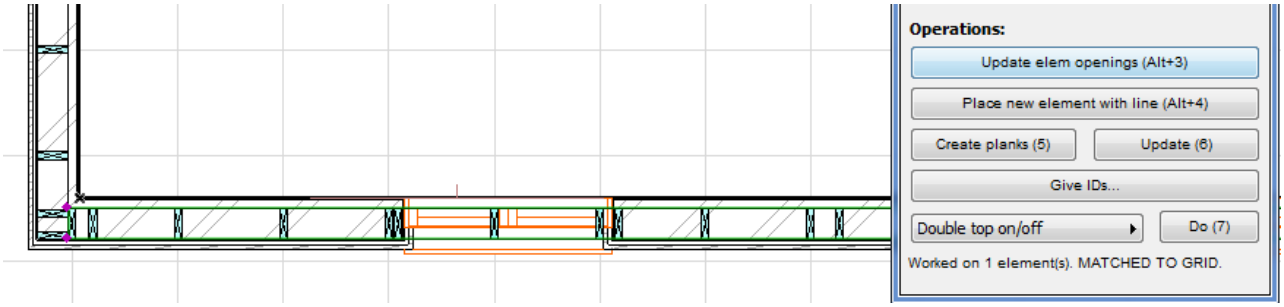


- *Swap viewing direction* rotates the element by 180 degrees and mirrors the element polygon. In short, it swaps the viewing direction.
- *More settings* opens Element settings dialog.
- *Create new element from selection* uses the AC-selection to create a new element object. The selection may contain many source elements. Every element is projected to the first element's plane and only one element object is created from the selection. This design allows constructing element object from many ArchiCAD elements and on the other hand forces user to create each element object separately. ArchiFrame will ask for direction of the rafters/joists before creating the new structure. The direction is given by a line.
- *Place new element with line* places new element based on entered line. Note that element will be viewed from the right side of the line and that is the standard side for exterior side. The tool always uses the tilt value from Force tilt. If there is a wall where the line is entered, its geometry is taken into element projection. If there is no wall, the level values from Bottom and Top fields are used:

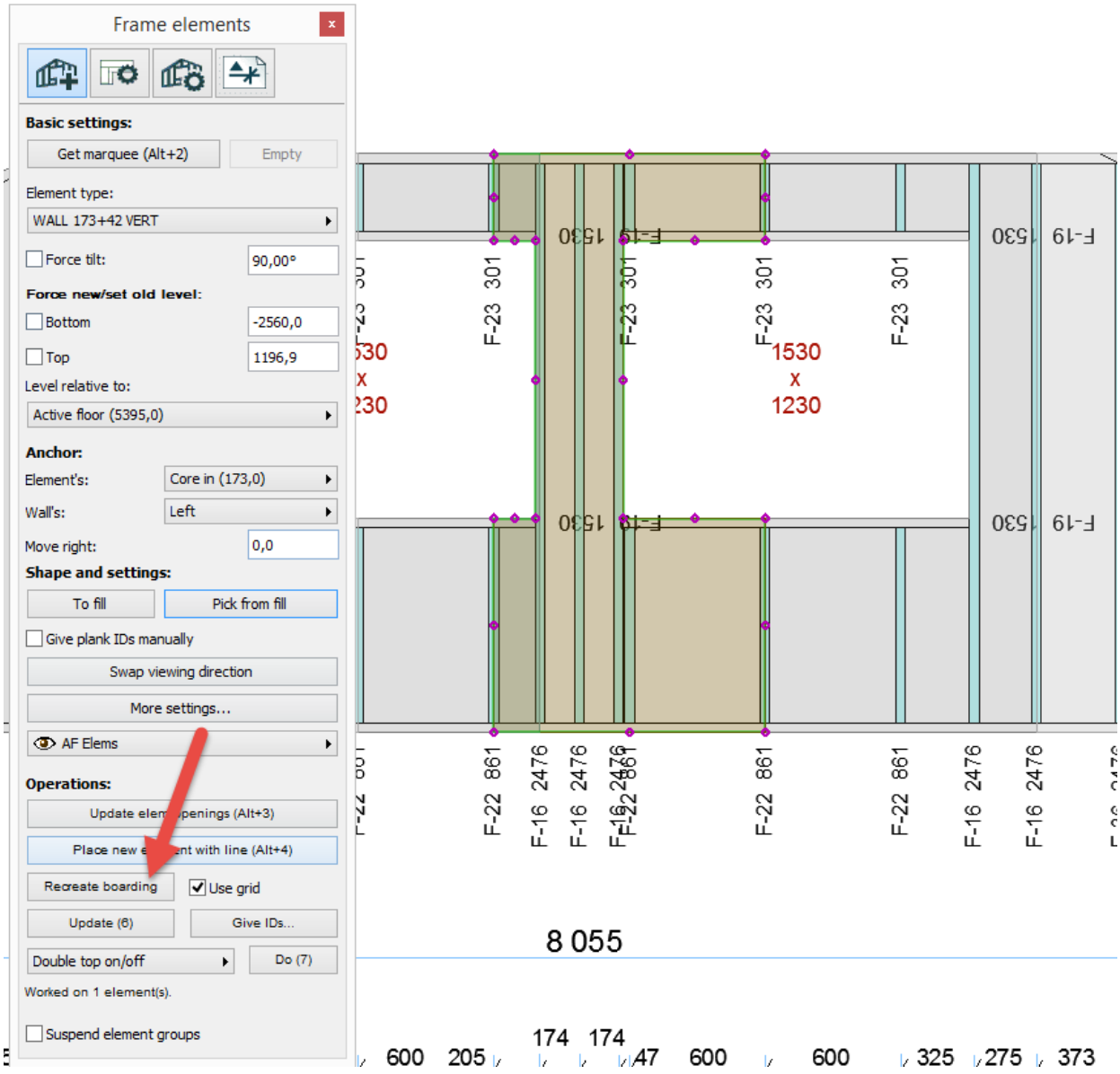
- *Create planks* deletes all existing planks except locked ones and recreates the framing taking the locked planks into account. The plank can be locked from element plank tools or from ArchiFramePlank object settings:



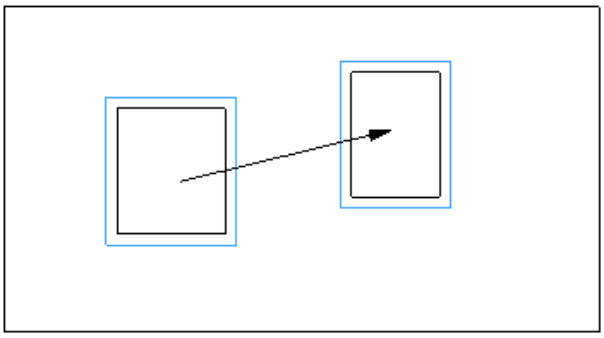
If there is a construction grid bigger or equal to stud spacing and *Use grid* is checked when creating the planks, ArchiFrame will place the studs to the grid if possible:



Just a single boarding layer is recreated by selecting a board and then clicking *Recreate boarding*. In the picture below the board shape has been edited, the board locked from object settings dialog and then the boarding is recreated to be compatible with the locked board:



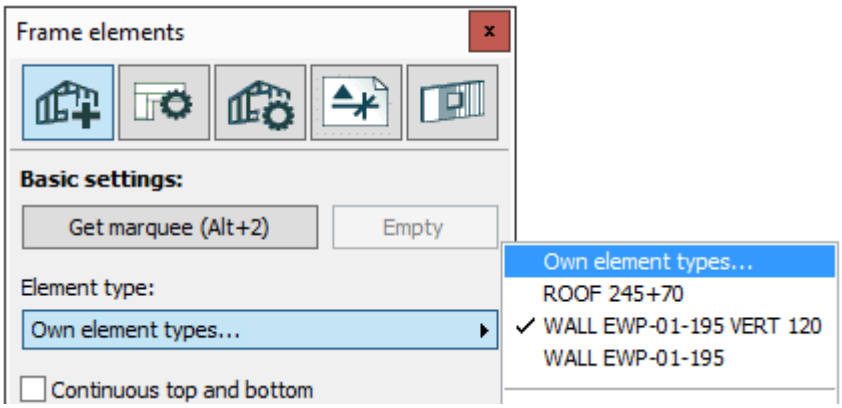
- *Crate planks* is changed to *Update elem openings* if there is an element or any piece related to an element selected. It will do nothing if there are no changes in the openings including any changes in force oversizes settings. In case of changes, ArchiFrame updates only the area of changed openings extended by two times the stud material width. The update area is marked with a blue line for the moved opening. Any plank having reference line intersecting with update area is recreated:



- *Update* updates elevation, its dimension lines and cut lists to ensure the projections match original planks. The 3D-model remains untouched except new IDs can be assigned.
- *Regenerate* forces ArchiFrame to re-create all projection items with current settings from xml-file. Planks and boards are kept in current positions. The 3D-model remains untouched except new IDs can be assigned.
- *Regenerate allowing reposition*, ArchiFrame resets the element elevation layout. The 3D-model remains untouched except new IDs can be assigned.
- *Check framing for opening dimensions*, if checked on, ArchiFrame will subtract framing polygons from ArchiFrameElement's polygons when creating opening dimension lines. Unchecked it will work like version 2017 and earlier and take openings directly from related ArchiFrameElement-object.
- *Update dimension lines* allows keeping the manual changes to the dimension lines when updating the elevations. If not checked, the dimension lines are kept as they are when updating. Regenerating or createing planks again will always update dimension lines.
- *Cleanup elevations* cleans up:
    - Overlapping plank IDs.
    - Board IDs overlapping with planks and other board IDs.
    - Cross dimension overlapping with any ID or planks.

- *Update opening text markings* defines whether the text related to openings are updated during the update. Setting it off allows manual editing of those texts.
- *Give IDs* opens renumber dialog. More information here.
- The last button is an operation that affects the whole element. For example, it can add a top beam either to inner/outer side of the element or create grooves to the bottom/top wood. To execute the operation, an element or any plank related to the element is selected.
- *Suspend element groups* affects multi-layer elements. Unchecked changes will affect every layer and when checked, changes apply only to selected element objects. The setting affects these operations:
    - Changing *Bottom* and *Top* levels will affect only the selected layers instead of every element layer. To limit, for example, inner boarding top level at gable walls it is useful to check *Suspend element groups* and edit *Top*-value.
    - *To fill* creates fills from single element only when checked and from every layer if unchecked.

10.1 Own element types/Custom element tools



The first item in the element type list opens *Custom Elements*-dialog. Below it ArchiFrame shows the 10 last used custom element types. To select one that is not in this most recently used list the *Custom Elements*-dialog must be opened – it will show all added types.

10.1.1 Custom Elements dialog

Define wood structures 1: https://vimeo.com/179027834
https://player.vimeo.com/video/179027834

Define wood structures 2: https://vimeo.com/173878793
https://player.vimeo.com/video/173878793

Define metal structures: https://vimeo.com/173751192
https://player.vimeo.com/video/173751192

This dialog is used to define a single or a multilayer element types. It resembles ArchiCAD's composite structures dialog. Please note that also single layer types are defined here – those have just a single layer.

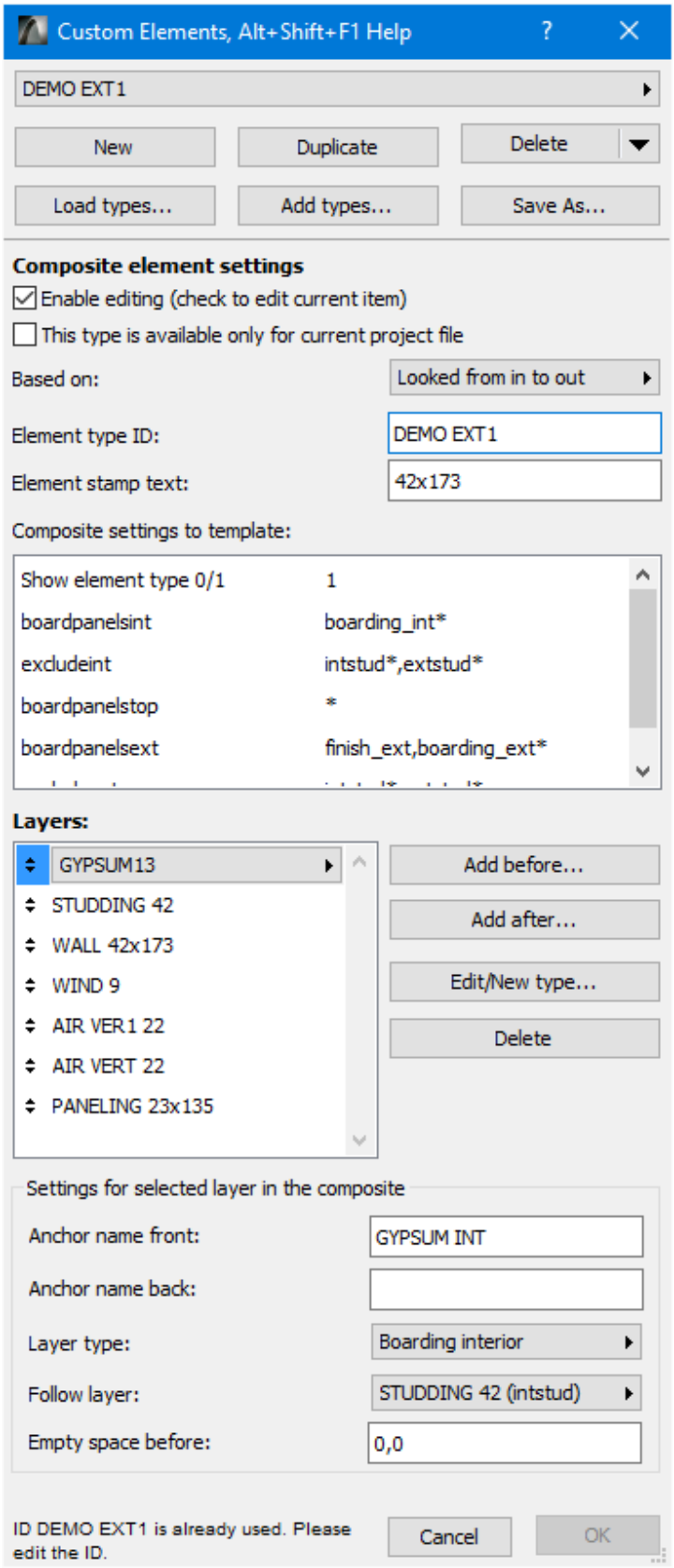
Editing the type here does not change all ArchiFrame element objects as changing ArchiCAD composite structure does. This limitation is due to performance reasons. The changes are applied only for selected ArchiFrame element objects.

The main watching direction of the layers go from top to bottom. For example, if there is an element containing framing and single board layer it should be defined the board layer first and then framing. Then having just single projection from front will show the boards in front of the framing as it is assembled in the factory or at building site.

The global custom element types are saved into user specific folder into file ArchiFrameTemplates.xml (in Windows 10 the folder is C:\Users\[user_name]\AppData\Local). Current project file specific types are saved only into the pln-file.

Before editing custom element types, it is recommended to save a backup of all definitions using *Save As*-button. The backup file can be taken into use again by clicking *Load types*-button. Canceling the dialog reverts all changes.

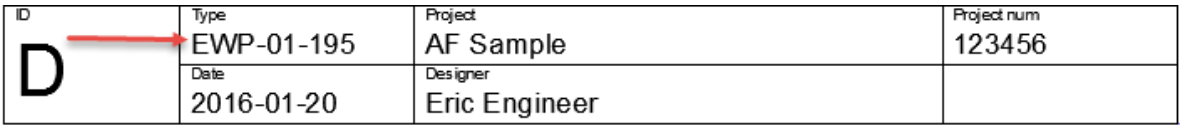
The template source file from the *data*-folder's file ArchiFrameElements.xml contains *replace*-tags in syntax [setting_id; prompt=Shown to the user; default=default value; type=text/length/real/matid/angle/layer/panelid]. Type angle is a special case: It is saved in the template and perhaps applied to the objects as radians but shown to user as degrees. All unknown types are treated as text.



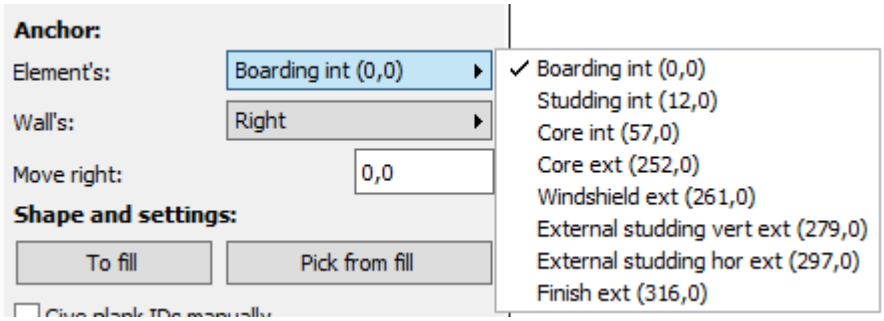
Parts of the dialog from top to bottom:

- *Element type* dropdown list, this list contains all the defined types. When closing the dialog the type here is set to any selected element object.
- *New* creates new composite structure without any layers.
- *Duplicate* duplicates current composite structure for further editing.

- *Delete* deletes current composite structure. Be very careful not to delete any structure that is in use. The custom element types are shared among all projects as default. So delete a type only if you are absolutely sure that it is not needed any more.
- *Load types* deletes all the custom composite types and single layer types and loads everything from an external file. This is very useful for example if there is a single person in the company taking care of the structures and shares the types to other users.
- *Add types* is as *Load types* except it does not delete current types. It merges the types from the file using any type having the same id from the given file.
- *Save as* saves all custom types to an xml-file to be used for *Load types* or *Add types*. The file can be edited using text editor for example to limit types to import to another user.
- *Enable editing* must be checked on before making any changes. Its purpose is only to avoid editing existing types accidentaly.
- Check box *This type is available only for current project file* defines that the type is available only in current ArchiCAD project file (pln-file).
- *Based on* defines the template to use for the composite structure. The templates are defined in the data folder's file ArchiFrameElements.xml and they define mostly the projections in composite structures. Layer templates define how the framing is done. Your supplier can help you with the element templates if new ones are needed or some custom framing is needed.
- *Element type ID* is the unique ID for the composite custom element type. The ID should never be changed if the element type is in use. It is advisable to design the ID types carefully before actually adding elements. For example, it could be WALL layer1+layer2+layerN (WALL GYP13+45+195+WIND9) or shorter form.
- *Element stamp text* is shown in element elevations here:

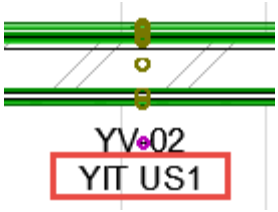
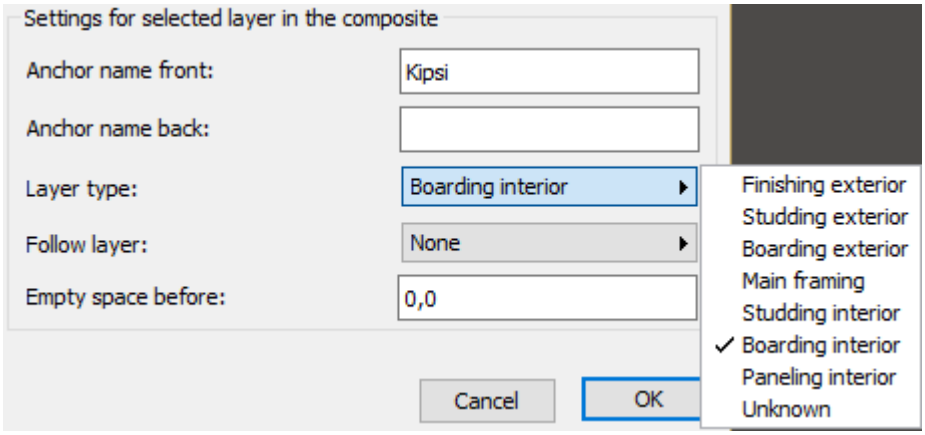


- *Composite settings to template* lists all editable parts of the template file. For composites, it contains mainly settings of which layers are visible in the different projections. By default all framing layers are visible and board/panel layers are not.
- *Layer list* shows the layers making up the composite structures. The order of the layers can be edited by dragging the item with mouse from the list's left hand side dragging icons. The best way to make a composite structure is to first add all the layers and after that set the dialog's bottom part settings per layer.
- *Add before* adds new layer before the selected one. *Add after* adds after the selected one.
- *Edit/New type* opens the single layer editing dialog. If there is no correct layer type defined already it can be added with this button.
- *Delete* deletes the selected layer from the composite structure.
- *Anchor name front* and *back* tells the anchor points to show here. The watching direction goes from the top of the layer list towards the bottom and *Based on* template defines if the watching direction is from inside or outside:



- *Layer type* tells ArchiFrame the usage of the layer. There must be exactly one *Main framing* layer. For projections, usually all layers before *Main framing* are defined as interiror types and after main framing as exterior. This rule is usually more important than interior/exterior classification.
- *Follow layer* is used for example for inner side additional studding. It should usually follow the main framing layer. Also for the boards this must be set to get the board edges on a stud.
- *Empty space before* can be used if some layers are omitted from the ArchiFrame composite structure (for example asphaltic felt in roof structures could be 3 mm thick).

10.1.2 Settings to element template for composites

Name	Description
Show element type	With value 1 the element type (YIT US1) is shown in floor plan and 3D in addition with the element ID (YV-02): 
Boards and claddings in projection ext/int	Layer type names to be shown in the related projections. Names refer to the layer type defined here:  The internal names are: • Finishing exterior, finish_ext. • Studding exterior, extstud. • Boarding exterior, boarding_ext. • Main framing, core.

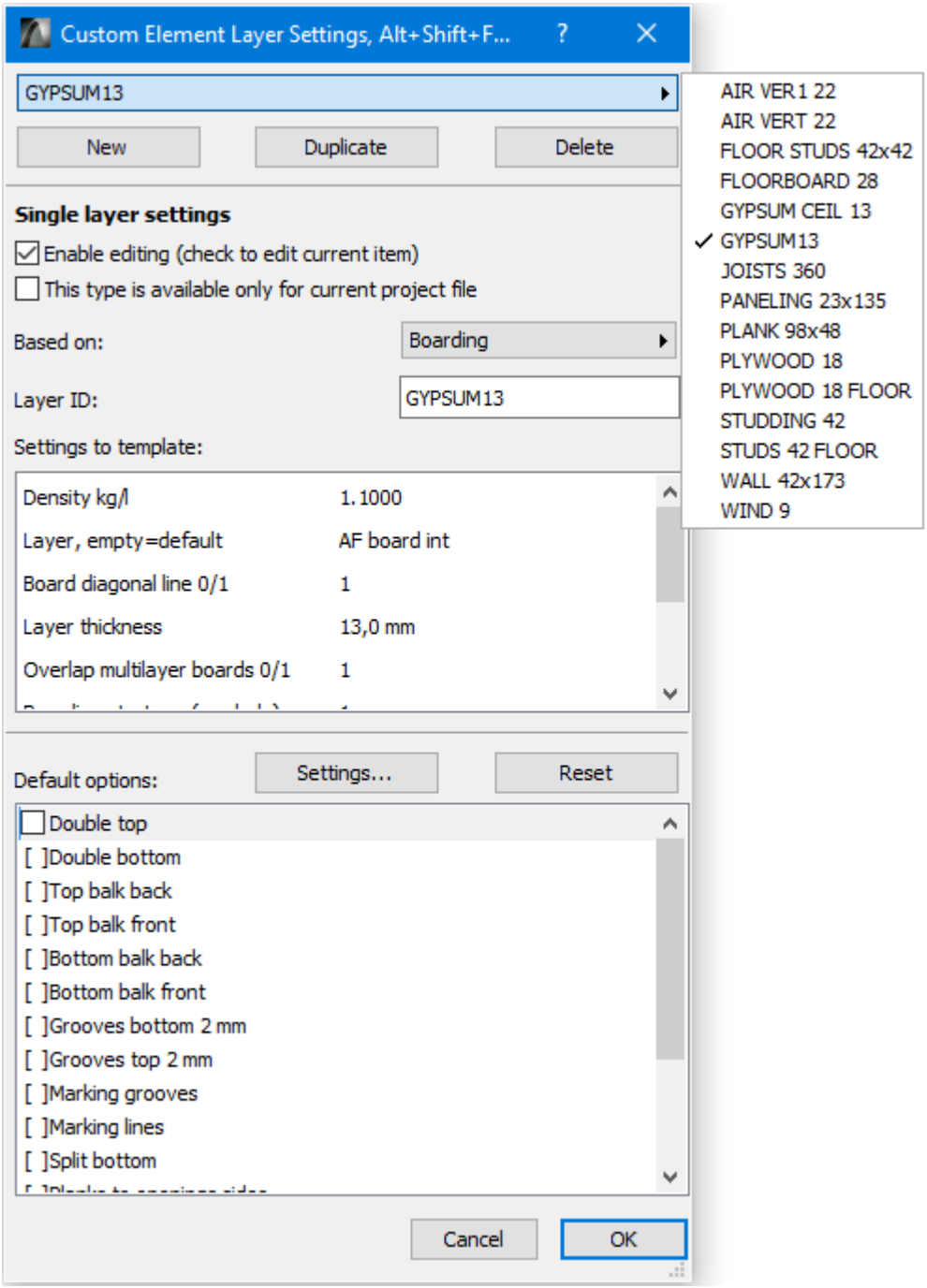
	• Studding interior, intstud. • Boarding interior, boarding_int. • Paneling interior, finish_int. If there are many layers of the same type, there will be a sequence number added to the end of the name. For historical reasons, the numbering starts from 2. For example: intstud2, intstud3. Wildcard * can be used here (intstud*).
Don't show in projection int/ext	As previous but defines layers to be excluded from the projection.
Show center of gravity	Possible values are: • 0, do not calculate. • 1, show the real position of the center of gravity. • 2, show above the elevation. • 3, show above the elevation and show separate value calculated. Without doors&windows if different.
Weight text	Text to set into the weight marker, (weight) is replaced with the calculated weight: 

Cnc-_elemtype	Wup cnc-output: Sets the element type line in wup-file:
	`VERSION 3.3;`
	`ANR 123456;`
	`ELB EXTERIOR;`
	Default is EXTERIOR if looking from outside and INTERIOR if looking from inside.

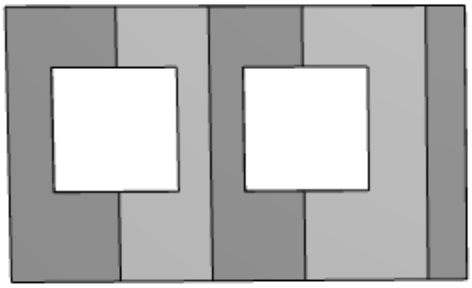
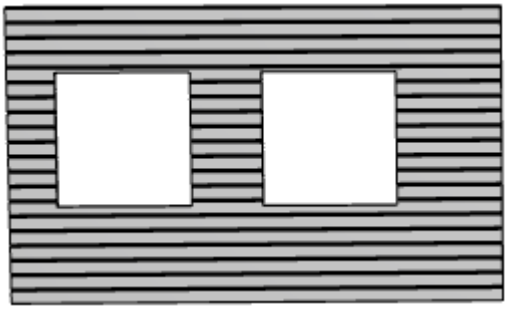
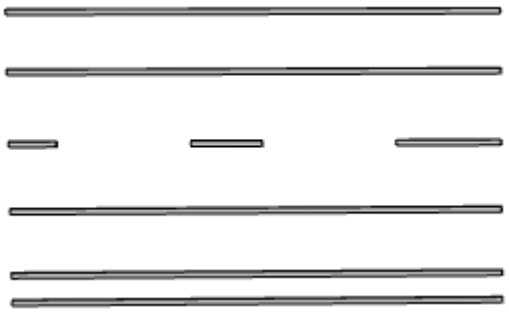
10.1.3 Custom Element Layer Settings dialog

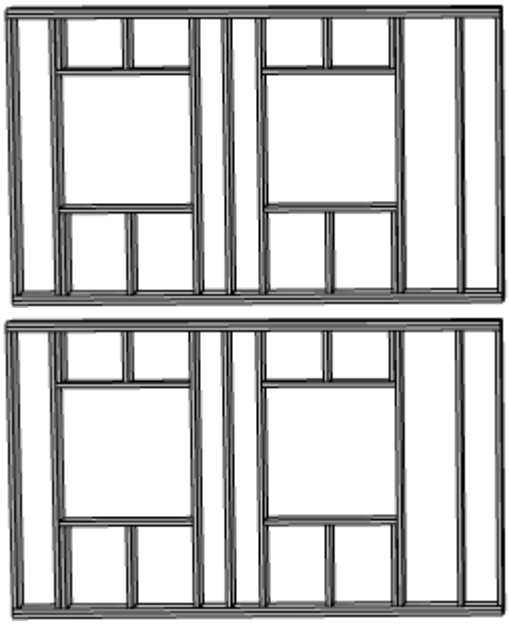
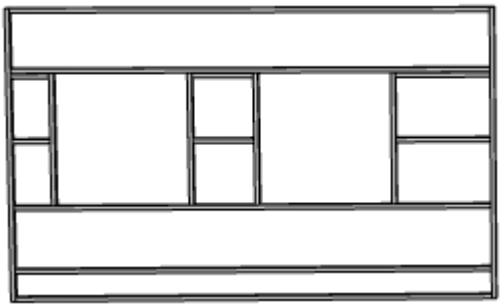
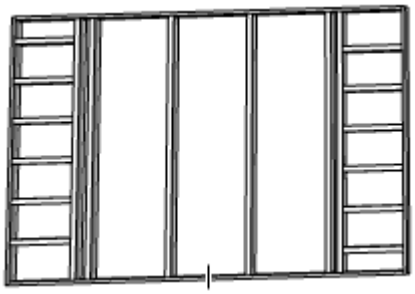
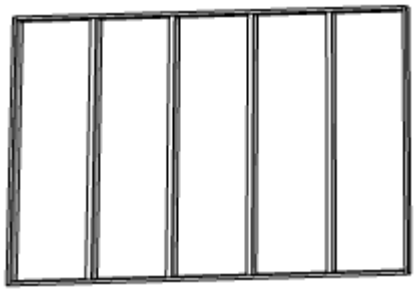
This dialog is used to define framing parameters or board types for a single layer. It resembles ArchiCAD's building materials dialog.

The panel profiles must be defined into *data*-folder file ArchiFrameElements.xml and are referred from a custom layer by its ID.



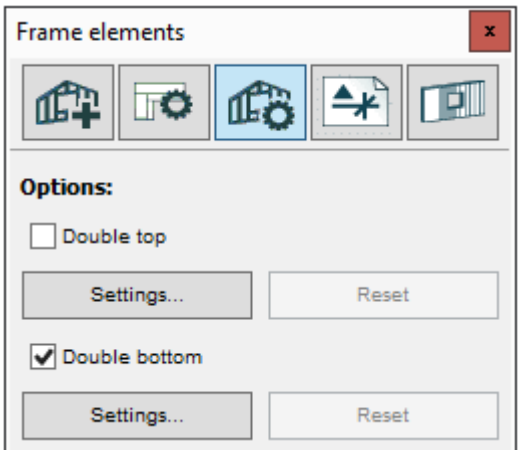
- At the top of the dialog there is the custom layer list like in the composite dialog.
- *New* creates new empty layer definition.
- *Duplicate* duplicates existing one for further editing.
- *Delete* deletes the current layer type.
- *Enable editing* must be checked on before making any changes. Its purpose is only to avoid editing existing types accidentaly.
- Check box *This type is available only for current project file* defines that the type is available only in current ArchiCAD project file (pln-file).
- *Based on* defines the template to use for the layer structure. The templates are defined in the data folder's file ArchiFrameElements.xml and for the layers they define the framing rules. Your supplier can help you with the element templates if new ones are needed or some custom framing is needed. The default types are:

Boarding	 See boarding strategies.
Paneling, paneling itself must be defined in ArchiFrameElements.xml manually.	
Air space strips vertical	
Air space strips horizontal	

Framing	 Please see Main framing rule/attribute framingrule.
Framing horizontal	
Floor/roof with sides	
Floor/roof no sides	

- *Layer ID* is the unique ID for the composite custom layer type. The ID should never be changed if the layer type is in use. It is advisable to design the ID types carefully before actually adding elements. For example, it could be WALL 45x195 telling the intended use and the main material for the framing.
- *Weight* opens Weight calculation settings.

- *Settings to template* lists all the editable parts of the template file. The settings include the layer thickness, material id for the framing, stud spacing and more. If the value begins with character @, it will point to another key in the template. For example, if there is common default material with key matid it can be pointed with value @matid. See Settings to element template.
- *Default options* is a list of all available options from the Element tools/Element options:

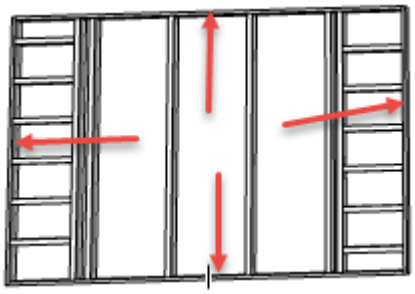


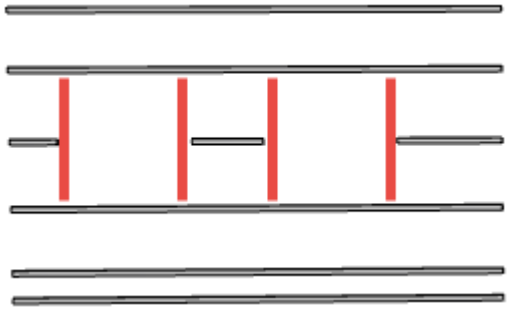
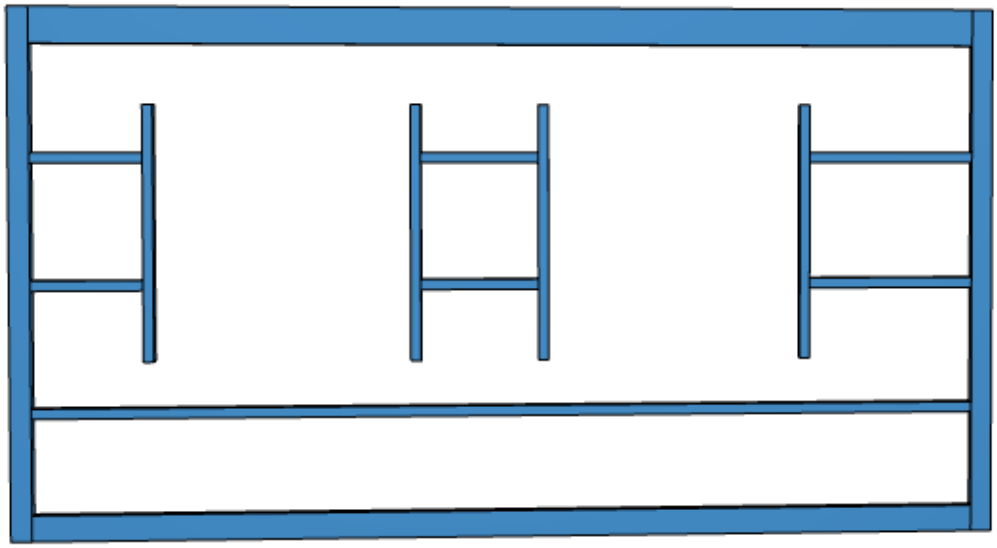
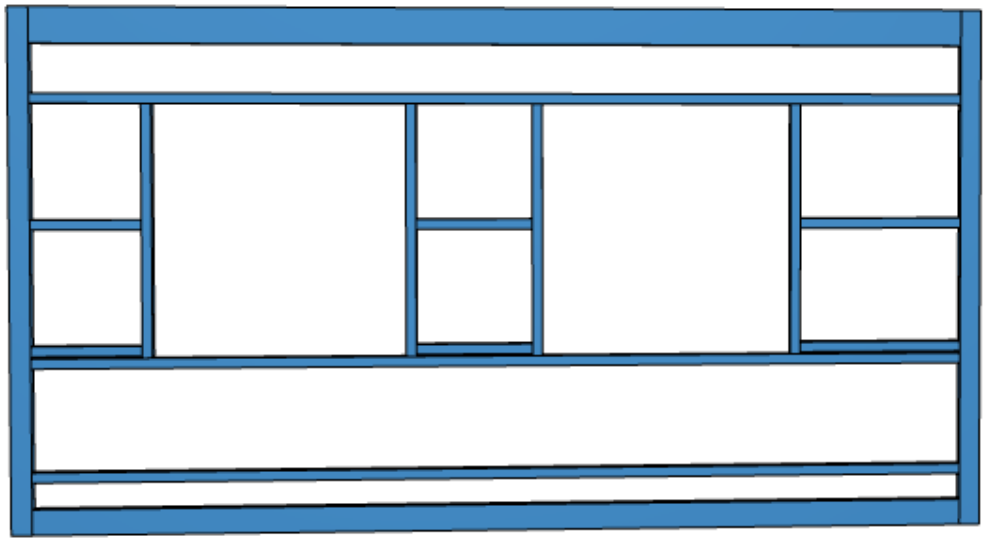
For example it may be useful to have double bottom as default on for the main framing but not for additional studding.

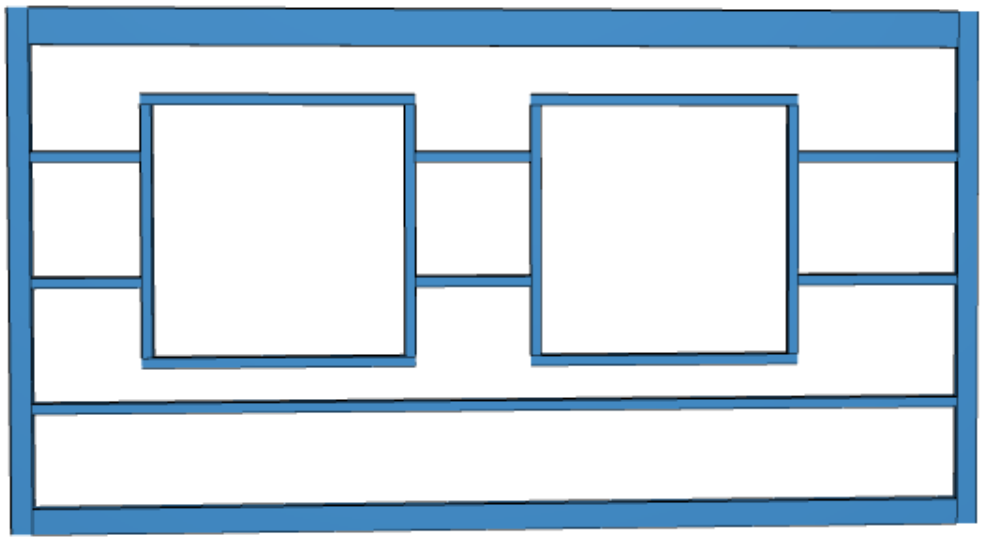
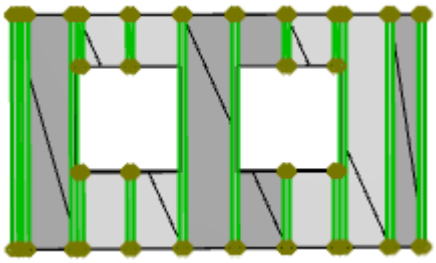
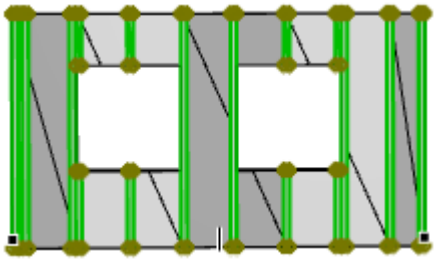
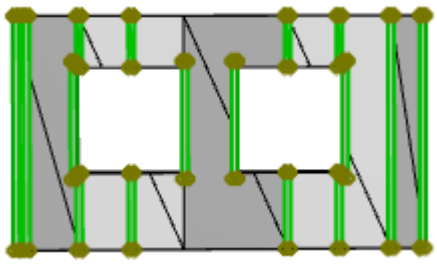
- *Settings* opens selected option's settings if available.
- *Reset* clears all given settings and uses option with default settings.

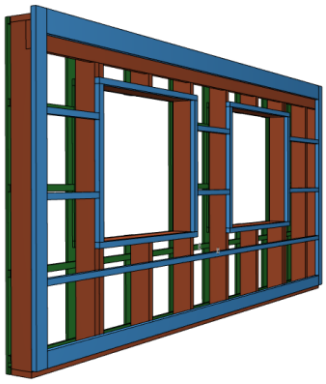
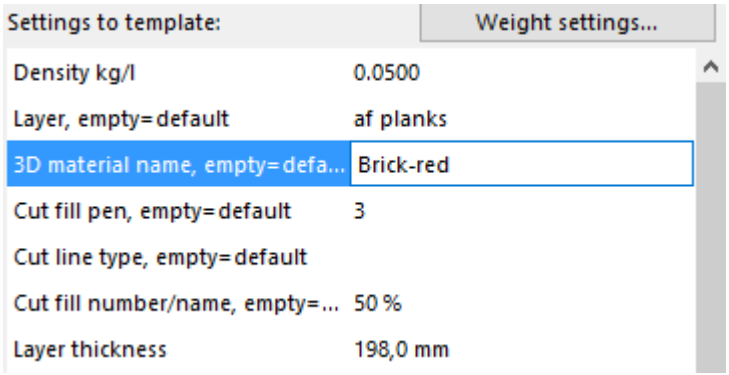
10.1.4 Settings to element template for layers

There may be any settings in the element template. Following table lists built-in tags:

Name	Description
Board type ID	Board type ID for listings.
Board width and height	Affects creating boards.
Create border	0/1, Used in floor structures: 
Density	Layer density for element listings.
Layer	The ArchiCAD layer where the target pieces are placed. You can choose a pre-existing layer from the drop-down menu, or else type in a new layer name. Typing the layer name creates a new ArchiCAD layer.
Layer thickness	Layer thickness. Usually equals to material depth in element's watching direction but in special cases may be different.
Material ID	Material to use. The material must exist in the *data*-folder's ArchiFrameBlocks.xml file.

Material ID top/bottom / left/right	Used in framing to define bottom or top plate material if different from other structures.
Planks to opening sides	Used in horizontal air stripes. With value 1 the red pieces will be created:  For framing horizontal, value 0:  1: 

	2:  For vertical air stripes the result is for value 0 (follows the main framing and that causes pieces to left and right hand sides of the openings):  Value 1:  Value 2: 
Spacing	Spacing distance in meters.
Spacing tolerance	How much for example a stud position may differ from the spacing rule before there will be a double stud. For example, value 0.01 will result in a double stud if existing stud's position is further than 10 mm from the spacing rule.
Rotangle	With value 0 the plank is placed the base line (middle bottom) on the element's front surface. Value 90 is useful for airstrips to get, for example, 20 x 50 mm piece so that it has 20 mm in depth and 50 mm width in element's watching direction.

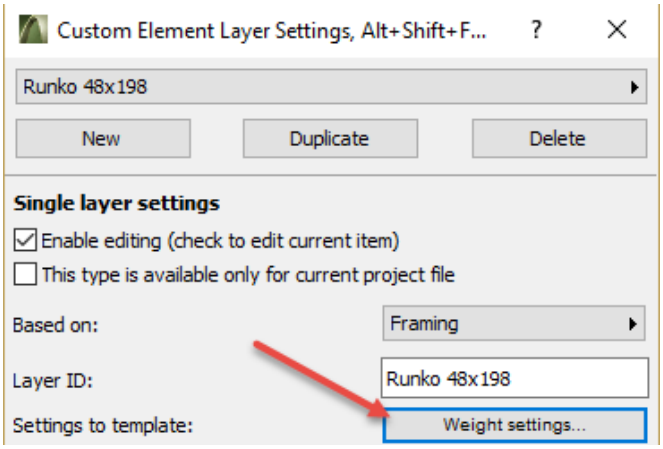
zoff	Text field that must be given in meters. Common value with rotangle 90 is mat_thickness*0.5 which defines that, for example, with 20 x 50 mm piece the placement position for plank's anchor point is 10 mm begind the element's front surface.
3D material name	Allows setting different 3D material for each layer. The material must be written exactly as it is defined (faster) so be sure to copy&paste material names to text editor before opening own elements dialog. Wildcards * (any string) and ? (any character) may be used but then ArchiFrame will scan attributes until match is found. For example, having different colours for different layers – single layer's settings:  
Cut fill pen	Sets pen used in sections for cut planks, must be given as number or empty=use default in this order: • What was set into object tool default settings defined when planks were created. • Value set in ArchiFrameBlocks.xml. • (Value given here).
Cut line type	Line type number for sections.
Cut fill number/name	Sets pen used in sections for cut planks. Can be given as number or as text like 50 %.
cnc_outputline	Reserved for future: To be used at btl-files telling the output slot.
cnc_nailgun	Wup-cnc-export: Default nail gun index number to use for nailings.
cnc_matindex	Wup-cnc-export: Material index number for the layer.
cnc_matname	Wup-cnc-export: Material name for the layer.
cnc_isframing	Wup-cnc-export: Is the layer framing layer, value 1 sets as framing and the planks in the layer are saved as QS, OG, UG, etc. Otherwise all pieces are saved as PLX. As default core and intstud* layers are saved with framing codes.

10.2 Weight calculation

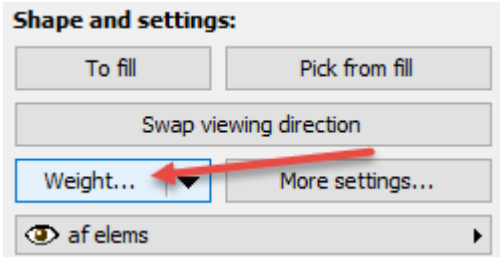
ArchiFrame contains weight definitions in own element settings per composite structure layer and in plank definition file ArchiFrameBlocks.xml in the *data*-folder. The order of getting a plank's or board's weight is (the latest definition will be used):

- Built in default which is:
    - 450 kg per cubic meter for planks (450 gr per liter), that is density for dry pinewood.
    - 1100 kg per cubic meter for boards (1,1 kg per liter) which is density of typical gypsum board.
    - 30 kg per cubic meter for insulation (30 gr per liter).

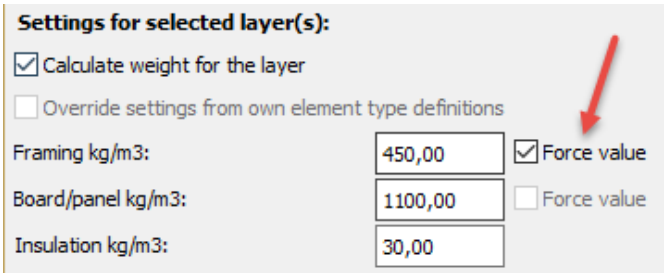
- Value set custom element layer settings:



- Overridden value from placed element settings:



- Value set for the plank type in ArchiFrameBlocks.xml unless Force value is checked in the weight settings dialog:

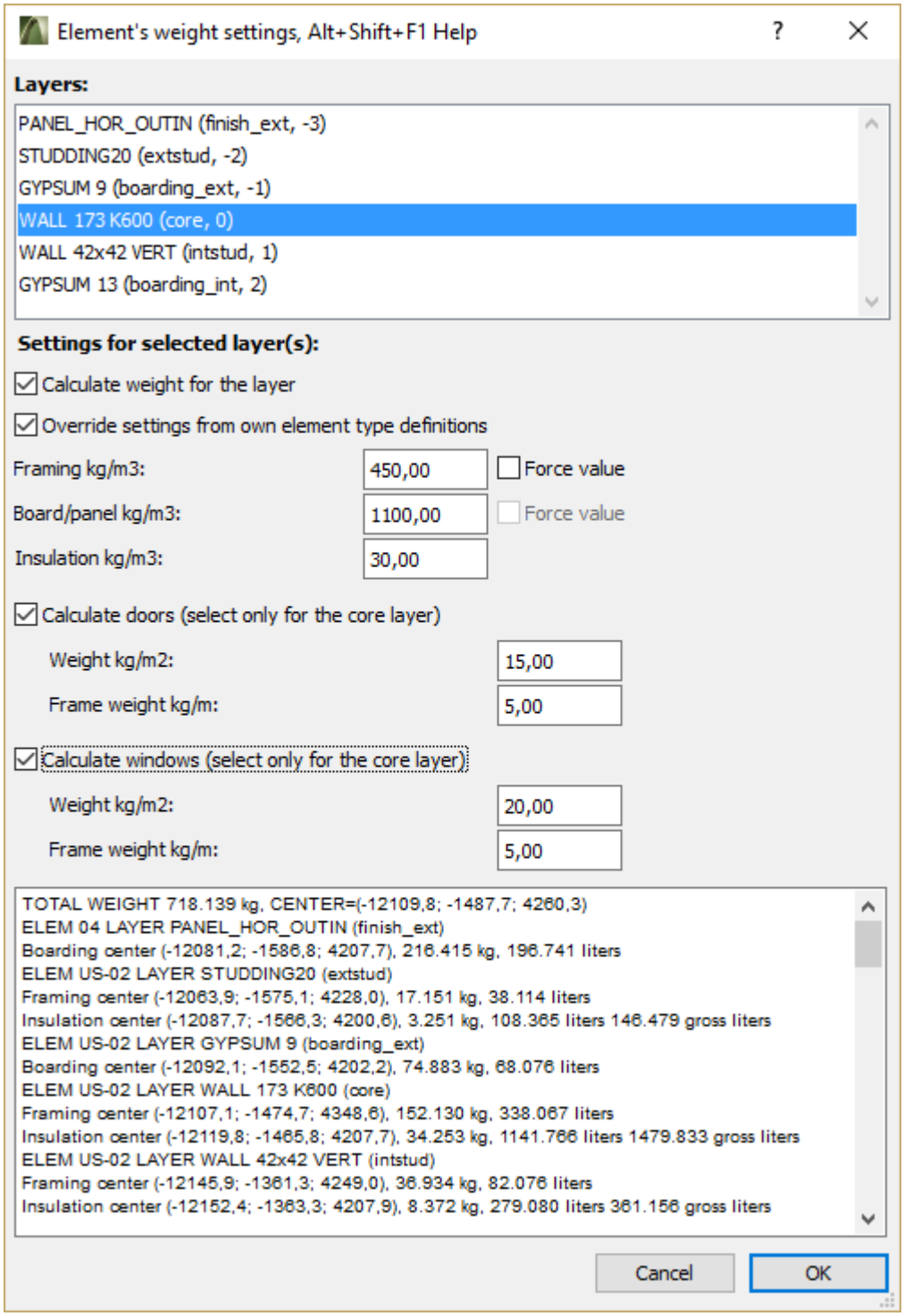


- Any ArchiCAD door/window/lamp/object having parameter FM_ObjectWeight nonzero will use that value. Related parameter FM_ObjectWeightUnit can be kg or lb.

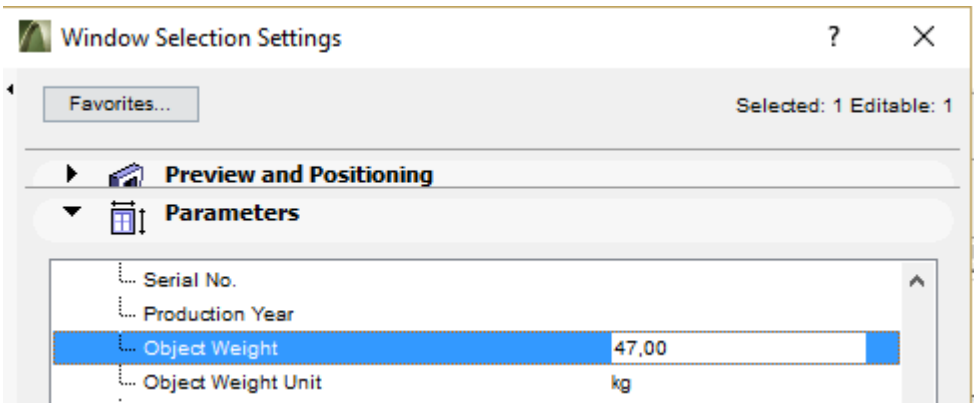
ArchiFrame can calculate weight for an element or for any ArchiCAD selection. In latter case, it will also place a 3D center of gravity marker. The gravity marker is not linked to calculated elements and if the weight is calculated again, there will be a new marker object.

Weight for doors & windows are calculated based on linked ArchiCAD-windows/doors. If ArchiFrameElement-object is not connected to any ArchiCAD wall and is vertical, all holes in the element are treated as windows. In that case a warning is issued: Element EW-01 - Copy is not linked with any ARCHICAD-element, weight calculating treats all element holes as windows.

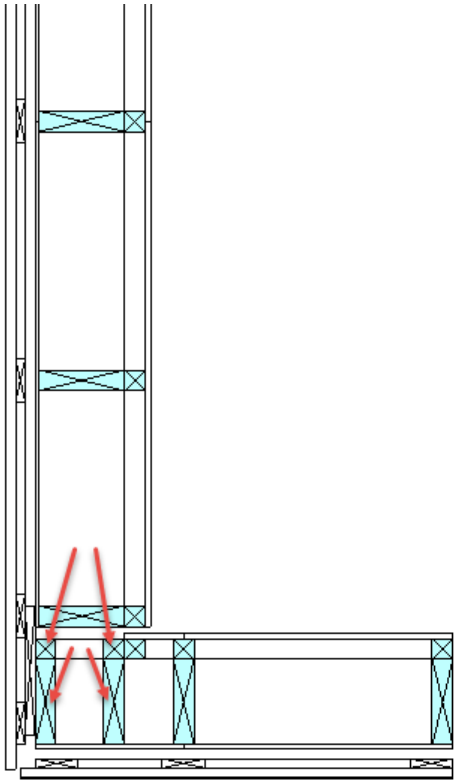
10.2.1 The weight settings dialog

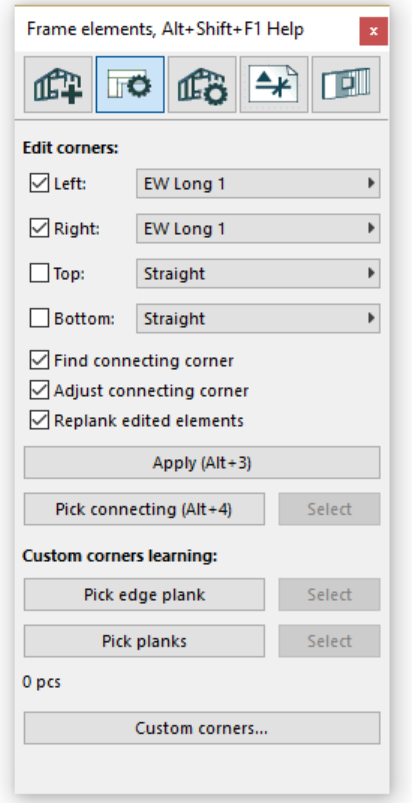


- *Layers* lists all different composite layers in selection. If opened from *Custom element layer settings*-dialog, it will contain just the current layer. It is possible to select multiple items from this list and edit the values to all selected composite layers.
- *Calculate weight for the layer* can be turned off, for example, if the related layer is built at the building site after installing the element.
- *Override settings from own element type definitions* enables setting values differing from the custom element layer settings for specific placed element. It is useful for example if a board type is changed in just a specific wall in the building.
- *Framing kg/m3* defines density of framing parts. For profile pieces like I-shaped beams the volume of the plank is calculated taking the profile into account. Please note that in ArchiFrameBlocks.xml the density for a plank type can be given as kg/liter, kg/m or lb/foot.
- *Force value* for framing will override the plank specific density given in ArchiFrameBlocks.xml.
- *Board/panel kg/m3* defines the density of the boarding layer. For non-exploded cladding this value contains also the air space caused by then planed profile.
- *Calculate doors/windows* enables calculating the weight of related ArchiCAD doors with the element layer. This option should be checked only for single layer in the composite structure. Anyway, ArchiFrame will not calculate the same part twice.
- *Weight kg/m2* defines the weight of the door leaf/window glass. ArchiFrame calculates the area for the door leaf following the door/window frame so the area is probably slightly oversized.
- *Frame weight kg/m* defines the weight for the door/window frame. For door and windows, it is possible to set ArchiCAD standard parameter FM_ObjectWeight nonzero value and it will be always used if set.



10.3 Element corner tools





The corner tool connects elements (both single and multi-layer) together and contains rules for placing special planks at corners. In the image above, the special planks, marked with arrows, are placed according to the corner rules. Corner rules also include layer offsets. The tool is split into two parts:

- Built in corners that are not editable from ArchiFrameElements.xml.
- Custom corners that user teaches to ArchiFrame.

Parts of the corner tool palette are:

- *Left/Right/Top/Bottom* check boxes control which sides of the element are set. Unchecked, the setting will not be changed for that side. If the check box text is **bold**, the setting is set in selected element.
- *Find connecting corner* defines whether the corner tool should search for related elements. If unchecked the target element's edge will not be moved. Otherwise it is moved to given distance from connecting element's surface. If operating in the floor plan only elements in the current storey are scanned – in the 3D all elements are scanned.
- *Adjust connecting corner* enables setting both parts at the same time. It requires that the selected corner type has defined the connecting corner's type.
- *Replank edited elements* automatically creates planks again if target elements are changed.
- *Apply* applies any changes that have been made.
- *Pick connecting* will pick the selected element object to be used as connecting element. It also defines connecting corner when teaching new custom corner type. To clear definition, click this button without any selection in ArchiCAD.

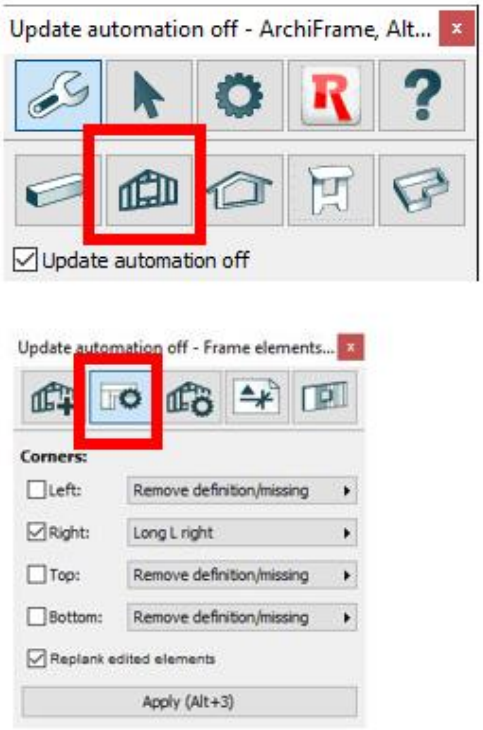
- *Pick edge plank* is only used when teaching new corner type to ArchiFrame. It must be a plank related to the main framing layer. ArchiFrame learns the closest element edge to this plank. To clear definition, click this button without any selection in ArchiCAD.
- *Pick planks* is only used when teaching new corner type to ArchiFrame. The picked planks' types and positions are included in the corner rule. To clear definition, click this button without any selection in ArchiCAD.
- *Select* next to any *pick*-button will select picked elements in ArchiCAD.
- *Custom corners* opens custom corner settings dialog where it is possible to edit old and add new corner types.

Video: https://vimeo.com/277423174
https://player.vimeo.com/video/277423174

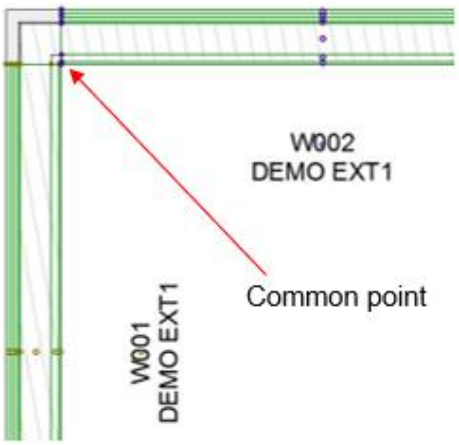
10.3.1 Follow these steps to model corners:

1. Open the corner tool by going to Element Tools → Corners and joints.
2. You must have two elements that share at least one point.

1.



2.

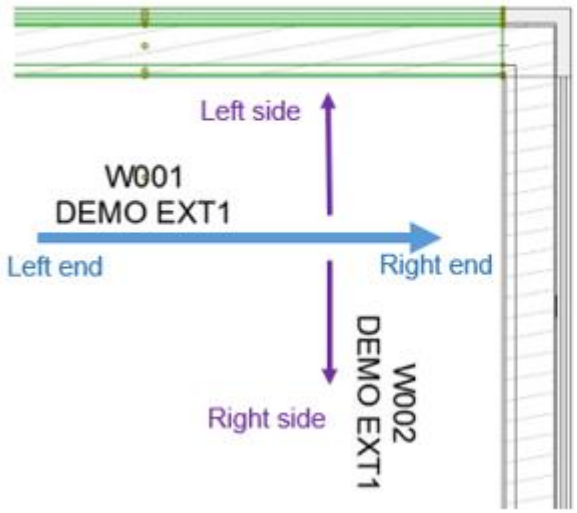


3. Select one element, choose appropriate corner settings and click "Apply".

   - In Picture 3, the selected element is W001 (highlighted in green).
   - The direction of W001 is shown by the blue arrow, and can also be determined from the direction of its text (W001 Demo Ext1).
   - W002 is therefore at the right end of W001 and on its right side.
   - Choose "Long L right" as the type of the element's right corner:

     - Right corner because the connecting element is at the right end of W001.
     - "Long L right" because we want W001 to make up the longer part of the corner and because the connecting element is on the right side of W001.

3.



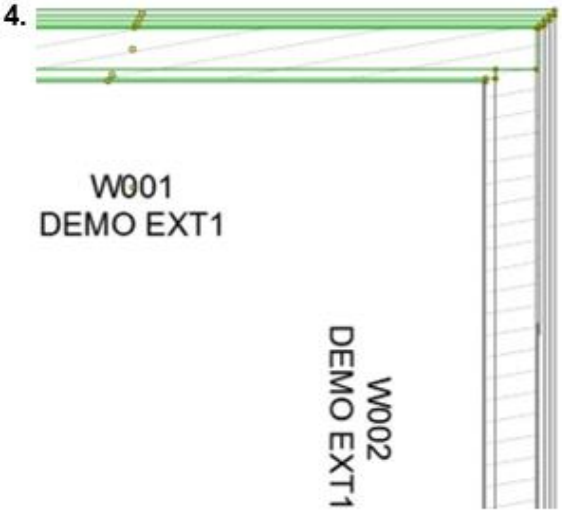
4. The result.

   - The wall element W001 now makes up the longer part of the corner.
   - Archiframe automatically sets W002's end to type "short".

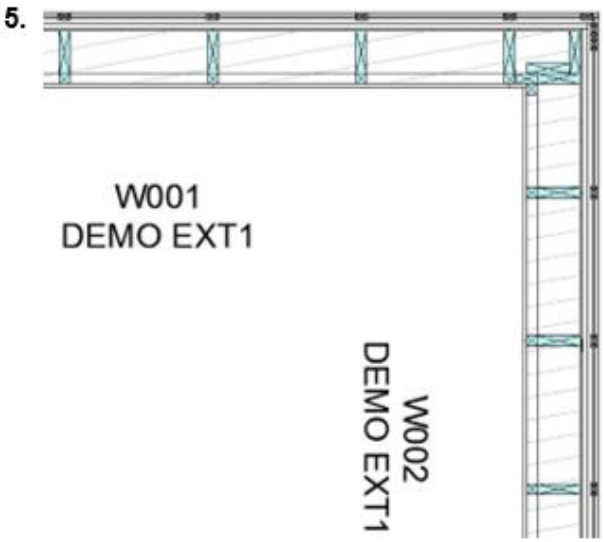
5. Select both elements and click "Create planks" to see how the corner planks are placed.

   - Create planks can be found in the "Add Element" window.

4.

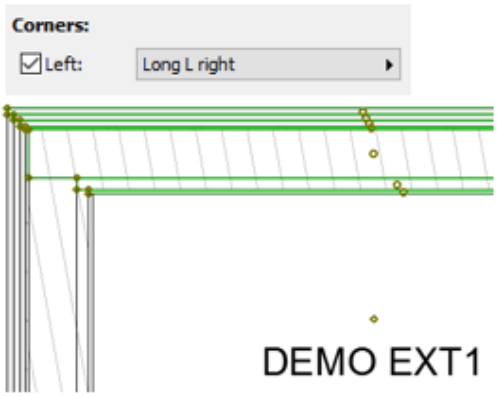


5.

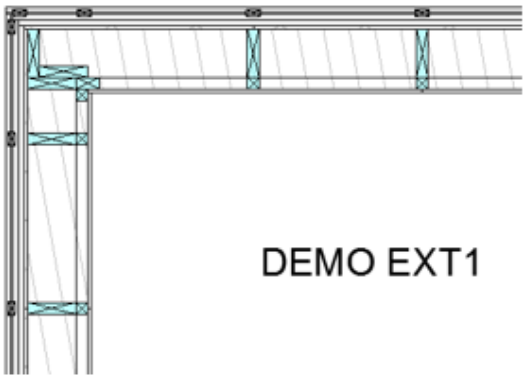


10.3.2 Examples of different corner types

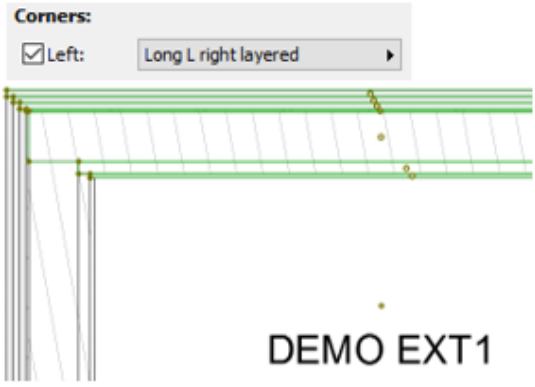
- Sometimes different corner types look the same when set for elements. In such cases the corner types' differences become apparent after planking is created.
- For example, styles *Long L right* and *Long L right layered* only differ after planking:



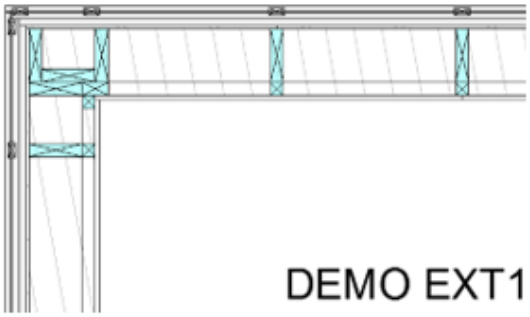
The selected wall element's left corner has corner style *Long L right*.



Result after planks have been created.

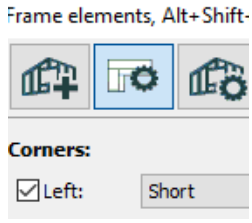
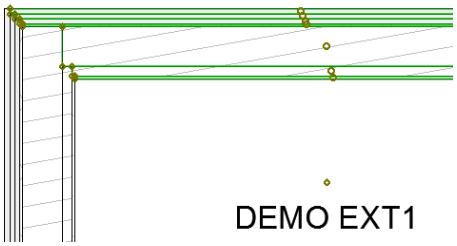
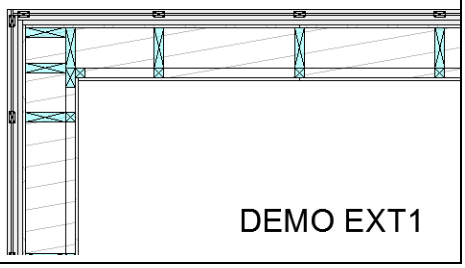


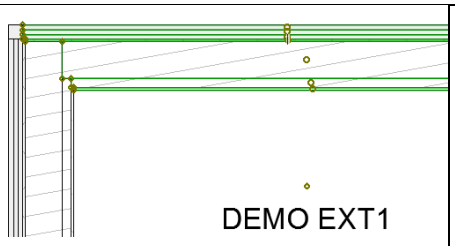
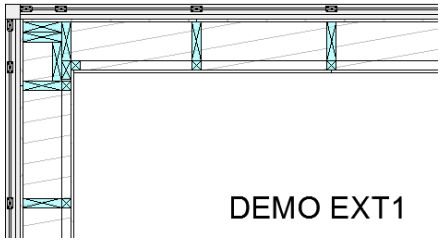
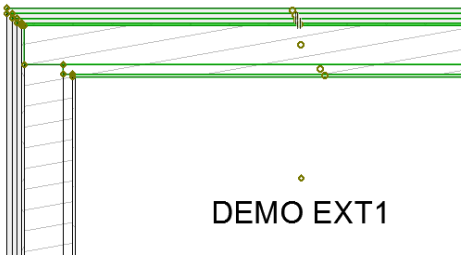
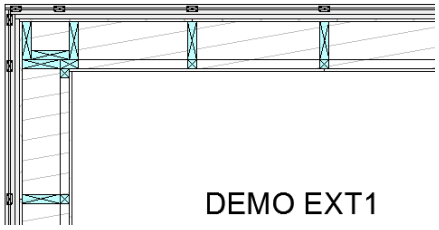
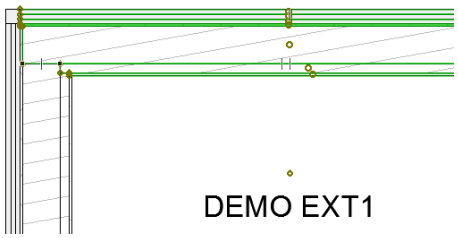
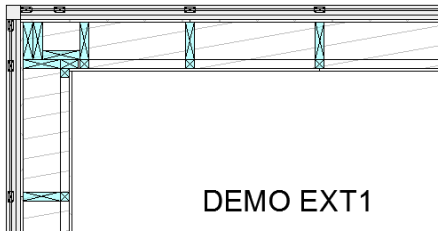
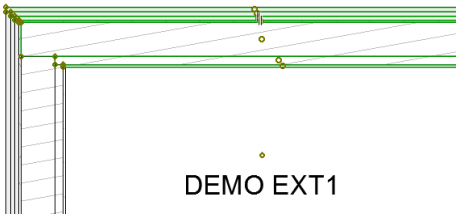
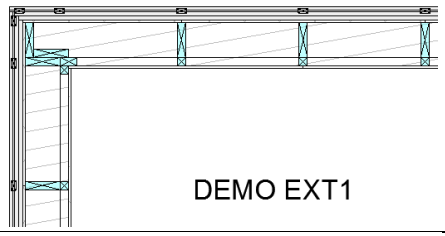
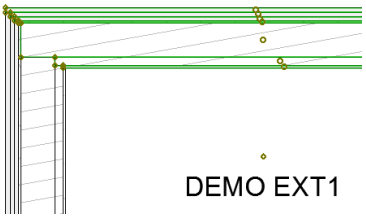
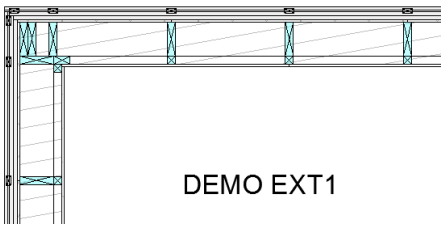
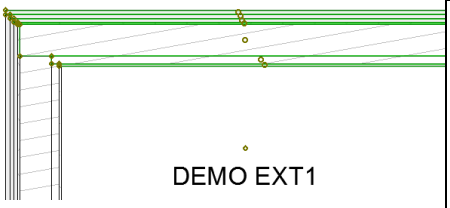
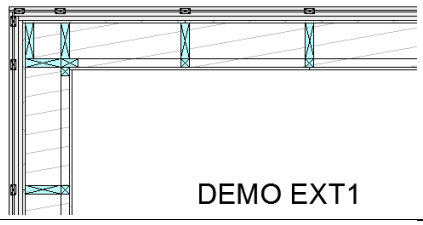
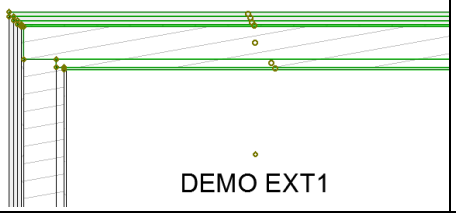
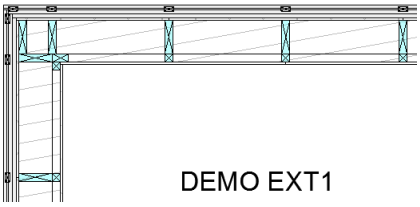
The selected wall element's left corner has corner style *Long L right layered*.

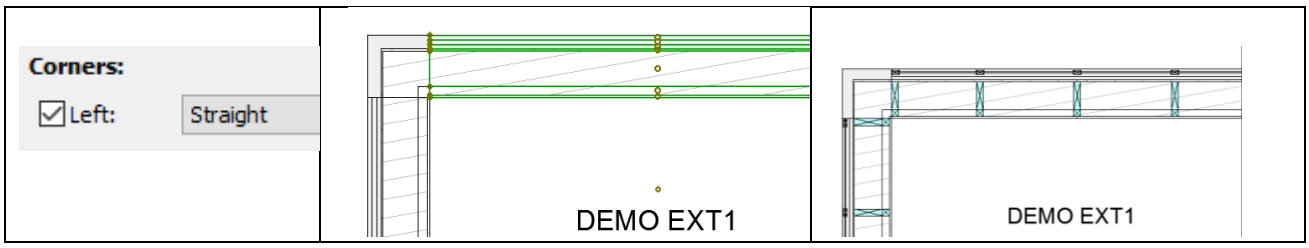


Result after planks have been created. The planking is different from the one above.

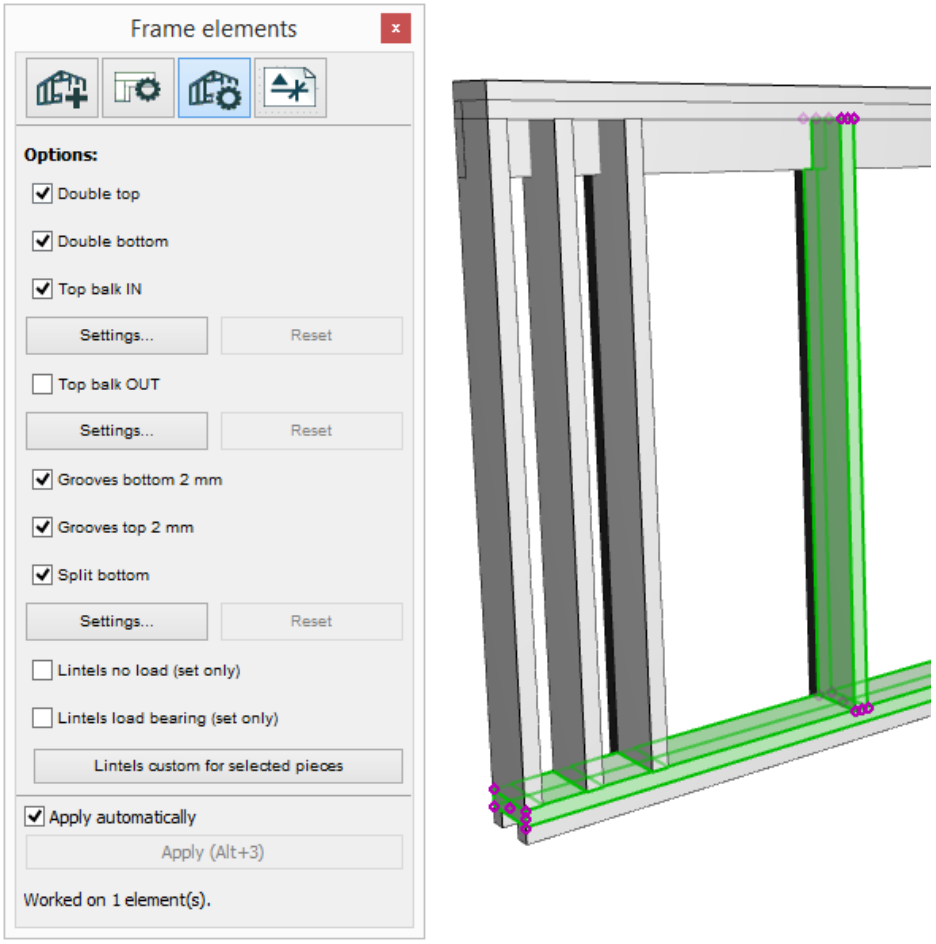
Here is a table of different corner types before and after planking. The wall element type used is called Demo Ext1 and contains (from inside out) gypsum, studding, framing, boarding, horizontal studding, vertical studding, and cladding.

Corner type	Before planking	After planking
		

Corners: Left: Short 3		
Corners: Left: Long L right layered Long L right layered		
Corners: Left: Long L right layered 3 Long L right layered 3		
Corners: Left: Long L right		
Corners: Left: Long 3		
Corners: Left: Long 2		
Corners: Left: Long		



10.4 Element options tools



Sets various options to the element. The options are automatically applied when recreating the elements. Some options like lintels cannot be unchecked. The options are defined in ArchiFrameElements.xml with Lua-scripts and can be customized.

When changing many options to a big element it may be useful to uncheck *Apply automatically*, set all the options and apply those once with *Apply*-button.

Planks to opening sides and lintels custom can be limited to only selected pieces (please select the top plank for lintels). The element should be replanked after changing option *Planks to openings sides* since the option affects studs from spacing rule. Or the option should be initially on before creating the planks.

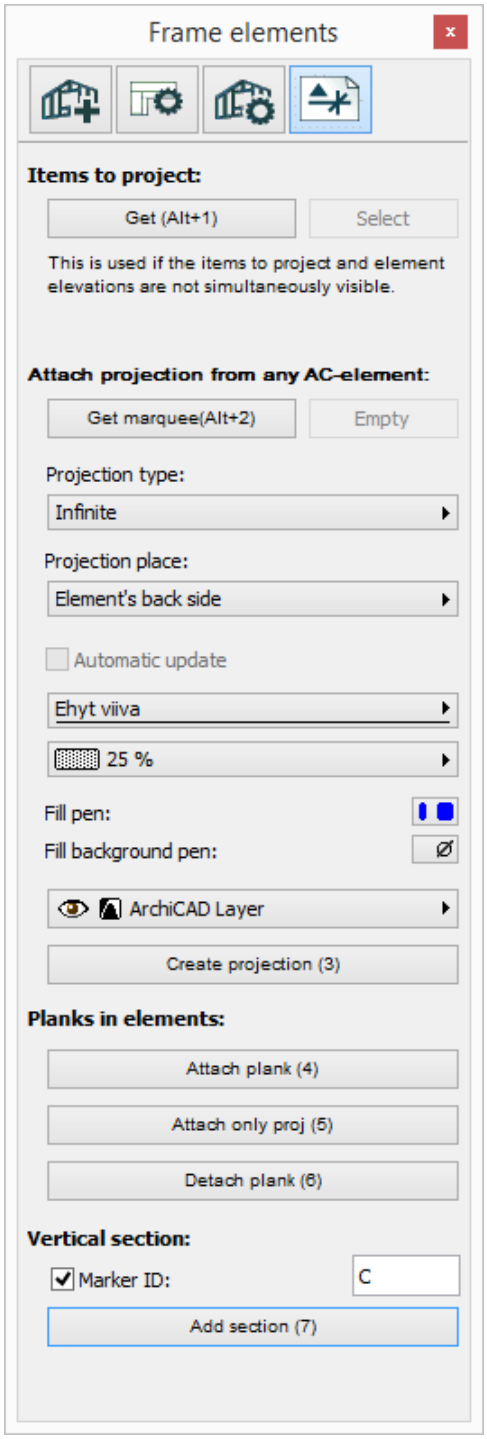
Links to videos:

- Double top and double bottom: https://vimeo.com/173890540.

10.5 Projection tools

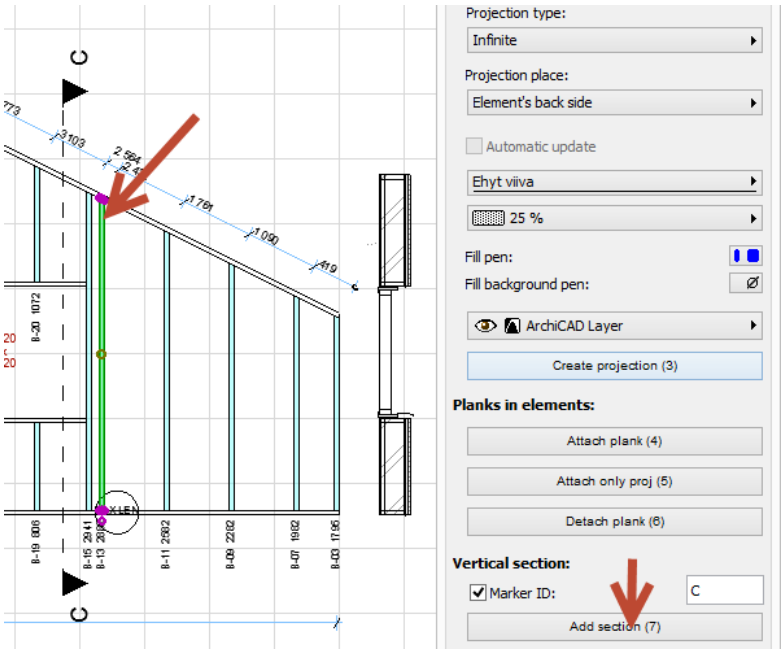
Projection tools are used to attach plank projections or projections from any ArchiCAD element to the element elevation. *Attach projection from any AC-element* creates the projection as fill and it will not be updated if the original element is changed and vice versa. Moving the projection fill will not move the original ArchiCAD element. *Planks in elements* tools makes fully updating projections from ArchiFrame planks only.

Projections can be added by selecting the elements or planks to project, at the same time as the destination projections where source elements should be projected. If this is not possible, for example, because the element elevations are in different storey than the elements to be projected, the *Items to project / Get* –command will pick the source elements. After this the projections are created with one of the commands *Create projection, Attach plank or Attach proj only.*



- *Get* will take source elements, for example, from another storey to be projected. All source elements to create projections or attaching to elements must first be picked with this. Can be used also to pick ArchiCAD walls/roofs/slabs to redefine sources for ArchiFrameElement-objects.
- *Get marquee* picks current ArchiCAD marquee to define projection clipping area. It is useful to show, for example, just a part of long beam in single element elevation.
- *Projection type* can be zero depth or infinite. For example, if there is a log joint at the projection plane (the element's front or back side), the log will appear as shaped from the joint if using zero depth type.
- *Projection place* is either on element's front or back side.
- Rest settings in the group define the resulting fill. ArchiCAD fill tool settings are used as well.

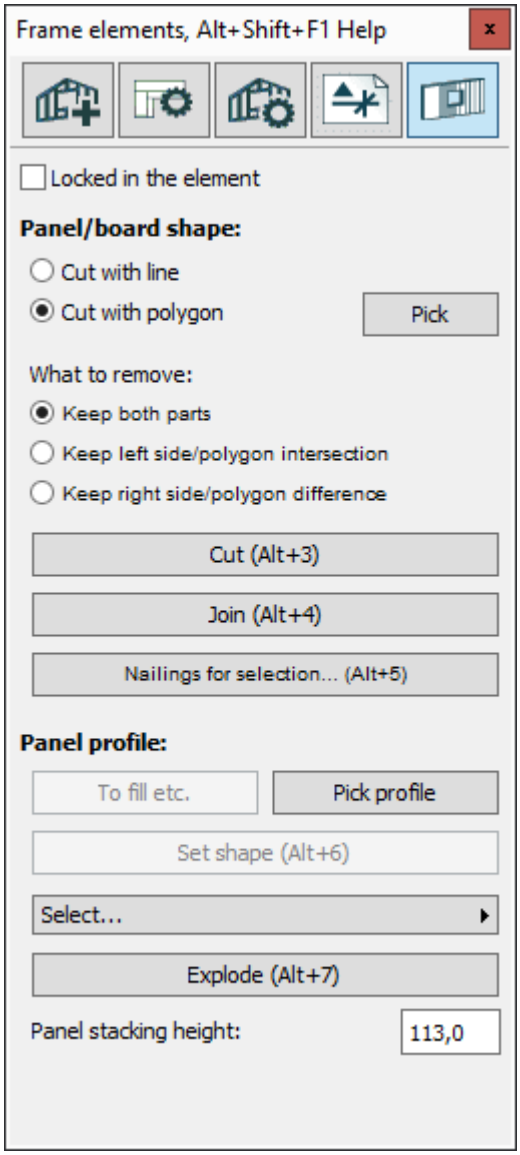
- *Create projection* creates selected or picked source elements' projections to selected element elevations as ArchiCAD fill.
- *Attach plank* joins selected ArchiFrame plank to be part of the element. The attached plank is deleted when element is replanked. *Attach only proj* places only projection to the element elevation. The original plank is not deleted if the projection is deleted or element is replanked. This is the recommended option to add projection of a beam to the element elevation.
  If the picked sources were ArchiCAD walls/roofs/slabs, attach will redefine sources for ArchiFrameElement-objects. This will allow ArchiFrame to for example update openings or calculate door/window weights for copied ArchiFrame-elements.
- *Detach plank* deletes selected plank projections without deleting original plank. In addition, the link from the original plank to the element is removed. This operation can be issued having only original plank(s) selected. In this case, it will remove every related projection and makes the plank independent from any element. Another option is to first use *Get* to pick planks/boards to be detached, then select ArchiFrameElement-objects or parts related to those. In this mode, planks will be detached only from related ArchiFrameElement-objects.
- *Add section* creates vertical section of the element. To use it select suitable layer combination (to see windows, the wall layer must be on), select one piece from projection, click section line, side and place of resulting section. ArchiFrame creates a temporary section and converts it into static line drawing and places the result to floor plan. The drawing is not automatically updating.



10.6 Board and panel tools

These tools are used to edit polygon shape of the boards and panels. The difference to for example planks' groove tool is that it makes a new machinings – this tool just changes the polygon. There may be a big difference in resulting cnc-.

Please note that element main tools' operations to edit the outline *To fill* and *Pick from fill* can be used to boards and panels as well as for elements.

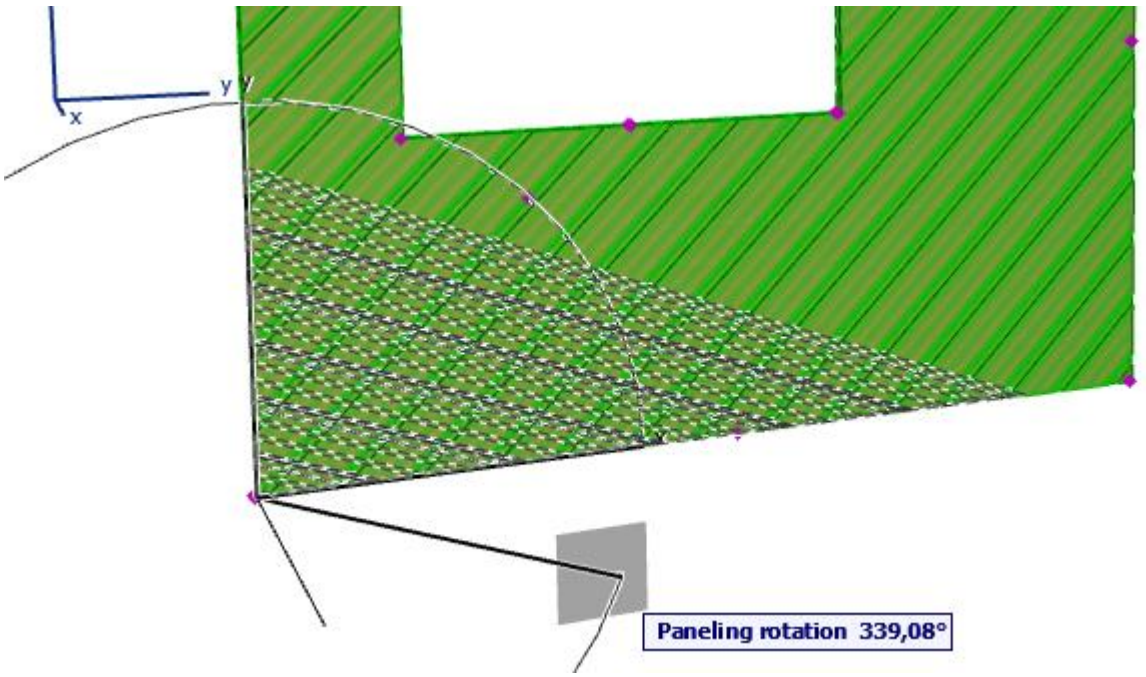


- *Locked in the element* sets or resets locking state for a board. When recreating the whole element or just boarding (having a board selected and clicking element main tools' button *Recreate boarding*) all locked boards remain at place and the rest is done according to locked ones.
- *Cut with line* allows splitting boards and panels into two or more pieces or just trimming existing ones.
- *Cut with polygon* first a working polygon is picked, then the target boards. This command is useful for example to change paneling just at part of an element (below windows for example). To use a polygon it must be first picked with *Pick*-button from existing fill or marquee.
- *Keep both parts* removes nothing. Instead with this option selected existing pieces are split into two or more pieces.

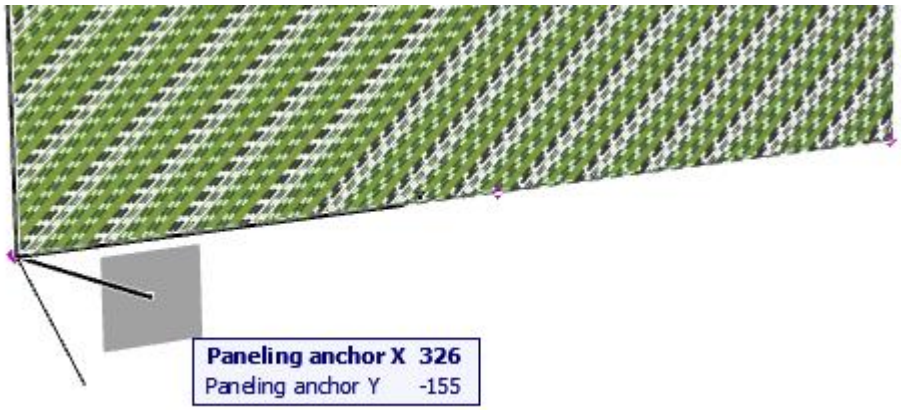
- *Keep left side/polygon intersection, Keep right side/polygon difference*, with polygon operations the target pieces must be selected. The operation results as (red one is a board/panel and blue one the picked polygon):

- *Cut* cuts boards.
- *Join* joins selected boards and panels together. The polygons of the boards to join must touch each other. If not, the outline needs to be edited prior to the operation.
- *Nailings for selection* places nail/screw lines to the boards. See Element nailings dialog.
- *Create saw cuts* makes board cutting objects to be used for cnc. Please see Panel cuts for more details.
- *Pick profile* picks the profile from existing panel or specific panel profile drawing. Pressing shift when clicking the button dumps the profile as text to be copy & pasted as a preset item into ArchiFrameElements.xml. Please see section Panel profile definition.
- *To fill etc* creates ArchiCAD fills, hotspots and lines for previously picked panel profile. These ArchiCAD elements can be edited to edit the paneling profile.
- *Set shape* sets previously picked shape for selected boards/panels.
- *Select…* sets a predefined panel type (in ArchiFrameElements.xml) for the selected boards/panels.
- *Explode* explodes panel object into separate pieces for further editing or cnc-production. Please see Panel cuts for more details.
- *Panel stacking height* sets the distance between exploded planks. This is used to match result with current cladding batch since actual height of the panel may vary.

Panel profile rotation is set with a hotspot in 3D, floor plan or element elevation:

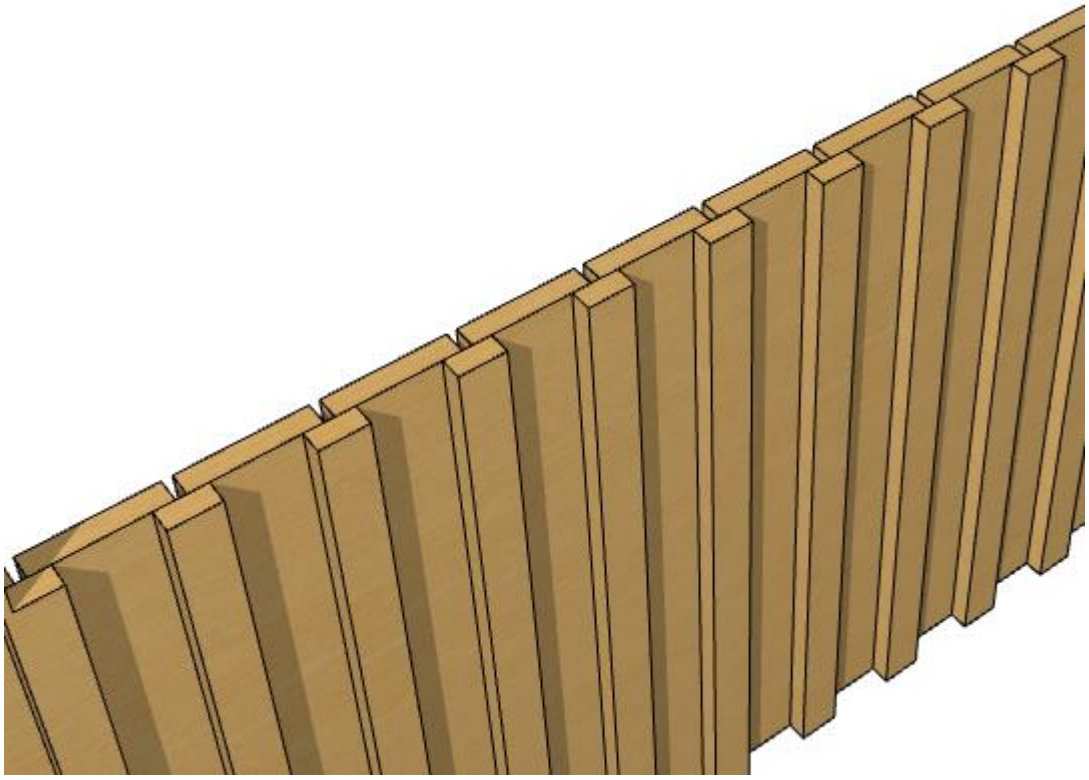


Also there is a hotspot to set the paneling achor point:



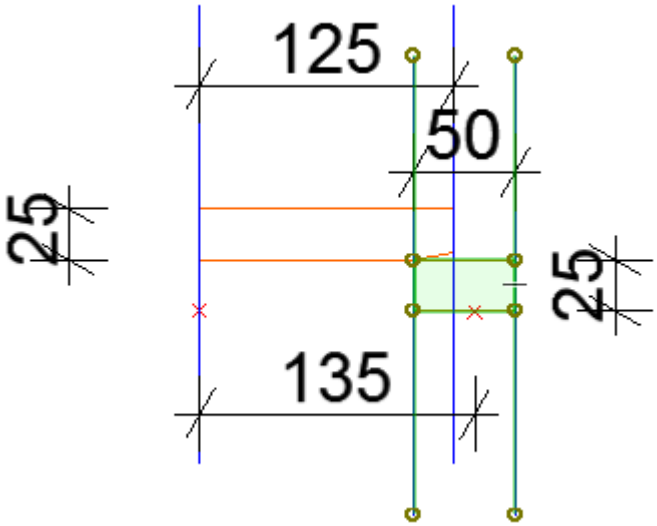
10.6.1 Panel profile definition and board/panel object's information

Let's use following profile as an example:



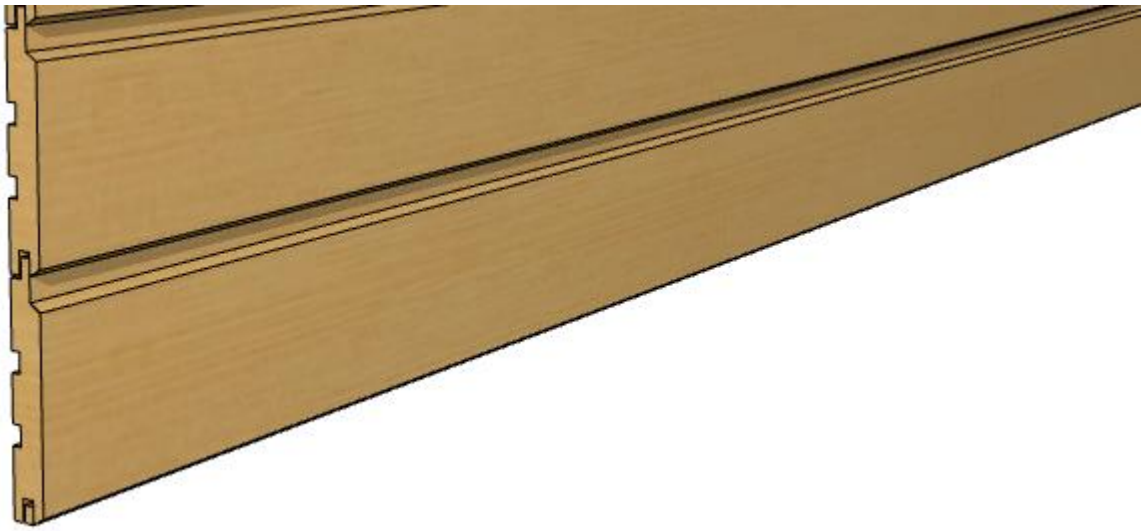
The paneling consist of two different pieces: 25 x 125 mm and 25 x 50 mm planks. The paneling profile definition contains definitions:

- Separate profile and placement for each plank type as ArchiCAD fills.
- Lines to show in section (not to use real profile producing possibly much too many lines) as ArchiCAD lines.
- Repetition anchor point and distance for the profile. In this example it is 135 mm. These are defined with ArchiCAD hotspots. The left most anchor point works also as origin.

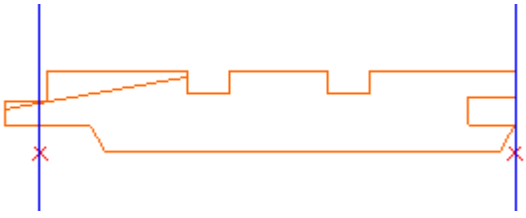


The image above has elements in two groups: First one contains the bigger one and it must contain also two hotspots defining the origin and the repetition distance (135 mm). All parts should have positive y-coordinates. The second group is selected here and it contains the profile fill and two lines used to show the profile in sections. Please note that the profile is mirrored along if the element is looked from inside. The default *data*-folder contains similar panel definitions for both watching directions.

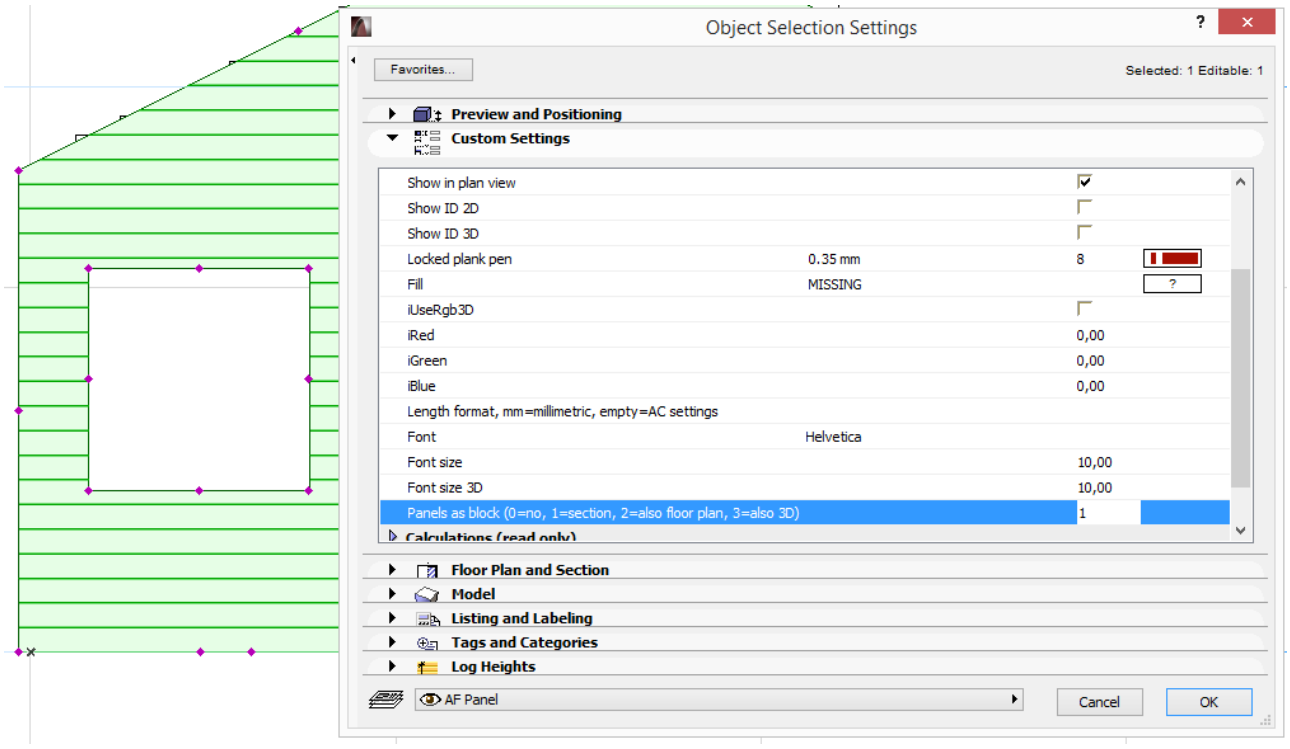
The meaning of the section lines is clearer for following paneling profile:



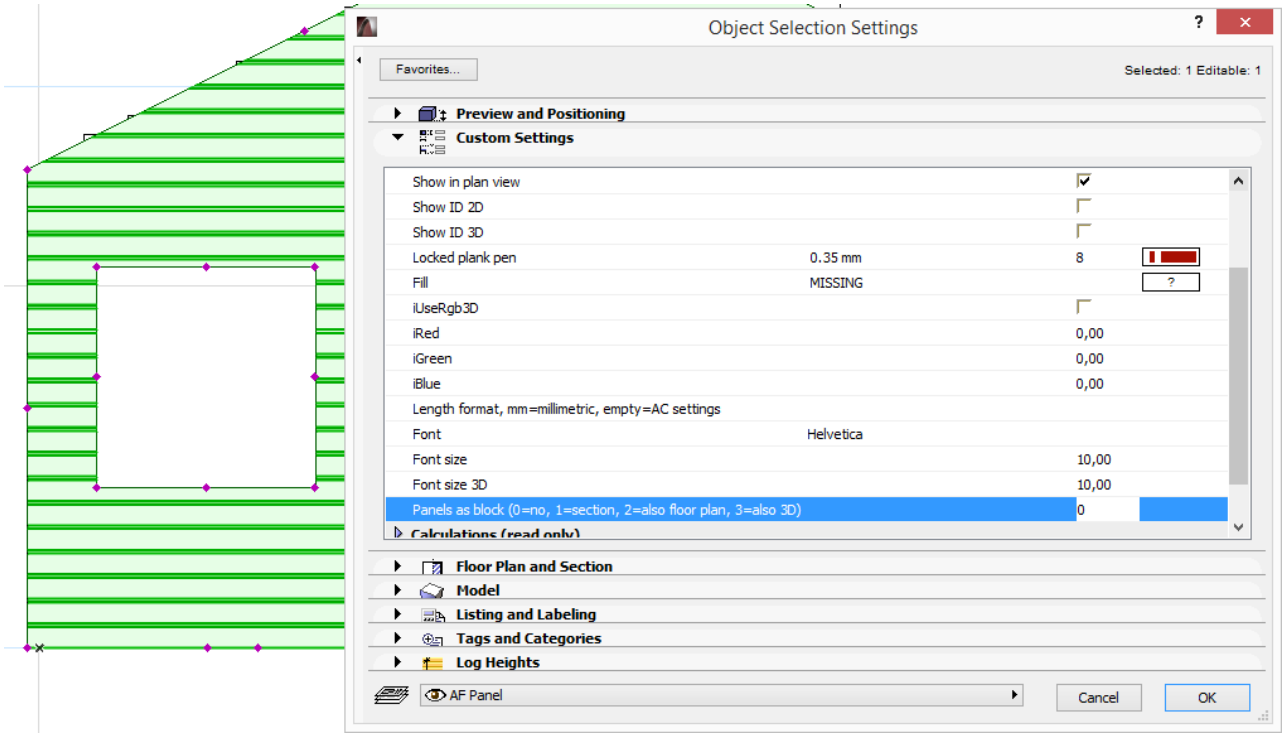
As panel definition:



With the default setting showing the lines only (Object tool/object parameters/View settings/Panels as blocks), the elevation looks like:



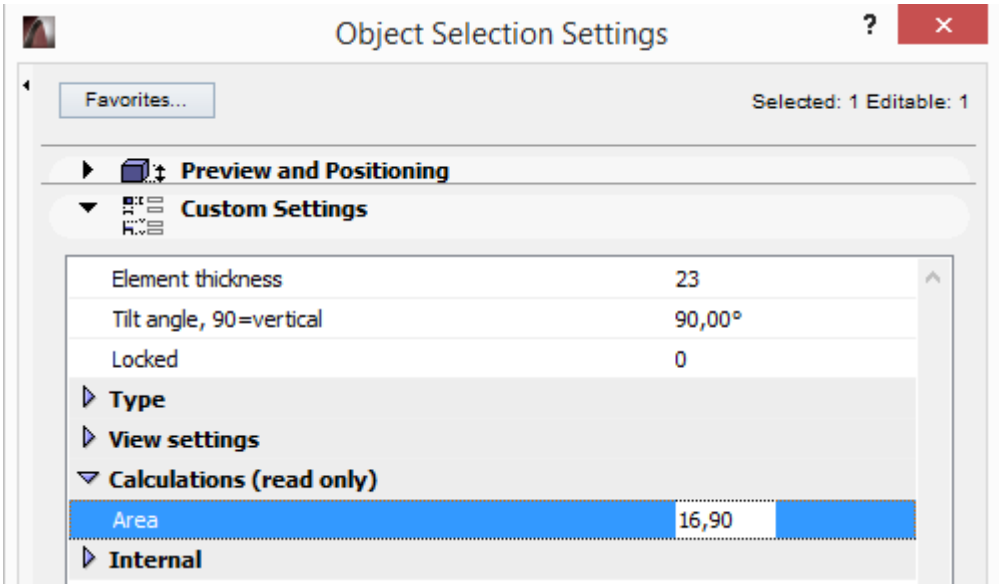
The same view showing the paneling with real profile is stuck with lines:



10.6.2 Technical information about the board/panel object

Preset profiles are saved in ArchiFrameElements.xml section <panelings>.

For calculations there is a parameter telling the net area of the board/panel (the area with holes reduced):



The panel profile is saved into ArchiFrameBoardPanel's table-parameter iPanelProf. It has triple items x, y, status. The special status codes are:

- -1, end of contour and next point is hole's first coordinate. Contour/hole starting point must be duplicated.
- -100, single panel profile ends, x,y unused.
- -101, size of current piece if modeled as a block.
- -102, Y-offsets from origin x = y1; y = y2.
- -103, X-offsets for lines in section x = x1; y = x2.

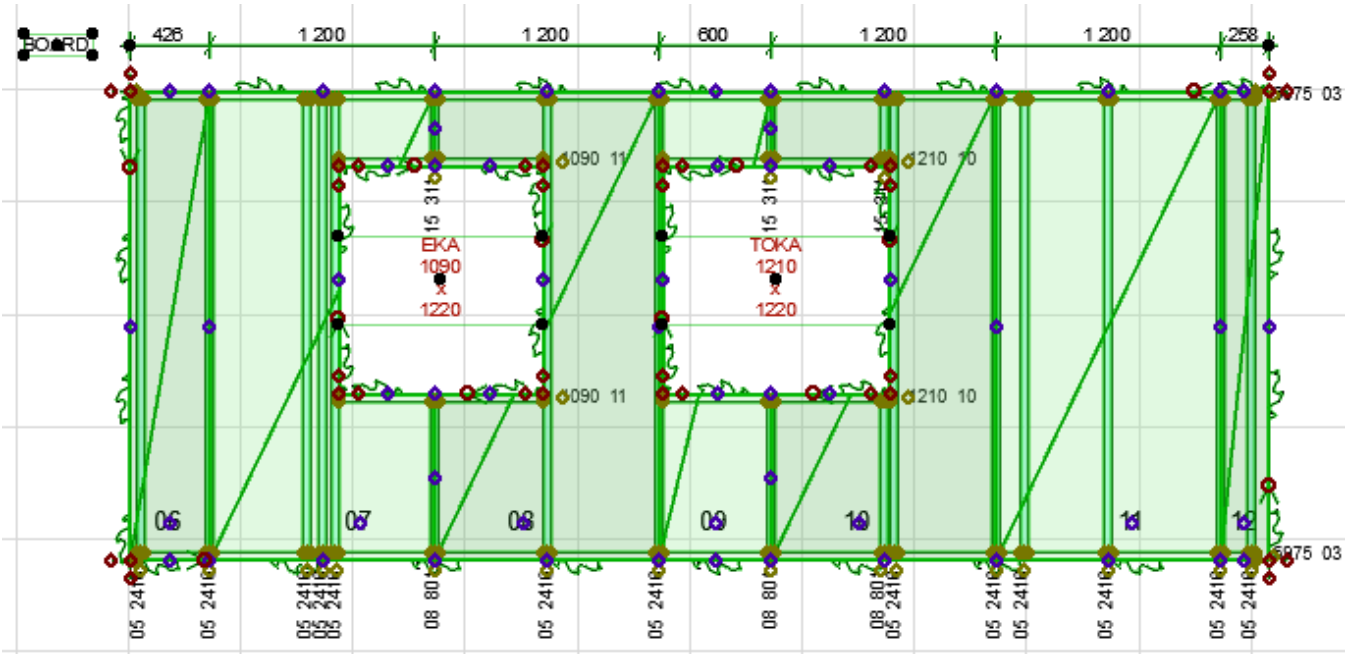
10.6.3 Panel/board cuts

There are two ways to define cuts for cnc-production: either use visible ArchiFrameSawCut-objects or rely on ArchiFrameBoardPanel edges. If there are any visible ArchiFrameSawCut-objects related to the target element layer (boarding or cladding), those will be used. ArchiFrame will warn about ArchiFrameSawCut-objects on hidden layers when creating wup cnc. If there are no ArchiFameSawCut-objects placed to the target boarding/cladding layer, the cuts are defined in the related ArchiFrameBoardPanel-objects.

10.6.3.1 Panel/board cuts with ArchiFrameSawCut-objects

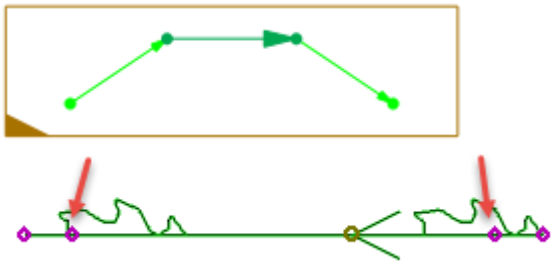
Saw cuts are placed by selecting the target layer boards and all existing saw cuts to remove those and then clicking *Create saw cuts* from the element board tool palette. Planks in selection are ignored. For example, to create saw cuts for the boards in this elevation it is ok to select everything

using selection rectangle – here existing saw cuts are removed, and new ones created:

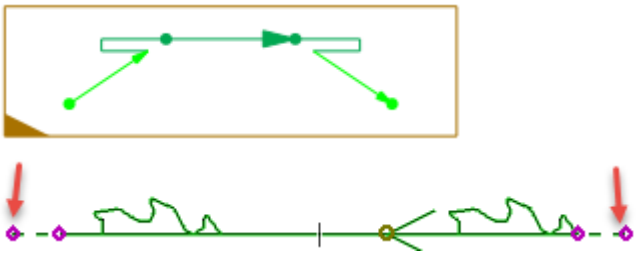


ArchiFrameSawCut-object has hotspots:

- At the ends to define whether it is an undercut:



ond overcut:



- Movable hotspots to drag the ends

To change blade side if automation fails or for special cases, the ArchiFrameSawCut-object can be mirrored. Deleting the saw cut removes the saw cut from the resulting wup-file.

ArchiFrame will automatically switch saw blade to router if the cut is undercut in both/other end and the cut is shorter than specified in saw blade default settings in ArchiFrameElements.xml <sawcut **routermaxlen="0.3"**>.

10.6.3.2 Panel/board cuts with ArchiFrameBoardPanel-objects

Panel cuts are used for cnc-production. When exploding paneling layer into separate planks, ArchiFrame leaves original ArchiFrameBoardPanel object in place and does following:

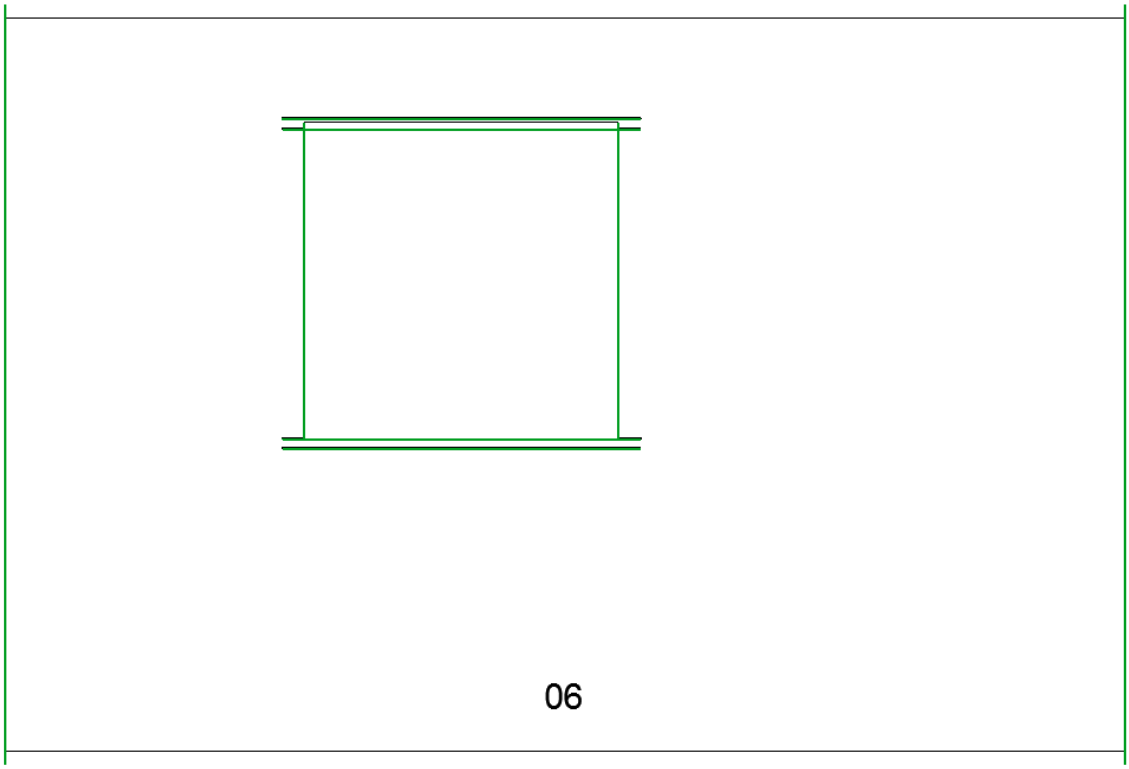
- Applies settings defined in ArchiFrameElements.xml path elem/settings/boardexploded. The default settings define that board object will not show the panel division any more, fill is

changed to Air space and parameter iSawLines is set to value 15 that causes ArchiFrame to set cutting lines.

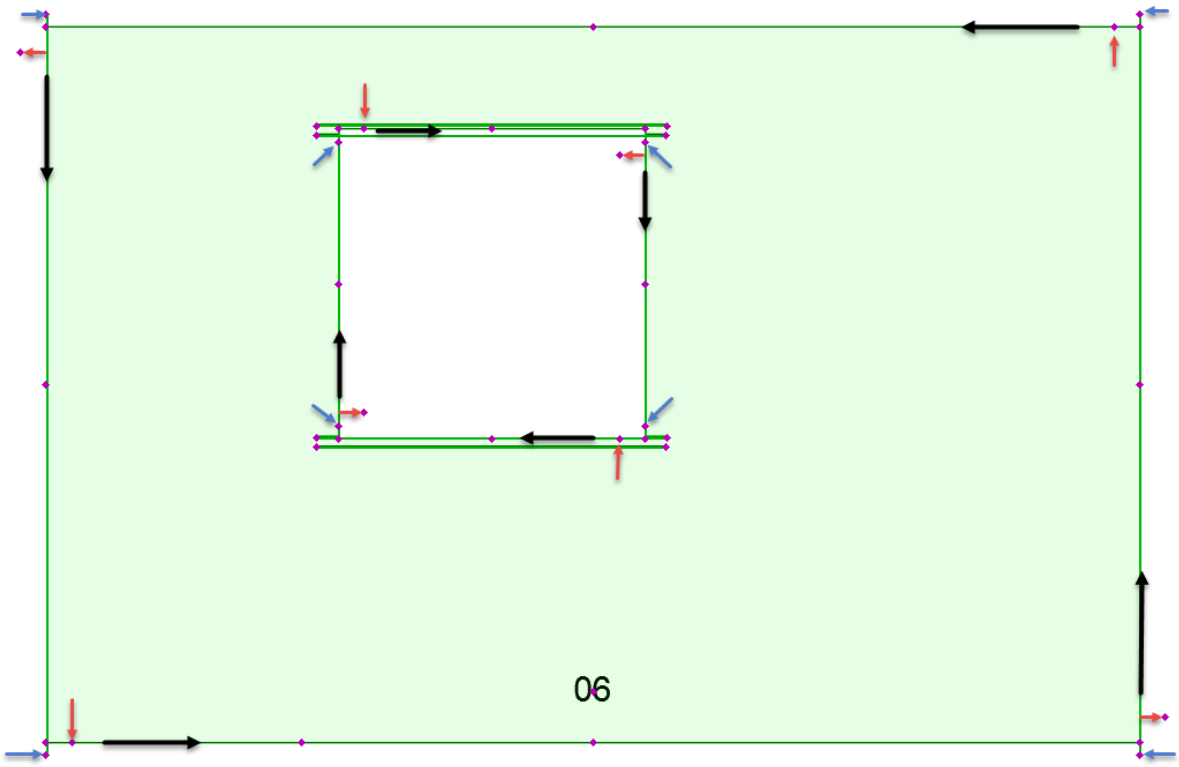
```
<boardexploded>
  <objparam name="iPanelBlock">4</objparam>
  <objparam name="iFill">Air space</objparam>
  <objparam name="iSawLines">15</objparam>
</boardexploded>
```

- There may be manufacturer specific processings in the *data*-folder that creates more cuts.
- Leaves the original board object(s) selected. The user can delete these objects by pressing Del after exploding.

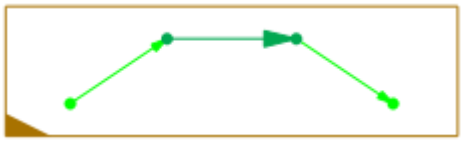
The image below shows the automatic cutting lines in green. ArchiFrame defines only the edges that are not parallel to paneling as cutting lines (in this case the paneling is horizontal). All opening edges will be cut lines even if parallel to the paneling. This example contains also manufacturer specific cutting lines for opening top and bottom:



The same with movable hotspots:



- Black arrows show the polygon edge lines direction and begin of the edge. Contour line is counter clockwise and holes are clockwise. Outside is always to right the hand side of the edge.
- Red arrows define whether the edge will be cutting edge or not: Having the hotspot on the line means no cut, having it to the right-hand side of the edge means that there is a cut. This hotspot is always 10 cm from begin of the edge.
- Blue arrows define whether the end of the cut edge should be overcut or undercut. Overcut means that the board/paneling cutting tool (saw blade or a small cutter/router) should travel over the corner to make sure that the cut is complete. For example undercut:

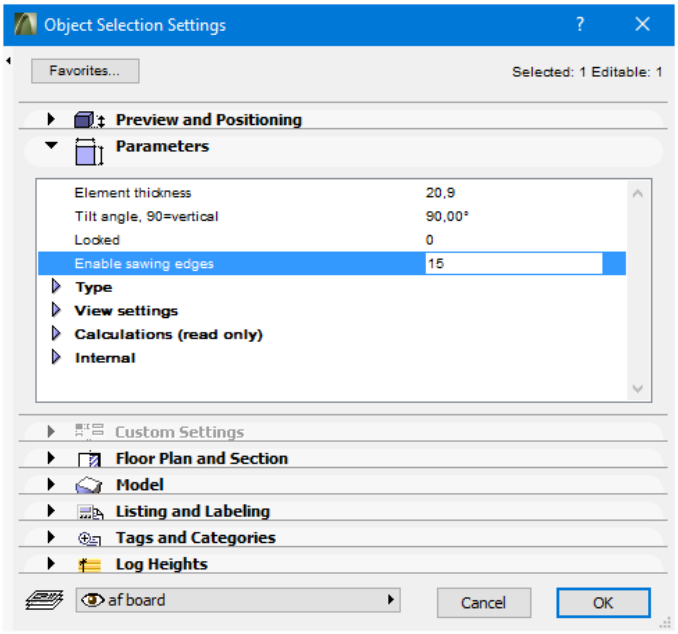
  

  And overcut:

- Other unmarked hotspots are to define the paneling anchor point and the polygon edges. The manufacturer specific special cuts also have editable hotspots.

These cuts are initially handled by ArchiFrame automation so you cannot edit those. To make the cuts editable there is parameter *Enable sawing edges* in the ArchiFrameBoardPanel-object:



It is sum of values:

- 1, let ArchiFrame manage the polygon edge cutting definitions.
- 2, let manufacturer specific script manage the special cuts.
- 4, reserved for future.
- 8, reserved for future.
- 256, enable cuts even if no automation is wanted.

Initial value 1+2+4+8 (15) enables all automation for the cut lines. To continue from initial cuts manually the parameter *Enable sawing edges* is changed to value 256. Now the user has full control over the cuts.

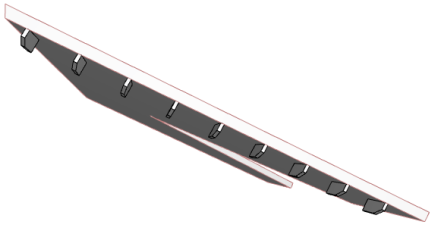
11 Modal dialogs

11.1 Multiply dialog

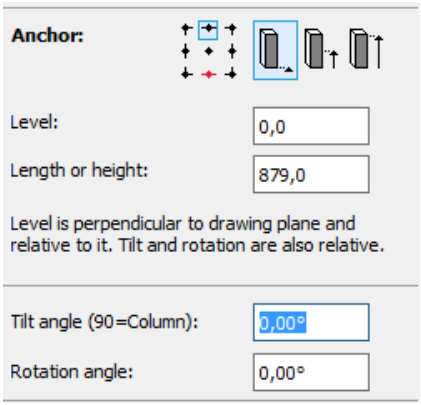
This dialog gives two extra features in addition to ArchiCAD's standard multiply-dialog:

- It follows ArchiFrame drawing plane if it is set.
- It is possible to define maximum distance between pieces and divide those with similar spacing on pointed line.

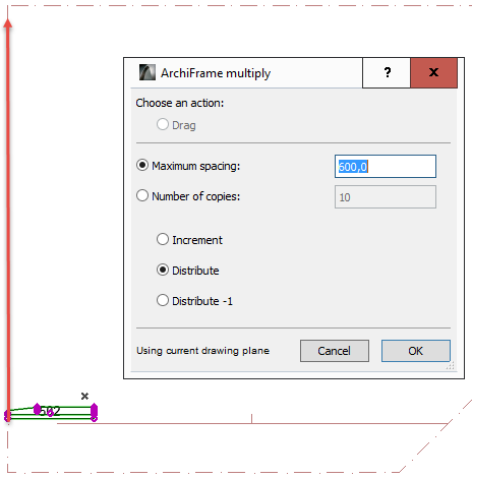
For example, to get the joists evenly spaced like this:



1. Pick drawing plane from the bottom of the roof.
2. Place the first joist setting the anchor point to middle top and set level, tilt angle and rotation angle to zero (these are relative to the drawing plane):

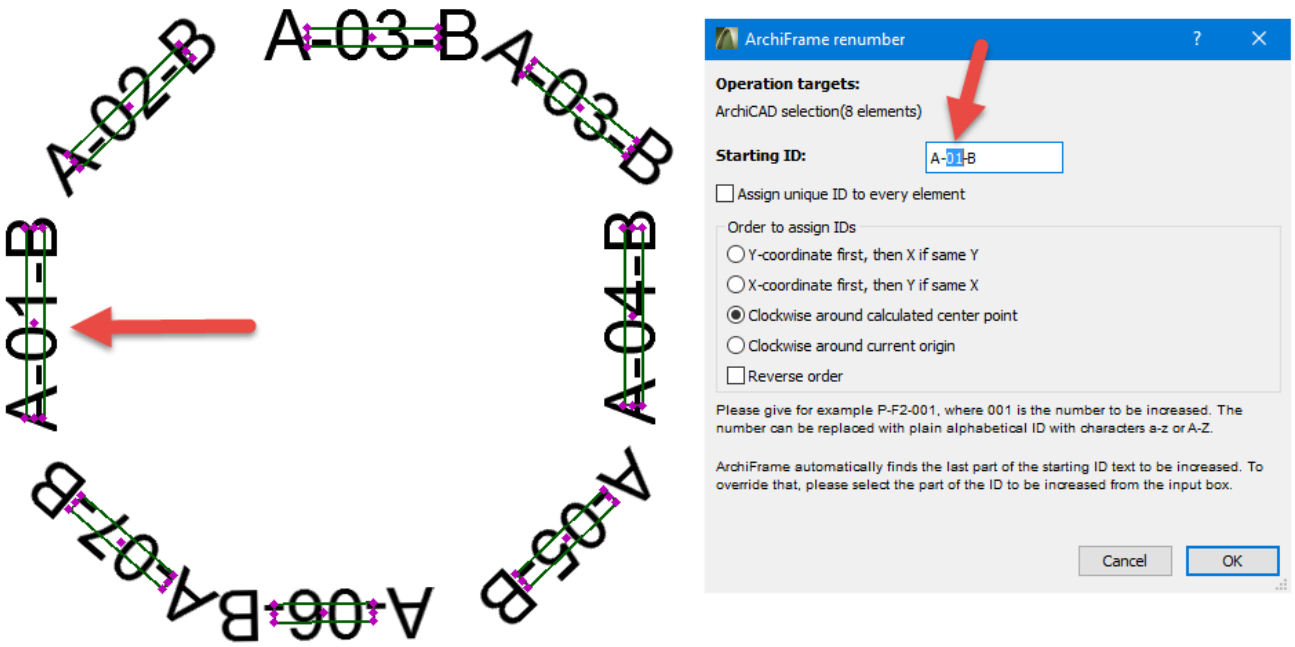


3. Select the first joist, open *Multiply* dialog and adjust settings like below. Finally show the line where the joists should be placed:



11.2 Assigining IDs

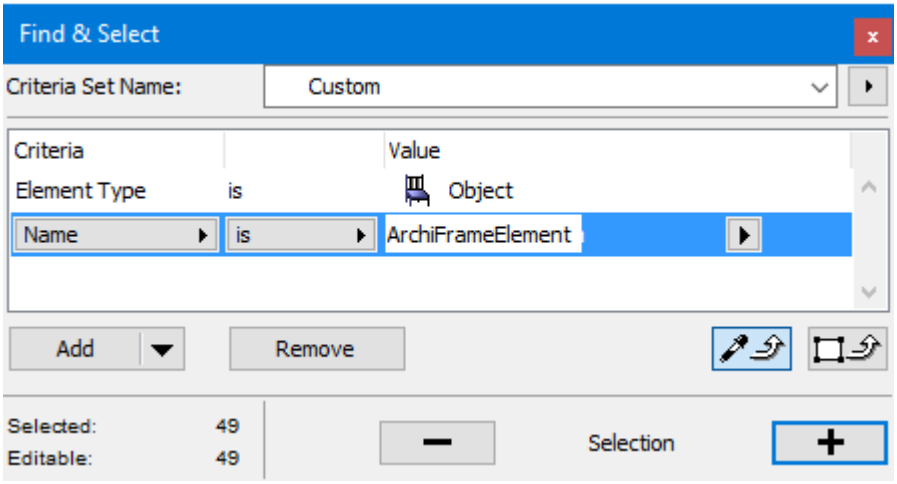
This dialog is used to assign IDs to ArchiFrame planks and element objects:



It is possible to define the part of the ID to be changed by selecting it before closing the dialog with OK.

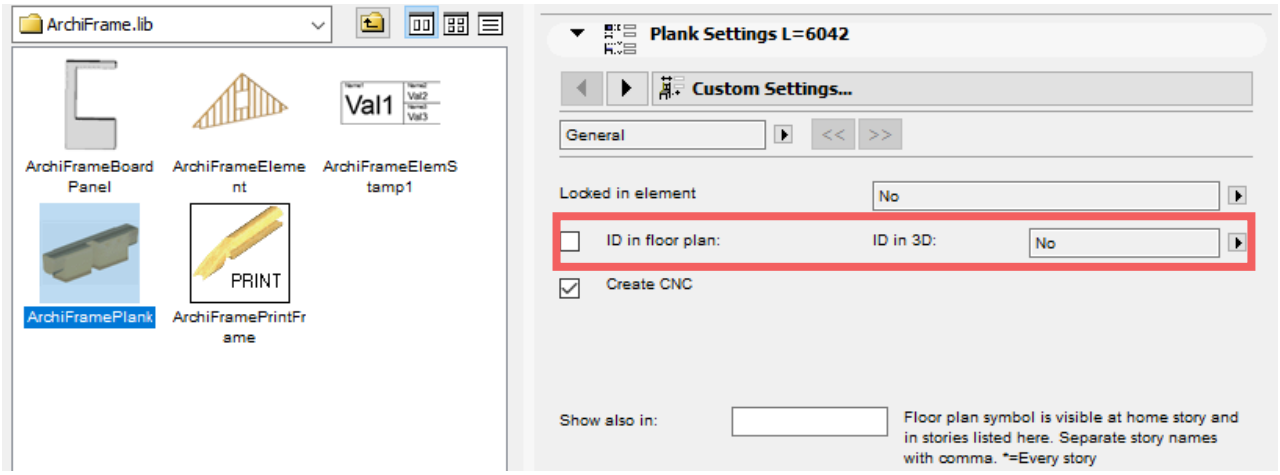
The last item selected will be the first one to get assigned ID. For element objects following procedure is helpful:

- Use ArchiCAD Find & Select to select all element objects:



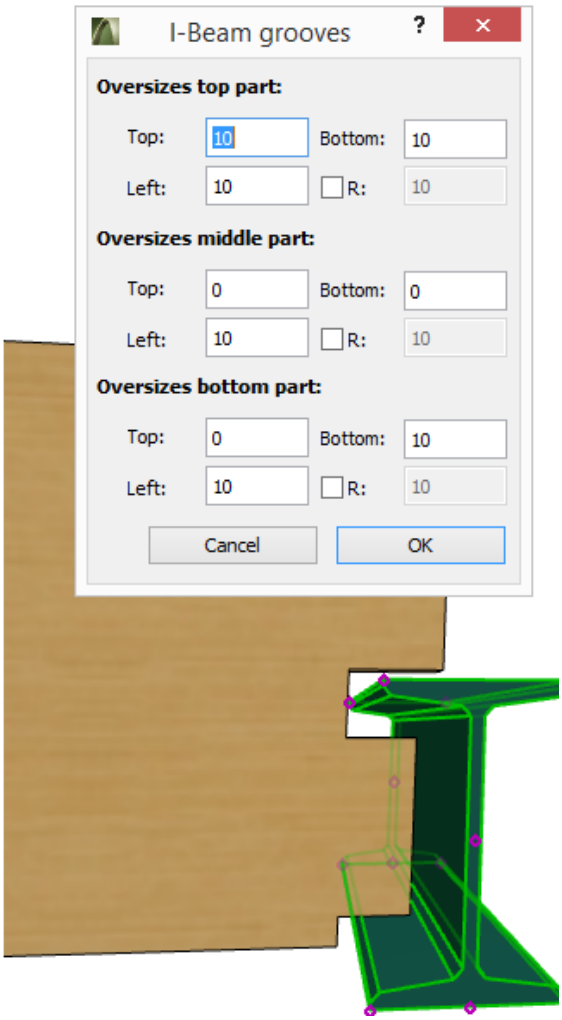
- Make sure that suspend groups is off, then de-select and select the first item again to be assigned shift-clicking it. Please note that selecting with rectangle will not work in this case.

In case the IDs don't appear in the floor plan or 3D view, go to the object's settings and make sure ID display is turned on. Object settings can be accessed by selecting the object and opening the ArchiCAD object tool (or pressing Ctrl + T).



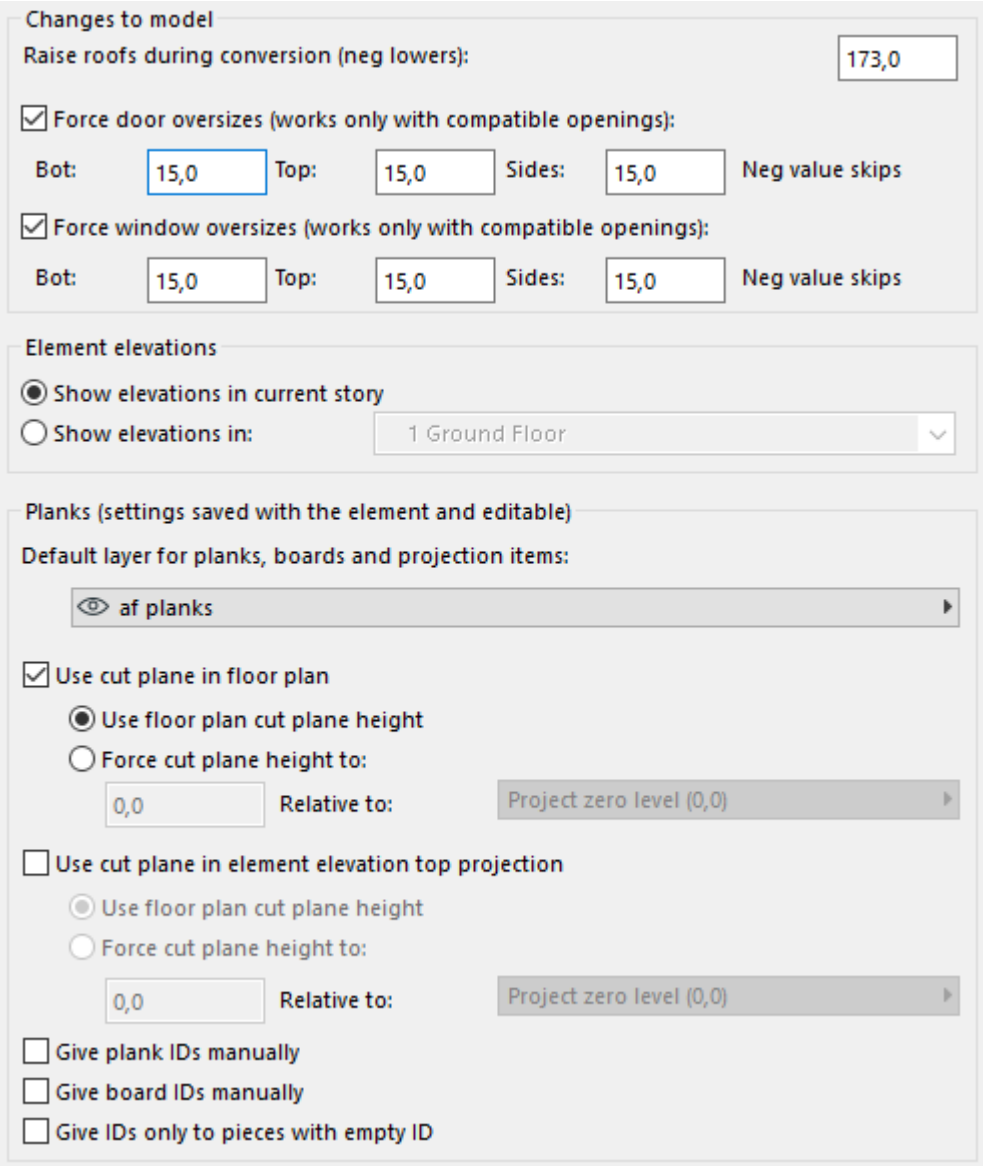
11.3 I-beam grooves

This dialog is used to make three grooves with three set of oversize-settings:



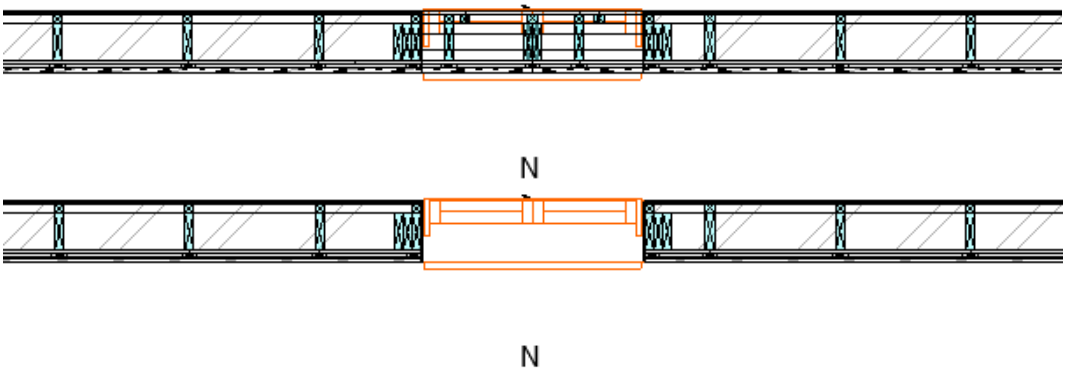
These oversize-settings are added to the main groove tool oversize-settings.

11.4 Element settings dialog

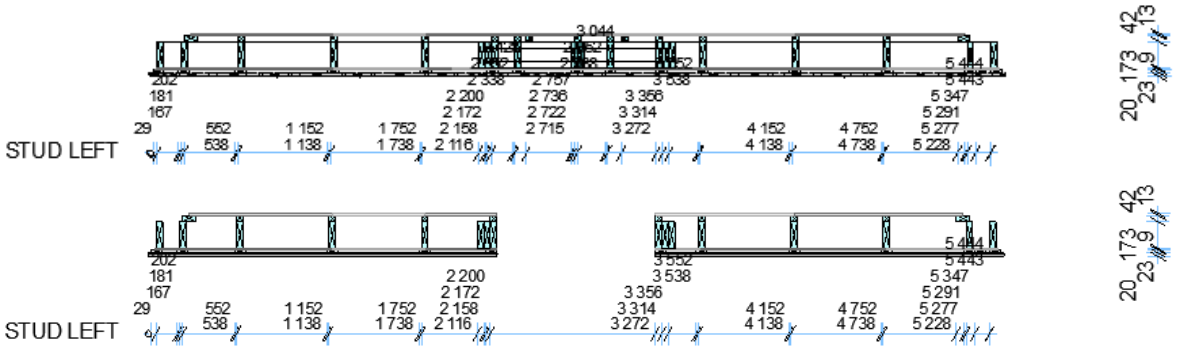


- *Raise roofs* lifts roofs temporarily during the element creation, perpendicular to the roof slope.
- *Force oversizes* sets oversizes to doors and windows that follow ArchiCAD standards. Some openings may use the standard parameters in their own way – it is important to verify that forcing the oversizes actually works with used library. Forced oversizes are set to library part parameters:
    - xl, xr, yu, yl: Used in some old openings, xl = x left/left side, xr = right side, yu = upper, yl = lower.
    - gs_left_oversize, gs_right_oversize, gs_upper_oversize, gs_lower_oversize.
    - ac_left_oversize, ac_right_oversize, ac_upper_oversize, ac_lower_oversize.
- Element elevations are perhaps best to be shown in separate storey with settings from *Element elevations*.
- Settings under planks-group can be edited for existing pieces. The changes affect all pieces belonging to selected element(s). Usually it is best to have suspend groups off and select every layer of multi-layered element and change the settings for all.

- *Default layer for planks, boards and projection items* is used when initially creating planks or regenerating projections under element tool's *Update*-button. This setting will not edit existing pieces since it is possible to define different ArchiCAD layers for each layer in the Custom Element Layer Settings dialog.
- Use cut plane in floor plan off and on makes this change (also the original ArchiCAD wall is visible here):

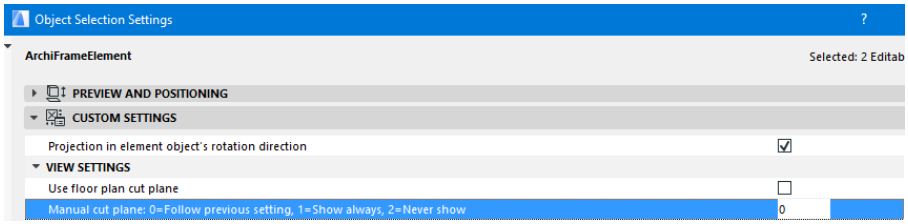


- In top projection the diffence is like below (the element must be updated with *Update*-button to get the dimension line(s) updated):

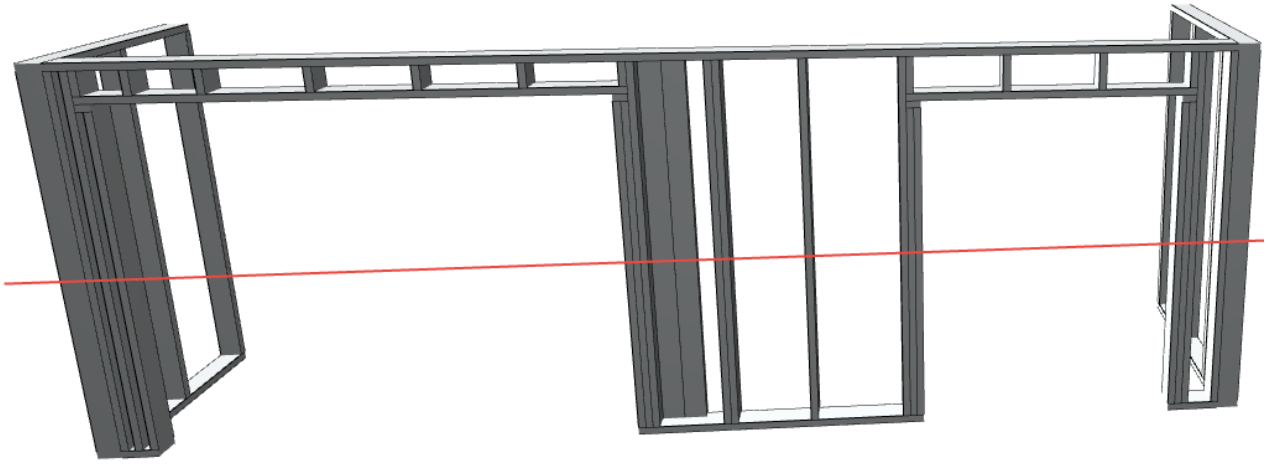


- *Give plank IDs manually* prevents ArchiFrame from assigning IDs to planks in elements when updating elements. This is useful to have same ID for similar planks in the whole building. But it requires that IDs are assigned manually just before creating end results like listings, cnc-file and layouts.
- *Give board IDs manually* does the same for boards.
- *Give IDs only to pieces with empty ID* is useful if there are changes after the prints have gone to production. The process of adding new planks using this option is:
    - Add new pieces and make sure their ID is cleared.
    - *Give IDs* and *Update* to assign IDs just to new planks and to update the cut list and dimension lines.

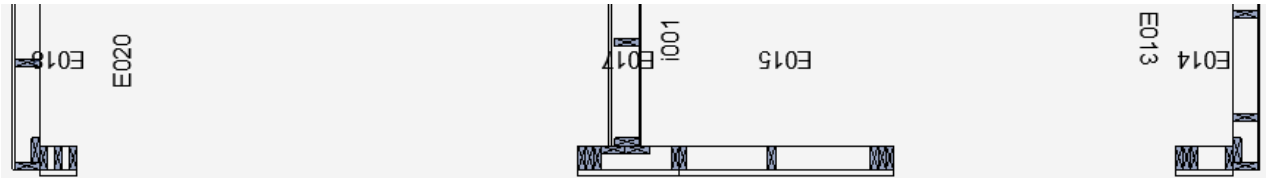
It is also possible to override the element specific cut plane setting for any element, plank or board object. They all have this setting:



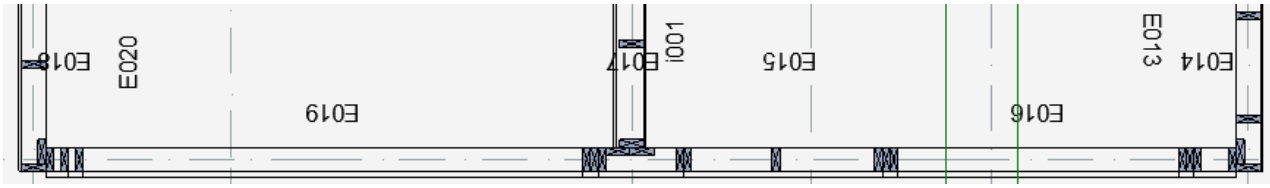
For example, in a case like this (red line shows the cut plane height):



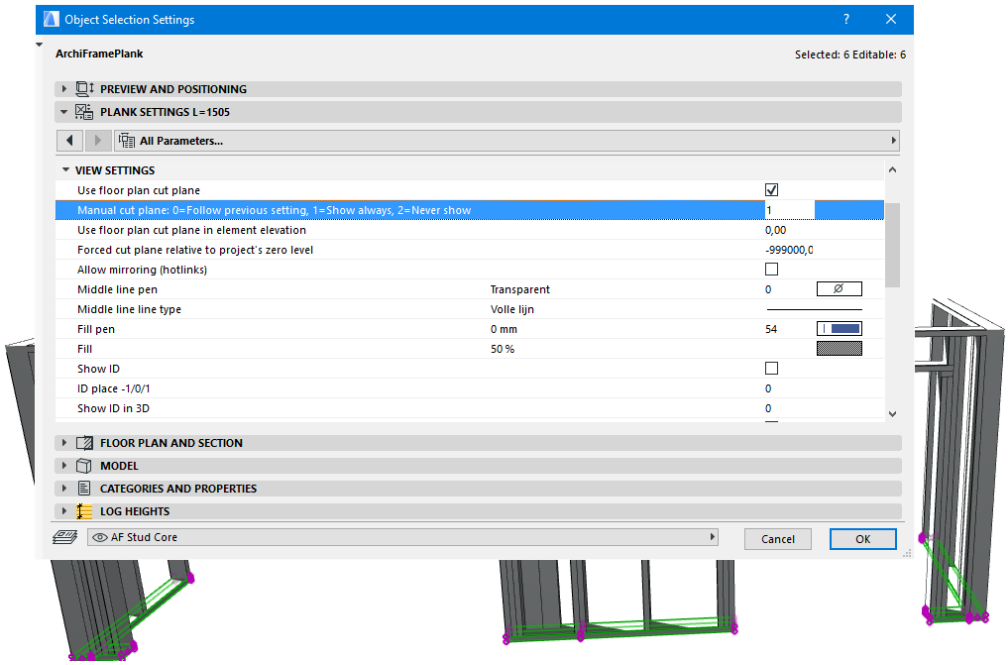
Using the cut plane, structure above the opening would remain hidden in the floor plan like this:



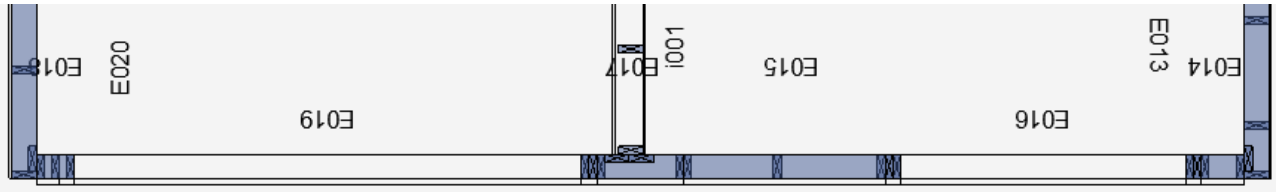
To have at least the element object visible in the floor plan *Manual cut plane* is set to value 1 for the element object and the floor plan looks like:



Another additional option is to set fill to bottom plates and force those visible also in floor plan:

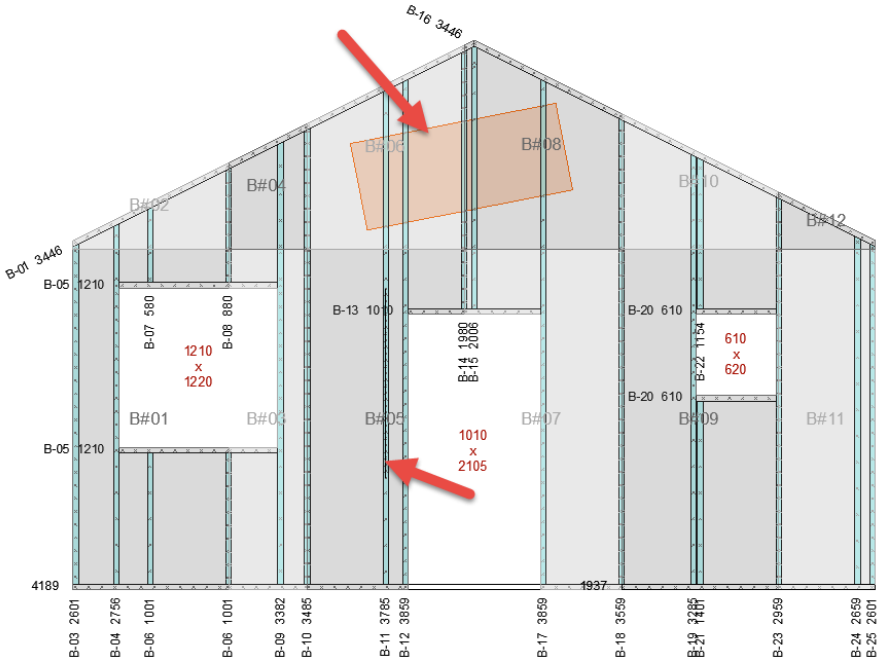


In floor plan it will look like this:



11.5 Element nailings dialog

This operation places nailing/screw lines to the selected boards based on selected framing planks. Nailing the planks is also supported.

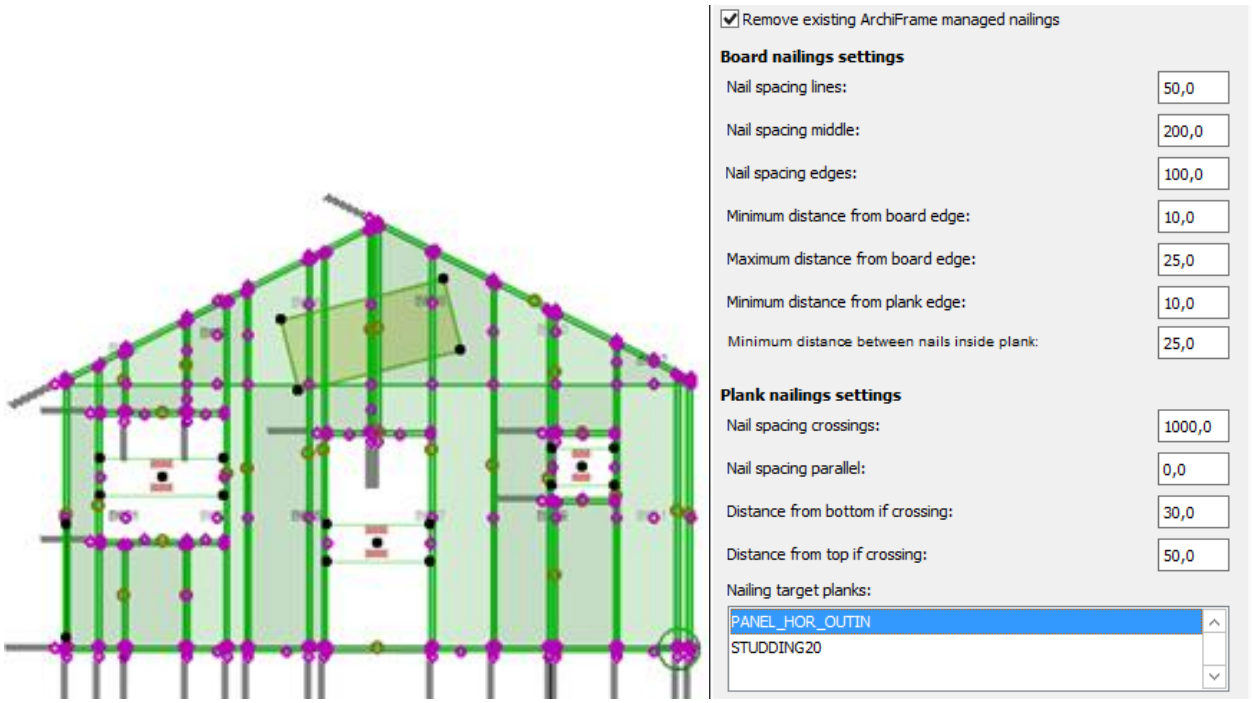


It is also possible to add ArchiFrameNailing-objects over the elevation to produce point nailints in any position. Please note that there must be a nailing target like a batten or stud behind the ArchiFrameNailing-object.

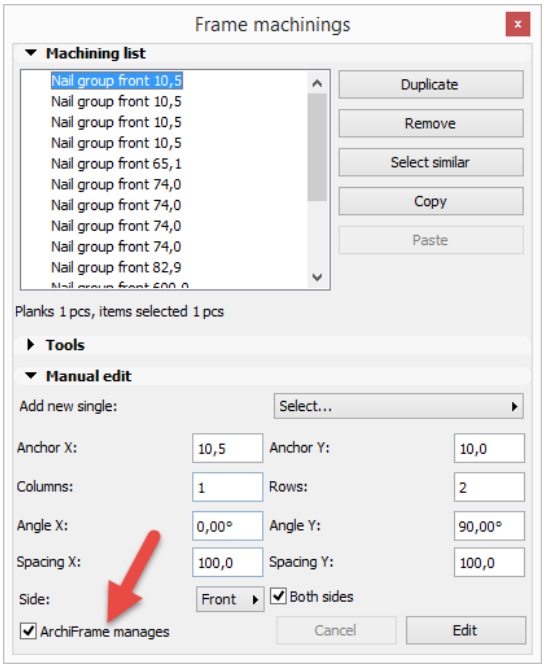
The process to add the nailings is to select all the target boards and related framing planks (usually everything in an elevation). If there are multiple studs next to each other, only the ones to have nailings to the boards should be selected. The selection may contain fills to exclude parts for nailing and lines to set specific type of nailings according to the lines. Also, selection may contain placed ArchiFrameNailing-objects to define individual point nailings. The nail gun number for nailing points can be forced in ArchiFrameNailing-object's settings. Only one layer should be done at a time since added nailings will be avoided when processing other layers if done for all layers at the same time.

No nailings at the same place: Nailings are placed so that minimum distance with the nails in the same element group is this:

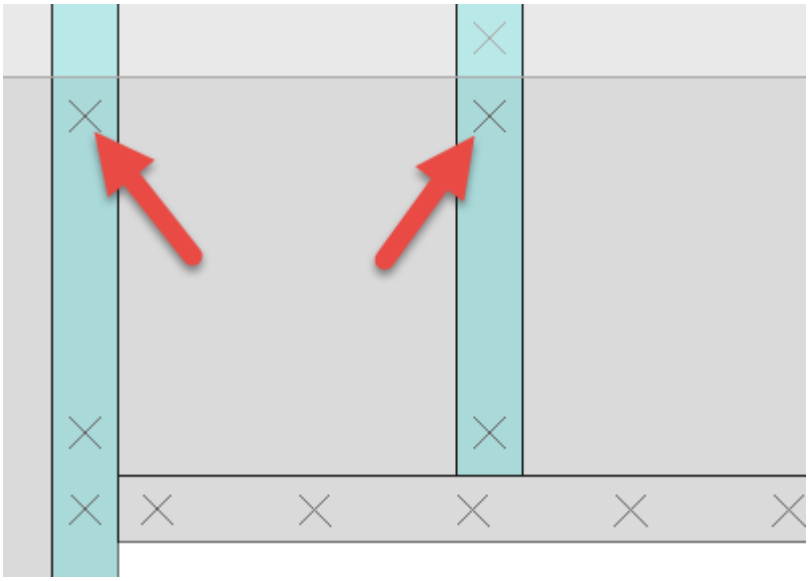
For example like this:



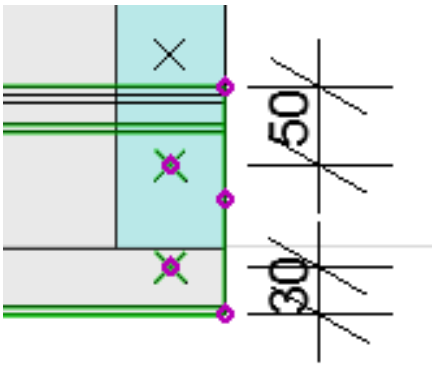
- *New* creates new preset
- *Duplicate* duplicates current settings for further editing.
- *Delete* deletes current item.
- *Load types* deletes all the presets and loads everything from an external file. This is very useful for example if there is a single person in the company taking care of the structures and shares the types to other users.
- *Add types* is as *Load types* except it does not delete current types. It merges the types from the file using any type having the same id from the given file.
- *Save as* saves all presets to an xml-file to be used for *Load types* or *Add types*. The file can be edited using text editor for example to limit types to import to another user.
- *Remove existing ArchiFrame managed nailings*, if checked all existing nailings having ArchiFrame manages on will be removed before adding new ones:



- *Minumum distance from plank edge* defines how close the nailing can be to the plank edge.
- *Minumum distance between nailings inside plank* is used to remove nailings, for example, if there are two edges at the same framing plank closer than this value. Also, this affects avoiding collision to existing nailings.
- *Maximum distance from nail target to framing* tells ArchiFrame to search for suitable framing pieces from selection no further than given value from the nail target surface.
- *Force nail gun number* sets a custom value to be used in CNC-output.
- *Max z-distance when searching existing nailing* defines the distance in nail's direction to limit searching of existing nailings. The measure is taken from nailing target surface. For example, this can be used when first the cladding nails are added to cladding with thickness 18 mm and airspace strips 32 mm (18+32=50 mm). When making nailings for the windshield boards, setting this value to 49 mm let's ArchiFrame to place board nailings ignoring cladding nailings.
- *No nails to first and last position* can be used when the airspace strips are nailed from the ends by CNC operator and the CNC-machine will shoot the remaining nails.
- *Nail spacing lines* defines the spacing used in the lines placed over the elevation. With value 0 no nalings are placed.
- *Nail spacing middle* defines the spacing used at the middle of the boards.
- *Nail spacing edges* defines the spacing used at the board edges including opening edges.
- *Minimum distance from board edge* defines no nailings are placed closer to the board edge.
- *Maximum distance from board edge* if there is a wide framing plank at the edge of the board, this value is used instead of the center line of the framing plank under the board. Also, the distance for nail from board edge if a framing plank is at middle of the board:

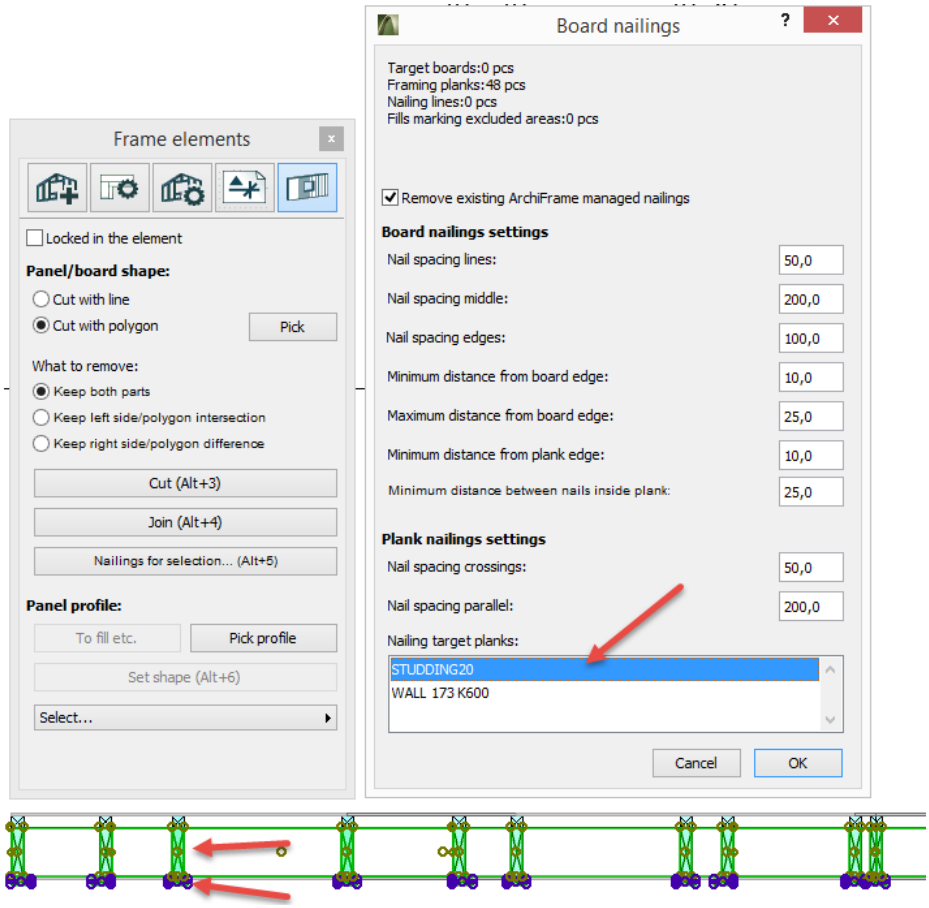


- *Nail spacing crossings* define the spacing used for crossing planks. If target plank is not wide enough to have two or more nailings, just one is placed at the plans' middle line intersection.
- *Nail spacing parallel* is the spacing used when the planks are parallel. With value 0 no nails are placed.
- *Nailing target planks* define the layer(s) to be nailing targets. It is possible to select many layers by pressing *Ctrl*-key when clicking the items.
- *Distance from bottom/top if crossing* is useful for panel/cladding nailings. Values given here will put single nailing given distance from the plank's top/bottom side. Note that single nailings are placed only if spacing crossing is bigger than the height of the piece. In the example the value is 1000 to do that. Value 0 means no special handling for the top or bottom.



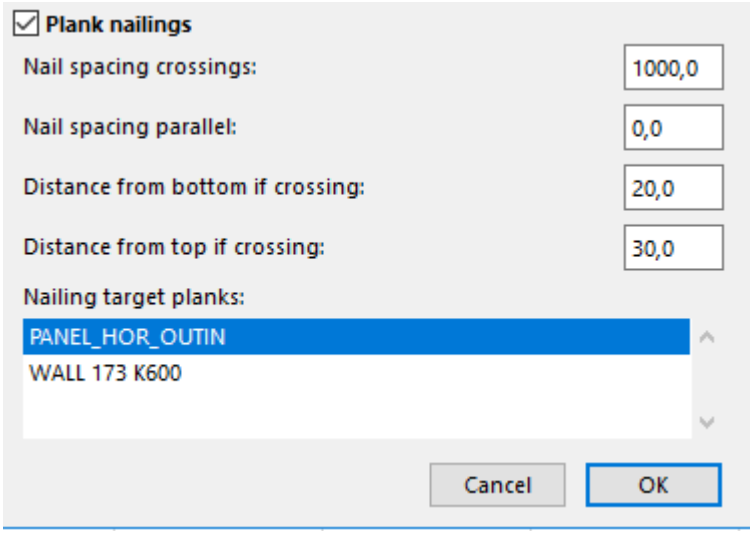
11.5.1 Example nailings for planks

Select two plank layers that should be nailed together:



Then select the nailing target layer.

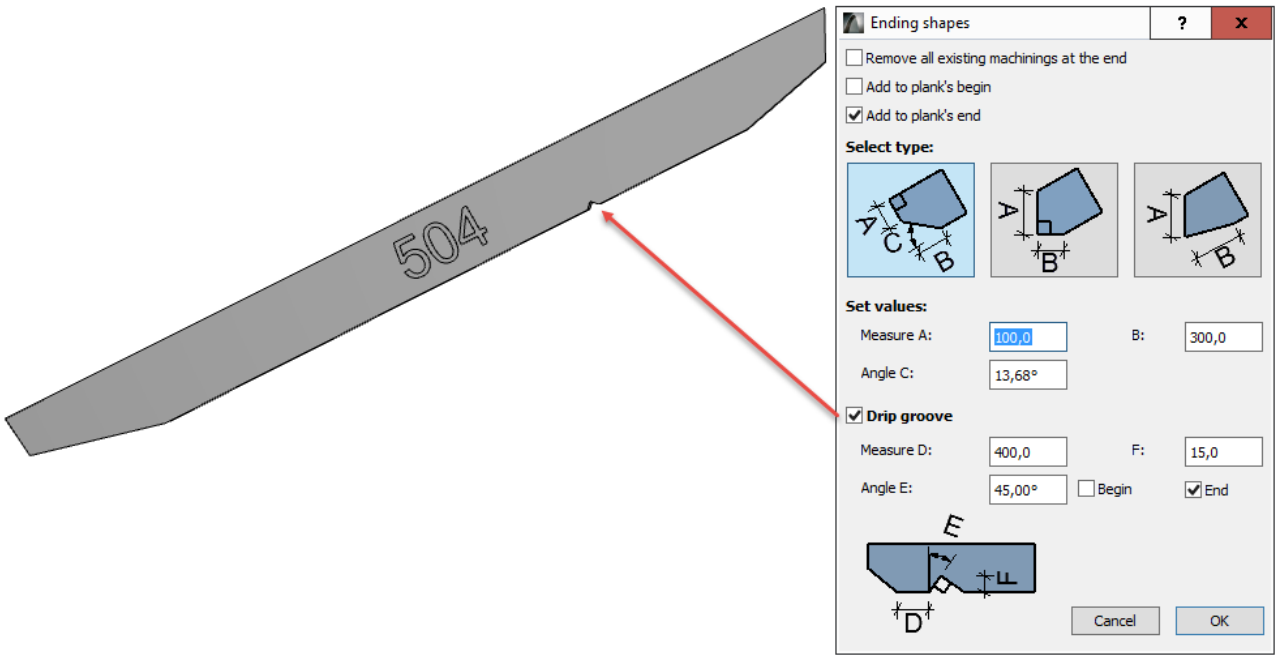
For cladding it is possible to define nailing distances from cladding piece top & bottom. For example, to get nailing 20 mm from bottom and 30 mm from profile top:



Here *Nail spacing crossing* value 1000 mm indicates that we want nails to studding and cladding intersection. 1000 mm is bigger than cladding piece height, so it will not produce any nail. *Distance from bottom/top if crossing* is used instead – settings produce nailing 20 mm from bottom of the cladding and 30 mm from top of the cladding profile.

11.6 Rafter endings

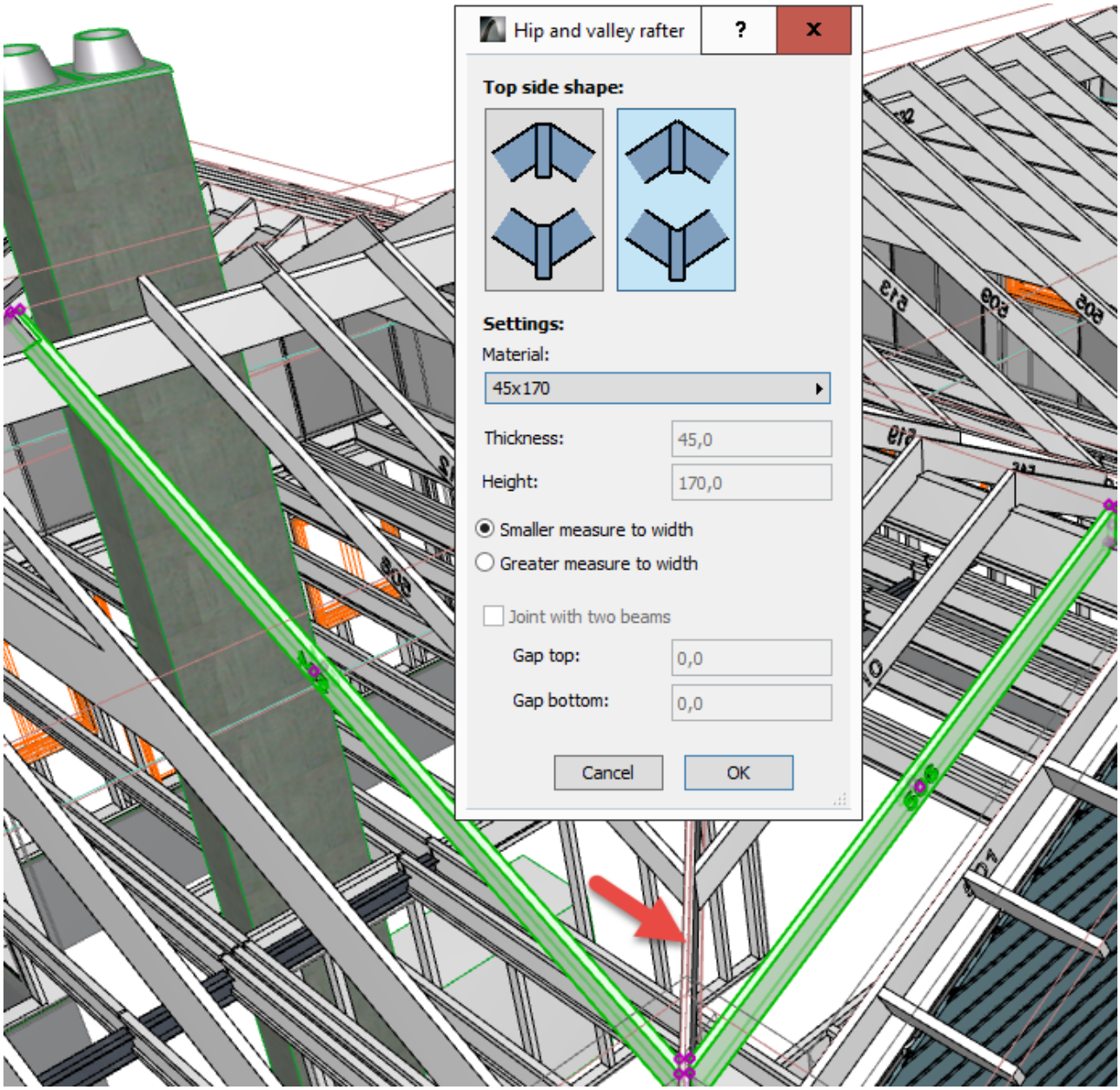
This dialog is used to to make typical endings for selected rafters.



- *Remove all existing machinings at the end* needs to be unchecked if adding different shapes to begin and end. The direction of the destination is shown by the ID text.
- *Add to plank's begin/end* define which end of the plank to set.
- *Drip groove* is handy at upper end of rafter if it is exposed to the weather and the rafter is made with cnc-.

11.7 Hip and valley rafter

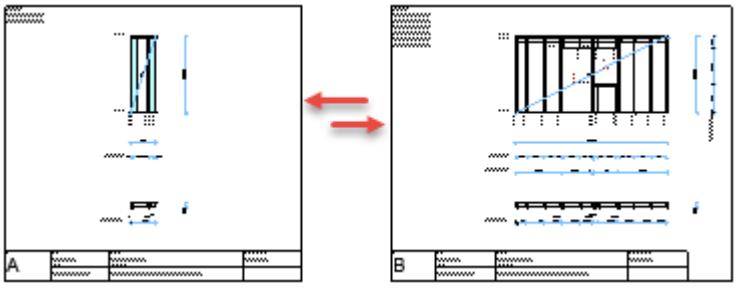
This tool helps placing these special rafters. To use it, select two existing pieces defining the two roof planes. ArchiFrame will calculate the position of the new piece.



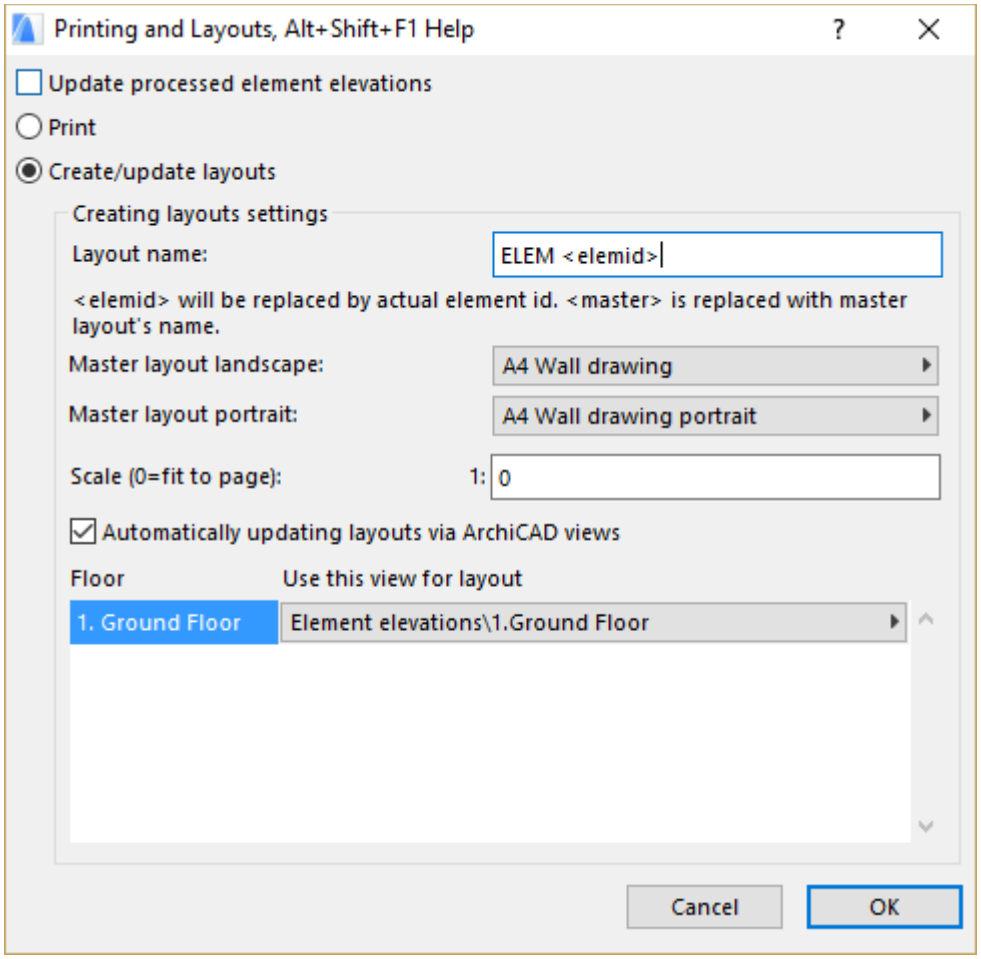
- Top side shape has two options: straight or V-cut. V-cut is useful if the planks are produced with cnc-.

11.8 Printing and layouts

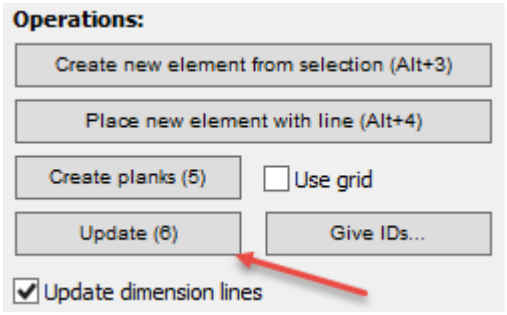
This tool handles areas in current storey marked by *ArchiFramePrintFrame*-objects:



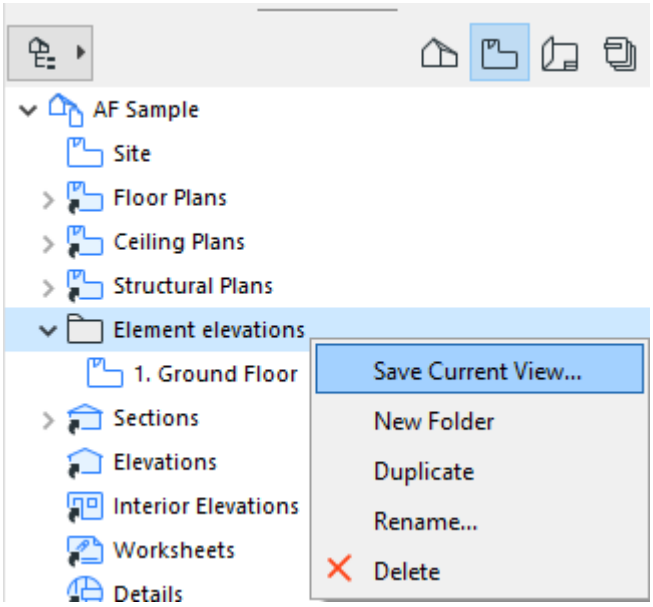
To limit the operation, select the target frame objects.



- *Update processed element elevations* does the same as Add & Edit element tool update:

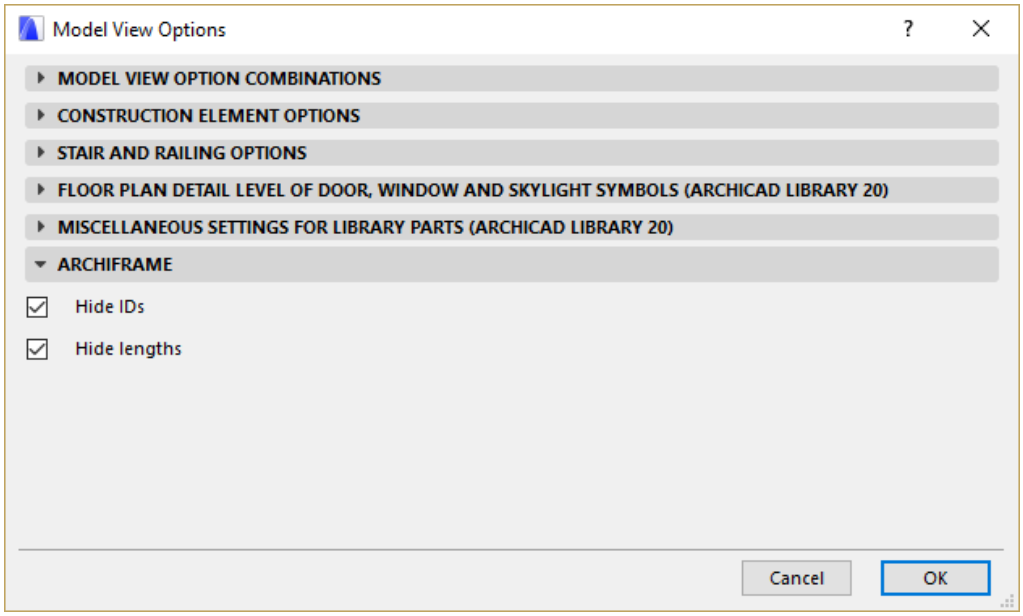


- *Print* prints the target frames directly.
- *Create/update layouts* creates ArchiCAD layouts and is the recommended way to handle printing.
- *Layout name* is the name in ArchiCAD layout book.
- *Automatically updating layouts via ArchiCAD views* will place a view on the layout and limit it to cover just the frame object. Due to technical limitation, there must be a view to do this. To create the views, make sure that at least the scale is correct (1:50) and needed layers visible. One way to organize the views is to create folder Element elevations (right click to the top-level item – AF sample in this example and select *New folder*). Then right click on Element elevation folder for each floor to save the views:



11.8.1 ArchiFrame Model View options

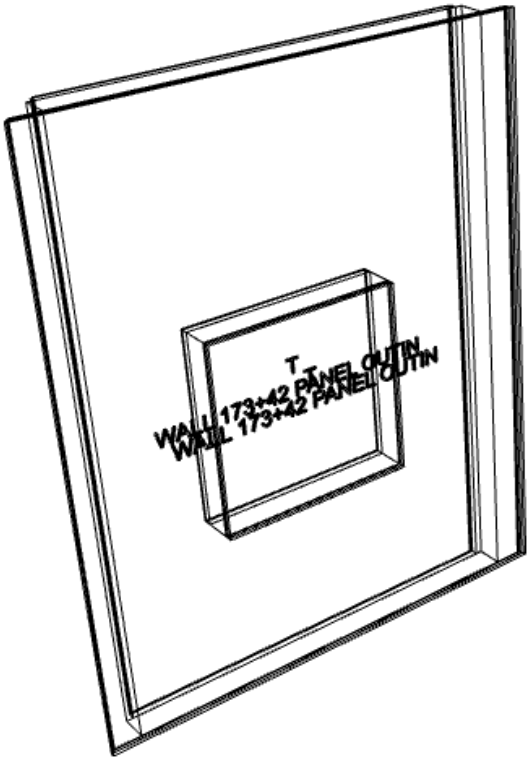
ArchiFrame has its own settings here (*Document-menu/Model View/ Model View Options*):



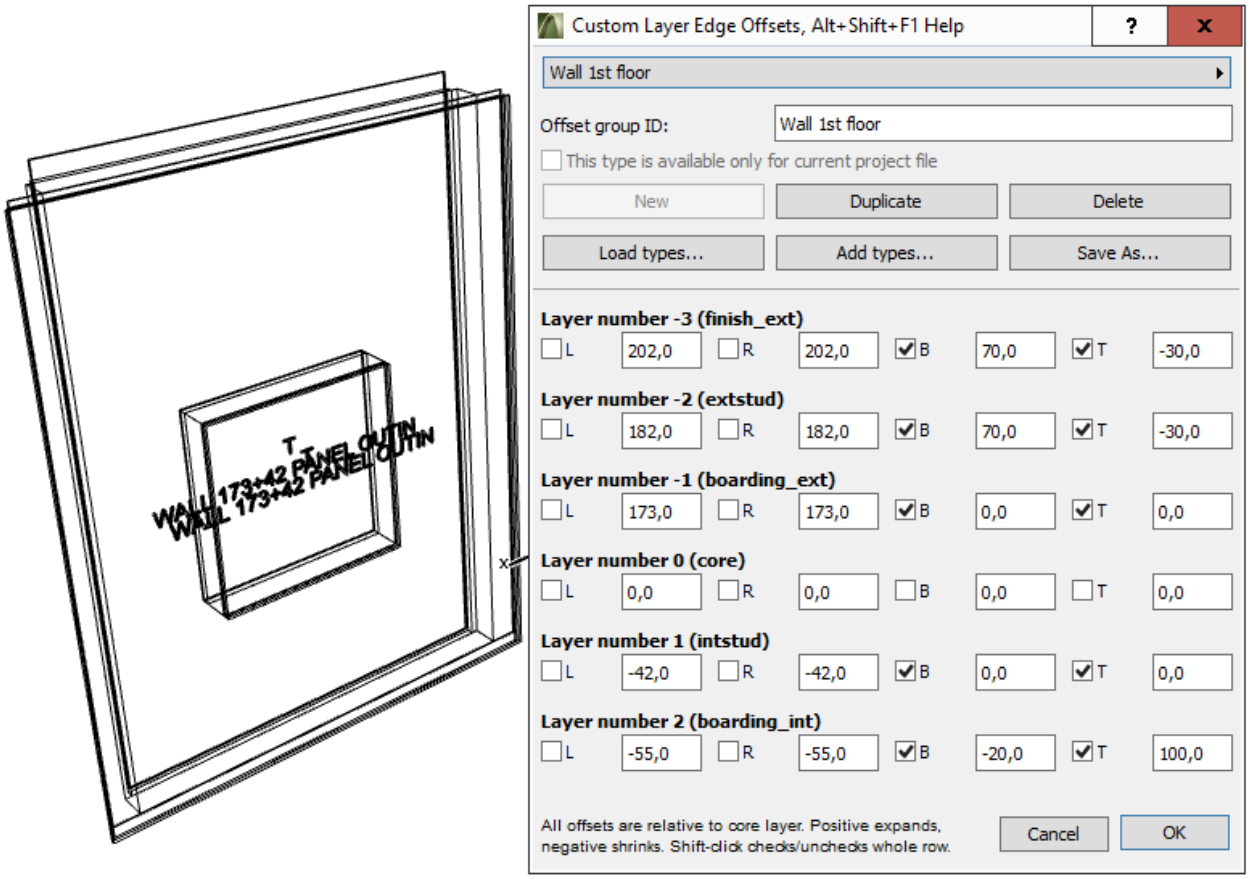
These options can be used for special cases. These options are saved and applied by ArchiCAD views.

11.9 Custom layer edge/opening offsets

This tool defines distances from other layer edges to the main core layer. For example, an element having corners already set with the Element corner tools but top&bottom offsets zero:

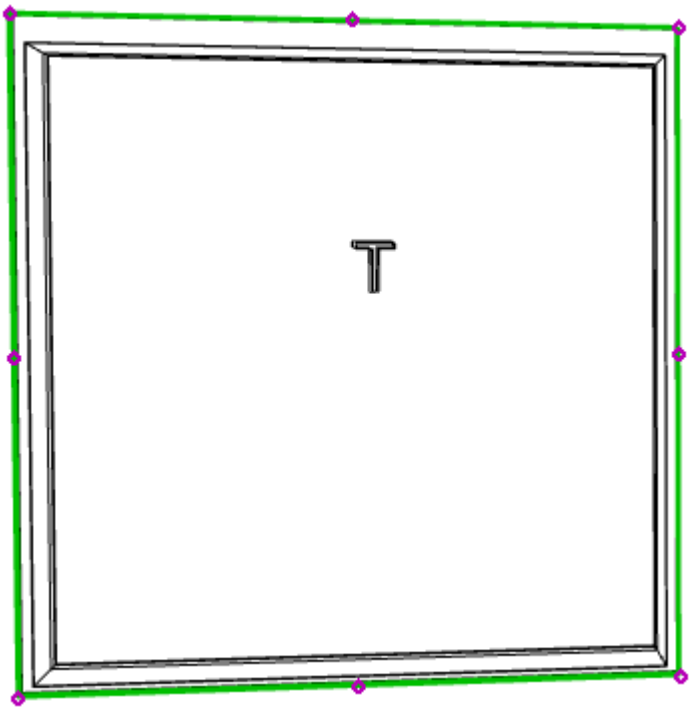


Extend bottom 70 mm and lower top by 30 mm for exterior finishing. And lift bottom by 20 mm and top by 100 mm for the inside gypsum:



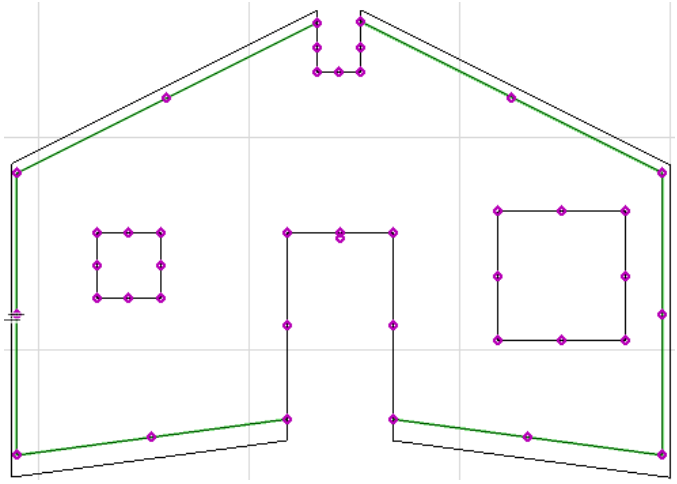
- Layer offset preset list contains all settings that have been given an ID.
- *Offset group ID* is the name for preset-list.
- Check box *This type is available only for current project file* is currently not available.
- *New* is currently not available.
- *Duplicate* duplicates current settings for further editing.
- *Delete* deletes current preset.
- *Load types* deletes all current presets and loads everything from external file. This is very useful for example if there is a single person in the company taking care of the presets and shares the types to other users.
- *Add types* is as *Load types* except it does not delete current list. It merges the types from the file using any type having the same id from the given file.
- *Save as* saves all presets to an xml-file to be used for *Load types* or *Add types*. The file can be edited using text editor for example to limit types to import to another user.
- Check boxes *L, R, B, T* and related length edit defines offsets for the left/right/bottom/top edges. Positive value extends the layer or the opening, negative makes it smaller. Only checked items will be applied to the element. For example, when defining top and bottom offsets, only T and B fields should be checked to let the corners be as they are.

Offsets for openings can be used for example to make the opening larger in the exterior finishing and air space layer:



ArchiFrame builds the layer list from the selected element or from the selected existing preset. ArchiFrame will use the definition to similar layers if applied to different element types. For example, definition for layer boarding_ext will be applied to boarding_ext1, boarding_ext2 etc if there are many boarding layers.

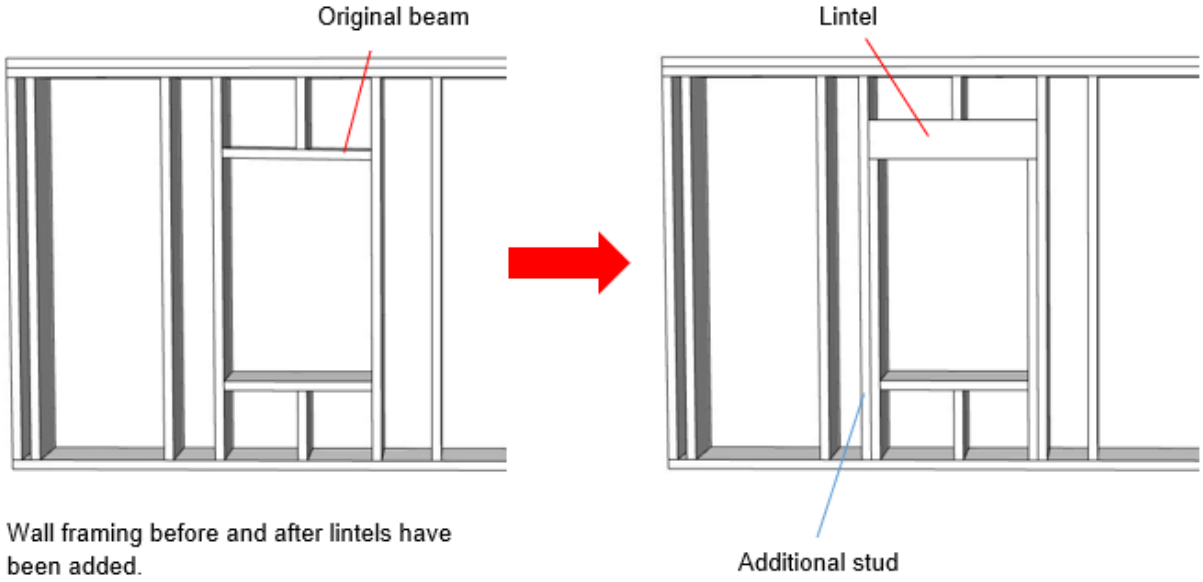
ArchiFrame will handle angled endings like below:



Here the rules are:

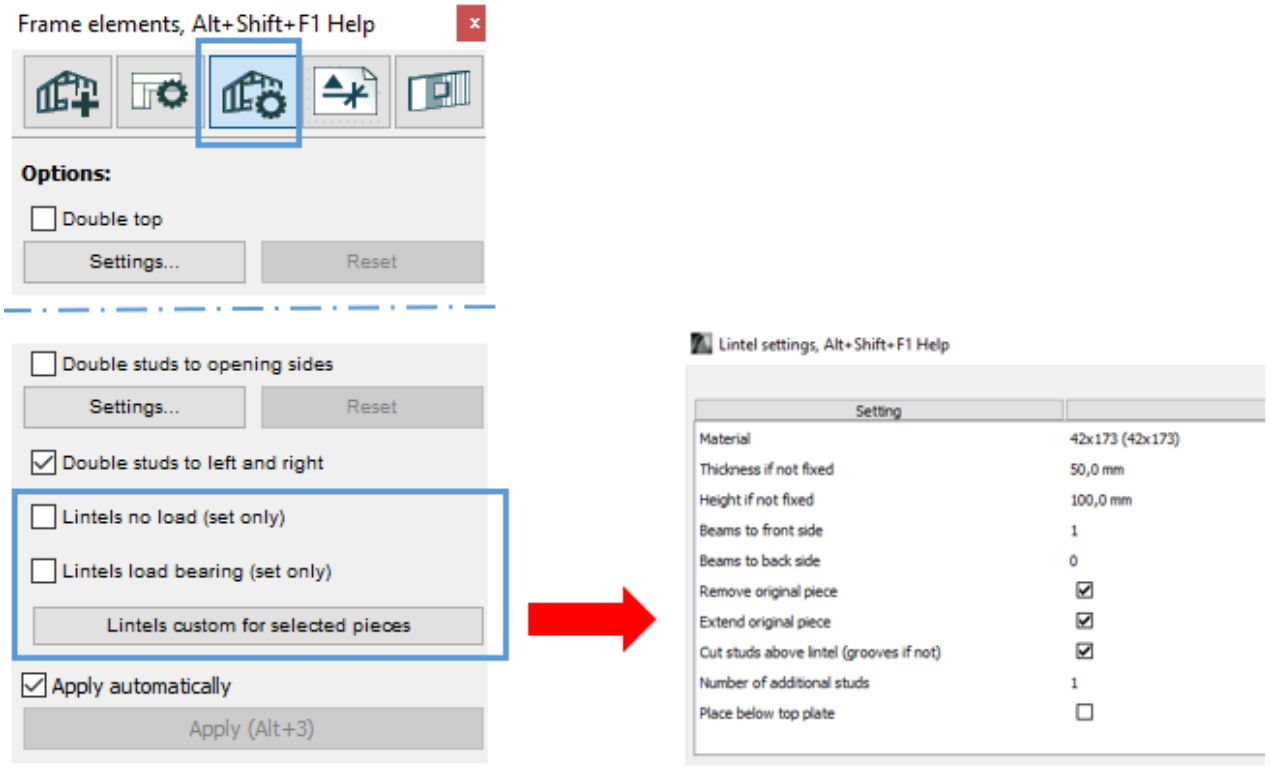
- Only top & bottom edges can have angled parts.
- Offset value is perpendicular to the line in question.
- Edge shape is taken from core layer and offset values are applied to that shape.
- ArchiFrame will find first and last part for the edge scanning for pieces that have angle -89…89 degrees to the edges normal (that is a vector pointing outside the edge).
- As a special case ArhiFrame will not apply edge offsets to three succeeding lines looking like door opening (like in the image above).
- Automation may fail in some cases. Then the element shape must be edited by hand using *To fill* and *Pick from fill* buttons or by other means.

11.10 Lintels



Wall framing before and after lintels have been added.

ArchiFrame has a special tool for adding lintels to wall openings. Lintels can be added either directly above the opening, or below the wall's top plate. Studding on either side of the opening can also be edited with this tool (Fig. 10.9.1). The tool can be found from the Set Element Options window:



11.10.1 How to create a lintel

Lintel settings can be set up for a wall element before planks are created:

- Select an element.
- Set the appropriate lintel settings.
- Create planks.

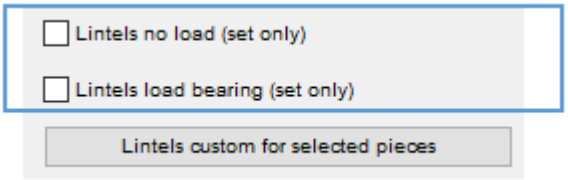
Lintels can also be added after planks have been created:

- Select the plank(s) above the opening you wish to add a lintel.
- Adjust the lintel settings.

11.10.2 Applying lintel settings from the XML file

Lintel settings are stored in the XML file in your data folder. To apply lintel settings from the file:

- Select a wall element or the piece above the opening.
- Check one of the boxes:



- Lintels will be added after planks are created.

Using settings from the XML file is useful if you are applying the same lintel settings to many different lintels, since this way the settings don't have to be set separately for each one.

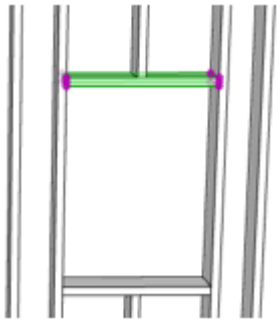
If you need to create a custom lintel type in the XML file, please contact us at ArchiFrame for help.

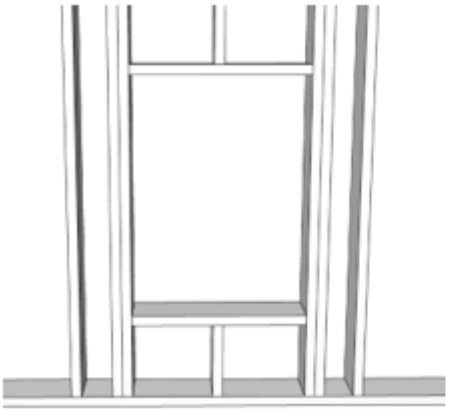
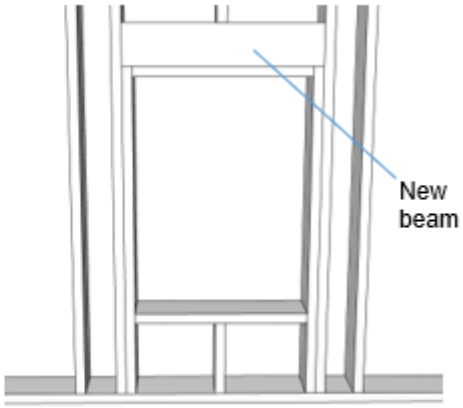
11.10.3 Custom lintel settings

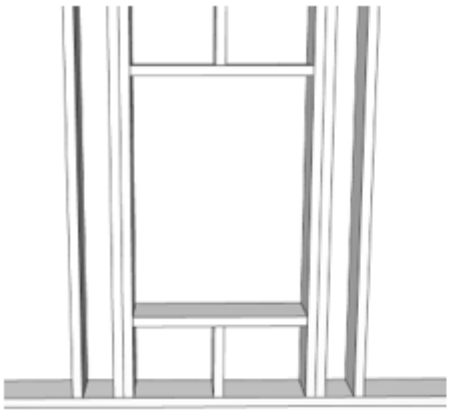
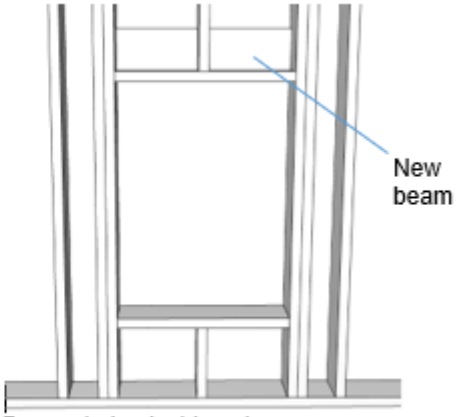
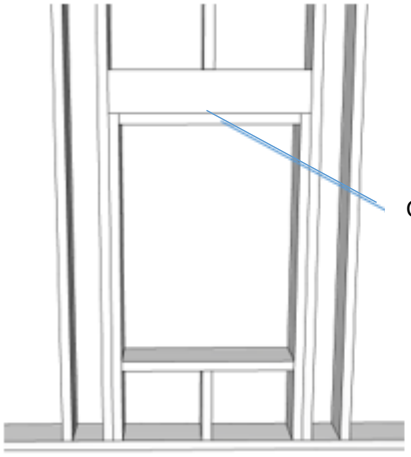
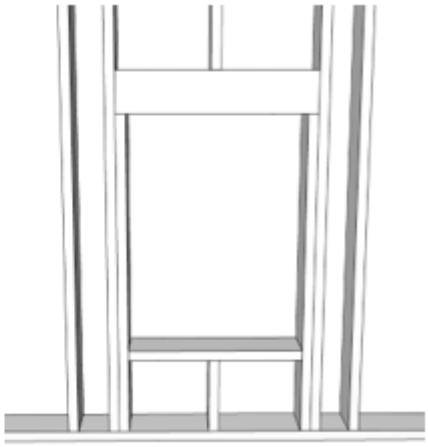
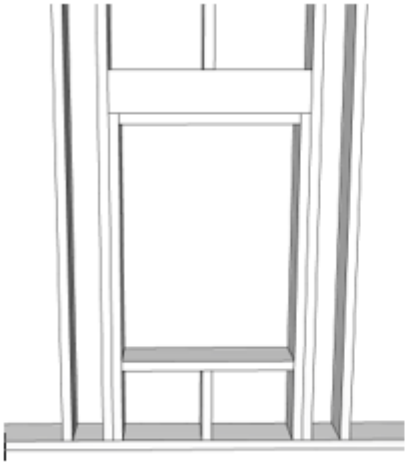
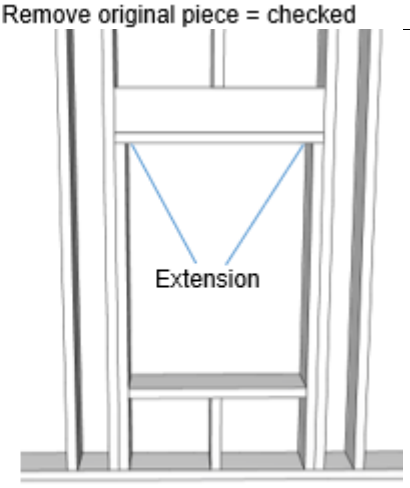
With the custom lintel settings, you can define a custom lintel type for an individual opening. To do so, first select the piece(s) above the opening (see Fig. 10.9.4), and then change the settings. Settings will be applied after you click OK.

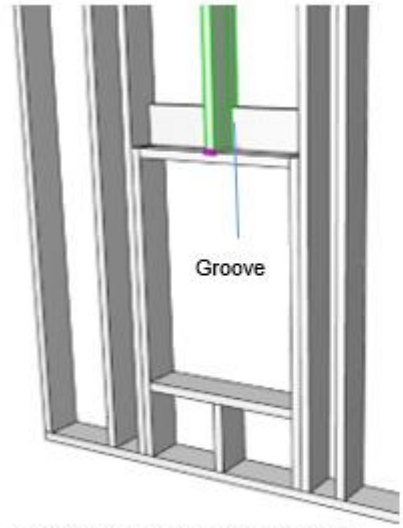
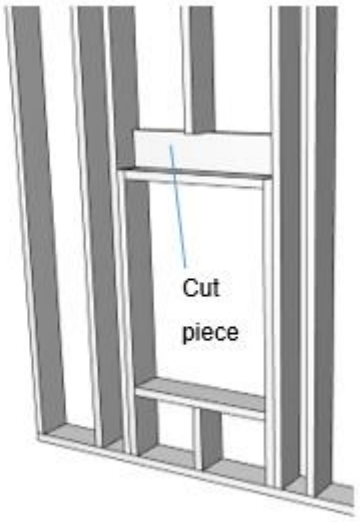
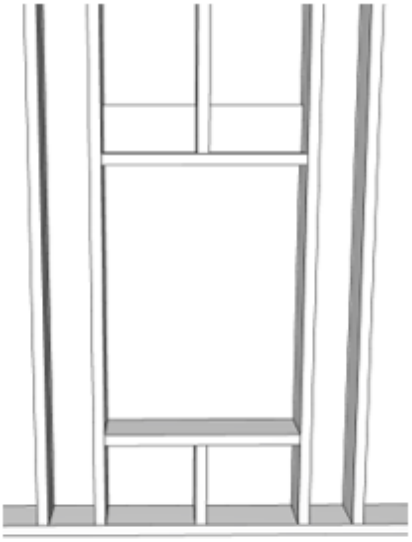
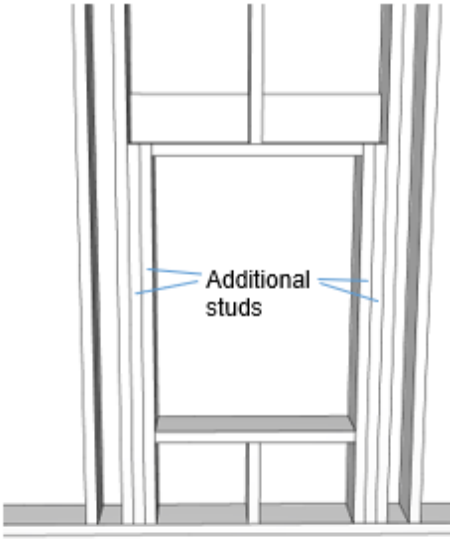
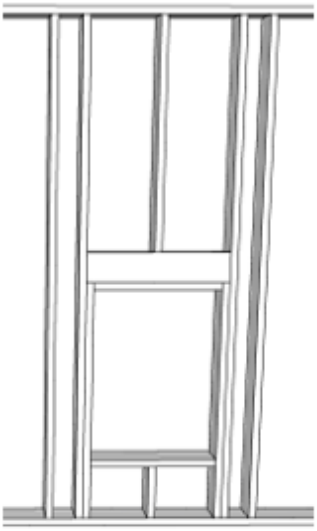
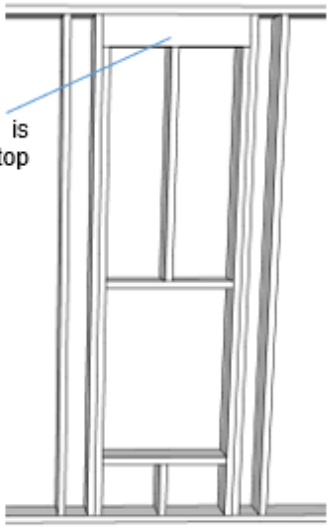
The effects of these settings are explained in the table below. All the figures in the table show the result after lintels have been created. The starting point is shown in figure 10.9.4.

Framing before lintels have been created. The original piece above the opening is selected.

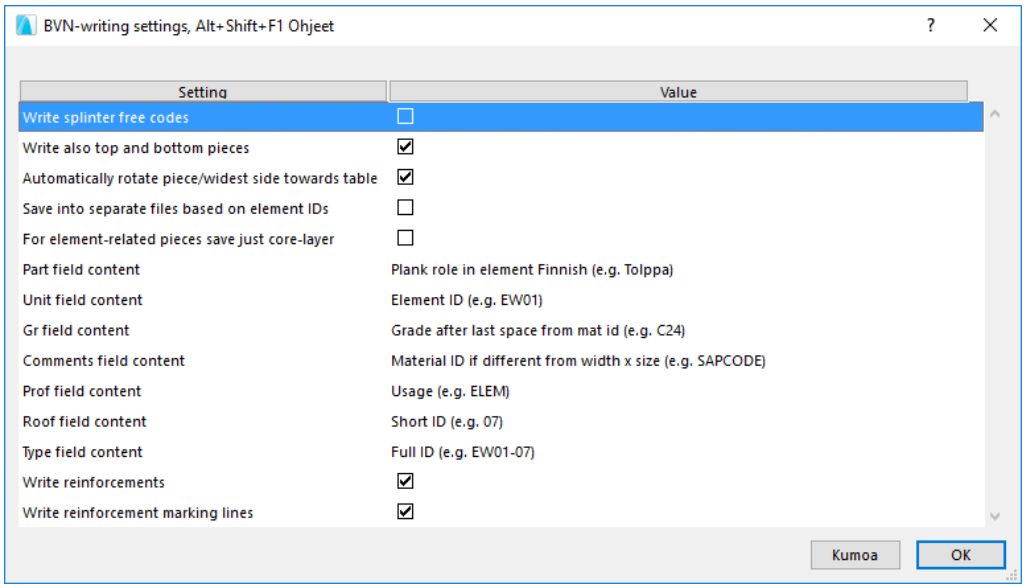


Setting	Explanation
Material	Determines the material of the new lintel piece.
Thickness	If the lintel's material was set to "block", with undefined dimensions, you can set the thickness here. • Benefit: you can create a custom lintel without defining a new material if the lintel plank is not found in the material list.
Height	See explanation for thickness above.
Beams to front side	Places the lintel on the front side of the element. In these examples, the front side faces the camera:  Beams to front side = 0  Beams to front side = 1

Beams to back side	Places the lintel on the back side of the element. In these examples, the front side faces the camera:  Beams to back side = 0  Beams to back side = 1
Remove original piece	 Remove original piece = not checked  Remove original piece = checked
Extend original piece	 Extend original piece = not checked  Extend original piece = checked

Cut studs above lintel (grooves if not)	 Cut studs to lintel = not checked  Cut studs to lintel = checked
Number of additional studs	 Number of additional studs = 0  Number of additional studs = 2
Place below top plate	 Place below top plate = not checked  Place below top plate = checked

11.11 Hundegger bvn-saving options

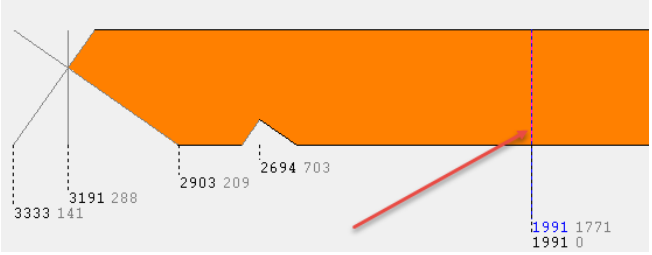


- *Write spliter free codes* defines if for example for a slot saw cuts sides of the slots before cutter to avoid splintering. Hundegger defines internally how the machining is done.
- *Write also top and bottom pieces*, if unchecked top&bottom plates are not included in the cnc-file.
- *Automatically rotate piece/widest side towards table* means that for example part 45x180 will be placed on the table the 180 mm wide surface down.
- *Save into separate files based on element IDs* will cause ArchiFrame to add element ID to given save as file name and put only pieces from single element to single file.
- *For element related pieces save just core layer* is used to skip for example air space strips from the resulting cnc-file.
- Rest of the settings define content to put into part related information fields:

Nb	Part	Req.	Cut	Width	Height	Length	Unit.	Gr	Comments	Prof.	Roof.	Type
1	Stud	1	0	36	198	2440	YV6c	C24	36x198 C24		5	YV6c-05 3

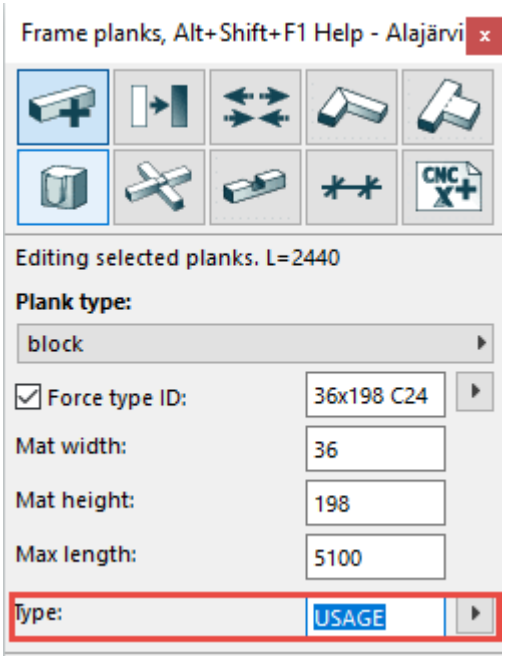
- *Write reinforcements* will write the pieces below into separate file with postfix _reinforce and it will add REINFORCED to comment field in bvn if the piece has reinforcements.

- *Write reinforcement marking lines* marks ends of reinforcements into the bvn-file:



The possible piece of information to select for each field are:

- The example piece is part of element EW01, its full ID is EW01-07, its type is 48x198 C24 and it is a stud. Usage is defined here:

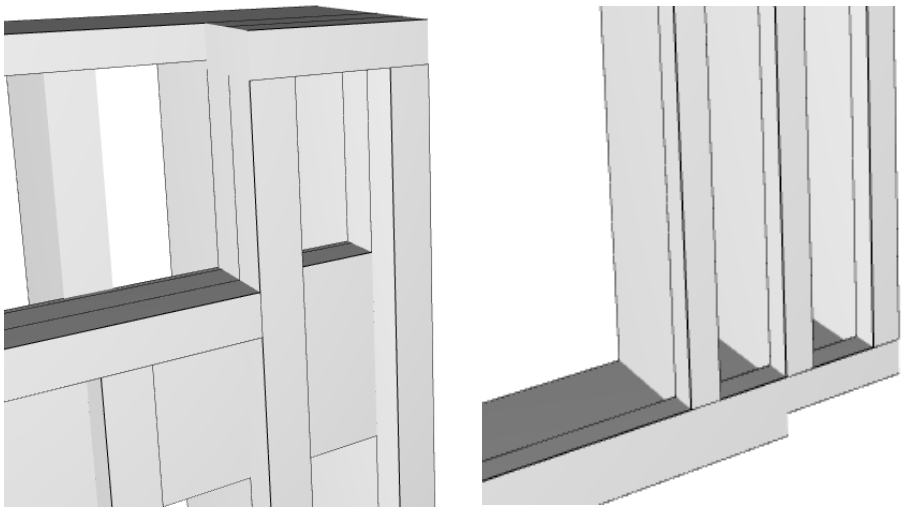


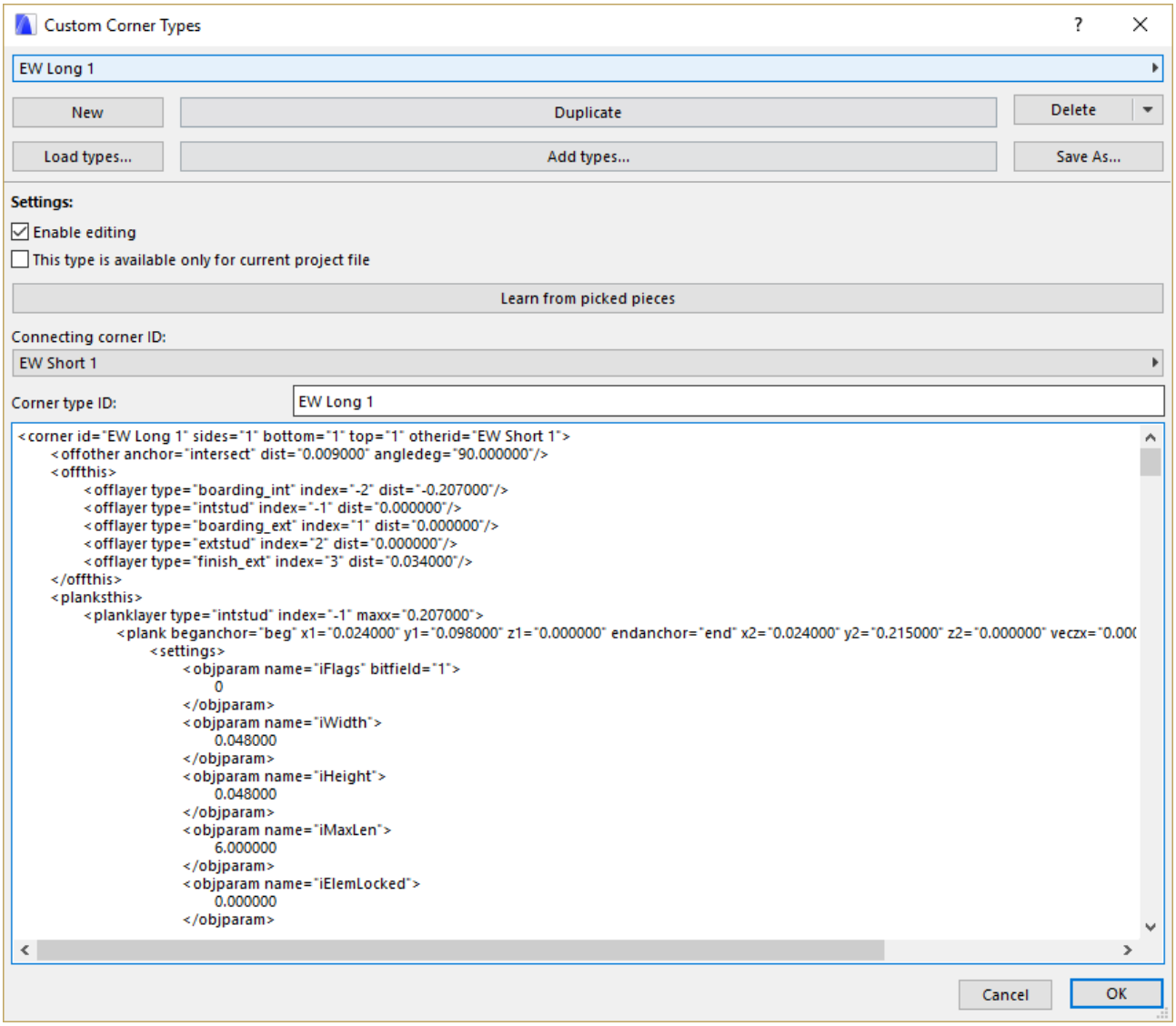
- Full ID plus usage: EW01-07 (USAGE).
- Short ID: 07.
- Full ID: EW01-07.
- Element ID: EW01.
- Usage: USAGE.
- Full mat id: 48x198 C24.
- Short mat id cut to last space: 48x198.
- Grade after last space from mat id: C24.
- Material ID if different from width x size: 48x198 C24.
- Plank role in element: Stud.
- Plank role in element Finnish: Tolppa.
- Plank role in element Norwegian: Stender.

11.12 Custom corner types

Video: https://vimeo.com/277423174
https://player.vimeo.com/video/277423174

Ability to teach corner types allows very complicated connections. For example, like below:





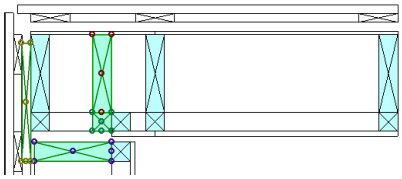
Dialog parts are:

- Corner type dropdown list, this list contains all defined types.
- *New* creates new empty corner type definition.

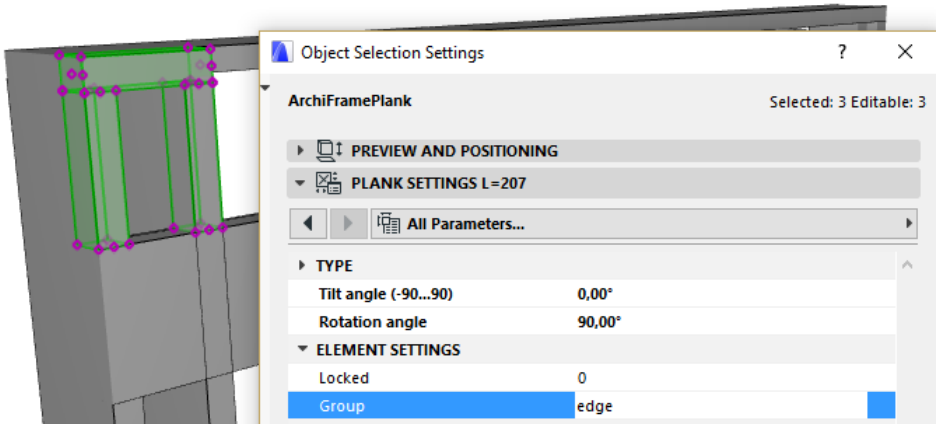
- *Duplicate* duplicates current type for further editing.
- *Delete* deletes current type. Be very careful not to delete any structure that is in use. The custom corner types are shared among all projects as default. So, delete a type only if you are absolutely sure that it is not needed any more.
- *Load types* deletes all custom corner types and loads everything from external file. This is very useful for example if there is a single person in the company taking care of the structures and shares the types to other users.
- *Add types* is as *Load types* except it does not delete current types. It merges the types from the file using any type having the same id from the given file.
- *Save as* saves all custom types to an xml-file to be used for *Load types* or *Add types*. The file can be edited using text editor for example to limit types to import to another user.
- *Enable editing* must be checked on before making any changes. Its purpose is only to avoid editing existing types accidentaly.
- Check box *This type is available only for current project file* defines that the type is available only in current ArchiCAD project file (pln-file).
- *Corner type ID* is the unique ID for the custom corner type. The ID should never be changed if the corner type is in use. It is advisable to design the ID types carefully before actually adding elements.

Steps to teach a corner:

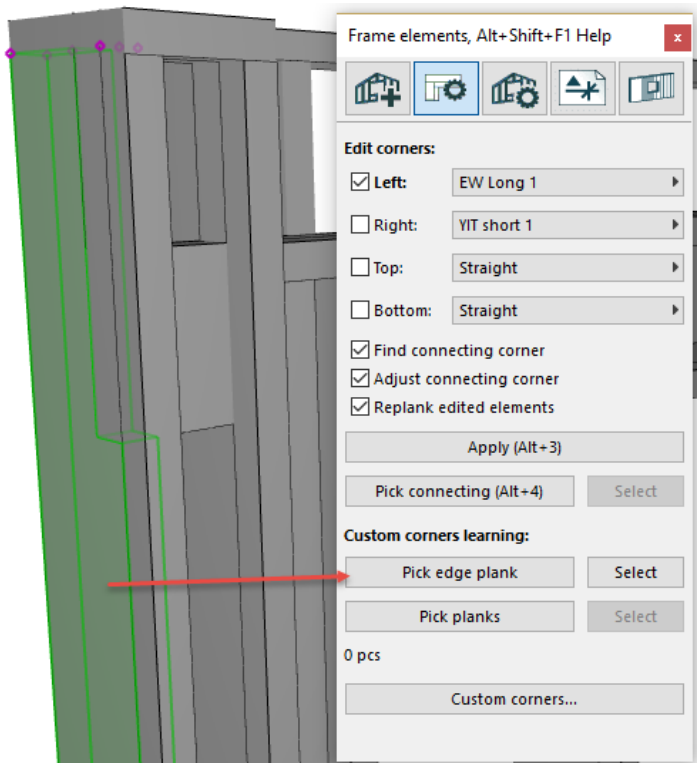
1. Model the corner as it should be (we will first teach the long/horizontal wall). *Important! ArchiFrameElement-objects must be first adjusted to have correct layer offsets – moving just the planks will not work.*



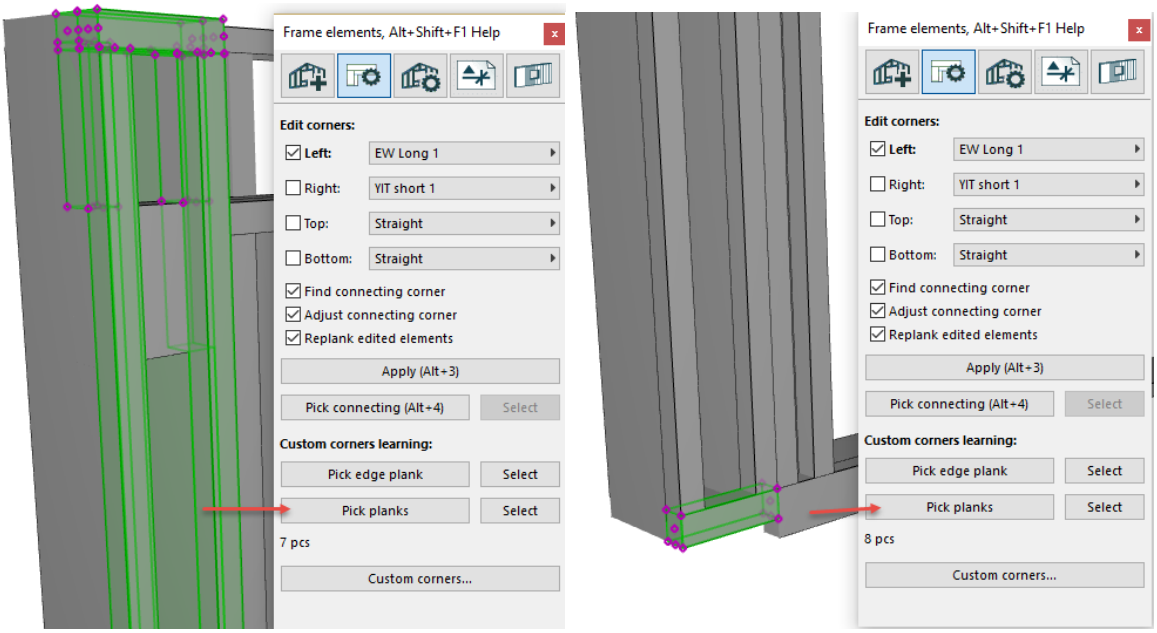
2. Change the element group of special pieces to value edge. If these were original (top_force, vertical_force and vertical_y) ArchiFrame would trim the pieces away when applying the framing rules. Normal studs should not be changed to allow for example *Marking grooves* and *Double top* options to work:



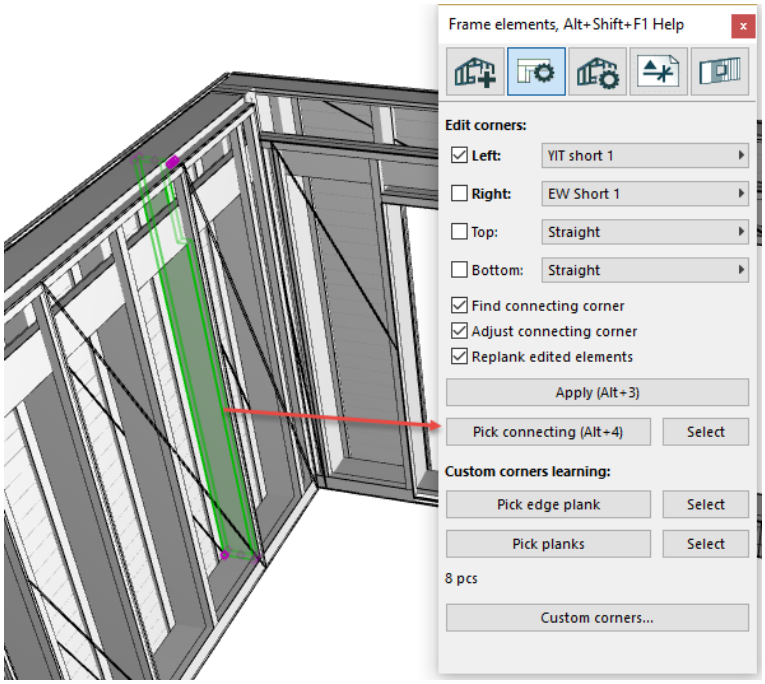
3. Pick the edge plank that tells ArchiFrame which edge to learn from:



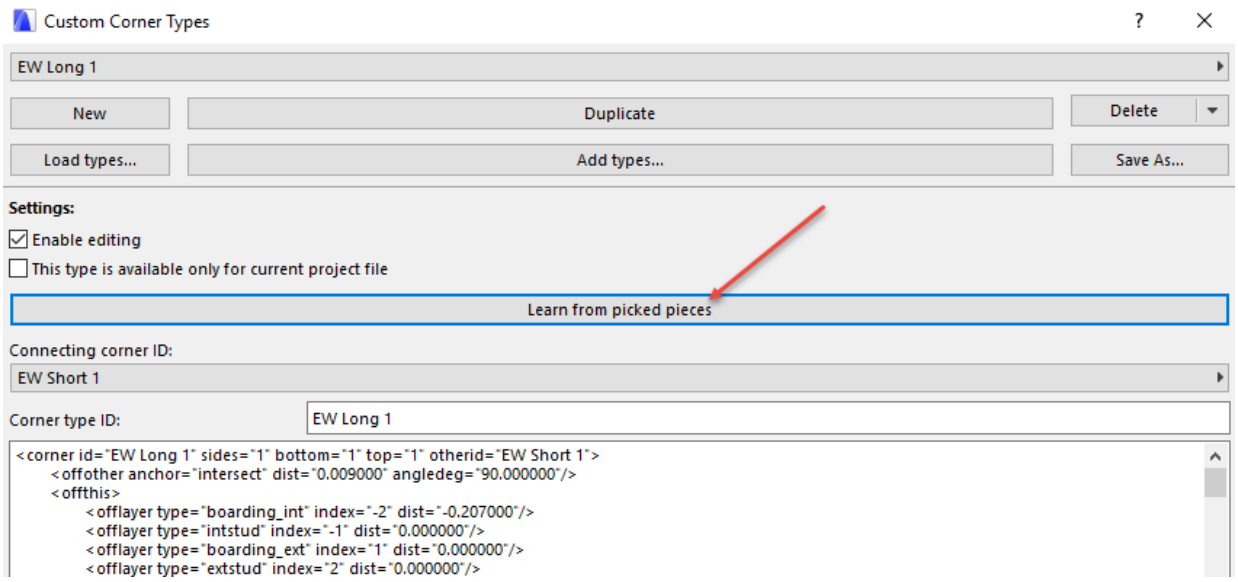
4. Select planks to learn, *Pick planks* is used twice – first at top and then at bottom of the wall:



5. Pick the connecting element (having any plank/board relateted to the connecting element is ok):



6. Open *Custom corners* dialog and click New to make new corner type (or select old one to edit its definitions), then click *Learn from picked pieces*:



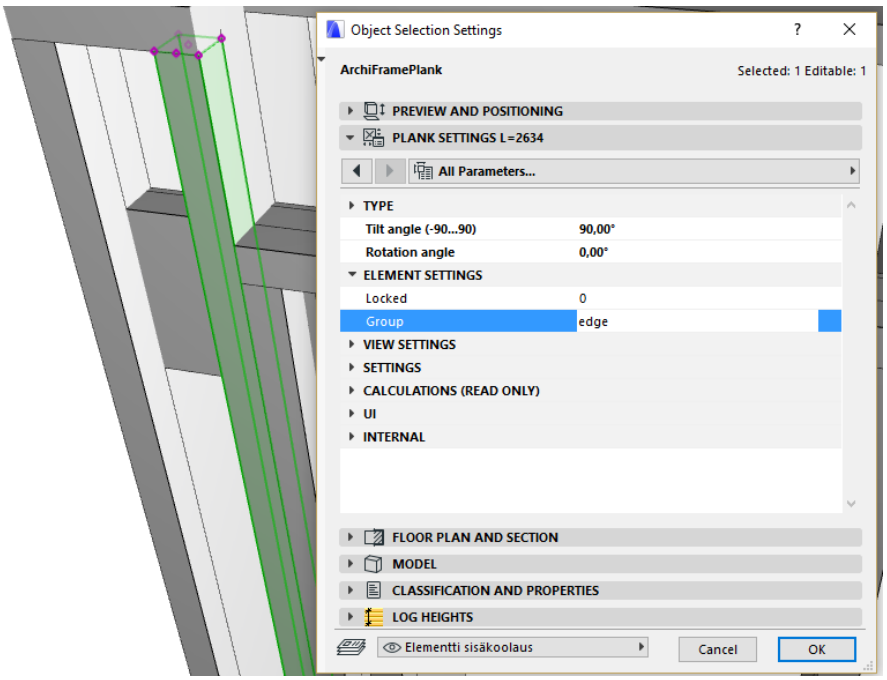
In special cases it is possible to edit the resulting xml-definition. ArchiFrame will save this information about the corner:

- Distance to connecting element's core-layer and angle (0…90 deg) between edge normal vector and connecting element's X-axis. This angle is used when finding related *ArchiFrameElement*-objects – only parts with matching angle are picked.
- Current element's layer offsets (for example, how much far the cladding extends from core layer).
- For each layer the handled area where other framing rules may not add planks. Attribute maxx="value" defines the distance from the edge that marks no framing area. If there was a plank to be created that intersects that area, the plank will not be created. For example, the definition below sets the area to be 485 mm from the edge of the element layer:

<planklayer type="core" index="0" **maxx="0.485000"**>
If the edge rule is applied to top/bottom of the element, this rule will prevent creating spacing-rule studs at all so you will need to edit the value to:
<planklayer type="core" index="0" **maxx="0"**>.

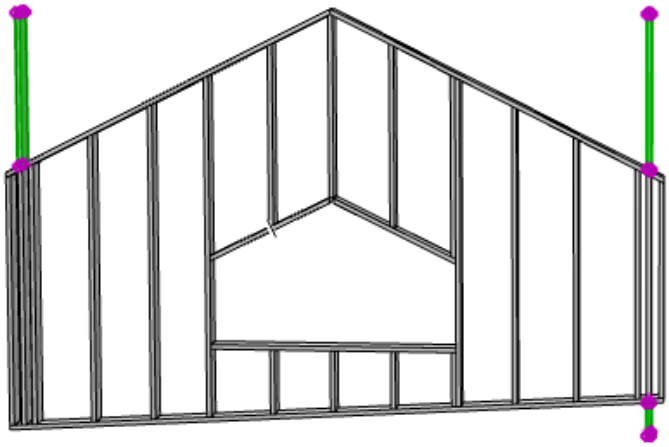
- For planks following information is saved:

    - Begin and end position relative to source edge's top and bottom coordinates.
    - Placement relative to source edge's geometry.
    - Numerous parameters.
    - The element role is set to edge unless it contained force originally:



This group setting affects how planks are handled in element options tool.

- Does the learnt piece intersect with its owning parent element? This information is used to automatically clean up pieces outside the element in cases like a gable wall.

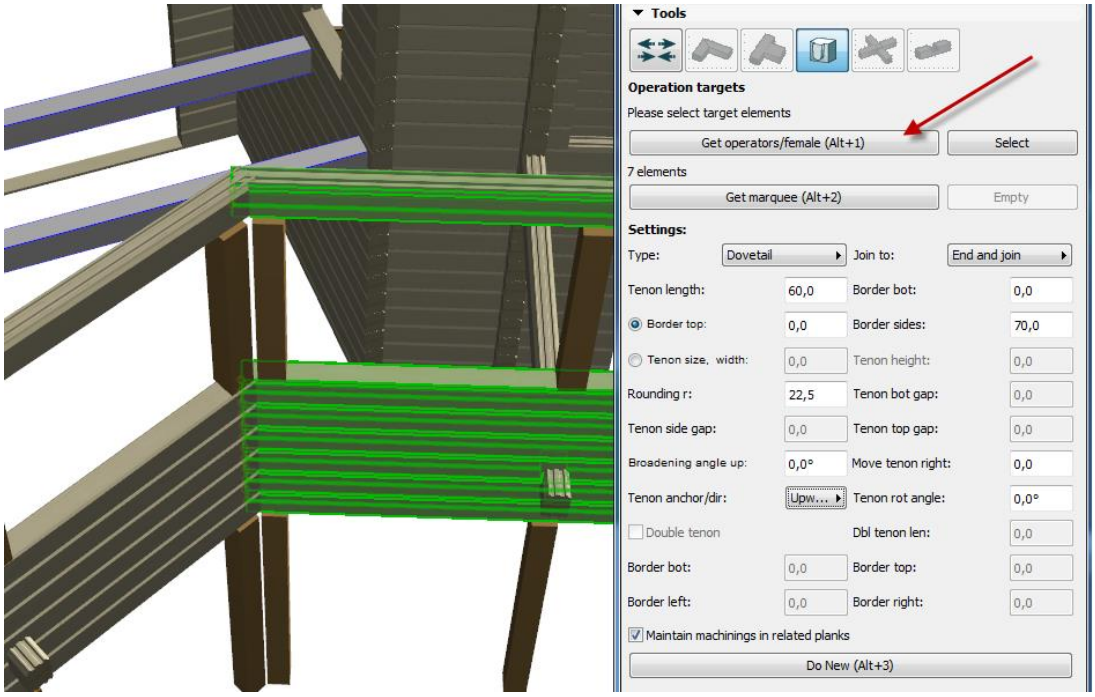


This is defined by attribute xelem = "0" (part does not intersect the element – don't cleanup) or xelem = "1" (part intersects with element, delete all parts in target that are outside the element).

12 Examples of using log structures

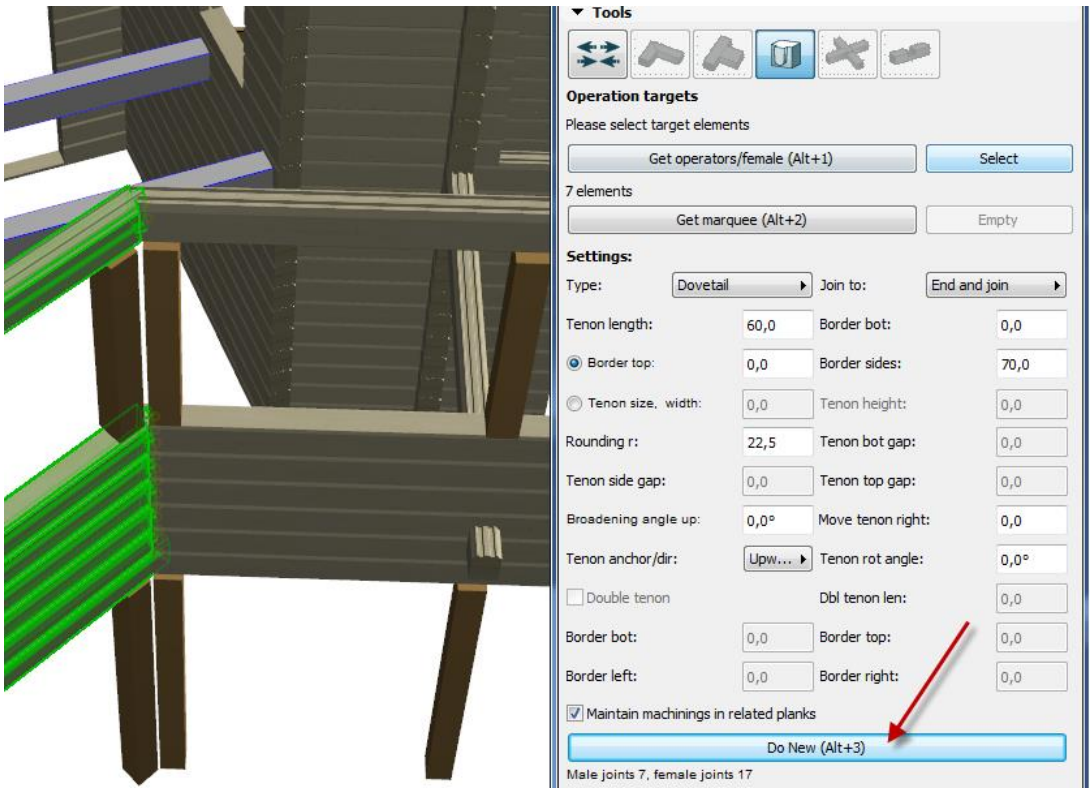
12.1 Joining beams or logs from ends with continuous dovetail

Select pieces to have DT-mortises:



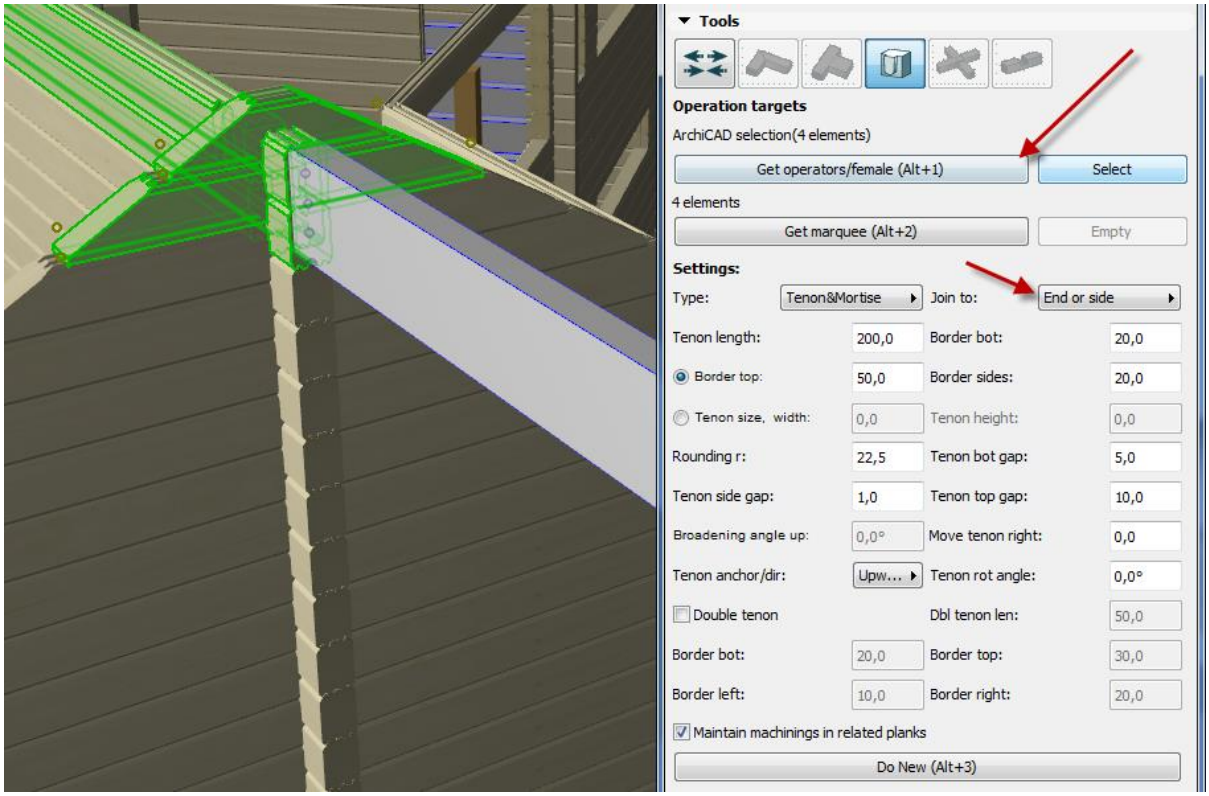
Adjust settings like above (zero at border bottom and top and at broadening angle up will produce wanted continuous dt-joint).

Then select male parts and create the joint (please notice that *Join to* be set to *End and join* – it will halve the angle between planks automatically):

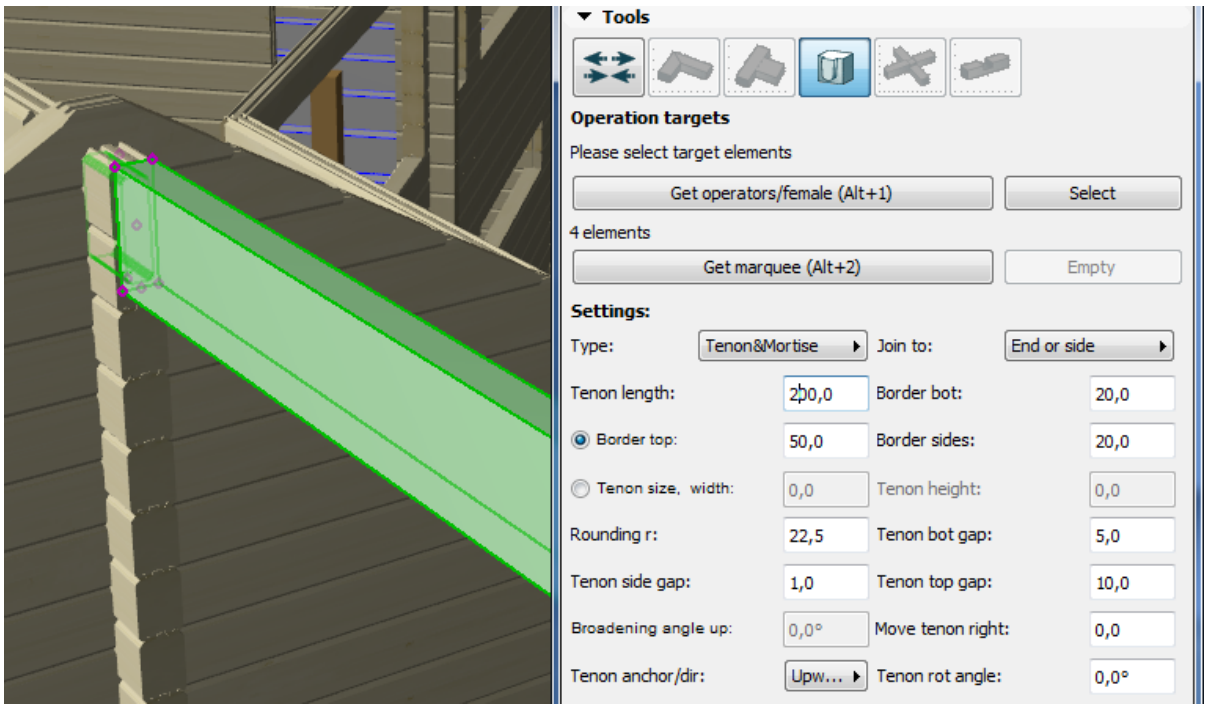


12.2 Join balk to the end of logs with tenon & mortise

Select the female parts and adjust settings (please note Join to-setting is *End or side*):



Select the male part and create the joint:

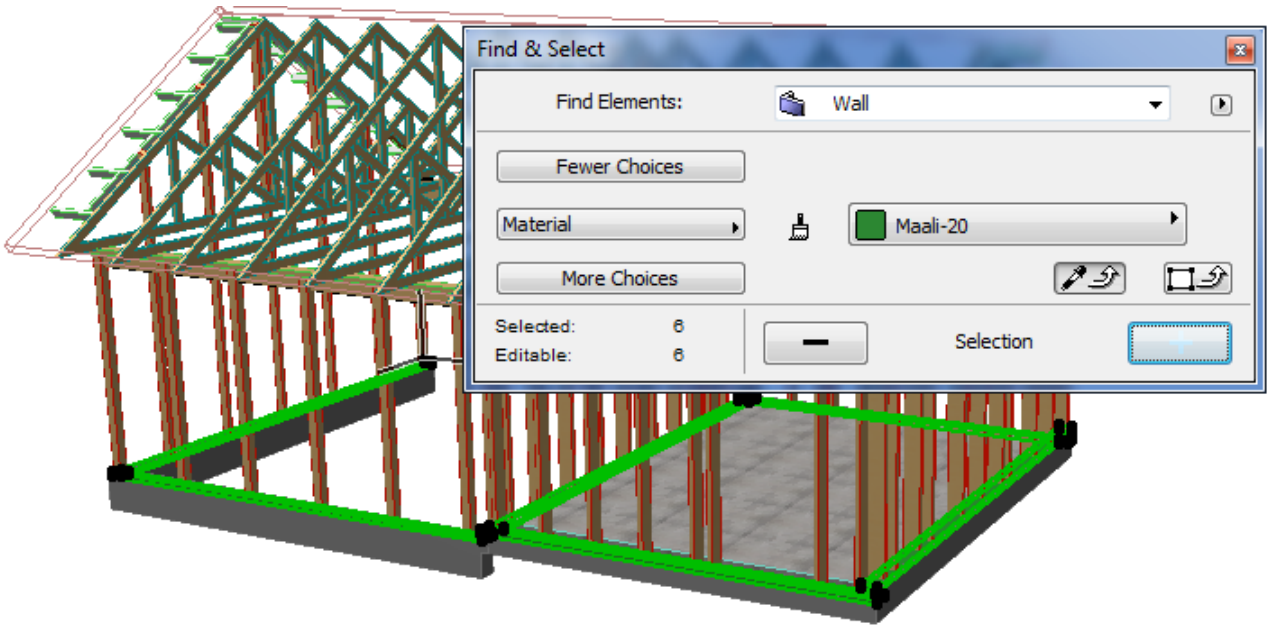


13 Examples of using frame structures

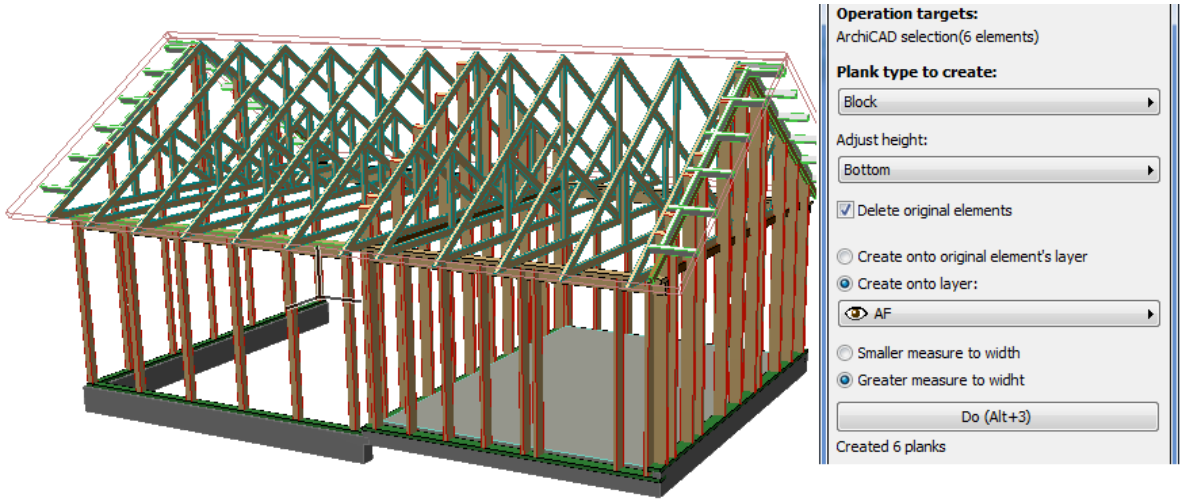
All these examples use model ArchiFrameDemo.pln which is installed as default to folder C:\ArchiLogs\samples.

13.1 Converting ArchiCAD-walls into ArchiFrame planks

1. Use ArchiCAD Find & Select tool to select elements to be converted:

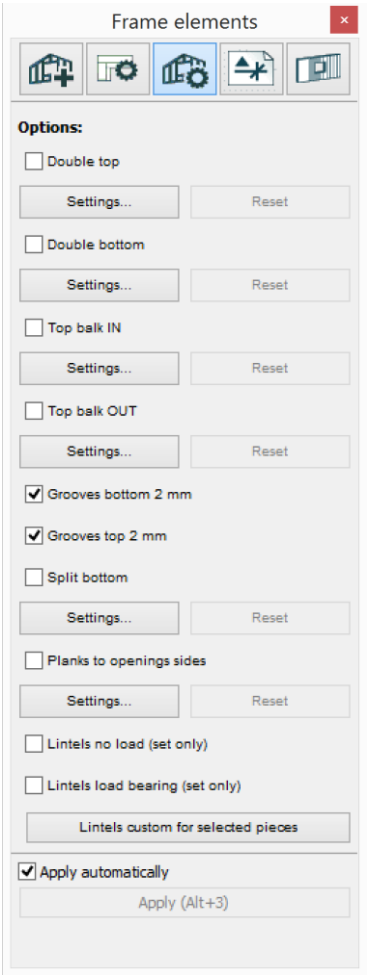


2. Convert the walls into ArchiFrame plank objects deleting original walls. Notice radio button selection in *Greater measure to width*. This way the resulting planks will have rotation angle of 0 degrees.



13.2 Adding grooves to sole plates

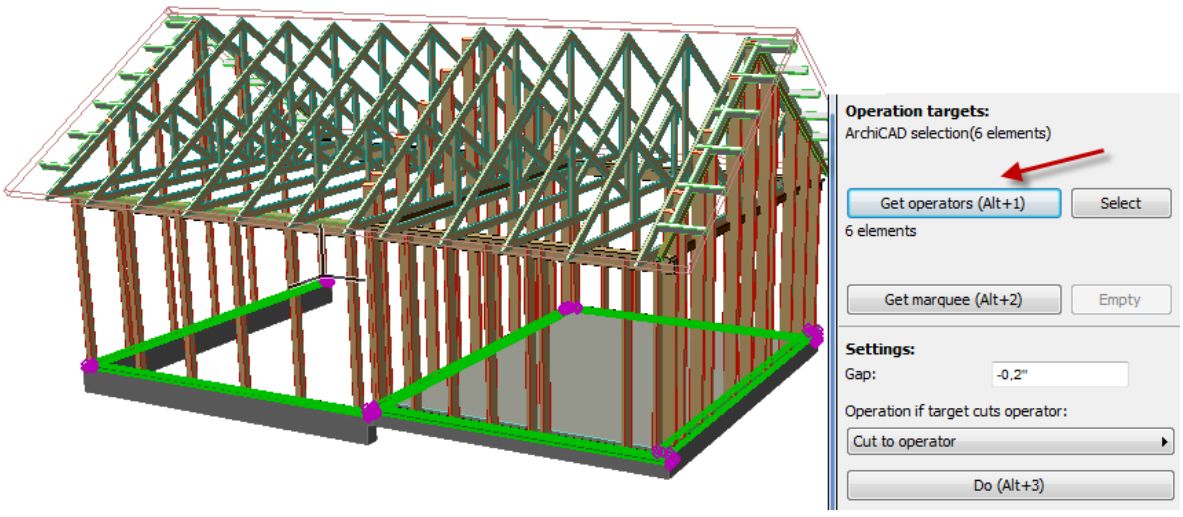
Update: There is an option tool in element tools that can be used to produce standard grooves with just a single click:



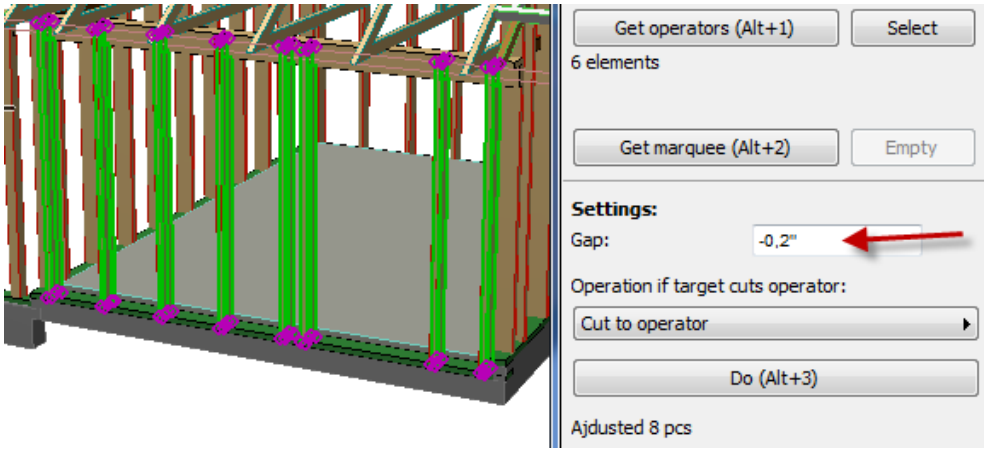
Following describes the steps to do this manually and with possibility to adjust any options.

Extend all studs 0.2” from bottom with *Adjust to operator’s* tool.

1. Select all sole plates as operators (use Find & Select tool):

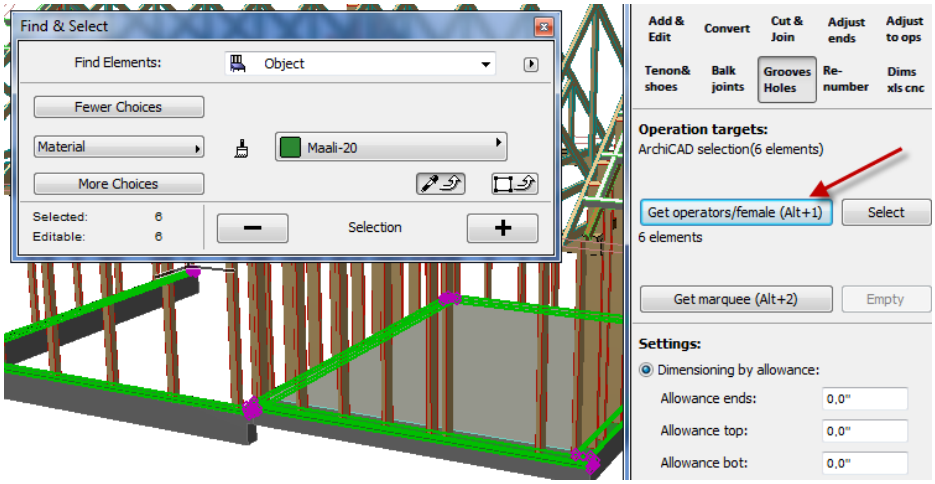


2. Adjust studs to sole plate with negative gap:

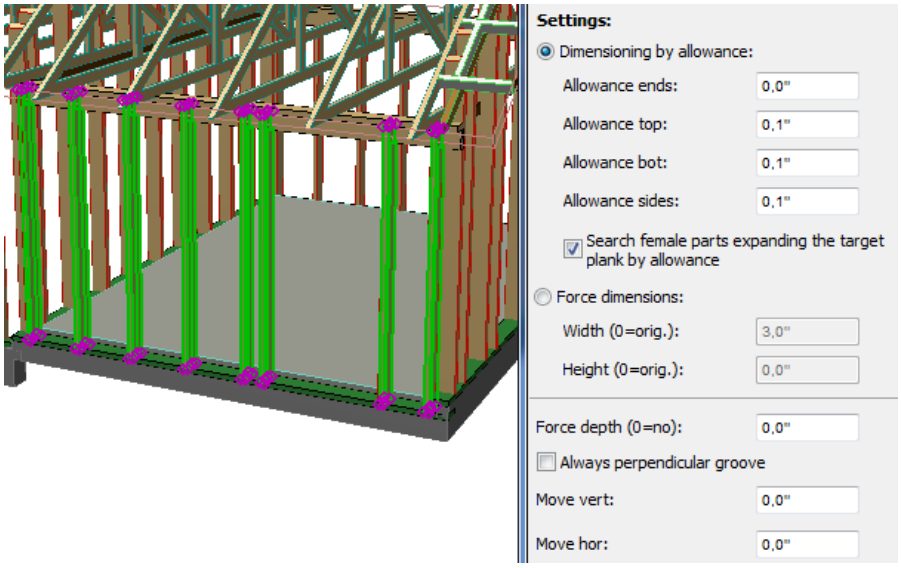


Repeat phase 2 to every stud and after that we will use *Grooves and Holes* tool. Please note that picked operators are reset when the tool is changed:

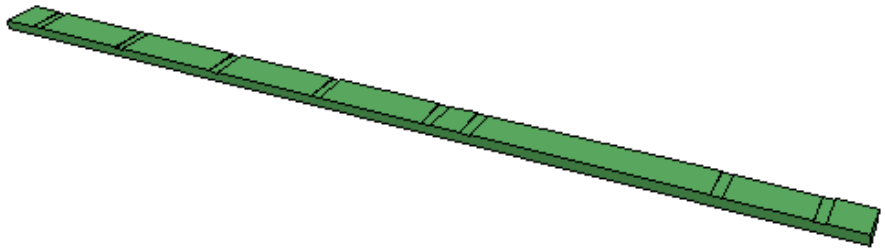
1. Select all sole plates as operators:



2. Select studs and click *Do*. Allowance ends is zero to make just 0.2” groove to sole plate. Other allowances are 0.1” to make the groove 0.2” wider than the stud:



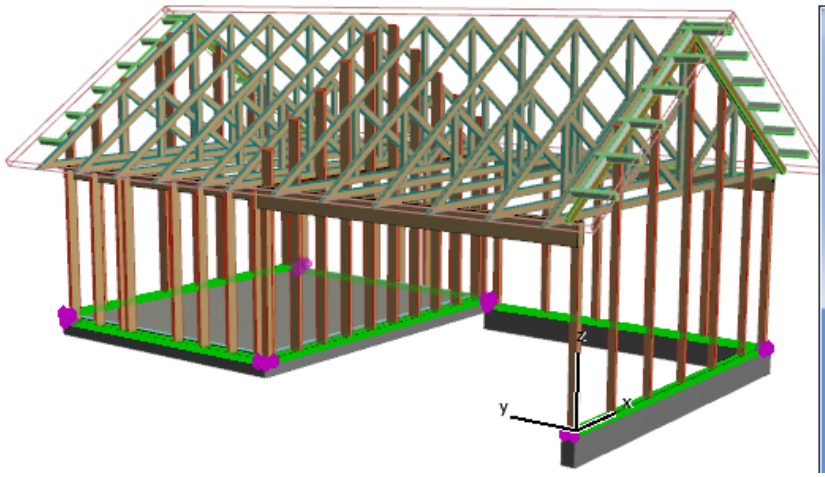
At the end the studs are 0.2 inches longer and sole plates have grooves:

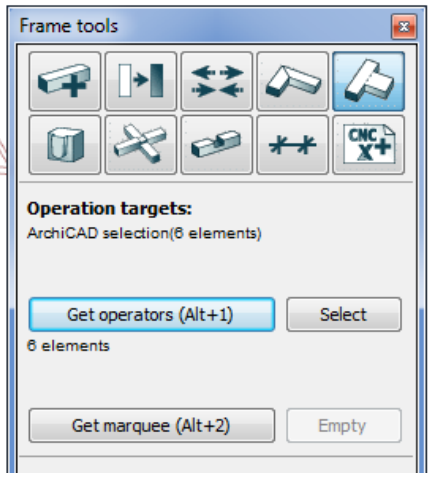


13.3 Adding markings instead of grooves to sole plates

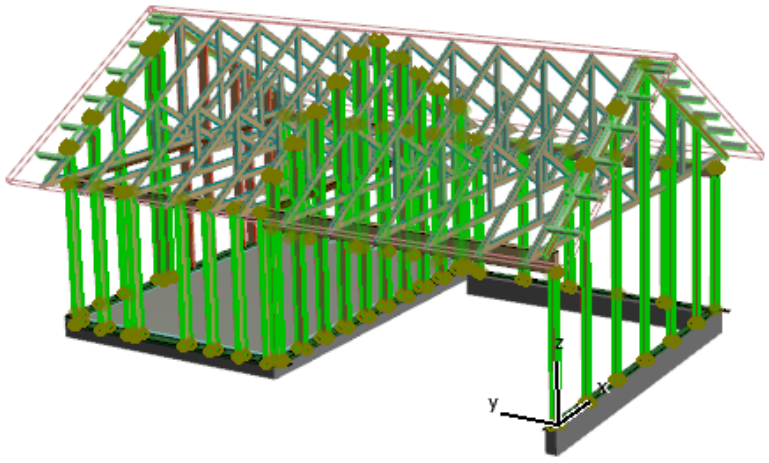
Mark places and lengths of studs to sole plates.

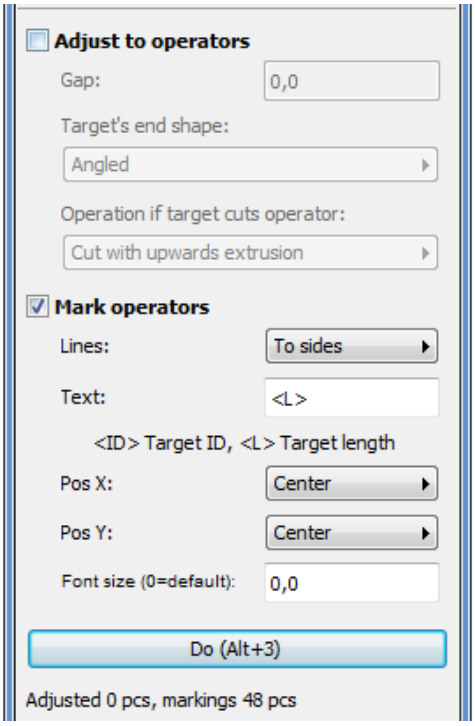
3. Select all sole plats as operators:





4. Select the studs and adjust markings settings:



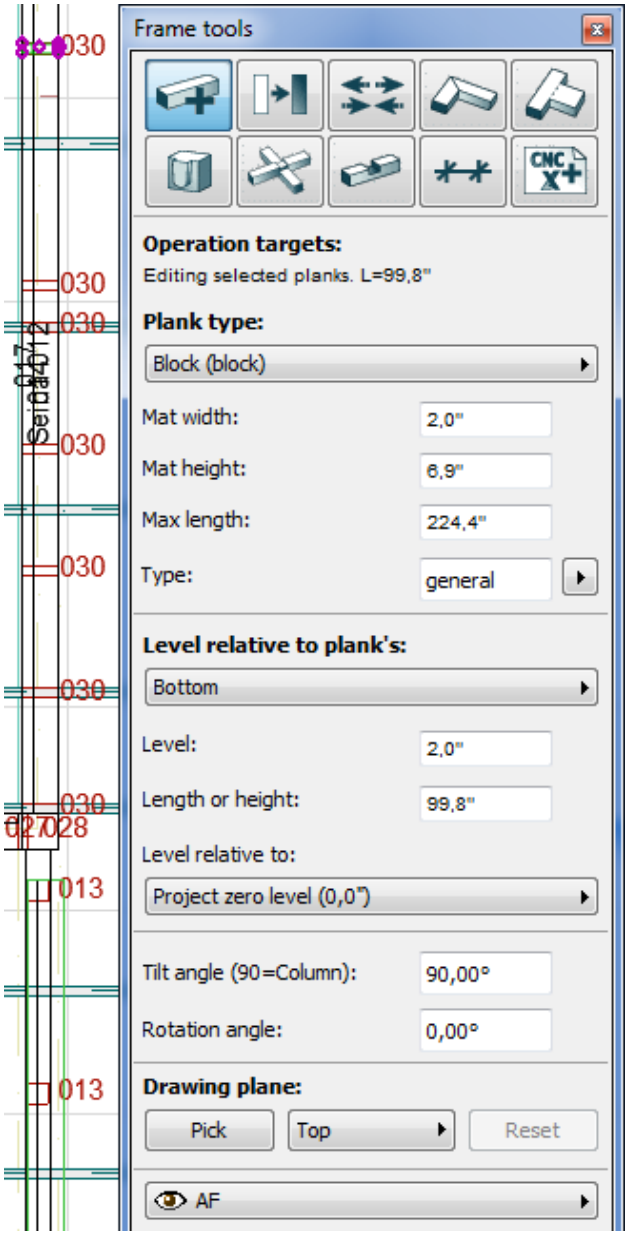


The result contains lines marking the stud places and the stud length next to the stud. Printing the text requires support from the cnc-machine.

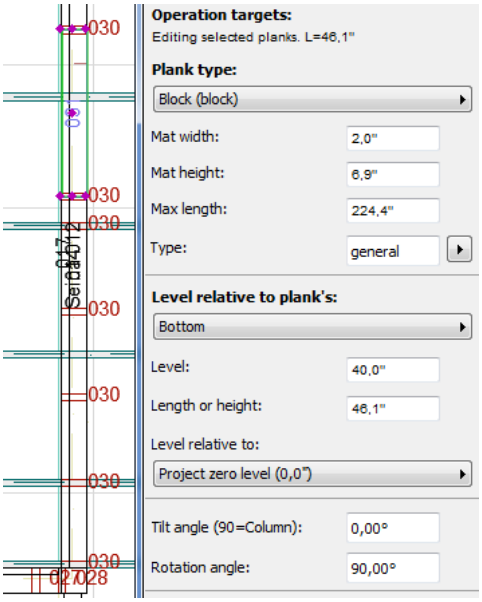
13.4 Adding horizontal runner below window

Let's use the same material as is used on frame studs. The level for the horizontal runner's upper side is 40 inches.

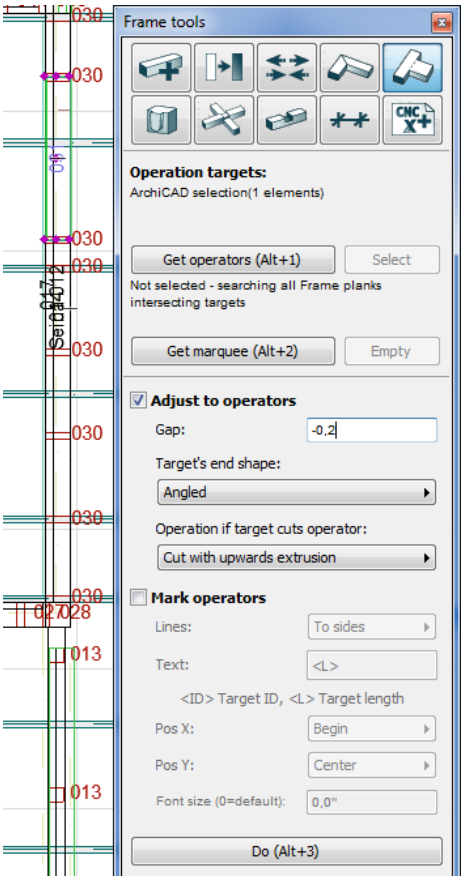
1. Activate ArchiFrame's *Add & Edit* tool and select one stud to get its settings. Also get its settings to object tool by Alt-clicking it. For example, material settings come from standard ArchiCAD object tool:



2. Remove the AC selection by pressing Esc. Then adjust tilt angle to zero (horizontal plank) and rotation angle to 90 degrees since material size is 2,0 x 6,9 inches. Also adjust the level and add the plank with *Add (Alt+3)* between studs. Press Alt to have the plank selected when adding.

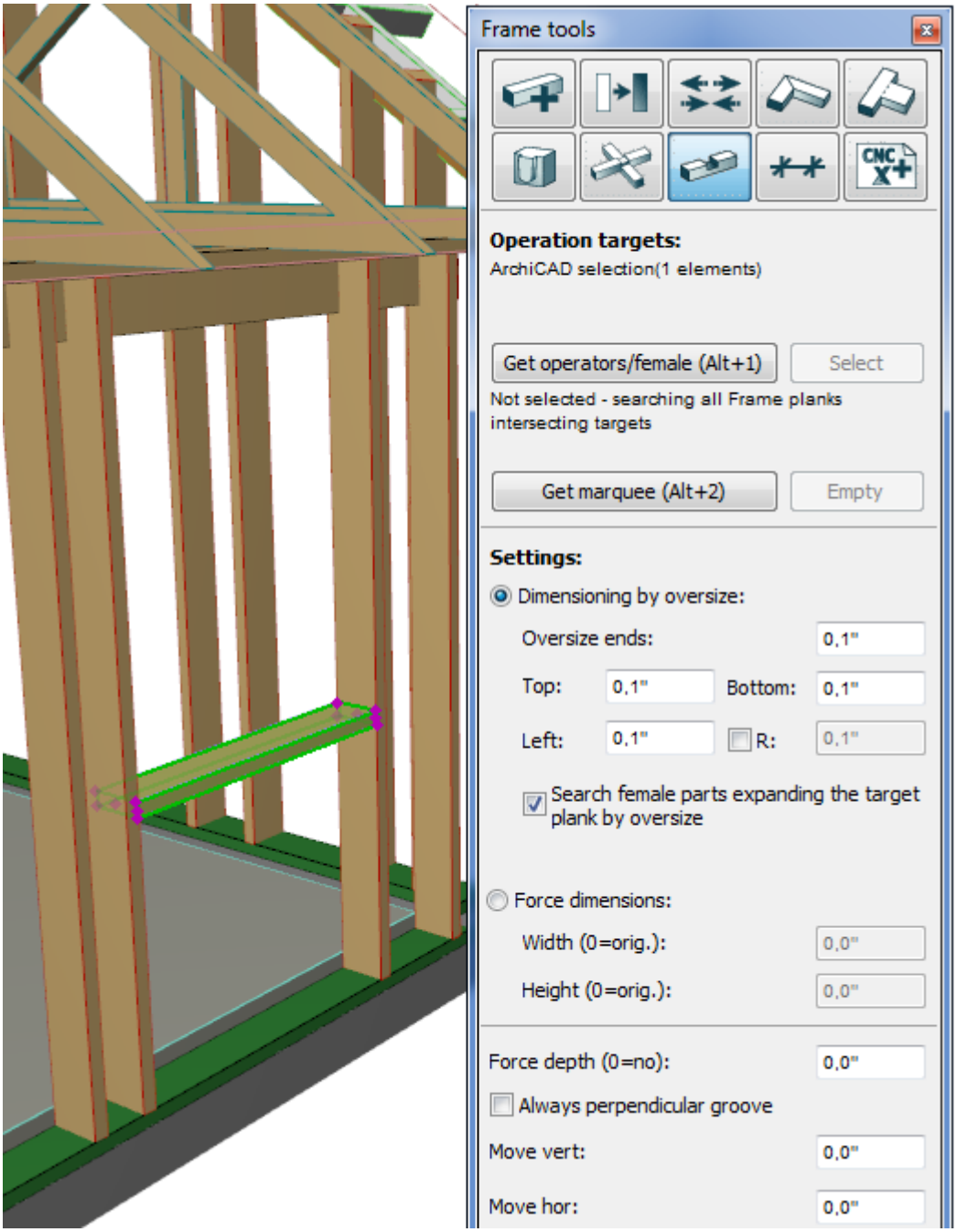


3. Adjust the horizontal runner to extend 0.2 inches inside the studs with *Adjust to operators* – tool. In this case there is no need to get operators since the horizontal runner intersect with just two studs. Before clicking *Do* make sure the horizontal runner is the only selected element:



4. Create grooves to studs. Keep the horizontal plank selected. Use 0.

5. 1-inch oversize everywhere:



13.5 Half lap joint to corner or straight joint

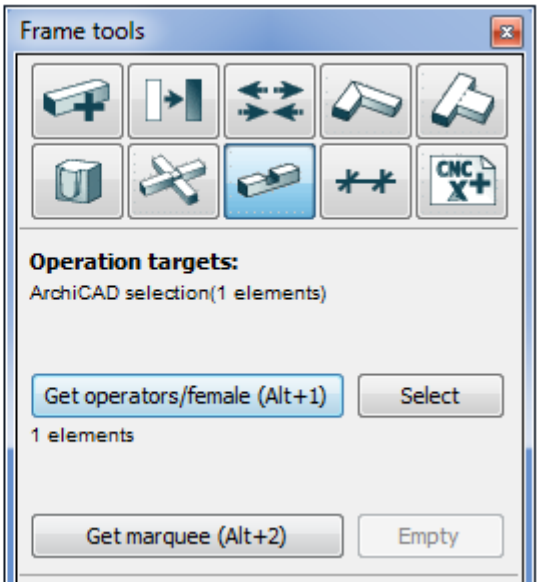
Let's make a joint to sole plate corner. Check how the plank is placed with *Add & Edit* –tool:

Tilt angle (90=Column):	0,00°
Rotation angle:	90,00°

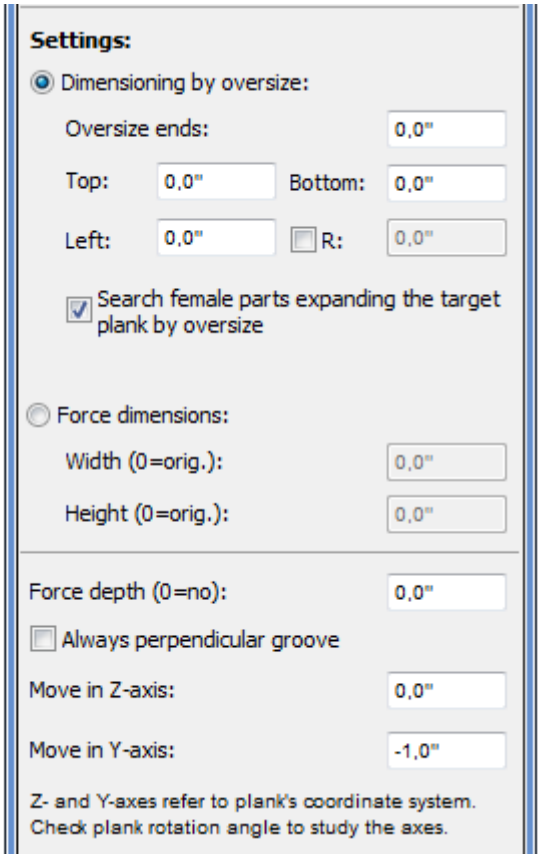
Rotation angle of 90 degrees means that the width is now the vertical extent and the height is on the XY-plane:

Mat width:	2,0"
Mat height:	6,9"

1. Select another of the corner planks and create a groove moving the plank one inch up (here the rotation angle must be considered). If the target plank intersects with many planks, the desired operator plank that gets the groove must be selected with *Get operators/female:*

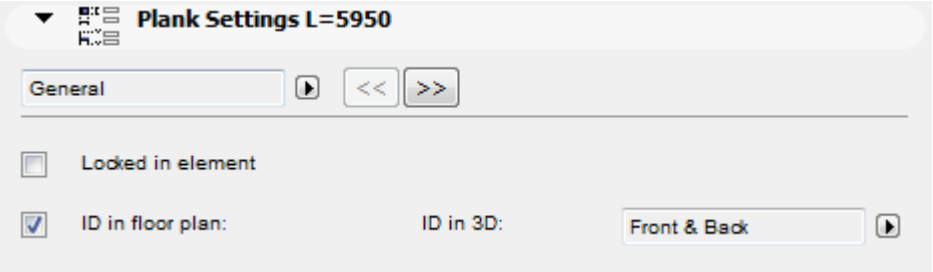


2. Then select another plank and set Y-movement to -1". The value moves the cutting plank upwards 1" since it is rotated 90 degrees and the begin point is at the closer end:



3. Do the same swapping operator and target planks and changing Move Y to 1".

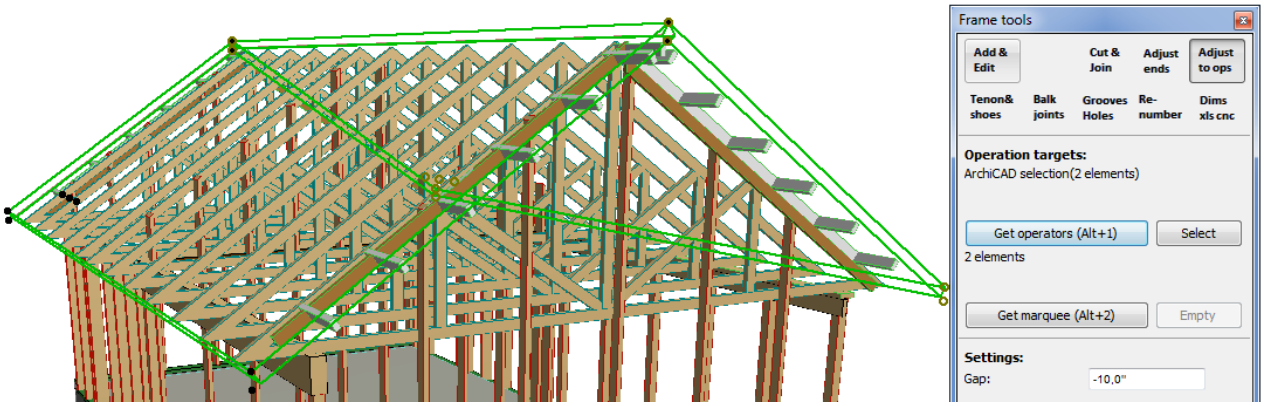
Plank direction can be seen only from its ID. It can be set to be visible also in 3D from object settings:



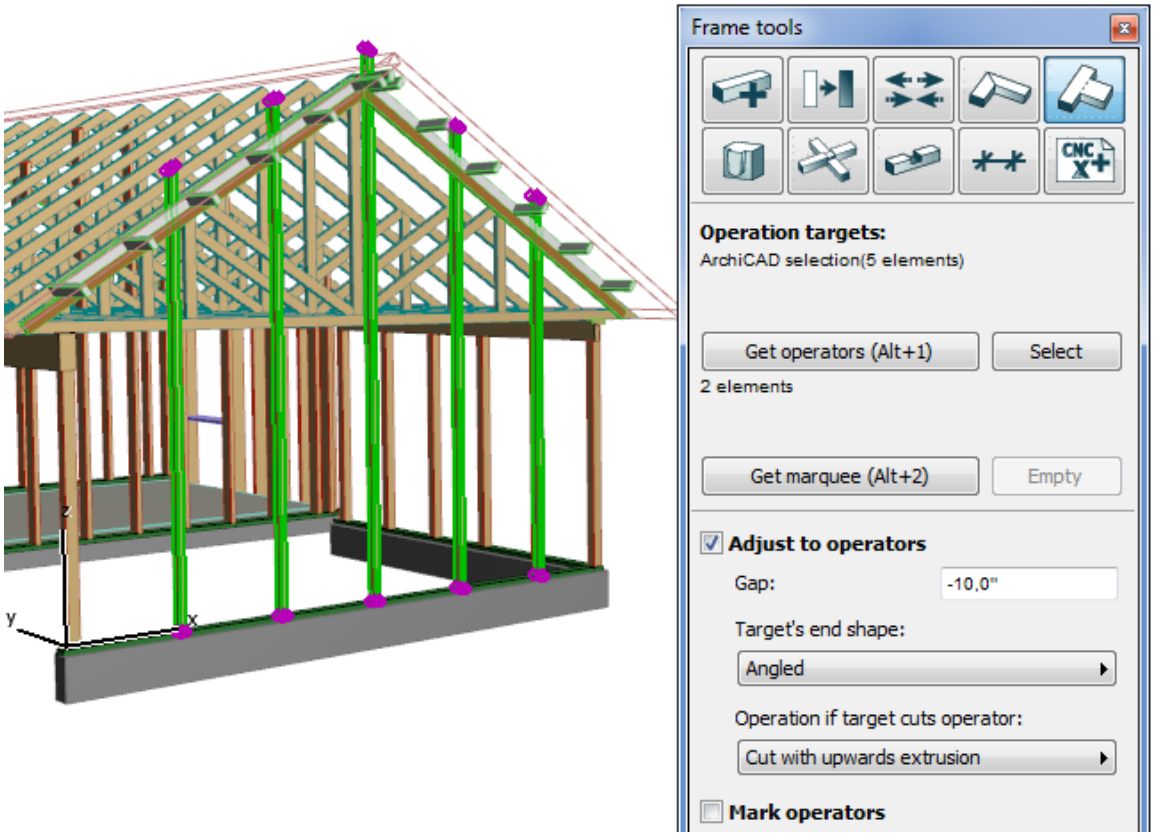
13.6 Adjust studs with roof

The same mechanism could be used to adjust planks with ArchiCAD-walls. Now we will adjust the studs to extend 10 inches over to roof bottom side.

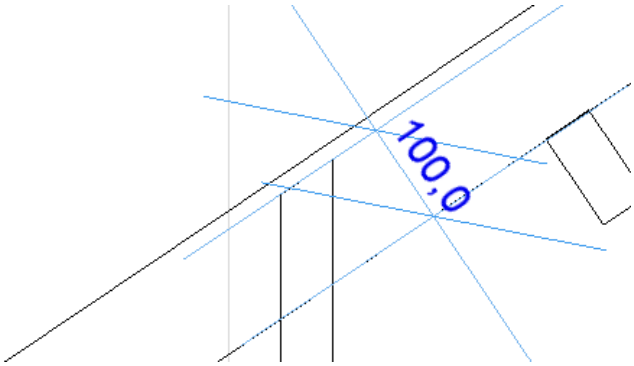
1. Get two roofs to operators:



2. Set gap to -10 inches. Positive gap would cut the studs below the operators (roofs):



Gap is perpendicular to roof slope:

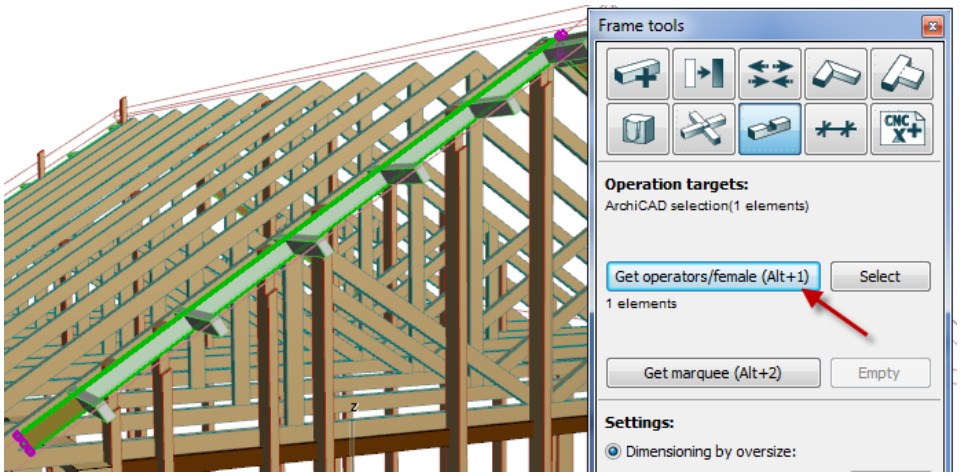


In vertical direction the distance is value / cos (roof angle).

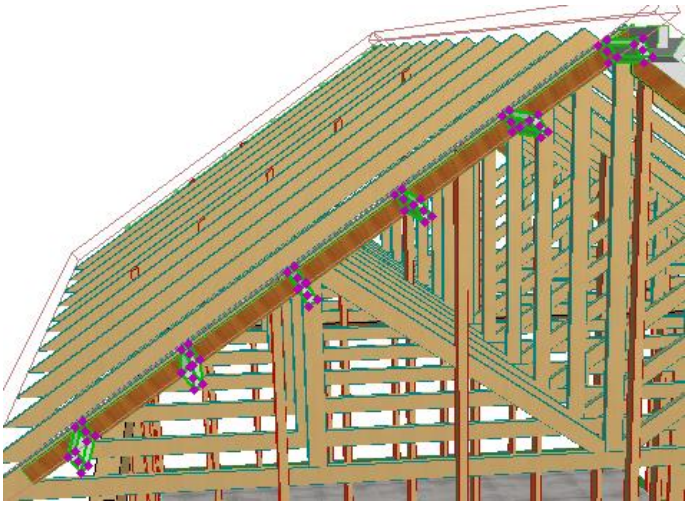
13.7 Cutting the plank with targets

In this example a solid plank is cut with selected targets using *Grooves*–tool.

1. Pick the plank to be cut as operator:



2. Select the cutting targets and click *Do*:

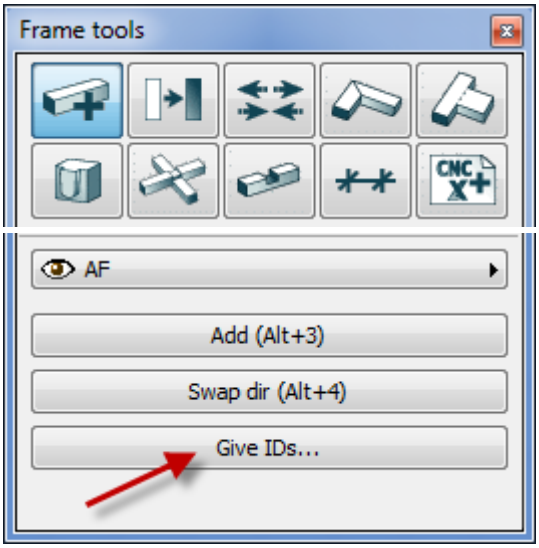


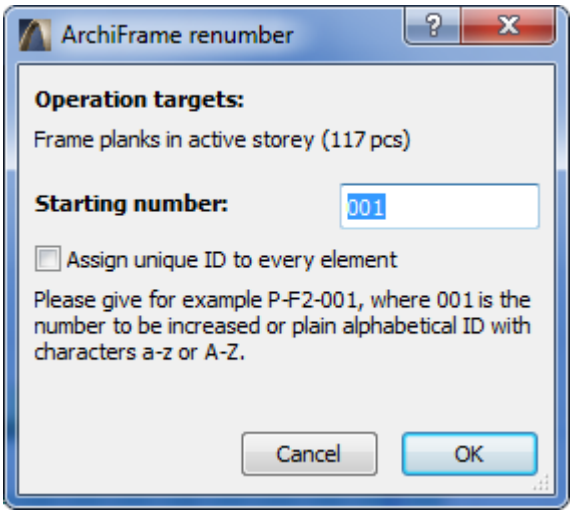
3. ArchiFrame cuts the operator plank. If cutting pieces (operation targets) did not cut the operator, it would have got grooves.

13.8 Operations just before manufacturing

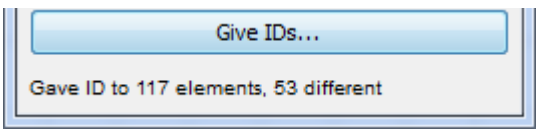
The last operations are giving the IDs (renumber), creating Excel listing and saving Hundegger BVN-file.

1. Give IDs. Nothing is selected and active storey is 1:

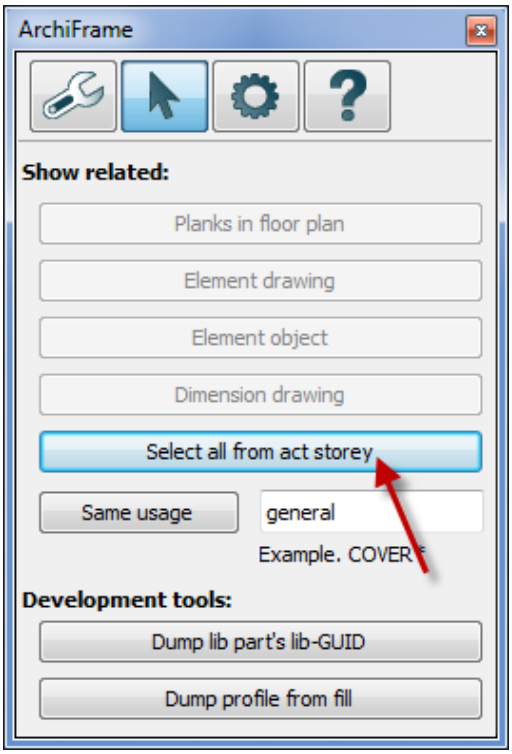


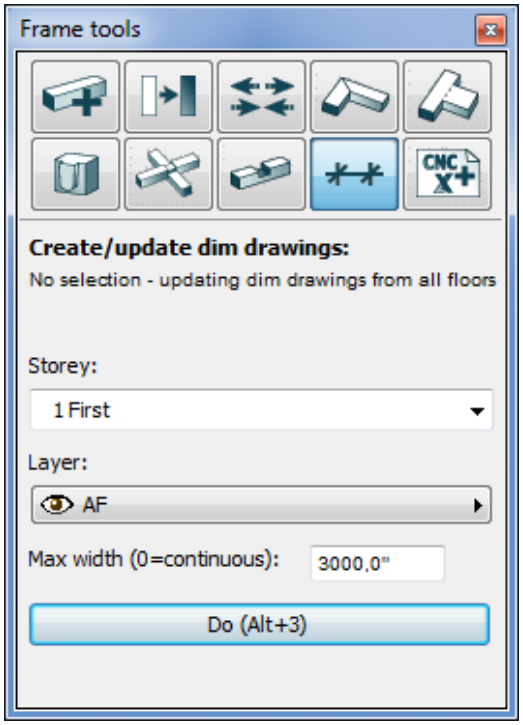


Status is here:

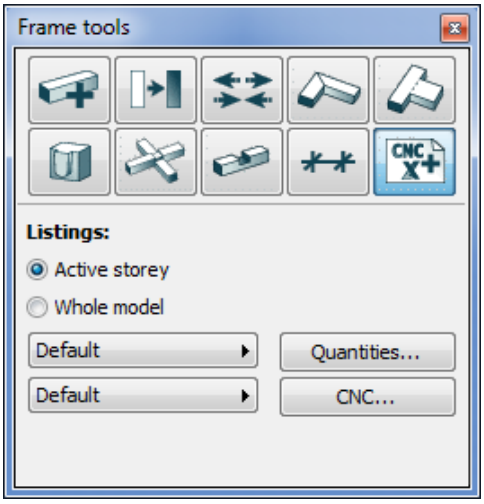


2. Create dimension drawings to first floor. For that select all the planks and carefully check the settings before clicking *Do*:





3. Save the listing to Microsoft Excel. If there is no selection, the listing is created from every plank visible in active floor:



1	**Project:**	Set project info 2								
2	**Project number:**	Set project info 3								
3	**Bvn-file**	Set project info 3.bvn								
4	**Number**	**Name**	**Mat**	**Width**	**Height**	**Pieces**	**Length**	**Len tot**	**Type len**	**Type m3**
5	009	general	5,5"x12,4"	140	315	1	5,400	5,400	5,400	0,238
6	002	general	2,0"x6,9"	50	175	1	5,137	5,137		
7	015	general	2,0"x6,9"	50	175	1	5,100	5,100		
8	020	general	2,0"x6,9"	50	175	1	4,570	4,570		
9	024	general	2,0"x6,9"	50	175	3	4,395	13,185		

etc… In the listing, every different plank is in its own line and total sum of material type is calculated. The dimensions are in mill metric and metric form because Excel does not have build-in support for imperial dimensions.

4. Create Hundegger cnc-file:

```
    Aseta Projektin tied          0              0       0       0       0       0            0   0
000001 general (A1-01)                          0
000001               26       0    1000     500    6300
000001 0100 03        0       0       0     900     900       0       0       0       0       0 000
000001 0100 -3     6300       0       0     900     900       0       0       0       0       0 000
000002 general (A1-02)                          0
000002                1       0    1750     500   51375
000002 0100 03        0       0       0     900     900       0       0       0       0       0 000
000002 0100 -3    51375       0       0     900     900       0       0       0       0       0 000
000003 general (A1-03)                          0
000003                2       0    1000     500   39382
000003 0100 03        0       0       0     900     900       0       0       0       0       0 000
000003 0100 -3    39382       0       0     900     900       0       0       0       0       0 000
000003 0100 -3    39382     500    1000     900    1240       0       0       0       0       0 000
```

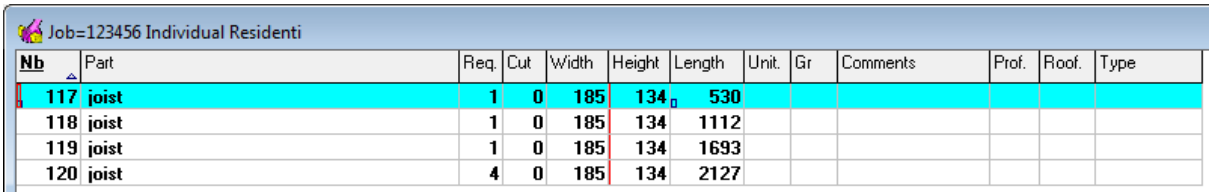
etc…

14 Listings

14.1 Hundegger cnc-file

File extension is bvn. The planks will get the ID like below:

- In case of fully numeric ID, it will be copied to Hundegger's number column and Part column will contain the set usage type:



Nb	Part	Req.	Cut	Width	Height	Length	Unit.	Gr	Comments	Prof.	Roof.	Type
117	joist	1	0	185	134	530						
118	joist	1	0	185	134	1112						
119	joist	1	0	185	134	1693						
120	joist	4	0	185	134	2127						

- If the ID contains non-numeric digits, ArchiFrame will assign consecutive numbers to the number columnd starting from one. Part column will contain ID (usage type):

Nb	Part	Req.	Cut	Width	Height	Length	Unit.	Gr	Comments	Prof.	Roof.	Type
1	C-001 (ELEM)	2	0	195	41	127						
2	C-002 (ELEM)	1	0	195	41	6000						
3	C-003 (ELEM)	2	0	195	41	2886						
4	C-004 (ELEM)	2	0	195	41	2918						

14.2 Editing listings

Note! Always edit a copy of the *data*-folder, for example C:\ArchiFrame\Data_own. This way the changes will remain during installation of updates.

The file to be edited is ArchiFrameListing.lua inside the data folder (see Lua scripts). In addition to the default listing there are special listings defined in global variable gtblListings. One item in the table contains fields:

Field	Description
strName	Name of the listing.
strUIClass	Special listings can be invoked from another place in the UI other than the default listing tool:

	• Cover, listing for weatherboards.
strOnInitFunc	Function that is called at the start of listing.
strOnSaveListFunc	Function that creates the actual listing.

strFilterFunc	Function that filters planks to collect, "" = no filter. For example:
	`function FilterPlank(sGuid)`
	`  local sUsage`
	`  ac_objectopen(sGuid)`
	`  sUsage=ac_objectget("iUsageId")`
	`  ac_objectclose()`
	`  if sUsage==nil then`
	`  return false`
	`  end`
	
	`  return string.upper(sUsage)=="BUCK"`
	`end`

nSorting	Sorting criteria: • 0, by plank ID. • 1, material id, height, thickness, ID, length. • 2, material id, height, thickness, length, ID.
nAllowSameId	Many different planks have the same ID: • 0, no (default). • 1, may have.
nCollectElem	Collect all planks related to the elements (use possible selection to collect every plank from elements related to the selection): • 0 or missing, no (default). • 1, collect all.
nCollectType	ArchiFrame collects planks only to table gtblPlanks. This field is used to change that. • Missing, default and collect planks only. • 0, don't pre-collect anything. • 1, collect planks. • 4, collect boards. • 1+4, collect planks+boards.

ArchiFrame also sets Lua-global gbPlanksSel to tell if source for the operation was element selection (true) or not selection (false). If listing script sets Lua-global gnStatusCount, it will be shown in tool palette's status text after the operation. Script may also set global gbCnc-IncludeNoCnc- to process also planks having parameter icnc- set to zero (usually skipped).

15 Material list ArchiFrameBlocks.xml

This file contains all pre-set material types and their settings. Editing the file is best to start from Materials <materials>/<material>.

15.1 Settings <settings>

```
<archiframe>
  <settings>
    <misc>
      <manufacturer name="xxx"></manufacturer>
```

Manufacturer name affects possible customisations programmed into ArchiFrame.

15.1.1 Printing settings

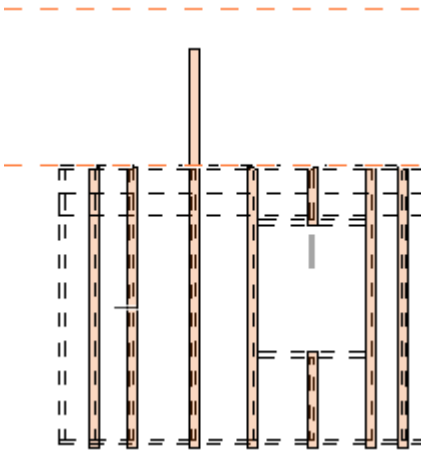
```
<print layoutname="Elem drawing &lt;elemid&gt;" frame="0"
       masterland="A4 Wall drawing" masterlandxsize="297" masterlandysize="210"
       masterport="A4 Wall drawing portrait" masterportxsize="210" masterportysize="297">
  <empty_custom name="MyMaster" left="0" top="0" right="100" bottom="0"></empty_custom>
</print>

<printframe dimtitle="Frame ">
  …
</printframe>
```

Tag <print> defines master layout names and dimensions for landscape and portrait types. Both can be defined with same dimensions to have always the same orientation.

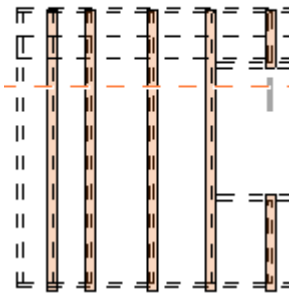
<empty_custom> defines how much space should be left at each side of the print area. This is useful if there is a stamp in master layout that should not be tampered. The example leaves 100 mm of the right side of the layout untouched. This setting is used if master layout name matches with name-attribute. Name may contain wildcards (name = "*") will match everything. It may also contain attributes:

Xlm-attribute	Description
valign	Align vertically in the master layout: • -1, top. • 0, center. • 1, bottom (default).
halign	Align horizontally in the master layout: • -1, left (default). • 0, center. • 1, right.

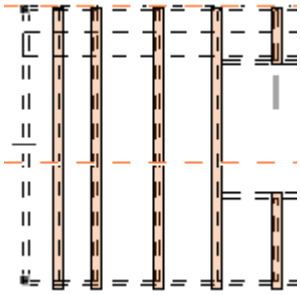
An example of different vailgn values for a projection with layoutminheight="5",
To bottom, valign="1":



Center, valign="0"



Top. valign="-1":



<*printframe*>-tag contains XML-settings for an ArchiCAD element.

Xlm-attribute	Description
layoutname	Default text for print dialog.
frame	Values: • 1, include the frame (default). • 0, no frame.
skiplibpart	If given the listed library parts are skipped when creating the layouts, for example, to skip standard element stamp, specify skiplibpart="9FD03D30-6A12-4B0F-AB3A-01410A529E38". Add multiple values using comma as a separator.
masterland	Name for landscape drawing (width>height).
...xsize, ...ysize	Paper size in millimetres.
masterport	Name for portrait drawing (height>width). Both landscape and portrait definitions must exist even if not used.
dimtitle	Title for single plank dimension drawings. List of plank Ids will be added after the title.

15.1.2 Plank dimension drawings <projections>

```
<projections layer="xxx" maxwidth="80.0" margboxes="3.0" margleft="1.0" margtop="1.0"
margright="1.0" margbot="1.0" margproj="0.7">
```

Xlm-attribute	Description
layer	Layer to place the dimension drawings. If the layer does not exist, it is created. Empty or missing layer name places dimension drawings to the same layer as the original plank.
maxwidth	Dimension drawing line maximum length in meters. If drawing does not fit to current line, it is placed to the next line above.
margboxes	Marginal between the dimension drawing frames.

margxxx	Marginal between the frame and markings inside it.
margproj	Vertical marginal between dimension drawings from different plank sides.

Global settings for any projection element in dimension drawings and element projections:

```
<projbase>
  <!--layer>Name, auto created</layer-->
  <!--elemparam name="pen">1</elemparam-->
</projbase>
```

It is possible to list any XML-settings for an ArchiCAD element. In this place as exception unknown lines do not cause any message. That allows setting for example object parameters even for dimension lines.

Tag projplank contains settings for the projection. It may be useful to set the line type to solid instead dash & dot etc:

```
<projplank>
  <elemparam name="linetype">1</elemparam>
  <objparam name="#useobjlinetype">0</objparam>
</projplank>
```

Dimension settings:

```
<dimsettings>
  <drill midout="0"></drill>
</dimsettings>
```

Xml-tag	Description
drill	Settings for drill dimensions, attributes: • Midout, default is 1. With value 0, drillings outside the plank surface will not have dim line for Y-coordinate.

15.1.2.1 Dimension drawings titles <projections>/<text_title>

This section defines the titles and text settings for dimension drawings from different plank sides.

```
<text_title yoff="0.10">
  <settings>
    <elemparam name="fontname">Arial</elemparam>
    <elemparam name="fontsize">1.2</elemparam>
    <elemparam name="fontstyle"> </elemparam>
    <elemparam name="pen">11</elemparam>
  </settings>
  <side1>
    <text>TOP</text>
  </side1>
  <side2>
    <text>FRONT</text>
  </side2>
  <side3>
    <text>BOTTOM</text>
  </side3>
  <side4>
    <text>BACK</text>
  </side4>
</text_title>
```

Xlm-attribute or tag	Description

yoff	Vertical marginal between the plank projection and the text.
<settings>	Settings for the text element, see XML-settings for an ArchiCAD element.
<sideX>	Text for different surfaces.
lang	To be used with *<text>*-tag. If current ArchiFrame language matches with the setting, that tag will be used. Otherwise first *<text>*-tag is used.

15.1.2.2 Dimension line settings <projections>/<dim_xxx>

Contains settings for different kinds of dimension lines, see XML-settings for an ArchiCAD element.

Dimension line types are:

Xlm-attribute or tag	Description
<dim_main>	Main dimension line.
<dim_mcmain>	Machinings dimension line.
<dim_angled>	Angled machining.
<dim_mcdet>	Additional machining dimensions.
<dim_mcdetangle>	Angled machinings additional dimensions.
<text_mcdet>	Settings for text details. Additional attributes for this tag: • Diapre = "Ø ", text to show before actual drill diameter.
yoff	Dimension line vertical distance to the plank.

15.1.2.3 Dimension drawings summary text <projections>/<text_summary>

Summary text is placed below the dimension drawings. See XML-settings for an ArchiCAD element.

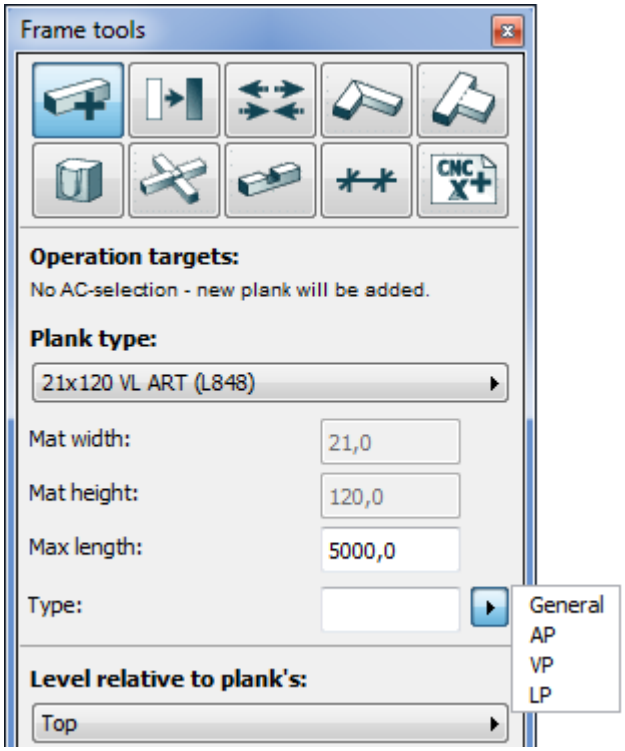
```
<text_summary height="1.4" minwidth="3.0">
  <settings>
    <elemparam name="fontname">Arial</elemparam>
    <elemparam name="fontsize">1.5</elemparam>
    <elemparam name="fontstyle"></elemparam>
    <elemparam name="pen">11</elemparam>
  </settings>
  <text lang="fin">Mat: [mat]\nLaatuluokka/Quality: [quality]\nNimi/Name:
[usage]\nSamanlaisia/Similar: [count]\nTunnukset/IDs: [idlist]</text>
</text_summary>
```

Text may contain parts to be replaced:

Part	Description
[mat]	Material name and ID, for example 41x195 (XYZ 123).
[quality]	Quality class from the plank object.
[usage]	Usage text from the plank object.
[count]	Number of similar planks.
[idlist]	IDs of similar planks.
\n	Line feed.

15.1.3 Plank usage pre-set list <usagetypes>

Use the list here:



Attribute name is the name for the list and id is the actual value to save. Name and id should be the same unless there is a need to use very short abbreviations.

15.1.4 Hidden balk joint list <balkhiddenjoints>

This section defines machinings for different hidden balk joints and accessory codes to be used in custom listings.

```
<balkhiddenjoints>
  <balkhiddenjoint name="Hidden balk joint 1" id="HIDDENBALKJOINT1" cutwidth="0.050"
cutheight="0.150" cutdepth="0.005" grodepth="0.100" growidth="0.008" drilly1="0.050"
drilldist="0.080" drillsize="0.010" drillspace="0.040" drillcount="3">
  </balkhiddenjoint>
</balkhiddenjoints>
```

Xlm-attribute	Description
name	Joint name.
id	Beam shoe id for custom listings.
cutwidth	Shoe back plate width.
cutheight	Shoe back plate height, shoe is machined into the plank from top surface downwards.
cutdepth	Show back plate thickness.
grodepth	Length of the shoe part that actually holds the beam (goes into the beam).
growidth	Thickness of above.
drilly1	First drilling's distance from top surface.
drilldist	Drilling distance from plank's end.
drilldist2	The same if there is another row of drillings.
drillsize	Drilling diameter.
drillspace	Distance between the drillings.
drillcount	Number of drillings.

15.1.5 Mortise and tenon joint default dimensions <mortisetenon>

```
<mortisetenon len="0.056" lenrelative="0.5" bordertop="0.02" borderbot="0.005"
borderside="0.01" roundingr="0.0225" endgap="0.001" sidegap="0.001" topgap="0.010">
</mortisetenon>
```

Xlm-attribute	Description
len	Tenon length.
lenrelative	Overrides len, use tenon length relative to plank width.
bordertop	Border top.
borderside	Border sides.
roundingr	Rounding radius for corners. Affects directly 3D and Hundegger cnc- so that zero means rectangular and any other value will create rounded tenon (rounding radius is defined by the machine).
endgap	Mortise bottom gap.
sidegap	Mortise side gap.
topgap	Mortise top gap.

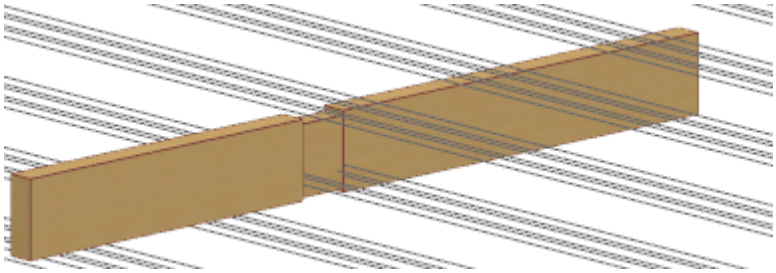
15.1.6 Dovetail joint default dimensions <dovetail>

```
<dovetail len="0.038" bordertop="0.00" borderbot="0.050" bottomwidth="0.06"
roundingr="0.0225" endgap="0.000" sidegap="0.000" angle="6" >
</dovetail>
```

Xlm-attribute	Description
len	Tenon length.
bordertop	Border top.
borderbot	Border bottom.
bottomwidth	Dovetail tenon width at bottom (in pocket the width at plank surface).
roundingr	Rounding radius, dovetail is always rounded – this affects only the 3D.
endgap	Pocket bottom gap.
sidegap	Pocket side gap.
angle	Opening angle.

15.1.7 Narrowed balk default dimensions <balkborder>

Narrowed balk is used for example when beam goes through a log wall:



```
<balkborder bordertop="0.02" borderbot="0.005" borderside="0.01" overtop="0.001"
overbot="0.000" overside="0.001" overlen="0.000" >
</balkborder>
```

Xlm-attribute	Description
bordertop	Top side narrowing.
borderbot	Bottom side narrowing.

borderside	Side narrowing.
overtop	Oversize top.
overbot	Oversize bottom.
overside	Oversize sides (to be used to both left and right).
overleft	If different on the sides.
overrigh	If different on the sides.
overlen	Lengthwise oversize. For example if the wall is 200 mm thick and this value is 1 mm, the narrowing will be 201 mm long.

15.1.8 Intact balk default dimensions <balkcut>

As narrowed balk but no machinings in the balk.

```
<balkcut overtop="0.01" overbot="0.000" overside="0.001" >
</balkcut>
```

15.1.9 Default settings <plankdefaults>

Plank default settings. When adding new planks these settings are set to the plank. See XML-settings for an ArchiCAD element.

```
<plankdefaults defaultlayer="Wall drawings.2d" plankminlen="0.01">
  <elemparam name="pen">11</elemparam>
  <objparam name="iLengthFormat">mm</objparam>
  <objparam name="iFontSizeBase">10</objparam>
</plankdefaults>
```

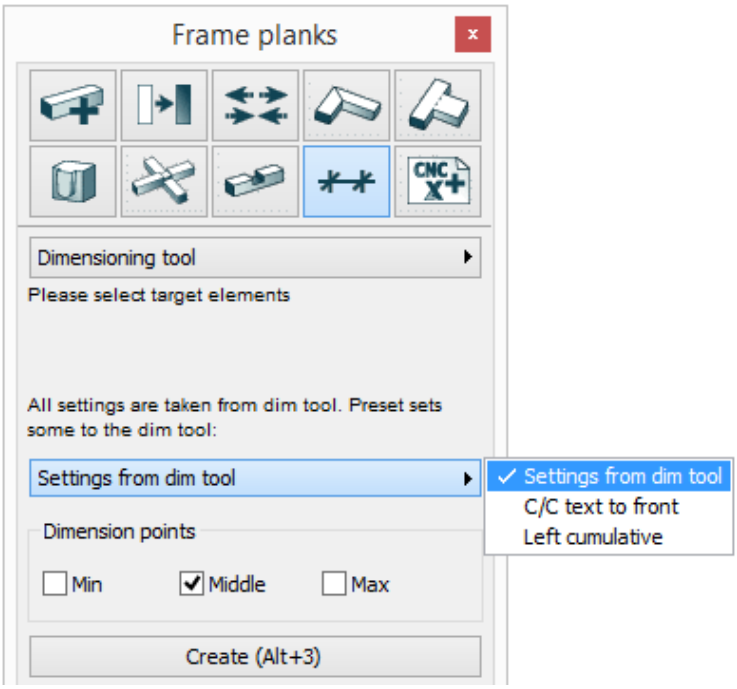
Xlm-attribute	Description
defaultlayer	Default layer for add & edit tool palette. If the layer is forced, it is given the normal setting.
plankminlen	Shorter planks are not created and are removed automatically.

15.1.10 CNC-limitations <cnc-limits>

Xlm-attribute	Description
gromaxdepth	Groove maximum depth. Groove tool palette uses this value to select groove target side in the plank. The side that does not exceed the maximum is better than the one that exceeds the maximum value.
nailmindistplank	Nail minimum distance from plank edge.
nailmindistboard	Nail minimum distance from board edge.
elemgeo	Value: • 0, no cnc- for element geometry, give same ID if planks are similar mirrored or direction swapped. Using this value still assigns different ID to the planks in case of certain machinings like angled ending having angle <> 90 degrees. Give value 2 if it is OK to combine these also. • 1, element geometry cnc- needed – give similar IDs only if similar unmirrored and not swapped (default). • 2, see value 0.

15.1.11 Dimension tools presets

These definitions affect here:



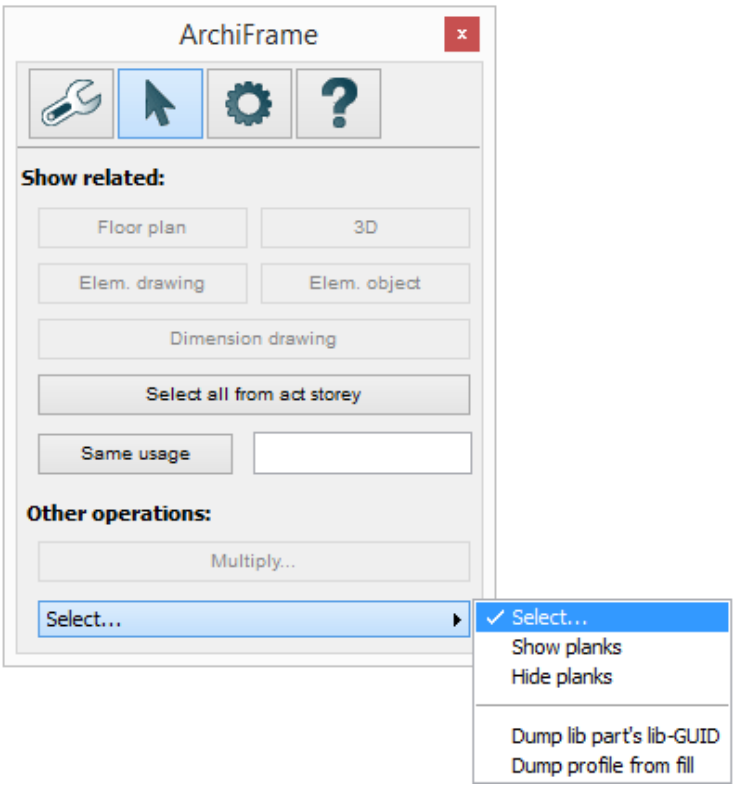

```
<dimpresets>
  <dimpreset name="Settings from dim tool" name_fin="Asetukset mittatyökalusta">
  </dimpreset>

  <dimpreset name="C/C text to front" name_fin="C/C teksti alkuun" setmin="0" setmid="1"
setmax="0">
    <dimlinesettings>
      <elemparam name="dimensiontype" >linear</elemparam>
      <elemparam name="fontname">Arial</elemparam>
      <elemparam name="fontsize">1.5</elemparam>
      <elemparam name="fontstyle"></elemparam>
    </dimlinesettings>
    <text content="C/C" textid="0" anchor="6" marg="0.2">
      <elemparam name="pen">11</elemparam>
      <elemparam name="fontname">arial</elemparam>
      <elemparam name="fontsize">1.5</elemparam>
      <elemparam name="fontstyle"></elemparam>
    </text>
  </dimpreset>

  <dimpreset name="Left cumulative" name_fin="Vasen juokseva" setmin="1" setmid="0"
setmax="0">
    <dimlinesettings>
      ...
    </dimlinesettings>
    <text content="LEFT" textid="0" anchor="6" marg="0.2">
      ...
    </text>
  </dimpreset>
</dimpresets>
```

Xlm-attribute	Description
name	Preset name in the tool.
setmin	Default value for check box Min.
setmid	Default value for check box Middle.
setmax	Default value for check box Max.

It is possible to attach text element to the dimension line using <*text*>-tag, for details please see Single dimension line settings <dimline>.

15.1.12 Selection tool automations

These definitions affect here:




```
<selautomations>
  <script name="Show planks" name_fin="Näytä kapulat" openundo="0">
    <![CDATA[
ac_environment ("layer", "af *", 1)
ac_environment("rebuild", 1)
]]>
  </script>
  <script name="Hide planks" name_fin="Piilota kapulat" openundo="0">
    <![CDATA[
ac_environment("layer", "af *", 0)
ac_environment("rebuild", 1)
]]>
  </script>
</selautomations>
```

The items may contain any kind of Lua scripts. ArchiFrame provides interface to ArchiCAD. Please see Lua scripts.

15.2 Materials <materials>/<material>

Material list contains pre-set materials. In addition to these the default material list contains freely adjustable block, planed and round materials. Single material is defined with <*material*>-tag and inside it is possible to give Settings <settings> except <projections>-part. In the example the pen for the material is set to value 22.

```
<material id="XYZ200" name="Weather A" thickness="0.025" height="0.120" shape="block"
maxlen="5.0" m3factor="0" fixlen="0.210" tolist="0">
  <settings>
    <plankdefaults defaultlayer="Wall drawings.2d" plankminlen="0.01">
```

```
      <elemparam name="pen">22</elemparam>
    </plankdefaults>
  </settings>
</material>
```

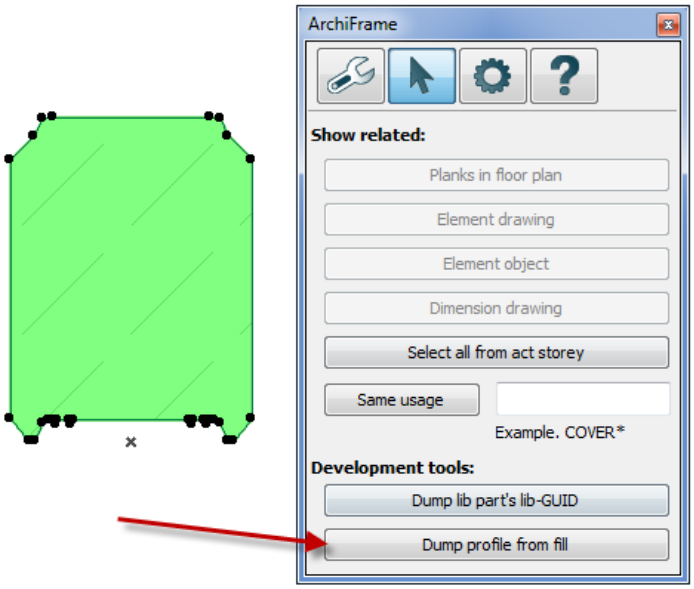
Xlm-attribute	Description
id	Material internal ID. Name can change in different languages but the material can still be matched by the same ID.
name	Material name to be listed for the user.
thickness	Material thickness, usually the smaller extent is given here.
height	Material height.
shape	Material basic shape: • Block, rectangular. • Plane, planed from corners. • Round, circular shape. Basic shape must be given even if material profile is provided.
maxlen	Maximum length. If exceeded the pank displays MAX LEN-remark.
m3factor	For listings to define volume: material volume = meters*m3factor.
fixlen	If given, the material length is always fixed.
tolist	Value 0 removes the material from Add & Edit tool palette. Material can still be used for example for weatherboards.
density	Material density (kg per dm3), default value is 0.460 which can be used for dry pine.
kgpermeter	Another option to give weight by kg per meter.
lbperfoot	Another option to give weight by lb per foot.
noswap	Do not allow ArchiFrame to make the plank go from left to right or bottom to top. Useful for example for weatherboard pieces having profile.

15.2.1 Material profile <material>/<profile>

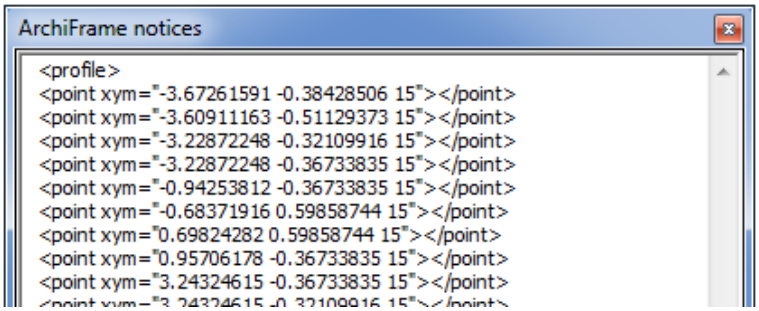
The default material list contains material XX 204x275 which contains profile.

```
<material id="XX 204x275" name="XX 204x275" thickness="0.204" height="0.275" shape="block"
maxlen="99" m3factor="0.000">
  <profile>
    <point xym="-0.10200000 0.23900000 15"></point>
    <point xym="-0.10200000 0.02000000 15"></point>
…
    <point xym="-0.10200000 0.23900000 -1"></point>
  </profile>
</material>
```

Profile is picked from fill placed to the origin. Origin defines the position of plank's reference line (bottom middle):



Profile is copied from ArchiFrame notices window and pasted to material's <*profile*>-tag.



Avoid using arcs that exist in the example as they make the 3D window slower. Arcs can be modelled with lines by changing the status code from 15 to 79. This will produce a 3D profile without unwanted lines.

15.2.2 Material lengthwise profile <material>/<profile_xz>

Lengthwise profile makes it possible to model decorative boards. However, there won't be cnc- for any profile definition. The example below contains fills for plank beginning, multiplied middle and end parts. Please note the point selected for picking is at the begin part's lower left corner:

Lengthwise profile is picked as normal profile and is placed under following tags:

```
<profile_xz>
  <beg>
    <point xym="0.00000000 0.27500000 79"></point>
    …
    <point xym="0.00000000 0.27500000 -1"></point>
  </beg>
  <mid>
    <point xym="2.00000000 0.27500000 15"></point>
    …
    <point xym="2.00000000 0.27500000 -1"></point>
  </mid>
  <end>
    <point xym="0.00000000 0.27500000 79"></point>
    …
    <point xym="0.00000000 0.27500000 -1"></point>
  </end>
</profile_xz>
```

16 Element definition file ArchiFrameElements.xml

This file is used to define all element types, element elevation layout and other settings related to placing the planks to the element. All dimensions are given in meters and a decimal separator is shown as a full stop, for example 1.234.

16.1 Settings <setting>

Contains general element related default settings.

16.1.1 Plank related settings <plank>

```
<misc minlen="0.050" idstr="%s-01" sort="xy" topbotunid="1"></misc>
```

- Minlen, shorter planks are not created.
- Idstr, ID string form, %s is replaced with element ID. Default is %s-001.
- Boardidstr, ID string for boards. Format is like idstr.
- Sort, sorting when giving IDs, possible values.
    - YX, assing from bottom to top then from left to right (default).
    - XY, assign from left to right then from bottom to top.
    - GlobXYZ, compare global bound box, first x, then y and finally z.
- topbotunid, if set to 1 top & bottom including double parts will not get the same ID as any other piece even if those are similar (will always have unique ID). Default is 0.

16.1.2 Default settings for new element <newelem>

See section XML-settings for an ArchiCAD element.

16.1.3 New plank's default settings <newelemplank>

Defines settings for original plank. Original plank is the one that is visible in 3D. See section XML-settings for an ArchiCAD element.

16.1.4 Element elevation from front and back side defaults <newelemprojside>

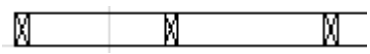
Defines settings for side projection planks. For example, ID is good to be shown here but not at the top projections. See section XML-settings for an ArchiCAD element.

16.1.5 Element elevation from top defaults <newelemprojtop>

Defines settings for top projection planks. See section XML-settings for an ArchiCAD element.

To get the column cross lines also to top projection, add following setting:

```
<objparam name="iColLines">3</objparam>
```



16.1.6 Dimension lines settings <dimlinesettings>

Base settings for every dimension line is read here first. See section XML-settings for an ArchiCAD element.

16.1.7 Pre-settings for dimension lines <dimsettings> ja <dimlines id="id">

This section contains a dimension line definitions used in many element types. The section contains <dimlines id="id"> tags. Xml-attribute id is the id used to refer to the settings, for example,

front side projection uses id "wall_elevation". Definition contains one or more single dimension line (<dimline>).

To use Lua-script for finding pieces to add to the dimension line attribute sript=”xxx” is added:

```
<dimlines merge="addpre" id="maindims" script="yit_projplanks">
```

This is a reference to <script id="yit_projplanks"> in xml-path archiframe/elem/settings/scripts.

16.1.8 Lua script to find relevant pieces to add to the dimension line

Used for maximum flexibility to control the content of the dimension line. An example at xml-path archiframe/elem/settings/scripts:

```
        <script id="yit_projplanks" merge="addpre">
          <![CDATA[
-- Sets ID to middle for horizontal planks
function DoProjSettings()
  local i, z

  for i=1,gnPlanks do
    ac_objectopen(string.format("#%d",i))
    -- Vertical (studs) with different fill
    z=ac_objectget("iEndZ")
    if z and math.abs(z)<0.001 then   -- nil if board
      -- Horizontal piece, ID to middle
      ac_objectset("iIDPlace", 0)
      ac_objectset("iXoffID", 0)
    end
    ac_objectclose()
  end
end

-- Dim line creator uses to find top and bottom plates only
function FindTopBottomPlanks()
  local i, s, n
  local tblRes, tbl

  tblRes={}
  n=0
  for i=1,gnPlanks do
    ac_objectopen(string.format("#%d",i))
    z=ac_objectget("iEndZ")
    if z and math.abs(z)<0.001 then   -- nil if board
      s=ac_objectget("iElemGroup")
      if s and (string.match(s, "^bottom_force%a*") or string.match(s, "^top_force%a*"))
then
        tbl={}
        tbl.index=i
        tbl.anchorfirst=1
        tbl.anchormid=0
        tbl.anchorlast=1
        n=n+1
        tblRes[n]=tbl
      end
    end
    ac_objectclose()
  end
  return tblRes
end
]]>
        </script>
```

Find function returns 1-based table where each item has fields:

- index, the 1-based (projection) plank index.

- anchorfirst/mid/last, overrides to any settings in the xml-attributes.
- anchorfirstskipnext, anchormidskipnext, anchorlastskipnext: Value 1 will not add a dimension to next point from current point.

16.1.9 Lua script to build all dimension lines

Allows script to use sophisticated logic to create only needed dimension lines and to add points using more logic than just for example adding all studs to the dimension line.

```
<dimlines id="dim_wallsect_nowin_horint_10" script="dim_wallsect"
scriptcreate="SectDimLines">
  <!-- Base dim lines could be added before script based -->
</dimlines>
```

As previous but will define everything related to the dimension lines:

- The xml to create dimension line optionally with text
- Dimension points to add

The dimension line creation script returns 1-based table each cell containing fields:

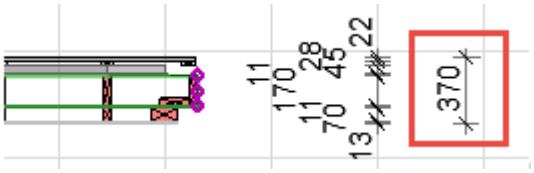
- xmldim, the xml defining the dimension line
- dimpoints, 1-based table of dimension points each containing one of the two options:
  - index, anchorfirst, anchormid, anchorlast as in previous section
  - x, y The static point to add to the dimension line

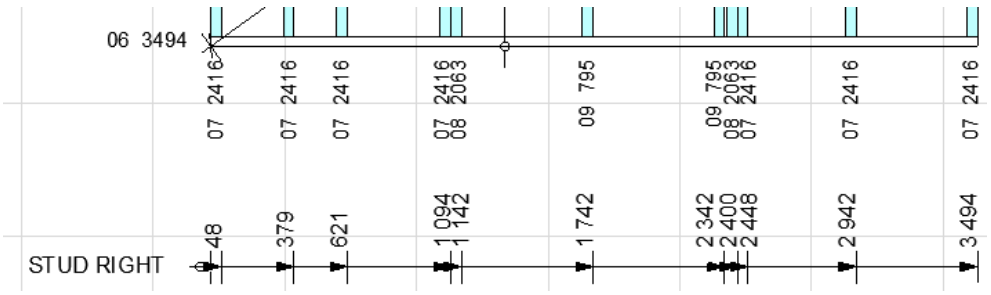
Please search for text SectDimLines in default data folder's ArchiFrameElements.xml for details (C:\ArchiFrame\data\ArchiFrameElements.xm).

16.1.10 Single dimension line settings <dimline>

```
<dimline side="bottom" margin="1" elemvecx="1" elemvecy="0" elemvecz="0" addminmax="1"
addx="0" addy="0" addz="0" anchorfirst="0" anchormid="0" anchorlast="0" exclude="extstud">
  <dimlinesettings>
    <elemparam name="fontsize">2</elemparam>
    <text content="STUD MID" textid="10" align="6" marg="0.2">
      <elemparam name="pen">11</elemparam>
      <elemparam name="fontname">arial</elemparam>
      <elemparam name="fontsize">2</elemparam>
      <elemparam name="fontstyle"></elemparam>
    </text>
  </dimlinesettings>
</dimline>
```

Xlm-attribute	Description
side	Where to place the dimension line from the projection/elevation. Value is one of "left","right","bottom","top".
margin	Marginal between projection and dimension line in meters.
marginproj	An alternative to margin=xxx, will place the dimension line given distance from the projection. ArchiFrame expands total bound box with the resulting dimension line.
elemvecx,y,z	Dimension line direction in the element coordinates. Note! In wall element Y-axis goes up, Z-coordinate is along the watching direction. Length of the direction vector must be 1.
scriptfind	The function name in the referenced Lua-script used to find planks relevant for this dimension line. For example: scriptfind = "FindTopBottomPlanks".

addminmax	The element's contour line minimum and maximum position are added to the dim line. For horizontal the left and right side are added and for vertical the top and bottom sides. Special values: • 2, add just min. • 3, add just max. To get total thickness like below, following values are used:  • 11, add total composite min&max. • 12, add just min. • 13, add just max. Related attributes: • addminplankmindist = "0.050" adds dimension point only if the closest plank dimension point is further that 50 mm. For min end. • addmaxplankmindist = "0.050" for max end. Other way round (do not add element edge dim if plank is closer): • addminplankmindistfrom = "0.050" adds dimension point for plank only if closest plank dimension point is further that 50 mm from end. For min end. • addmaxplankmindistfrom = "0.050" for max end.
addopeningmin	For horizontal dim line: Add opening's left side to the dim line. For vertical dim line: Add opening's bottom side to the dim line.
addopeningmax	For horizontal dim line: Add opening's right side to the dim line. For vertical dim line: Add opening's top side to the dim line.
addsidesminmax	Add first and last point from the element sides. For vertical dim line both left- and right-hand side corner points are added to the dim line.
addx,y,z	Direction of the planks to be added to the dim line. Direction is defined in element's local coordinates. For example, in a wall element the vertical studs have direction addx = 0 addy = 1 addz = 0 and horizontal planks have direction addx = 1 addy = 0 addz = 0. Z-direction is probably never used in wall elements.
anchorfirst	Is the plank's first side added to the dim line. Useful for example if dimensioning from element's left side to plank's left side.
anchormid	Is the plank's middle point added.
anchorlast	Is the plank's last side added.
limitdist	To have different dim lines for studs at bottom and top. The maximum distance from plank's end to element polygon contour line.
minpoints	If defined, there must be at least given number of points for at dimension line to be created.
exclude	Exclude planks and element layers on given type = “xxx” layer. For example exclude = “extstud”. `<layer ref="WALL 19" anchorname1="Ext studs ext" type="extstud" studs="core">`

include	To be used with exclude = “*”, include layer types to handle here. Also must be given to include boards: include = “*boarding*”.
includeelem	To include only element object, if given, overrides include when processing element objects.
onlyelem	If given onlyelem = "1", no planks/boards are included in the dimension line – just the *ArchiFrameElement*-object making up the layer. Useful to create layer offset dimension lines.
textrot	Setting textrot = "90" will produce following result (angle is global angle):  Please note that this setting causes the text to be manually placed and ArchiCAD/ArchiFrame will not clean up overlapping texts – that will remain as manual work. Also notice that dimension text layouts are scale specific and the dimensions are rotated only in the scale the dimension line was originally created. To force recreating the dimension line it must be deleted and recreated by element tools/*Update*-command.
textrotoff	May be used with textrot-attribute. Defines the distance to move the text to left from default position. If not specified, the default value of 75 mm is used.
textrotanchor	If given, the dimension text anchor point is: APILbl_MiddleAnchor= 0 APILbl_TopAnchor= 1, APILbl_BottomAnchor= 2, APILbl_Underlined= 3 (default)

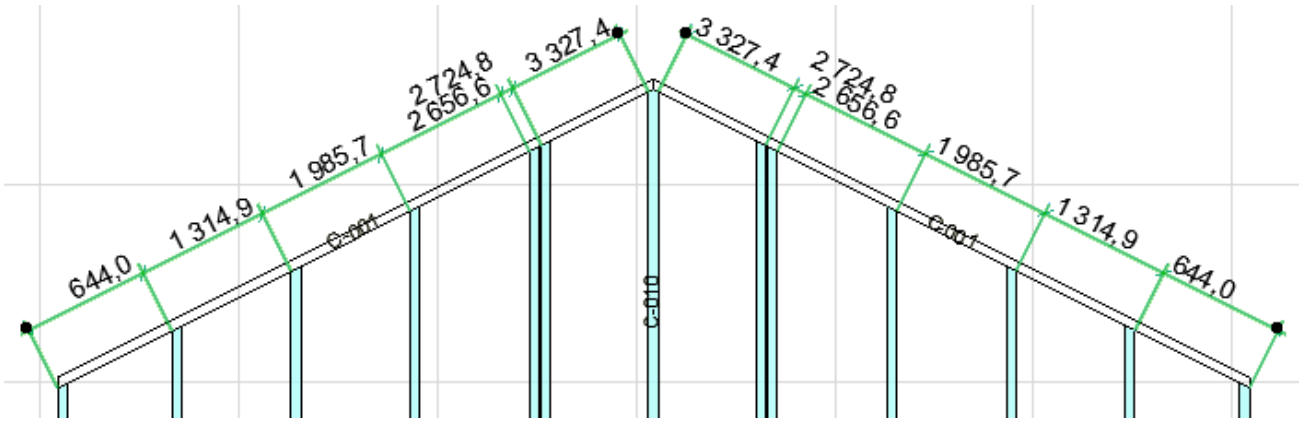
Definition may contain Settings <setting> inside <*dimlinesettings*>-tag that affect just this single dimension line. In addition it may contain text related to the dimension line. Text-element may contain attributes:

Xlm-attribute	Description
content	Text.
textid	Text ID that must be unique inside the whole element drawing. The ID is used to detect unchanged dimension lines during update. Value is from range 1-254. Value can be given as an expression that uses variable for projection number. For example, `textid="projid*20+4"`.
anchor	Text anchor point and placement relative to the dimension line. For example value 6 places text to the left side of the dimension line and the text extends to left from the anchor point. Anchor points are: 123 456 789
marg	Distance to the dimension line in meters.

16.1.11 Dimension line for gable wall top planks

```
<!-- Dimension line for gable wall top tilted planks -->
<dimspec type="toptilted" margin="0.3" addminmax="2" anchorfirst="1" anchormid="0"
anchorlast="0" ltor="0">
  <dimlinesettings>
    <elemparam name="dimensiontype">cumulative</elemparam>
  </dimlinesettings>
</dimspec>
```

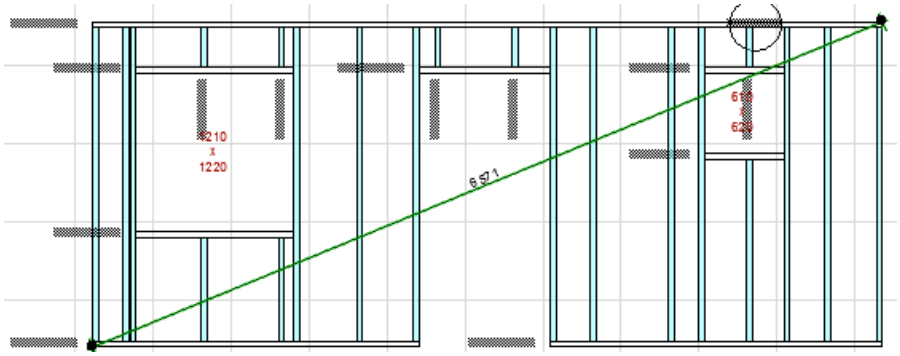
With *dimspec*-tag and attribute type="toptilted" it is possible to get the dimension lines like below. With attribute ltor="1" the dimension line is always from left to right. Otherwise it is from bottom to top.



16.1.12 Cross dimension line

```
<!-- Cross dimension for the element -->
<dimspec type="cross" anchor="lb">
  <dimlinesettings>
  </dimlinesettings>
</dimspec>
```

With *dimspec*-tag and attribute type="cross" it is possible to get cross dimension line for the rectangular shapes – other shapes must be dimensioned manually. Attribute anchor can have values "lb" (left bottom) and "rb" (right bottom).

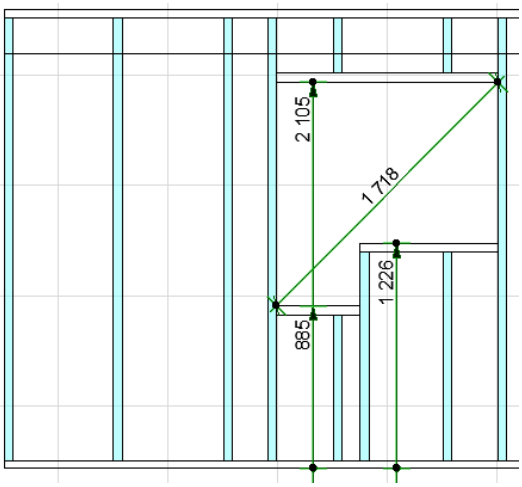


16.1.13 Openings dimension lines

```
<dimspec type="opening" include="core" vertdim="1" crossdim="1" vertanchorbot="top">
  <dimlinesettings_vert>
    <elemparam name="dimensiontype">cumulative</elemparam>
    <elemparam name="markertype">5</elemparam>
  </dimlinesettings_vert>
  <dimlinesettings_cross>
  </dimlinesettings_cross>
```

```
</dimspec>
```

With *dimspec*-tag and attribute type = “opening" it is possible to add individual dimension lines for each opening. Attributes *vertdim* and *crossdim* are used to enable/disable corresponding dimension lines.



Xlm-attribute	Description
include	Defines which class (es) of the planks visible in the projection are used to define the openings.
vertdim	Create vertical dimension 0/1.
crossdim	Create cross dimension 0/1.
vertanchorbot	Where to anchor the bottom of vertdim: default is bottom of bottom plate, value top anchors to top of the bottom plate.
vertanchortop	If missing (default), there is no dim line to the element’s top plate. If given, these values can be used: • "top", add dim point to topmost top plate point • "bottom", add dim point to lowest top plate point
vertopeningbot	-1=anchor to lower horizontal piece’s bottom side 1= anchor to lower horizontal piece’s top side (default)
vertopeningtop	-1=anchor to upper horizontal piece’s bottom side (default) 1= anchor to lower horizontal piece’s top side

16.1.14 Common marking settings to be used in multiple places <elemmarkings>

These settings are referred from element projections and settings are given here to avoid copy & pasting the same settings many times.

```
<!-- For element creation, markings in the element projections -->
<elemmarkings>
  <elemmarking id="mark_opening">
    <opening text="[width]\nx\n[height]">
      <elemparam name="pen">8</elemparam>
      <elemparam name="fontname">Arial</elemparam>
      <elemparam name="fontsize">1.5</elemparam>
      <elemparam name="fontstyle"></elemparam>
      <elemparam name="just">center</elemparam>
    </opening>
    <weight libname="{FD6C137D-C647-42FD-A904-181281909443}-{9060078A-0F41-48C8-8400-
8E414A838053}" create="1" text="(weight) KG" units="kg" movey="0.6">
      <settings>
        <layer>AF Markings</layer>
        <objparam name="iFill">*25*</objparam>
```

```
        <objparam name="A">0.250</objparam>
        <objparam name="B">0.325</objparam>
      </settings>
    </weight>
  </elemmarking>
</elemmarkings>
```

For openings following texts are replaced:

- [id], all doors and windows located in the element hole are collected here. There may be, for example, many small windows next to each other. If there is a script defined, [id] refers to script defined text.
- [width], [height] size of the hole in the element in setting's defined units.

For weight marker the attributes are:

Xlm-attribute	Description
libname	The library part object to show the weight
create	Can be used for example in element templates. If missing or value is 1, the weight marker will be created. Possible values are: • 0, do not calculate. • 1, show the real position of the center of gravity. • 2, show above the elevation. • 3, show above the elevation and show separate value calculated. Without doors&windows if different.
text	The text to show, (weight) will be replaced by the actual height
units	kg or lb
movey	Offset for marker's y-coordinate

16.1.15 Common default settings to be used in multiple places <pre-settings>

These settings are referred from element projections and settings are given here to avoid copy & pasting the same settings many times. Settings are referred with tag name, in this case with xml-attribute settingsref = "cutlist":

```
<presettings>
  <!-- Used for cutlists -->
    <cutlist tab1="0.5" tab2="0.8" tab3="1.2" tab4="1.5" tab5="1.8" boards="1" sortby="len"
plankid="-" boardid="#">
    <elemparam name="pen">11</elemparam>
    <elemparam name="fontname">Arial</elemparam>
    <elemparam name="fontsize">4</elemparam>
    <elemparam name="fontstyle"></elemparam>
</cutlist>

  <!-- Bottom wood markings -->
  <markings id="mark_default">
    <bottomwood>
      <stud line="3" id="2" idline="1" markxpos="2" markypos="1" marksize="0"
marktextdir="1"></stud>
      <opening line="1" windowtext="Window [width]x[height]" doortext="Door
[width]x[height]"></opening>
      <neighbour begtext="&lt;[id]" endtext="[id]&gt;" marksize="0"
marktextdir="1"></neighbour>
    </bottomwood>
```

```
  </markings>
</presettings>
```

<*cutlist*>-tag may have following xml-attributes (<cutlist tab1 = "0.6" tab2 = "0.9" boards = "0" sortby = "id">):

Xlm-attribute	Description
tab1	Tab stop for number of similar pieces in meters.
tab2	Tab stop for start of IDs, also tab3, tab4 …tab10 can be given.
plankid	Separator character for plank ids: • Missing or empty, use full plank ID (default). • Separator character -, include just the number part. • “No”, no plank IDs.
boards	Values: • “0”, boards are not included into cut list. • “1”, boards first then planks (default). • “2”, planks first then boards.
boardid	Include board ids to the cut list, possible values: • Missing or “no”, do not include (default). • Separator character #, include just the number part. • Empty, full ID.
sortby	As default the planks are sorted by length, sortyby=”id” changes this to ID.
exclude	Exclude planks on given type = “xxx” layer. For example, exclude = “extstud”. <layer ref=”WALL 19” anchorname1=”Ext studs ext” **type=”extstud”** studs=”core”>
matname	How to show material id and name in the cut list, default shows just the material id. String contains parts [name] and [id] which are replaced with the current values for the material. For example, matname = “[name] ([id])”. Used only if material id and name are different.
settingsref	Use settings from given tag.
elemid	Include element id to begin of the cut list text. Default value "1", "0" = do not include element id in the cut list. It may also be free text with tags to replace. For example, elemid = "#id# - #floorname#". See the tags here.
script	Construct whole content after possible elemid with given script, see Lua-script for cutlist.
libpart	Use library part for the cut list. For example to refer ArchiFrameTable, use libpart="{9056D768-6A80-423F-B548-30D0A09D857F}".
libscript	Must be used in conjunction with attribute libpart. Builds up the cut list.

16.1.16 Lua-script for cutlist

There are two options: to build the textual cut list or to build an object containing all needed information. To build the content of the cut list ArchiFrame calls Lua-function DoCutList (nSeqNum). Parameter nSeqNum (0…N-1) tells the sequence number of the cut list for current composite element. For multiple cut lists for an element, ArchiFrame keeps the Lua script in memory during processing single ArchiFrame element. This way the script may keep data in memory as Lua globals between calls to DoCutList ().

16.1.16.1 Textual cut list

Function must return 1-based table of text runs. Each item in the table may contain fields:

- Str, mandatory text. Use \n to add line feeds.
- Sizemul, multiplication factor for the size, default is 1.0.
- Bold, use bold style.
- Italic, use italic style.

16.1.16.2 Library part cut list

It is easiest to use these built-in definitions that pick just part of the pieces in the element:

- cutlist_table_coreall
- cutlist_table_corestuds
- cutlist_table_coreothers
- cutlist_table_extboards
- cutlist_table_extcladstuds
- cutlist_table_intboards
- cutlist_table_intstuds

These are referred for example from own composite element template definitions like below:

```
<group layout="horizontal">
  <cutlist layoutmargins="6.5,0.1,0.1,0.1" settingsref="cutlist_table_corestuds"
layoutminheight="3"></cutlist>
  <cutlist layoutmargins="0,0.1,0.1,0.1" settingsref="cutlist_table_coreothers"></cutlist>
</group>
```

The script fills given new cut list library part object. ac_objectopen(nil) will open the current table GDL-object to process (always new).

Cells may have tags before actual text for formatting. Possible tags are:

- <mulheight number>, multiplication factor for text size. Default=1.
- <merge number>, merge next number cells to current cell in this row.
- <nolines>, no lines for the cell.
- <b>, **bold.**
- <i>, *italic.*
- <u>, underline.
- <left>, left align.
- , center align.
- <right>, right align.
- <vtop>, <vcenter>, <vbottom>, vertical alignment.
- <fillrgb r g b>, background fill color. Values r g b between 0…1, red is <fillrgb 1 0 0>. Or if it contains just single value, then it defines the pen number directly.
- <penrgb r g b>, pen color.

Cut list object has parameter *Manage manually* (iManual in xml-file) that can have these values:

- 0, let ArchiFrame update content and position.
- 1, full manual – ArchiFrame will not to touch it.
- 2, keep position but let ArchiFrame update content. This value is useful if the elevation layout needs hand tuning.

For library part cut list function DoCutList() may return values:

- 0, do not create the list (list is empty)

- 1, normal operation (this is the default option if there is no return value from DoCutList)

Default cut list script cutlist_table_an11 supports giving settings from Lua global table gListSettings. It allows customizing the cut lists much further. This customization script gets environment set by the default cut list script, default one gives these:

- gLang, three letter language code: "eng", "fin" etc.

Table gListSettings may have following fields with the default cut list script:

Field	Description
title	Title for the cut list. Following replace-strings may be used: • #id# will be replaced by the element ID
funccollect	Function that defines if a piece should be included into the cut list. Function has single parameter: Data of the collected piece. Content of the data depends on the cut list script – default data is described in the following table.
addmc	Add machinings column to the table: 0/1
addlen	Add length column, for boards this should be 0.
cladaddmat	Add cladding material name to cladding piece ID and include it in grouping
col1str_plank	Replace first column's text with this text. Parts to be replaced: • #id#, plank's ID • #matwidth#, typically smaller measure • #matheight#, typically bigger measure • #len#, plank's length

Default cut list generator fills these fields for every collected part (can be used in the *funccollect*):

Field	Description
elemtype	Owning element layer's type: core, intstud, extstud etc.
elemtypesort	Internal coding that defines the part's role in the element. Possible values and the default names are: • 001, STUD • 002, OTHERS (just core layer non-vertical piece) • 010, INT STUDDING • 020, EXT STUDDING • 030, CLADDING • 040, BOARD EXT • 050, BOARD INT • 999, OTHERS (unknown layer type core/intstud/extstud etc)
id	Part's ID
plankinfo	af_request ("plankinfo") for the part
more…	Please see function CollectPlanksBoards() for more fields.

16.1.17 Lua-scripts to be used later <scripts>

Instructions are in section Lua-scripts with elements.

16.2 Element type <elemtypes> and <elemtype>

Element types are listed with tag <elemtype> inside container tag <elemtypes>. Content of the <elemtype> tag is <elemtype id = "short_id" name = "name for user">. Xml-attributes:

Xlm-attribute	Description
id	Element type ID that remains the same even if the language changes.
name	Element type name for the user. If missing, id is used.
idstamp	Text to put to element stamp at bottom of the elevation.
tolist	With value idstamp = "0" the element type is not added to the element tool palette.
density	Defines the insulation, boarding or panelling density in kg/dm3. Default value is 0.050 (50 kg per m3).

After this comes <*settings*>-tag that may contain settings to override common <settings>-part, for this element type.

Also, it is possible to set some options on as default by giving <*options*>-tag like below:

```
      <elemtype class="wall" id="WALL 173 K600" idstamp="42x173">
        <!-- Default settings for this element type -->
        <settings>
          <newelem>
            <elemparam name="pen">1</elemparam>
          </newelem>

          <newelemplank>
          </newelemplank>
          <newelemprojside>
          </newelemprojside>
        </settings>

        <!-- Could set some options on here as default, settings may be given inside
<option>settings</option> and format depends on the option -->
        <options>
          <option id="openingsides">
            <![CDATA[
mat=block
thickness=0.050
height=0.173
]]>
          </option>
        </options>
```

16.2.1 Element weight calculation

It is possible to give the weight definitions also to old style xml-definitions (instead of using own element type definitions which produce this kind of xml-definition):

```
<elemtype class="wall" id="WALL 173+42 PANEL OUTIN" idstamp="173+42">
  …
  <!-- Default element weights for this type -->
  <elemweight>
    <bools calc="0" door="1" win="1"></bools>
    <floats framingkg="400" boardingkg="1150" insulationkg="35" doorm2kg="10"
doorframekg="4" winm2kg="15" winframekg="4"></floats>
  </elemweight>
```

The attributes for bools are:

Xlm-attribute	Description
calc	Calculate weight for the element 0/1.
door	Calculate doors 0/1.
win	Calculate windows 0/1.

forceframing	Force framing weight to value given in floats-part (instead of using weights given in ArchiFrameBlocks.xml).
forceboarding	Force boarding weight as for framing.

And for floats:

Xlm-attribute	Description
framingkg	Framing weight in kg per cubic meter.
boardingkg	Boarding weight in kg per cubic meter.
insulationkg	Insulation weight in kg per cubic meter. Value 0 is good if it is air space.
doorm2kg	Weight for door area kg per square meter.
doorframekg	Weight for door frame kg per frame meter including bottom.
winm2kg	Weight for window area kg per square meter.
winframekg	Weight for window frame kg per frame meter including bottom.

Default weights set internally inside ArchiFrame are like this:

```
  <elemweight>
    <bools calc="1" door="0" win="0"></bools>
    <floats framingkg="450" boardingkg="1100" insulationkg="30" doorm2kg="15"
doorframekg="5" winm2kg="20" winframekg="5"></floats>
  </elemweight>
```

Then the weight settings are searched from each construction layers' *<elemweight>*-tag. Composite structure cannot have weight settings.

16.2.2 Element layers <layers> and <layer>

Now only the core is modeled. Layers can be defined to allow easy anchoring of the element's gypsum board inner surface with ArchiCAD wall side. Layers are listed starting from exterior.

```
<layer name="Core" visible="1" thickness="0.195" type="core" anchorname1="Core ext"
anchorname2="Core int">
```

Xlm-attribute	Description
name	Layer name that is visible in element tool's anchor list.
visible	For future.
thickness	Layer thickness.
type	Core must be specified with setting: type = "core".
anchorname1	Name of the layer's exterior side to the anchor list.
anchorname2	Name of the layer's interior side to the anchor list.

16.2.3 Planks to the layer <planks>

Defines single plank group. Group has a name that is used to for example to adjust plank groups together, creating grooves and other machinings.

```
<!-- Group name will have suffixes: _y=vertical planks, _x=horizontal planks -->
<!-- Group names can be used for example to make grooves for balks and groups may be
referred using wild cards: vertical* -->
<planks group="vertical" axis="y" opening_parallel="1" opening_normal="1" spacing="0.6"
spacingtolerance="0.01" gablehipsplit="0">
  <material id="L123" zoff="0.000" rotangle="0">
    <objparam name="iUsageId">XXX</objparam>
    <elemparam name="pen">8</elemparam >
  </material>
  <force>
```

```
    <parallel x1off="0.141"></parallel>
    <parallel x2off="-0.141"></parallel>
  </force>
</planks>
```

XIm-attribute	Description
group	Group name, ArchiFrame will add postfix at the end of given name (for example "vertical_y"): • _y, plank in Y-axis. • _x, plank in X-axis. • _opening, inclined edges in openings. • _force, forced planks. • _spacing, planks added due to spacing rule.
axis	Planks base direction in element's coordinates, either "X" or "Y" or "unused" to process unused contour edges. If used value axis="unused", two more attributes are available: • contour_unused = "1", handle contour line (default = 1). • opening_unused = "0", handle holes (default = 0).
skipaxis	With value "1" there will not be planks in the element's base direction.
opening_parallel	• "0", do not add. • "1", add planks to axis-direction edges in openings extending to the polygon. • "2", do not extend (useful for horizontal studding).
opening_normal	As previous but axis is rotated 90 degrees from the base direction.
spacing	Spacing distance in meters.
spacingtolerance	How much for example a stud position may differ from the spacing rule before there will be a double stud. For example, value 0.01 will result in a double stud if existing stud's position is further than 10 mm from the spacing rule.
stackingoffset	By default, the spacing will start from the edge of the element object. For example, with spacing value of 600 mm, the first stud will be placed 600 mm from the element edge. Setting this attribute makes the middle line of the first piece to be placed given distance from element's left/right edge. Axis is the base direction, for vertical (axis = "Y") positive starts from left and negative from right, for horizontal (axis = "X") positive starts from top and negative from bottom. This is useful to get evenly spaced pieces for log kind of structures. Also, it is useful if there is a piece with width 90 mm that should be spread evenly using spacing 100 mm.
gablehipsplit	Value is "0" or "1": Default value "0" will add a stud to every corner at the elements top side. With value "1" there will be two studs halving the given spacing from the corner position to left and right.
create	To be used in element templates. Value is "0" or "1" (default and used if attribute is missing). Value "0" will skip this definition.

16.2.4 Material settings <material>

Similar material settings are used with element and weatherboard settings. It is possible to give settings related to just these planks (see XML-element settings).

```
  <material id="block" thickness="0.015" height="0.075" zoff="0.000" rotangle="0">
    <objparam name="iUsageId">XXX</objparam>
```

```
    <elemparam name="pen">8</elemparam >
  </material>
```

Xlm-attribute	Description
id	Material id, "block" is general type and it also requires giving the material size.
thickness	Material thickness only if id is "block".
height	Material width/height only if id is "block".
xoff,yoff	Move in element's X- and Y-axes.
zoff	Move in element's Z-axis. For example value -0.050 places the plank's reference line 50 mm in front of the current layer.
linexoff	Move to right relative to current line, negative moves to left.
rotangle	Plank rotation angle in degrees. ArchiFrame sets the rotation so that the plank extends towards the watching direction. This value is added to that. For example plank with size 50 x 100 mm is placed with default settings to the layer's front surface and the plank extends 100 mm in the watching direction.

Following variables may be used inside <*material*>-tag:

Variable	Description
mat_thickness	Material thickness as given in ArchiFrameBlocks.xml, for example: `<material id="L128RL" name="21x120 VL" thickness="0.021" height="0.120"`
mat_height	Material height/width.

16.2.5 Forced planks <force>

It is possible to add planks in fixed positions, for example double studs at the left and right-hand sides of the element:

```
  <force>
    <material id="98x48" zoff="0" rotangle="0"></material>
    <parallel x1off="0.141"></parallel>
    <parallel x2off="-0.141"></parallel>
  </force>
```

<parallel> adds planks in element's axis direction and <normal> in the perpendicular direction. Material can be set for the forced planks with <*material*>-tag.

Xlm-attribute	Description
x1off	Plank is placed to the left side of the element relative to current axis. Middle line is moved the given distance to the right.
x2off	As x1off, but relative to the right side.
zoff	As in <material>-defintion.
extend	Either "0" or "1". Value "1" extends the plank to the element's contour.

16.2.6 Boards

```
<layers>
  <boarding thickness="0.013" overlap="1" boardingtype="1">
    <board id="GYPSUM 13x1200x2700" width="1.2" height="2.7"></board>
  </boarding>
</layers>
```

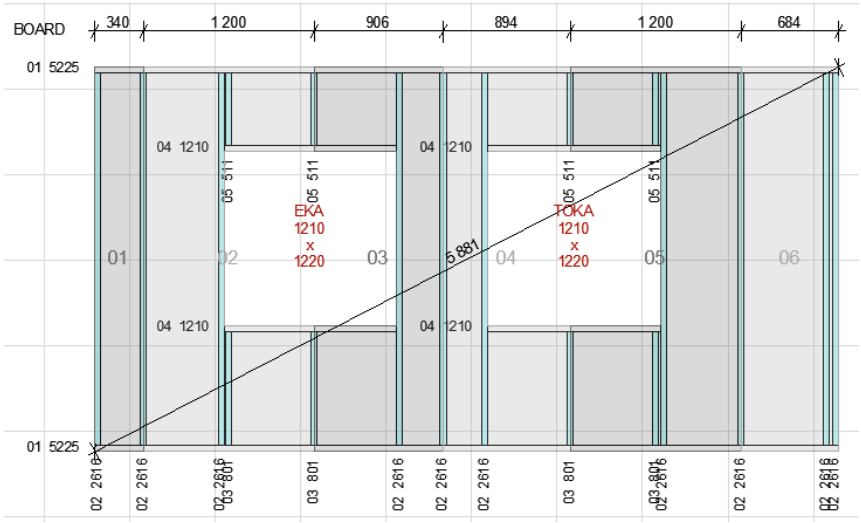
Xlm-attribute	Description

overlap	Affectes multiple layers of boards, values: • 0 or missing, no special processing. Board edges will be at the same place. • 1, avoid board edges at the same stud. If this is set and there is another board layer to process, no other boarding strategies are applied.

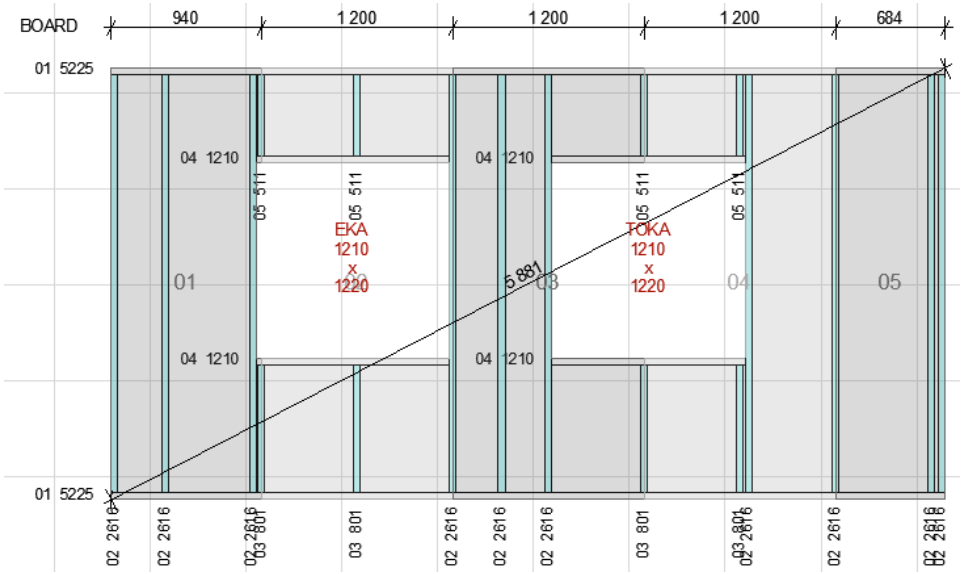
16.2.6.1 Boarding strategies/attribute boardingtype

Studding affects greatly how the boards can be placed. Therefore, the studding layer should have compatible framing rule with the boarding rule.

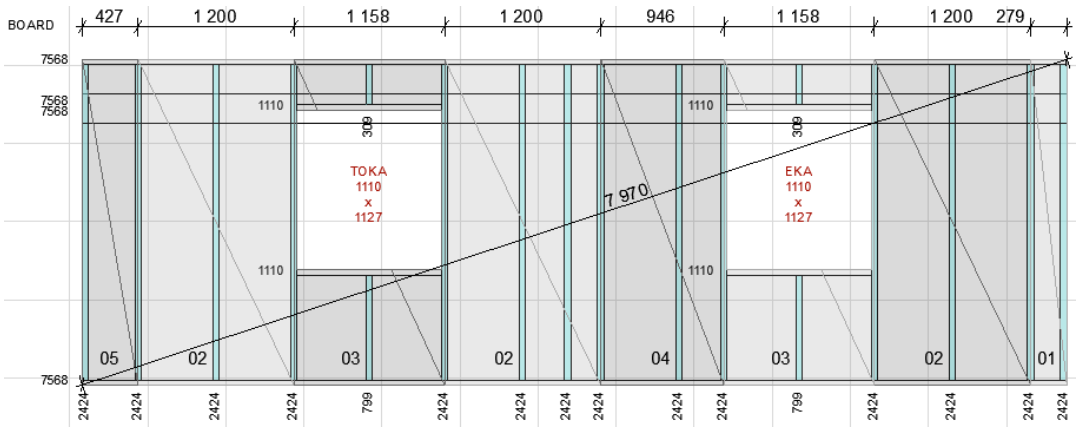
1 or missing (default), no board edge to opening edge.



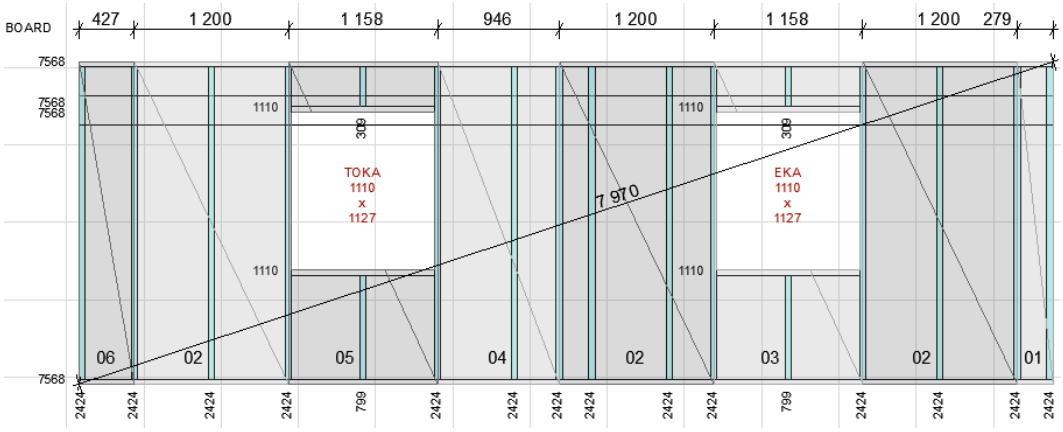
2 starts from left, no special handling for openings.



3 start from left and right-hand sides of each opening and then do the openings.



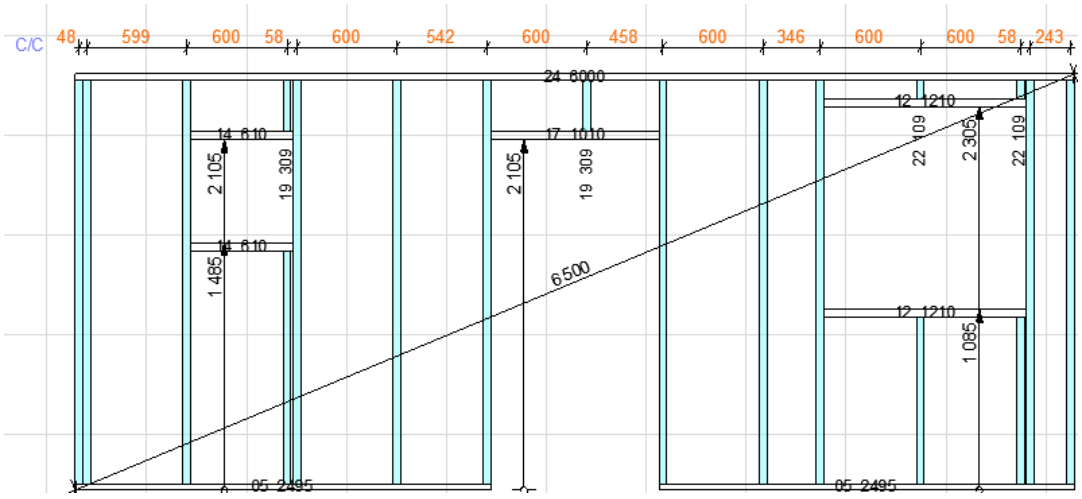
259 (3+256) same as 3 but start processing from right hand side.



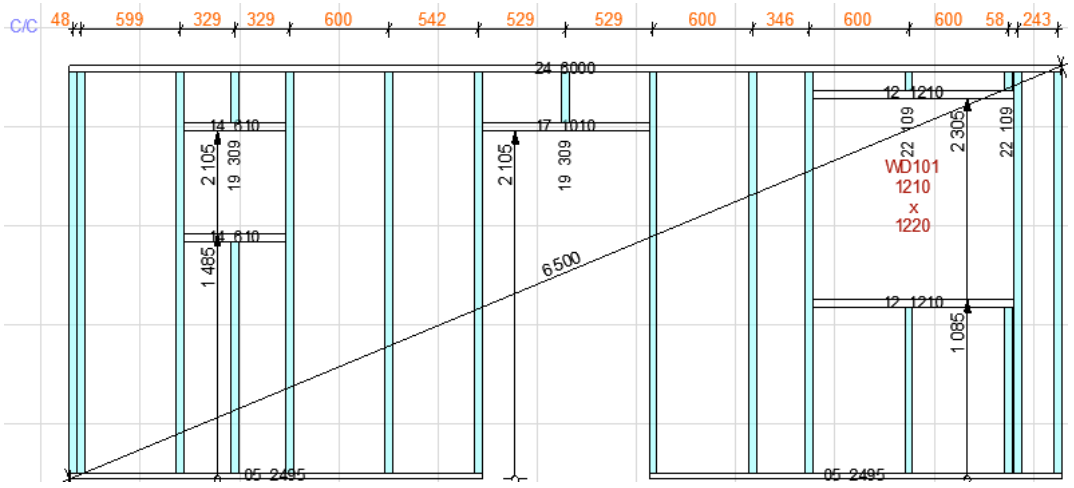
16.2.6.2 Main framing rule/attribute framingrule

Value zero or missing definition uses default rule: Try placing studs from every fixed stud (left & right, opening left & right) and select the one that uses least material.

Value 1, start applying spacing rule from left & right-hand sides of the opening. If there is more than 2 x spacing from the starting point to end of spacing area, the first spacing is spacing – stud width/2:



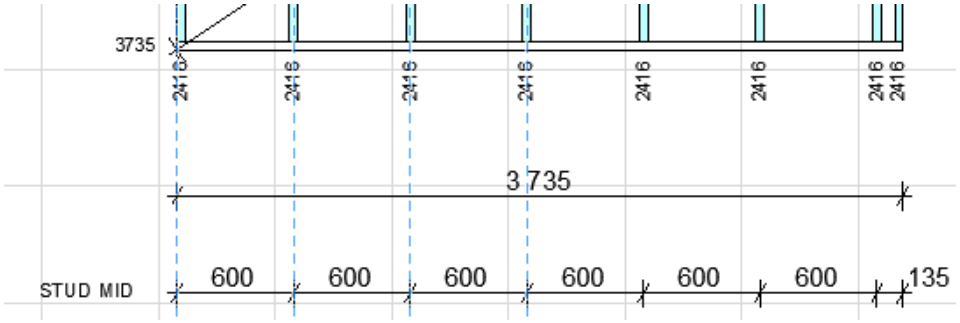
Value 2, same as 1 but place single stud to middle of opening instead of standard spacing if width less than 2 x spacing:



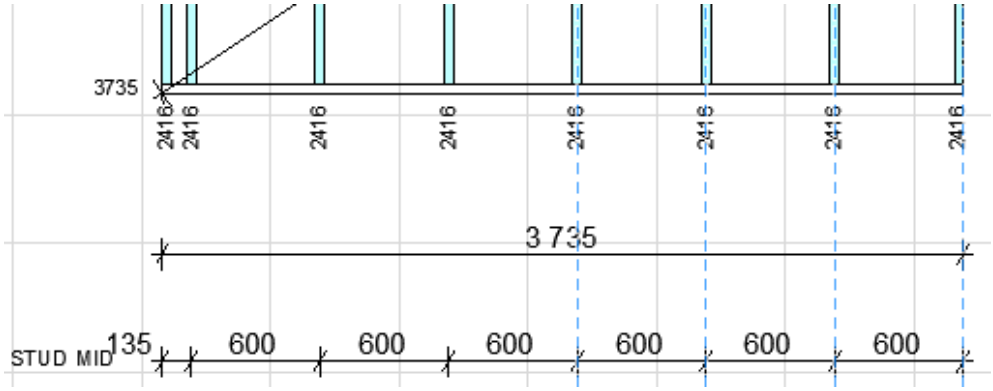
Value 3 is same as 1 but spacing is always from middle of the first stud.

Value 4 is same as 2 but spacing is always from middle of the first stud.

Value 5, start from left and from top for horizontal structures:



Value 6, start from right/bottom:



Value 7 as 5 but spacing starts from middle of the starting piece instead of its edge.

Value 8 as 6 but spacing starts from middle of the starting piece instead of its edge.

16.2.7 Operations between the planks <operations>

In this section the planks are adjusted together and grooves are added for beams. All operations use plank groups to define operators and targets.

```
<joinends target="horizontal*|inclined*" operator="horizontal*|inclined*">
  <joinends conntype="endtoend" jointgap="0.000"></joinends>
</joinends>
```

Xlm-attribute	Description
target	Name of the groups that are operation targets. It is possible to give multiple names using separator |. The name may contain wild cards * and ?, * replaces any text and ? single character.
operator	Operation operator group names.
targetskip	To be used by scripts: Comma separated lists of plank guids/ptrs to exclude from the group. Also group names with wild characters may be used.
operatorskip	The same for operators.

16.2.7.1 Limiting operation with plank connection type <xxx conntype="yyy>

In many operations it is possible to limit the plank connection types. Attribute conntype may have following values:

Value (yyy)	Description
endtoend	Plank reference lines must be connected from ends.
endtoline	Target plank reference line must connect with its end to middle of the operator plank's reference line. Possible additional definitions: • mindisttoend = "meters" defines target end's minimum distance from end of operator plank ends. • disttoend = "meters", as previous but length must match exactly. Useful for weatherboards.
linetoend	Operator plank reference line must be connected from end to middle of target plank's reference line. Additional definitions as in endtoline-connection.
linex	Target plank's reference line must intersect with operator plank. Note! It is ok if target reference line just intersects the operator – the reference lines may not intersect. Possible additional definitions:

	• maxdisttoend = "meters" defines maximum distance from intersection to the end of target plank. • maxdisttoendop = “meters” same as previous but distance from operator piece. Please note that if intersection point is outside the line, the distance is negative and max distance condition is always true.
hasx	The planks have any intersection.

16.2.7.2 Adjusting planks to another planks side <jointo>

Straight cut is made with <*cut*>-tag:

```
<jointo target="target_groups" operator="operator_groups">
  <cut jointgap="0.000" extendmaxlen="0.0" expandopfind="0.0" endshape="angled"></cut>
</jointo>
```

Xlm-attribute	Description
jointgap	Gap between the planks, negative value produces overlapping.
extendmaxlen	Maximum distance between target and operator plank to make the operation.
expandopfind	How much the operator plank is expanded from each side before checking the intersection. Pieces touching just from the corner will not be found unless for example 1 mm (0.001) is given here. 
endshape	Possible additional definition for the cut: • "angled", extend and cut to the operator (default). • "anglednoext", just cut without extending. • "angleddel", as angled but remove all existing cuts. • "angleddelfirst", as previous but delete all existing only for first joint. For next join operations old cuts are preserved (to keep two cuts at hip). • "straightshort", straight cut to first intersection. • "straightlong", straight cut to last intersection.
toend	Cut surface: • 0, trim to first intersecting surface (default). • 1, trim to further surface.
removepart	Remove part: • "up", remove upper part. • "down", remove lower part.
conntype	See Limiting operation with plank connection type <xxx conntype="yyy>.

Mortise and tenon joint is done with <*mortisetenon*>-tag:

```
<jointo target="targets" operator="operators/female parts">
  <mortisetenon len="0.056" bordertop="0.02" borderbot="0.005" borderside="0.01"
roundingr="0.0225" endgap="0.001" sidegap="0.001" topgap="0.010"></mortisetenon>
</jointo>
```

Dovetail with <*dovetail*>-tag:

```
<jointo target="targets" operator="operators/female parts">
  <dovetail len="0.038" bordertop="0.00" borderbot="0.050" bottomwidth="0.06"
roundingr="0.0225" endgap="0.000" sidegap="0.000" angle="6" ></dovetail>
```

```
</jointo>
```

16.2.7.3 Adjusting ends <joinends>

This operation connects two planks together halving the angle. Both targets and operators are handled the same way.

```
<joinends target="horizontal*|inclined*" operator="horizontal*|inclined*">
  <joinends conntype="endtoend" jointgap="0.000"></joinends>
</joinends>
```

Xlm-attribute	Description
jointagap	Gap between plank ends.
conntype	See Limiting operation with plank connection type <xxx conntype="yyy>.

16.2.7.4 Grooves and balk joint <groove>

This operation is used for example to make grooves to the studs for a beam inside the element.

```
<groove target="vertical_y|vertical_spacing|vertical_force" operator="balktop*">
  <groove overtop="0.000" overbot="0.000" overside="0.000" overlen="0.000"></groove>
</groove>
```

Xlm-attribute	Description
overtop	Oversize at operator plank's top side.
overbot	…bottom side.
overside	…sides, may be given individually with attributes overleft and overright.
overlen	…lengthwise.
forcedepth	To force depth as in groove tool.
force90	To force groove as perpendicular.
forcetargetside	Value 1...6 of forced target surface.
conntype	See Limiting operation with plank connection type <xxx conntype="yyy>.

Narrowed balk (adds machinings to operator plank too):

```
<groove target="vertical_y|vertical_spacing|vertical_force" operator="balktop*">
  <balkborder bordertop="0.02" borderbot="0.005" borderside="0.01" overtop="0.001"
overbot="0.000" overside="0.001" overlen="0.000"></balkborder>
</groove>
```

Xlm-attribute	Description
bordertop	Narrowing depth at operator-plank's top side.
borderbot	…bottom side.
borderside	…sides.
overlen	How much longer the narrowing part is than the related target plank. Other oversizes as in <groove>-operation.

16.2.8 Projections <projections>

Please see XML-file for details. Attribute layoutmargins defines marginal in order: Left, right, top, bottom.

```
<projections>
  <group layout="horizontal">
    <group layout="vertical">
      <!-- Only planks belonging to the element are included in cutlist -->
      <!-- layoutalign: -1=left/bottom, 0=center, 1=right/top -->
```

```
      <cutlist layoutalign="-1" layoutmargins="0.1,0.1,0.1,0.1" layoutminwidth="2.0"
layoutminheight="7.0" settingsref="cutlist"></cutlist>
    </group>
    <group layout="vertical">
      <projection type="front" layoutmargins="2,2,2,2">
        <dimlines>
          <dimline ref="wall_elevation"></dimline>
        </dimlines>
        <elemmarkings>
          <opening ref="mark_opening"></opening>
        </elemmarkings>
      </projection>
      <projection type="top" layoutmargins="2,2,0,2">
        <dimlines>
          <dimline ref="wall_top"></dimline>
        </dimlines>
      </projection>
      <projection type="back" layoutmargins="2,2,0,2">
        <dimlines>
          <dimline ref="wall_elevation"></dimline>
        </dimlines>
      </projection>
    </group>
  </group>
</projections>
</elemtype>
```

Xlm-attribute	Description
layoutalign	Obsolete.
valign	Align vertically in the parent: • -1, top (default). • 0, center. • 1, bottom.
halign	Align horizontally in the parent: • -1, left (default). • 0, center. • 1, right.
type	For projection tag, possible values: • front • back • top • bottom • topflipx (will mirror X-coordinates of the projection). • bottomflipx (will mirror X-coordinates of the projection).

<*projection*>-tag may contain settings for specific layer's projection plank objects with syntax:

```
<projection type="front" layoutmargins="2,0.5,0,1" exclude="*ext*"
boardpanels="boarding_int*">
  <dimlines>
    <dimline ref="wall_elevation"></dimline>
  </dimlines>

  <projplanksettings type="core">
    <elemparam name="linetype">*dash*</elemparam>
    <objparam name="iFill">0</objparam>
    <objparam name="iShowID">0</objparam>
    <objparam name="iShowLen">0</objparam>
```

```
  </projplanksettings>
</projection>
```

This is useful for example, if the core layer should be shown as trace only. Settings may contain anything described in section XML-settings for an ArchiCAD element.

To make more complicated settings, it is possible to add a Lua script for processing the projection planks. For example, the script at XML-path archiframe/elem/settings/scripts. The function name DoProjSettings() is fixed:

```
        <!-- Sets ID to middle for horizontal planks  -->
        <script id="yit_projplanks" merge="addpre">
          <![CDATA[
function DoProjSettings()
  local z

  for i=1,gnPlanks do
    ac_objectopen(string.format("#%d",i))
    -- Vertical (studs) with different fill
    z=ac_objectget("iEndZ")
    if z and math.abs(z)<0.001 then   -- nil if board
      -- Horizontal piece, ID to middle
      ac_objectset("iIDPlace", 0)
      ac_objectset("iXoffID", 0)
    end
    ac_objectclose()
  end
end
]]>
        </script>
```

And the reference to it:

```
<projection type="front" layoutmargins="2,0.5,1,1" exclude="*boarding*,*stud*,*finish*">
  <projplankscript ref="yit_projplanks">
  </projplankscript>
```

16.2.9 Projections/static texts

It is possible to include title/information texts inside <group>-definitions. For example, to add ID outside the print frame and text LOOKED FROM OUTSIDE to position X = 3, Y = -0.05 from current's group's top & left coordinates.

```
<group layout="vertical">
  <text content="#id#" anchor="7" x="0" y="0.1" update="1">
    <elemparam name="pen">1</elemparam>
    <elemparam name="fontname">arial</elemparam>
    <elemparam name="fontsize">30</elemparam>
    <elemparam name="fontstyle"></elemparam>
  </text>
  <text content="LOOKED FROM OUTSIDE" anchor="1" x="3" y="-0.05">
    <elemparam name="pen">1</elemparam>
    <elemparam name="fontname">arial</elemparam>
    <elemparam name="fontsize">1.5</elemparam>
    <elemparam name="fontstyle"></elemparam>
  </text>
```

Xml-attribute	Description
content	Text to add, following texts are replaced: • #id# by the element id, for example EW01. • #weightkg# by element weight in kilograms.

	• #weightlb# by element weight in pounds. • #elemtype# by composite element's type ID. • #floornum# by the home floor number. • #floorname# by the home floor name.
anchor	Text anchor point: 1 2 3 4 5 6 7 8 9
x,y	Offset to place text from current position.
update	By default, these static texts are created once – they are not moved, nor is content updated afterwards. By setting this to value 1, the position and content will be updated when element is updated.

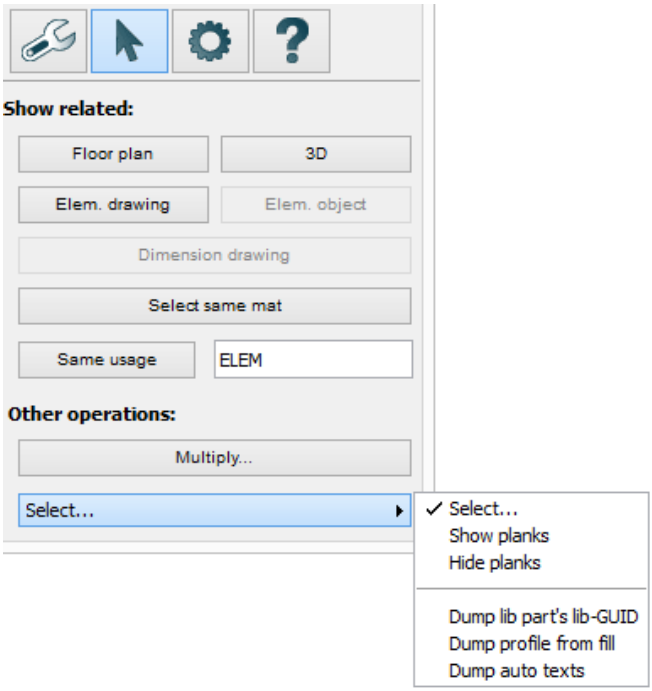
16.2.10 Projections/element stamp <projections>

It is possible to use any kind of stamp object in the element drawings. ArchiFrame library contains at least single stamp that can be slightly customized. As default the stamp looks like:

ID	Type	Project	Project num
A	50x100	AF Sample	123456
	Date 2013-12-17	Designer #CAD Technician	

The field names and values all come from ArchiFrameElements.xml. The default definition is:

```
<elemstamp layoutalign="-1" layoutmargins="-2.2,0,0,0" layoutminwidth="10.0"
layoutminheight="1.0" libname="{9FD03D30-6A12-4B0F-AB3A-01410A529E38}-{D420792D-C066-42CF-
988C-9596DD164EB7}" settingsref="elemstamp">
  <script ref="elemstamp"></script>
</elemstamp>
```

Xlm-attribute	Description
libname	Library part's name, please use GUID that you can get from ArchiFrame main tool palette/Selecion/Other operations: 

settingsref	Refers to common settings xml-tag in elem/settings/presettings.
script ref	Refers to xml-tag elem/settings/scripts. With element scripts multiple references may be given using attributes ref = “first” ref1 = “second” ref3 = “third”…

It is recommended to set the stamp field names in settings xml and the stamp parameters in Lua-script. For example the default settings for field names:

```
<!-- Used for element stamp -->
<elemstamp>
  <elemparam name="pen">1</elemparam>
  <objparam name="iFont">Arial</objparam>
  <objparam name="A">10</objparam>
  <objparam name="B">1</objparam>

  <objparam name="iName1">ID</objparam>
  <objparam name="iName2">Type</objparam>
  <objparam name="iName3">Date</objparam>
  <objparam name="iName4">Project</objparam>
  <objparam name="iName5">Designer</objparam>
  <objparam name="iName6">Project num</objparam>
  <objparam name="iName7"></objparam>
</elemstamp>
```

The default script sets field *Date* and *Designer* only if those are empty. To see available auto texts, open ArchiFrame main tool palette/Selecion/Other operations (picture above). To change the automatic values, select the stamps for example using ArchiCAD Find & Select and open object settings.

Please see default ArchiFrameElements.xml for all the details.

16.2.11 Markings into planks <markings>

```
<markings id="mark_big">
  <bottomwood setcnc-="0">
    <stud line="3" id="2" idline="1" markxpos="2" markypos="1" marksize="5" marktextdir="1"></stud>
    <opening line="1" windowtext="Vindu [width]x[height]" doortext="Dør [width]x[height]"
marksize="5"></opening>
    <neighbour begtext="&lt;[id]" endtext="[id]&gt;" marksize="10" marktextdir="1"></neighbour>
  </bottomwood>
  <topwood setcnc-="0">
    <stud line="3" id="2" idline="1" markxpos="2" markypos="1" marksize="3" marktextdir="1"
forceside="0"></stud>
  </topwood>
</markings>
```

Specifies markings to the planks in the element. Markings are set whenever the element is updated. Attribute setcnc- controls whether plank's cnc-parameter is set to 1 if there are markings added. Default value is 1. Currently it is possible to mark only the top and bottom wood (connecting elements, studs and openings):

Tag	Description
stud	Stud markings, attributes: • line, 1 = mark left, 2 = mark right, 3 = mark both. • id, 1 = mark whole id, 2 = strip elem id away, just the plank number. • idline, 1 = left hand side, 2 = right hand side, 3 = let ArchiFrame calculate. Position not under any stud. • markxpos, justify 1 = left, 2 = middle, 3 = right. • markypos, justify 1 = bottom, 2 = middle, 3 = top.

	• marksize, text size in centimetres, 0 = default. • marktextdir, 0 = in plank dir, 1 = in line dir. • forceside, 0=automatic target side, 1-6=force marking to given side
opening	Mark openings, attributes: • line, 0 = no line, 1 = line covering the opening. • windowtext, text for window, [width] and [height] are replaced with AC locale aware dimensions. • doortext, the same for doors.
neighbour	Mark connecting elements (only end connections): • begtext, text to connection at plank begin, [id] will be replaced with connecting element id. • endtext, the same for plank end. • mark*, the same as for studs.

16.2.12 Element options, <options>

All element options are defined in <options>-section of the *ArchiFrameElements.xml*–file. Single option definition is like:

```
<option id="openingsides" name="Planks to openings sides" applystate="prespacing"
ordernum="100">
  <script ref="openingsides">
    <![CDATA[
]]>
  </script>
</option>
```

Xlm-attribute	Description
id	Unique id of the option. Not visible to the user.
name	Visible name.
applystate	Specify to alter the phase when the option is applied. Default phase is after creating all element planks. Possible overrides are: • Prespacing, after all element edge pieces have been creates but spacing rule has not been applied yet. Use this if the option moves studs.
autoreset	Specified if other option(s) need to be reset when this one is set. For example autoreset = “openingsides”.
ordernum	Options may depend on each other. This sets the applying order for the options. Smaller number (default = 100) will be applied first.
reapplyupdate	If set reapplyupdate = “1” and option is checked on, the option is first cleared and then set again when the user clicks Update from the element tools. This is useful for example to add markings to the pieces.

<*reapply*>-tags may be given if some options should be reapplied when current one changes. For example, if double top is changed, grooves at top must be reapplied to the lowest top piece and balk top level may change:

```
<option id="doubletop" name="Double top" name_fin="Tuplayläjuoksu" name_nor="Dobbel
Toppsvill" name_swe="Dubbla hammarband">
  <!-- If there is already grotop for the element, remove existing and set after running
this option -->
  <reapply id="grotop"></reapply>
  <reapply id="balktopint"></reapply>
  <reapply id="balktopext"></reapply>
```

The option itself is defined with *Lua*-script, please see Lua-scripts with elements.

17 Examples of using elements and trusses

These examples use model *ArchiFrameElemDemo.pln* which is installed in folder C:\ArchiFrame\samples by default. Please make sure that ArchiFrame-library is loaded (default folder is C:\ArchiFrame\Lib).

There are more recent videos on this topic:

ArchiFrame elements: an overview, https://vimeo.com/178327076.
https://player.vimeo.com/video/178327076

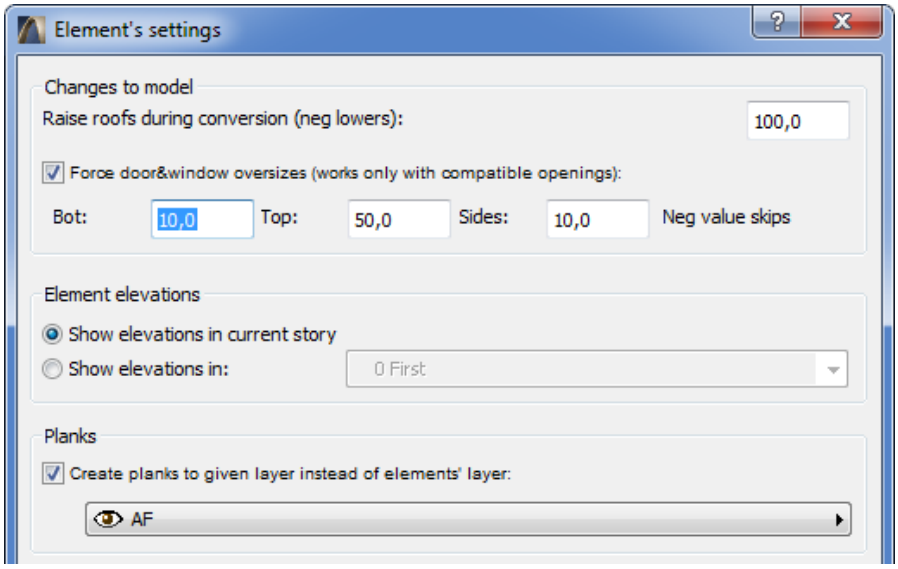
Placing rectangular wall elements, https://vimeo.com/180035580.
https://player.vimeo.com/video/180035580

Placing complex wall elements, https://vimeo.com/181255910.
https://player.vimeo.com/video/181255910

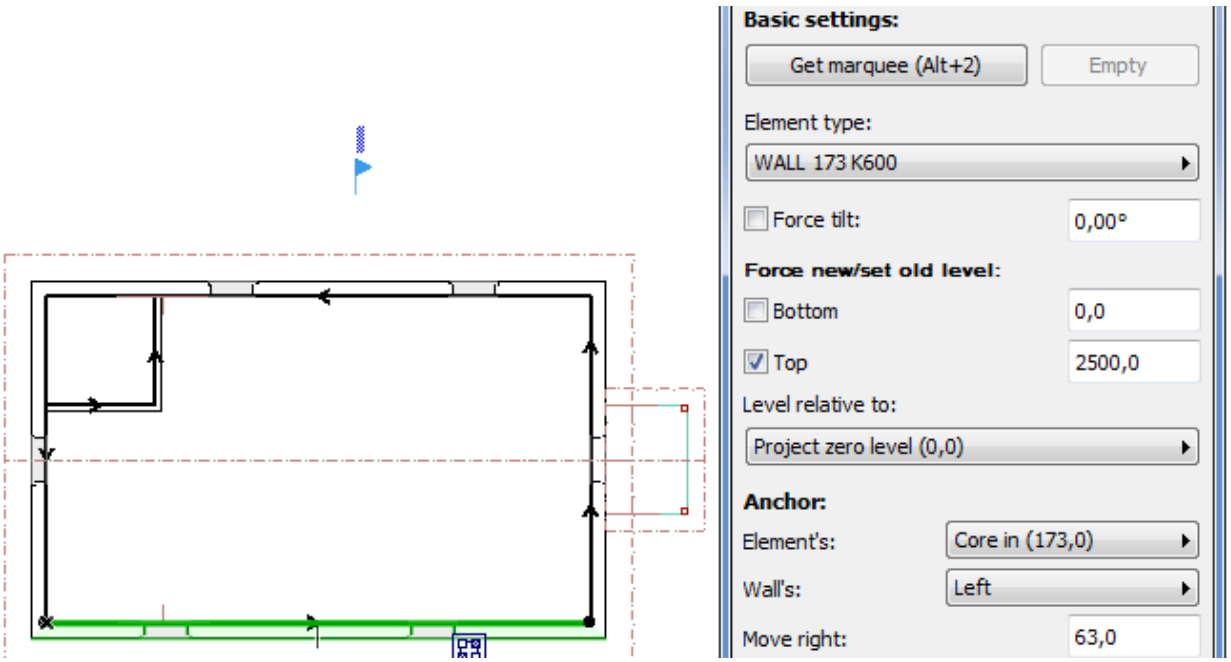
Adding a wall element with steel framing, https://vimeo.com/173754604.
https://player.vimeo.com/video/173754604

17.1 Creating wall elements/frames

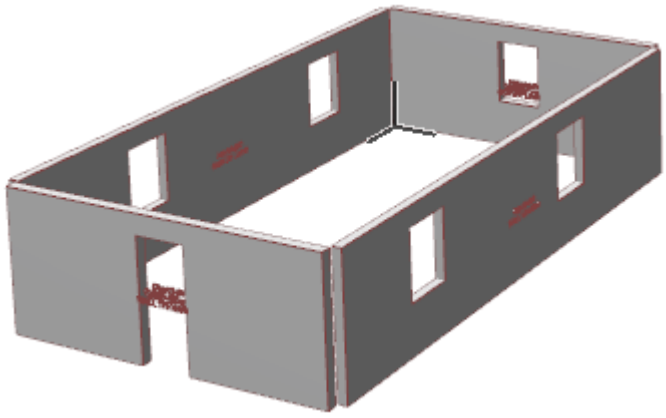
First create elements for the side walls. Specify the settings like below:



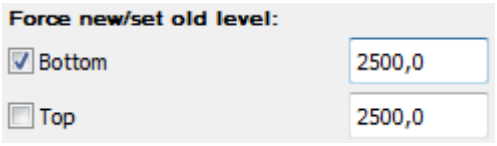
Let's make the side wall elements 2500 mm high (the trusses are placed over those) and adjust core inner side 63 mm outwards from architect's wall (13 mm gypsum board + 50 mm additional studding):



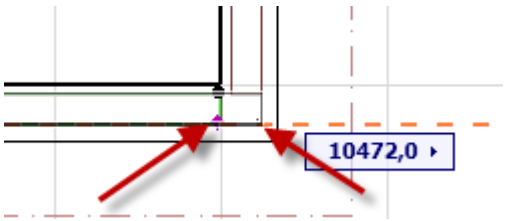
Create new element from selection creates element for the side wall. Do the same for the other side wall. Then make the gable wall elements with *Place new element with line* –operation. Before adding let's make sure that Force tilt has the value 90 degrees, element height is 2500 mm and move to right is 63 mm. Draw the line starting from side wall element's inner corners so that the exterior of the element will be on the right hand side (from bottom to top at right and from top to bottom at left hand side). After that the elements look like:



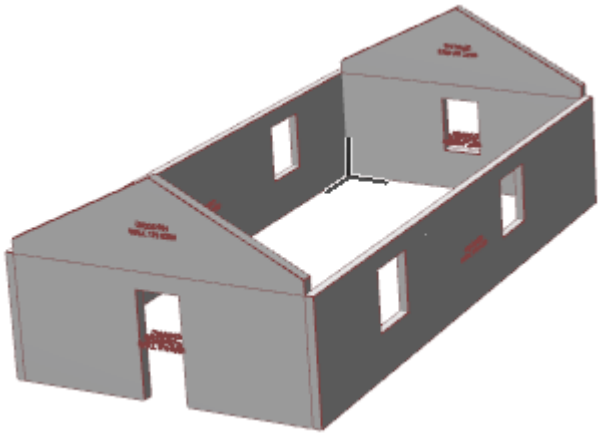
Add angled gable wall elements as previous ones but force bottom level to 2500 and remove forcing top level:



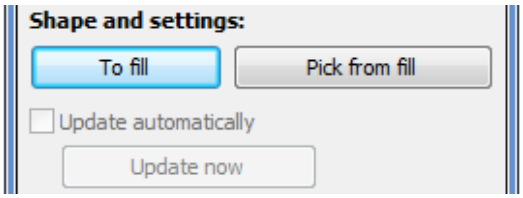
Stretch the side walls to fill the corner gaps:



The elements now look like this:

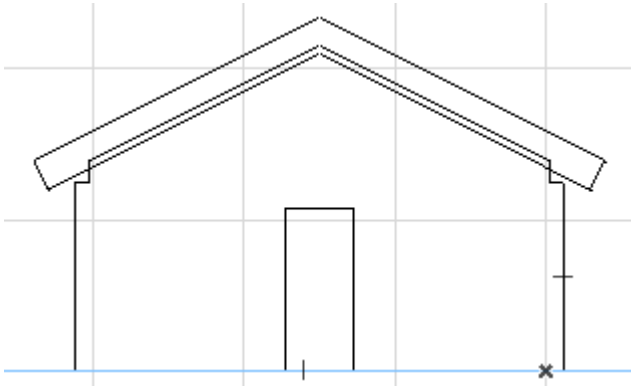


Now we could edit the top gable elements to extend over the side walls with *To fill* and *Pick from fill* –operations. The fill is edited with standard ArchiCAD *fill*-tools and the changed shape is picked back to element with *Pick from fill* button. Fill is deleted after editing.



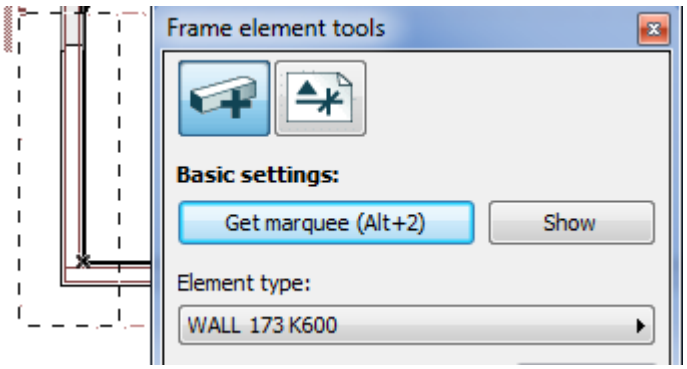
Finally the elements will be given IDs with *Give IDs*-operation.

This section shows that the gable element extends 100 mm inside the roof slope:

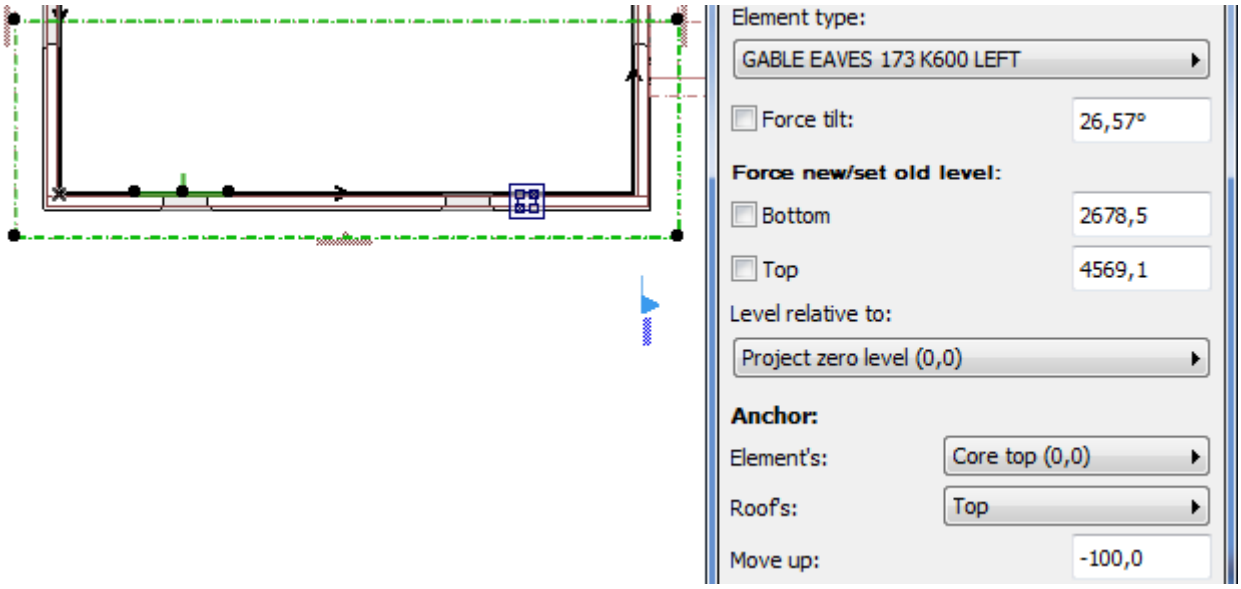


17.2 Creating eaves element from roof

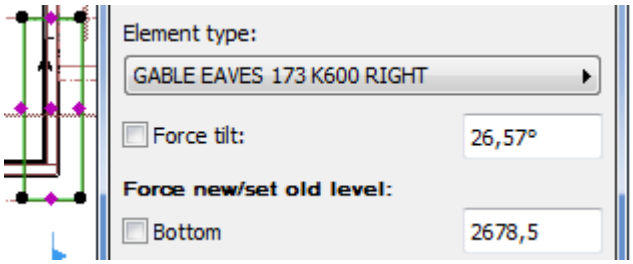
Draw 1200 mm wide marquee to the left hand side of the roof and pick it using the element tool (the marquee can be picked from any polygonal ArchiCAD-element):



Clear the marquee and select the roof and set the values as below to get the element top 100 mm below ArchiCAD roof. Click *Create new element from selection* to create the element:



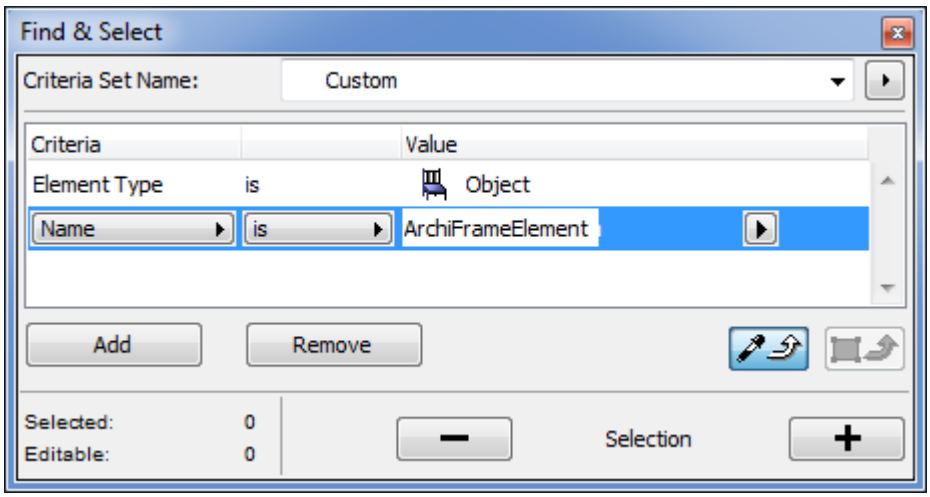
Move and copy the created element to the right side with built in ArchiCAD command (Ctrl+Shift+D) and change element type to right:



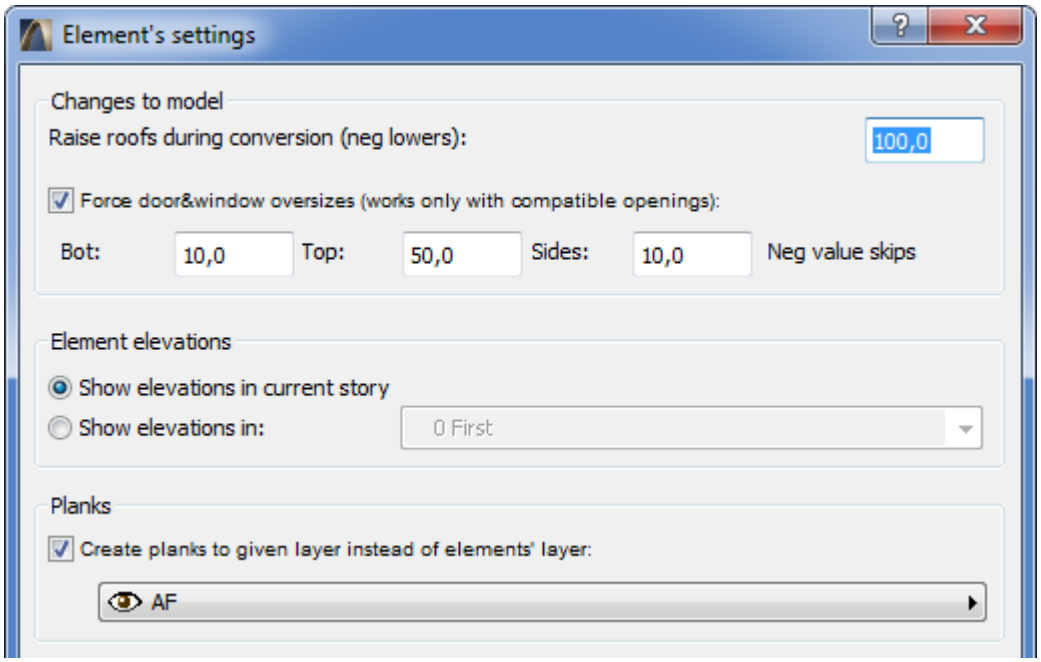
Finally *Give IDs* to new elements. Note that.

17.3 Creating and editing the planks

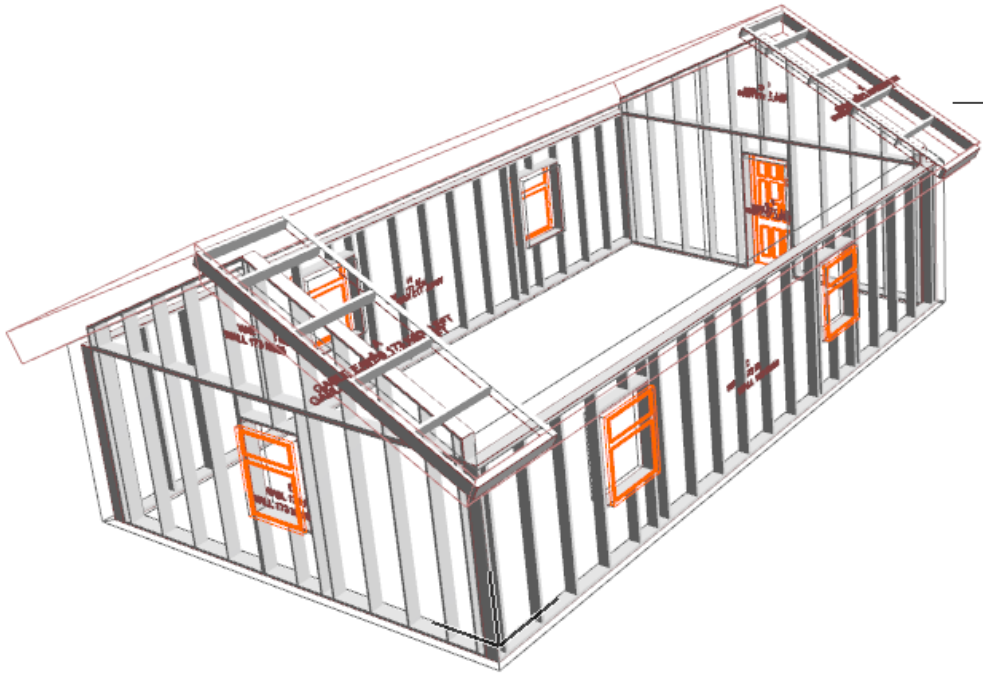
This is a simple phase: Select the element objects to work with. ArchiCAD Find & Select is useful for that:



Then click *Create planks*. Please note that the additional settings affect the planks and the elevations:

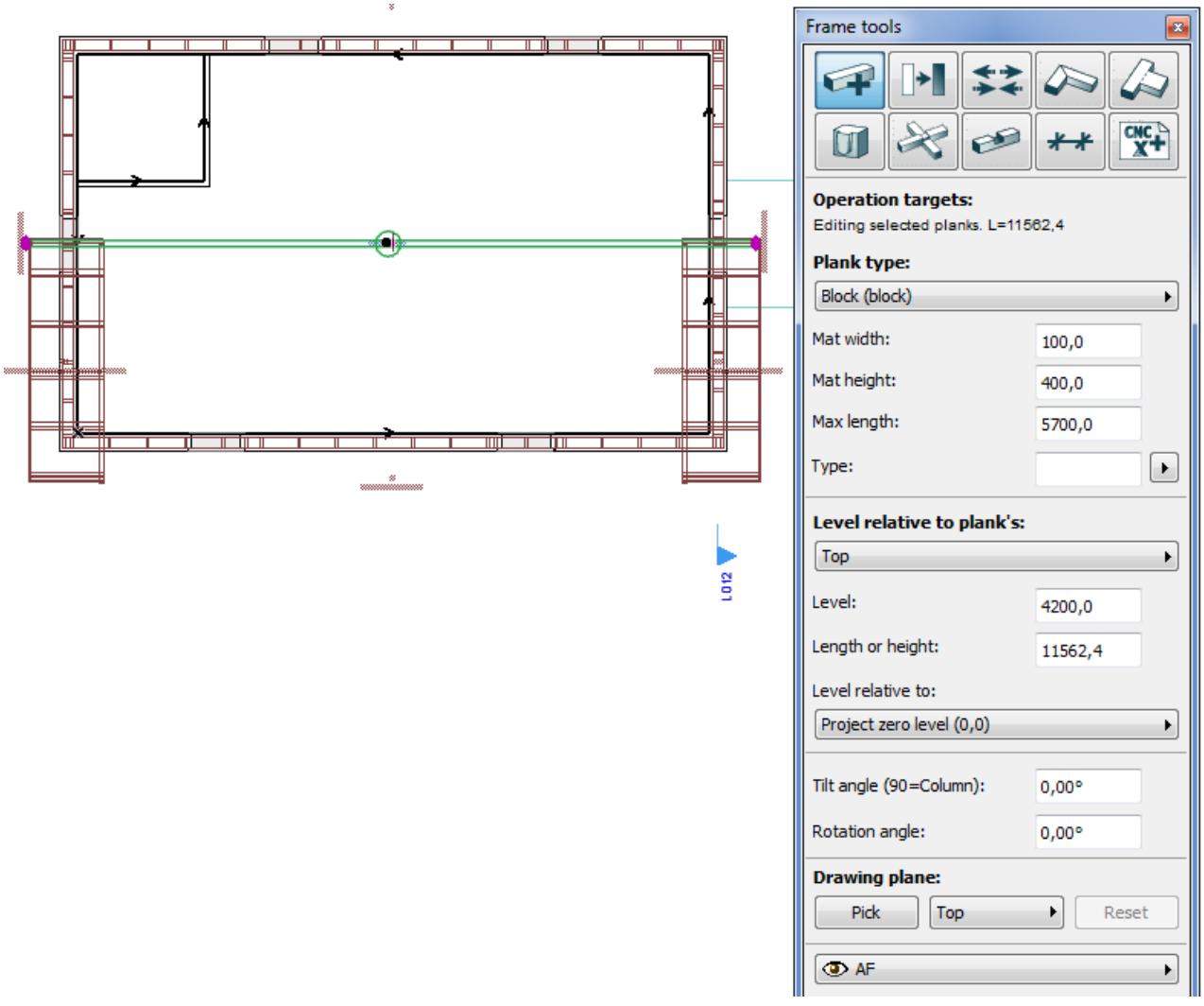


Planks can be edited with standard *ArchiCAD*-tools and ArchiFrame plank tools. Next image is the model after creating planks. The elements are on wireframe layer:

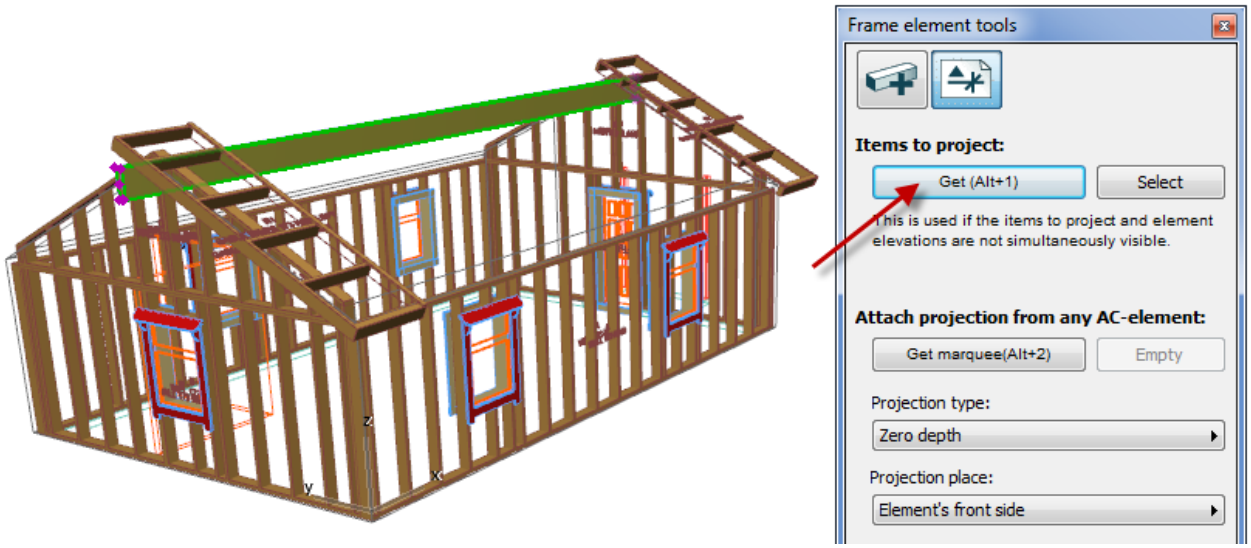


17.4 Adding ridge beam

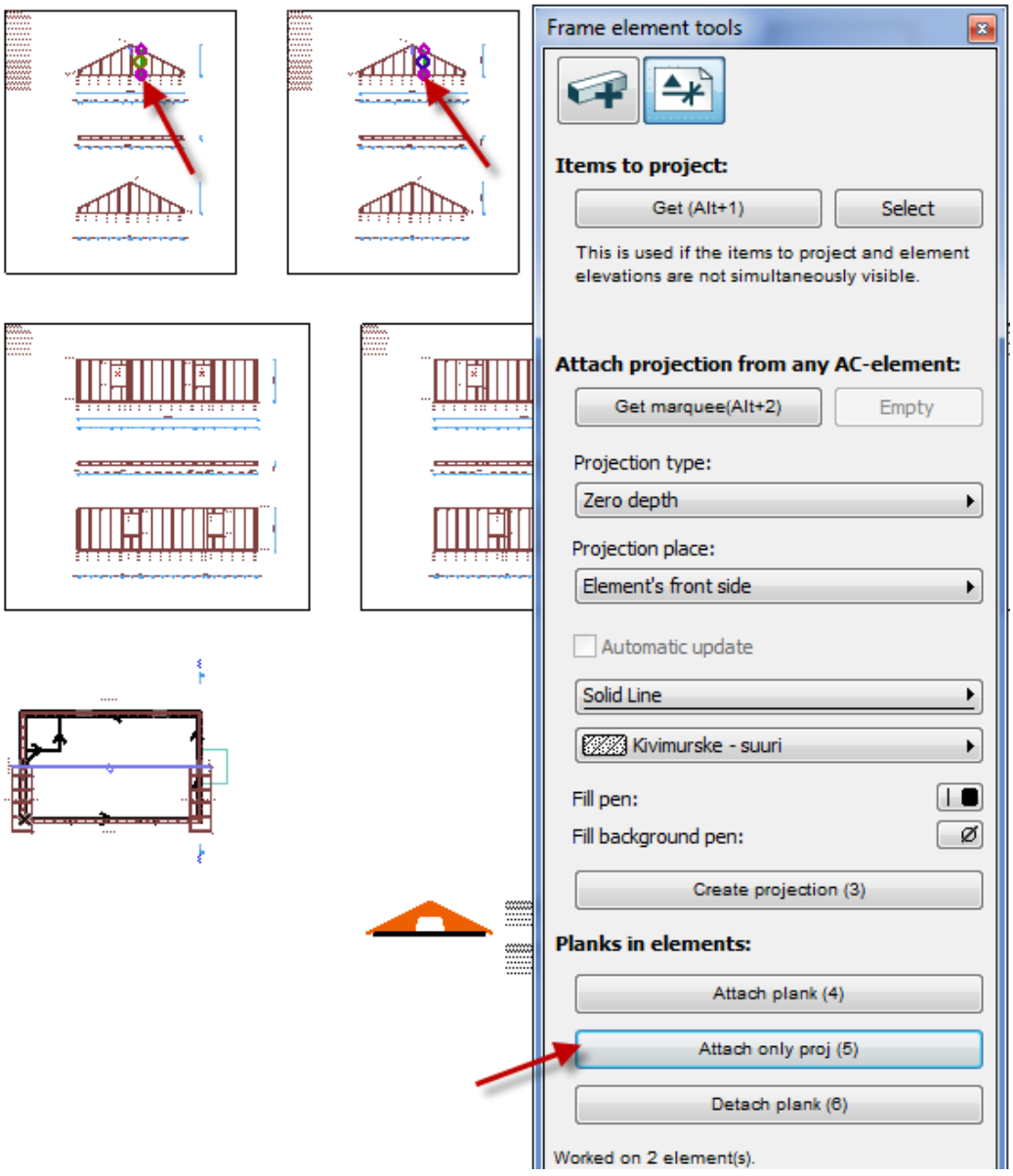
Ridge beam is not part of any element but it is useful to see it in the gable elements. Let's add the ridge beam with the following settings:



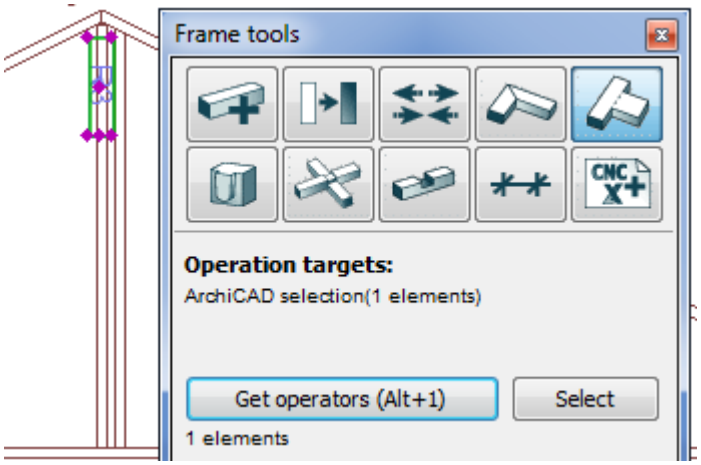
Then add the projections to the gable element. First pick the beam as an item to project from a 3D window:



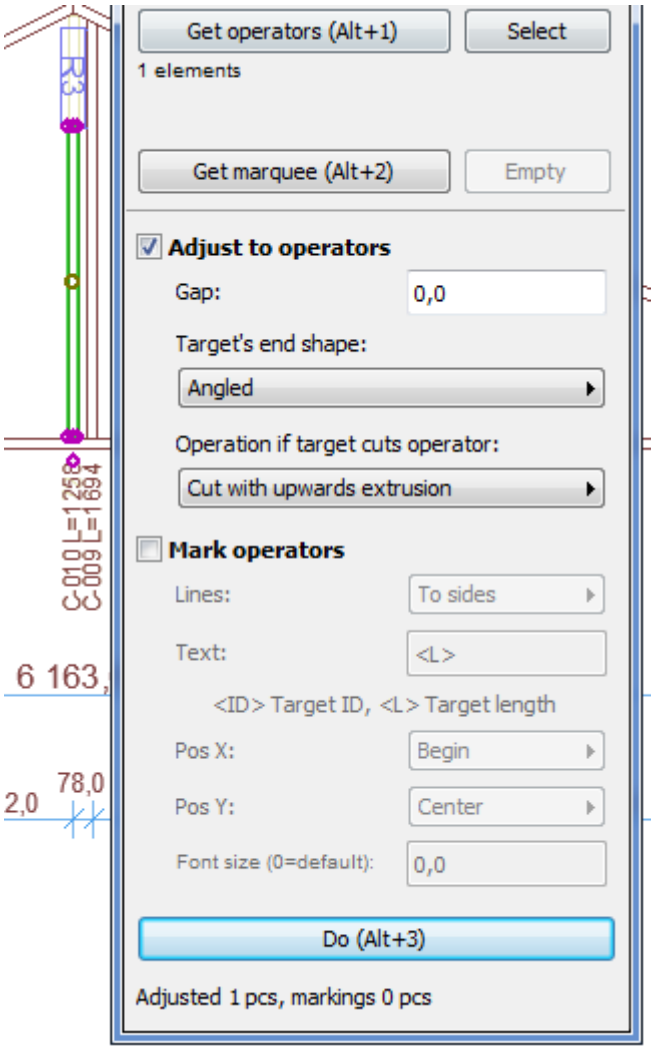
Then select the projections where the ridge beam should be visible and click *Create projection*:



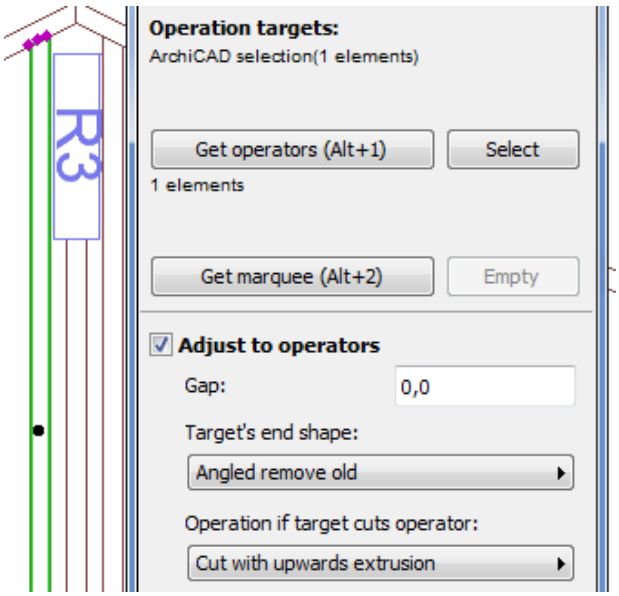
Add load bearing support for the ridge beam with plank tools. Let's select the ridge beam as operator:



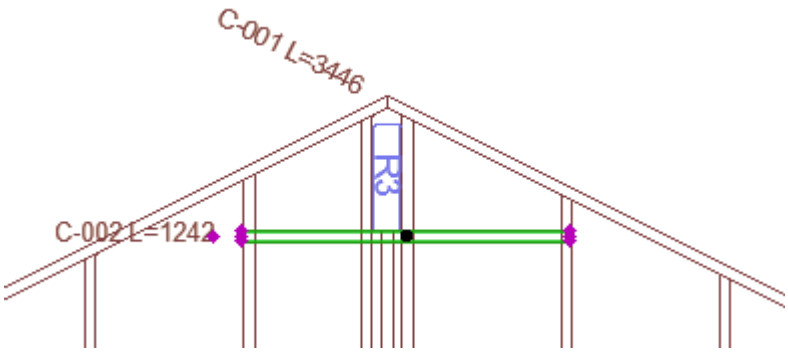
Adjust top of the stud to the beam's bottom side and remove remaining part of the stud above the beam:



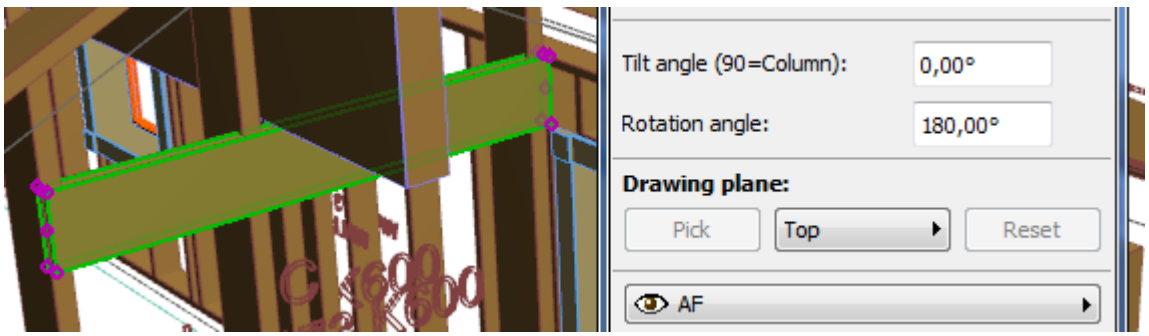
Duplicate the right hand side stud with the ArchiCAD drag & copy operation to the left and adjust it to element's top plank (select it first as operator). This time remove old cuts:



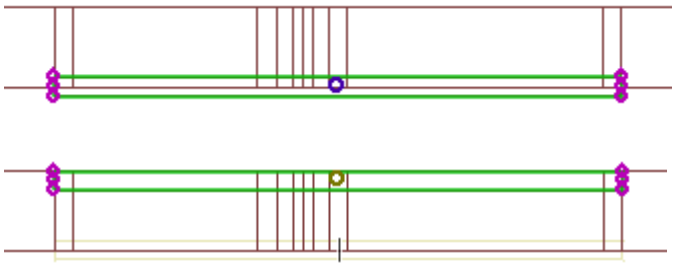
Then add horizontal support by copying the lowest plank of the element and stretching it to fit:



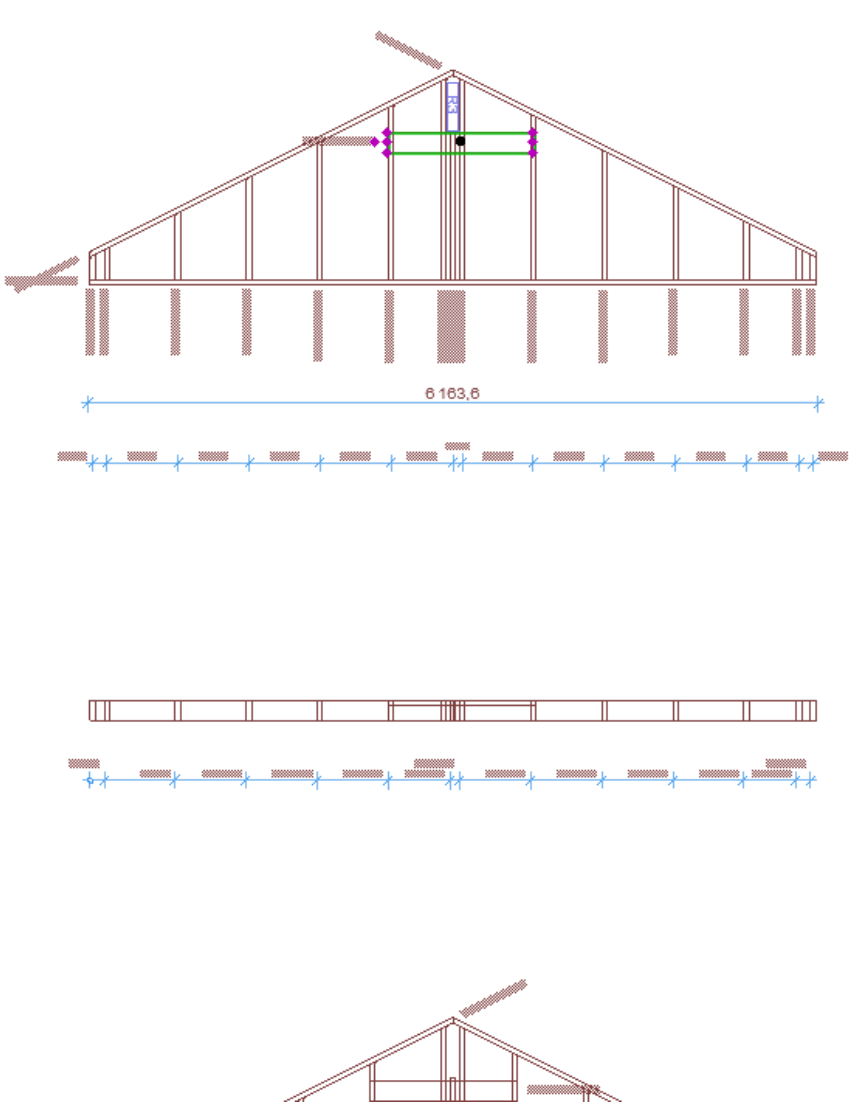
Adjust the rotation angle in a 3D window:

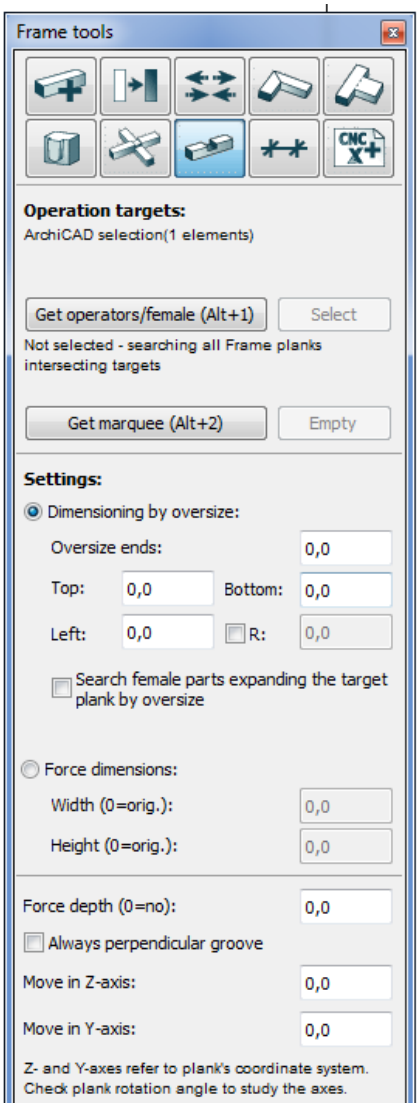


Move the plank to the inner side, either in 3D or using the top projection:

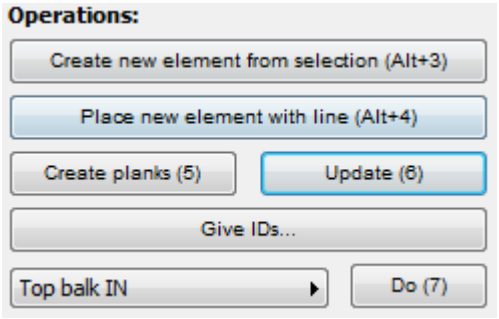


Change grooves to studs:



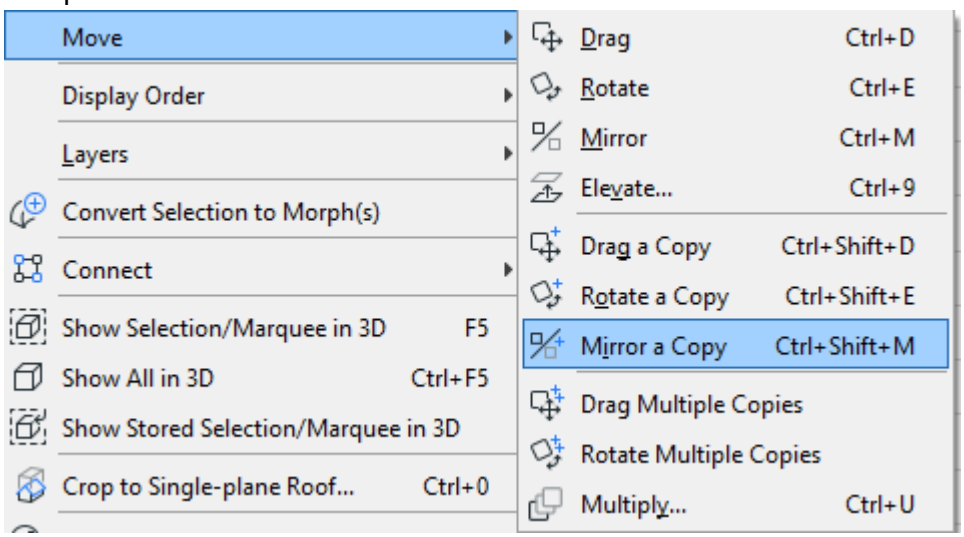


After the changes, update the cut list and dimension lines with element tools *Update* –operation:



17.5 Copying finished element to another place

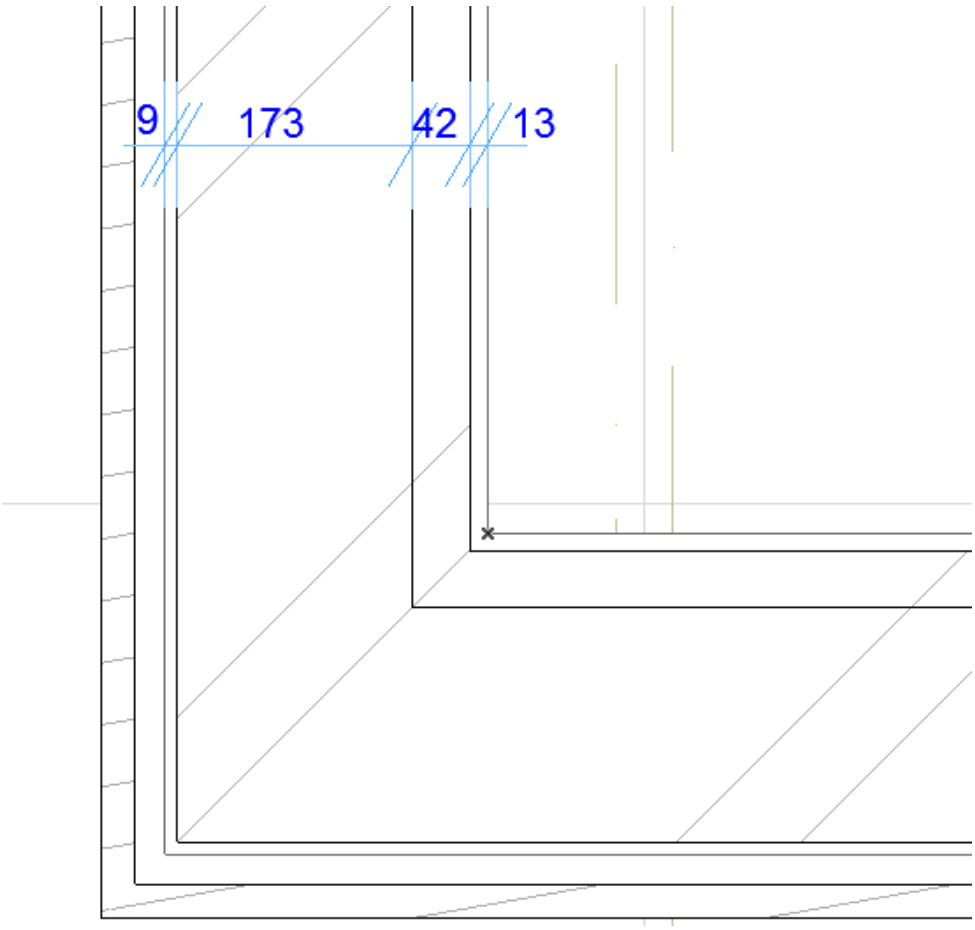
For example, the gable walls may be similar. In this case, only the other end is finished, and it is copied & mirrored to another end. To do that, the related ArchiFrameElement-objects are mirrored & copied with standard ArchiCAD-command:



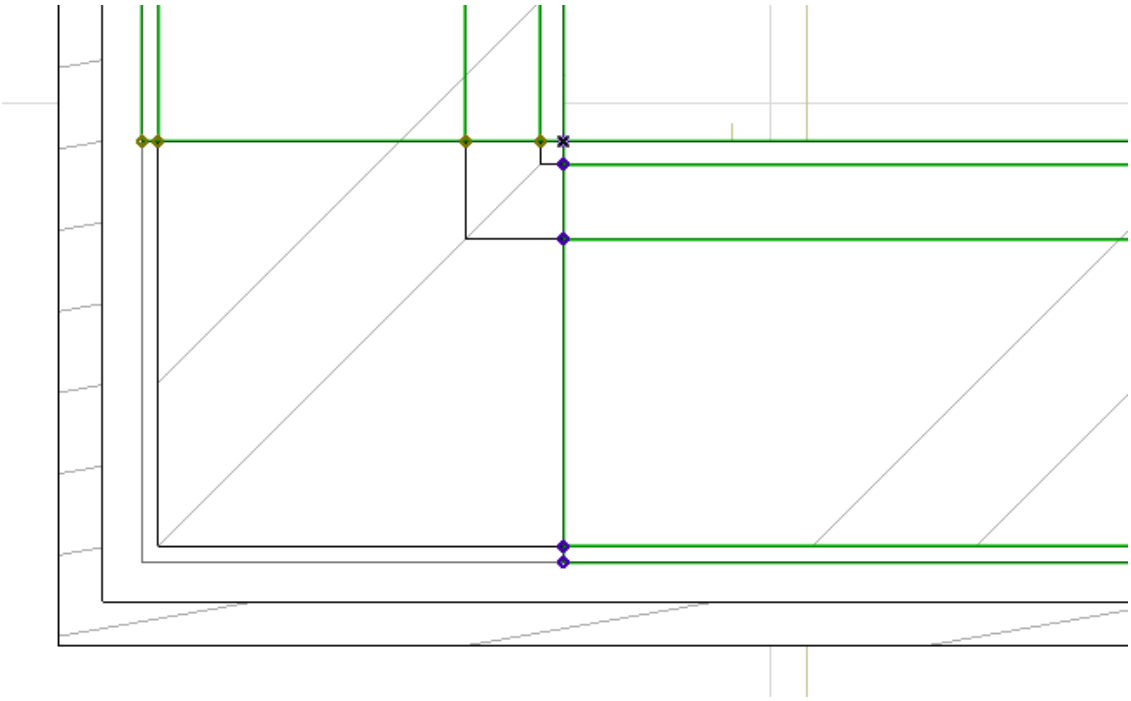
ArchiFrame will mirror & copy the related planks & boards with the ArchiFrameElement-objects. Copied ArchiFrameElements are not linked to ArchiCAD-walls (or slabs/roofs) in the new place. To do that, the ArchiFrameElement-objects must be connected to ArchiCAD-elements using projection tools. Connection is needed to be able to update element openings and for correct opening weight calculation.

17.6 Multilayer elements, corners and boarding

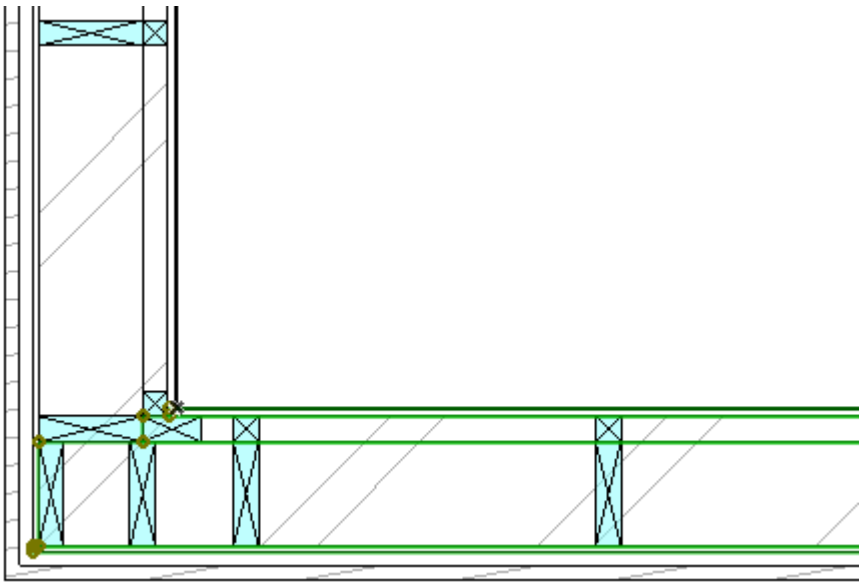
Example structure from default *data*-folder element type *WALL 173+42 VERT*:



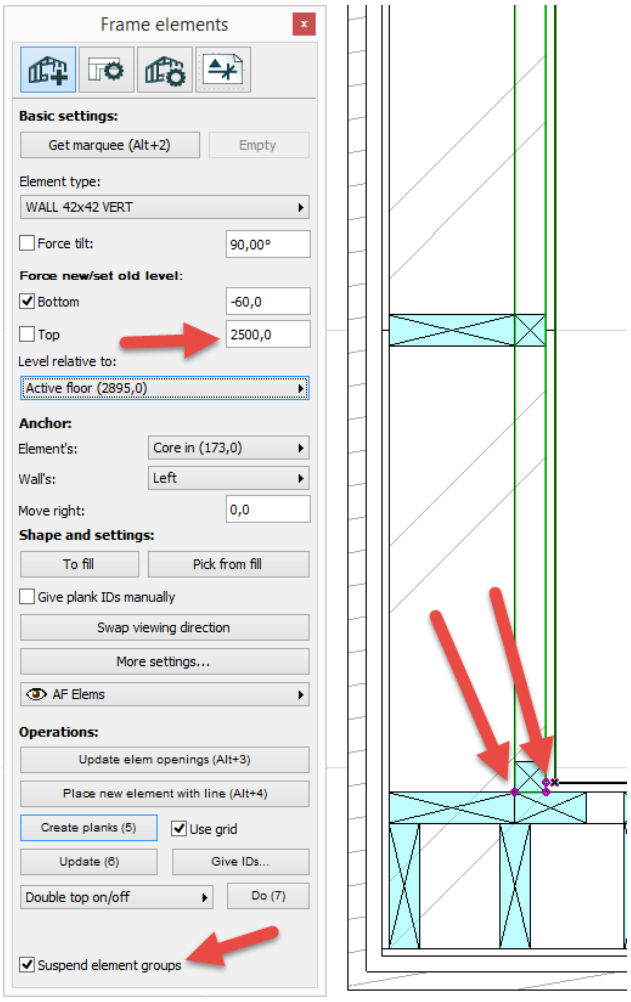
Exterior paneling and related framing are skipped. The elements are placed so that corners touch to allow ArchiFrame to connect the corners together. Corners may overlap if placed for example using the core in as anchor – the corner tool works that way also. IDs are given normally – ArchiFrame detects multilayer element and assigns just a single ID to the group.



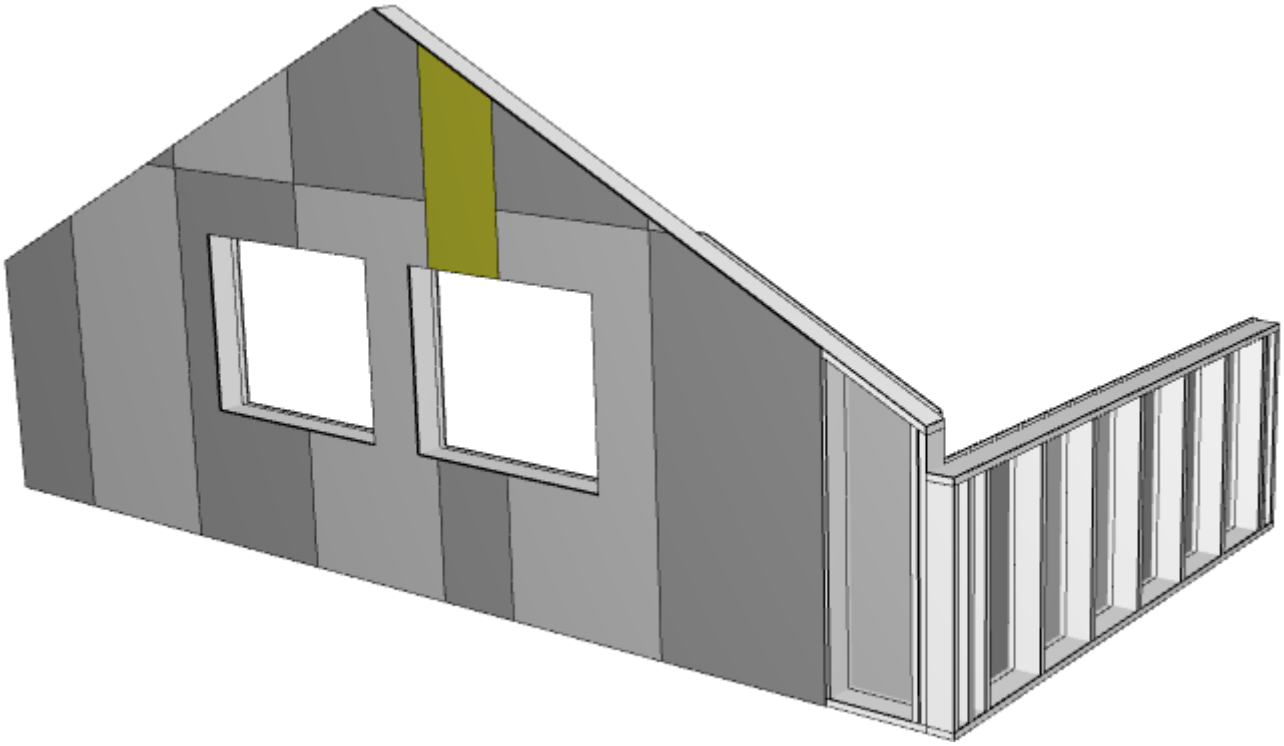
After setting corner type to *Long*, the other wall will be automatically set to *Short*. The result after creating planks:



Then the height of at least the inner boarding and extra studding needs to be limited for gable walls. It is done by selecting the corresponding element objects, suspending element groups to let the main studding and the exterior boarding to fill the whole gable and finally editing the level value. The same could be done with using *To fill*–button, editing the fills and clicking *Pick from fill*. Simple editing is also possible using element object's hotspots in 3D or section.



After creating planks (and deleting some boards to see the framing) the result is:



- ArchiFrame has assigned colours to boards so that they are visible.
- Projections have only exterior boards when looked from outside, interior from the other side.

Miscellaneous notes:

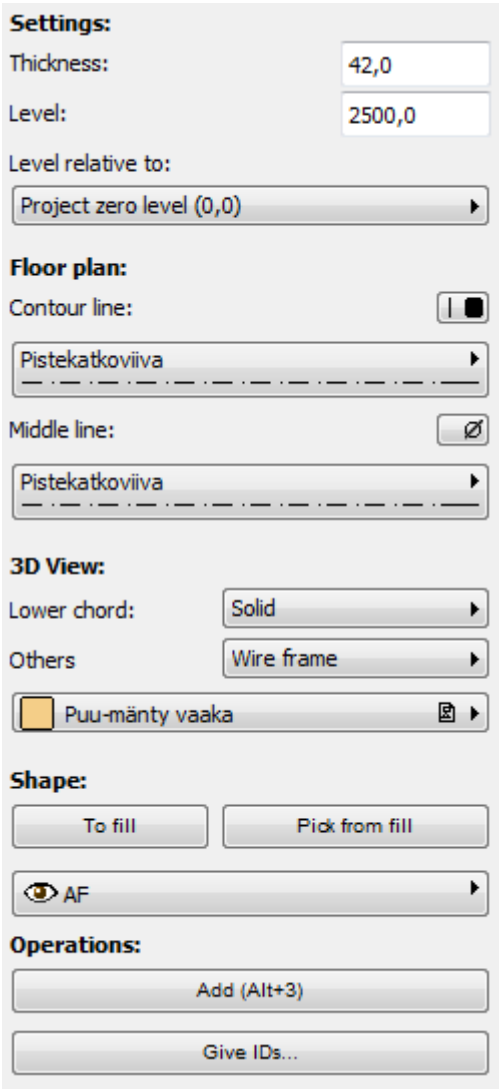
- To find definitions in *ArchiFrameElements.xml* please look for text MULTILAYER.
- To recreate boarding after changes in studding: select just a board belonging to an element and *Create planks* will be changed to *Update boards.*
- Attribute studs = “xxx” means this layer depends on layer xxx. For boards it is where seams can be made and for planks it means that place studs where there is a stud in referenced layer.
- Update element updates colours and RGB-values of adjacent boards to make sure the boards can be seen.

18 Trusses

Trusses are modeled with object ArchiFrameTruss. The work flow when adding trusses is:

- Make a truss section for each truss type and add dimension lines manually if needed.
- Pick the shape from the section to ArchiCAD fill. Use separate fill for the lower chord or other main part to allow getting its projection to elevations without the rest of the truss.
- Pick the truss shape from fill or fills and place the trusses. The truss shape is always adjusted with fill – the truss object cannot be stretched.
- Assign IDs to trusses and place the quantities wherever needed.

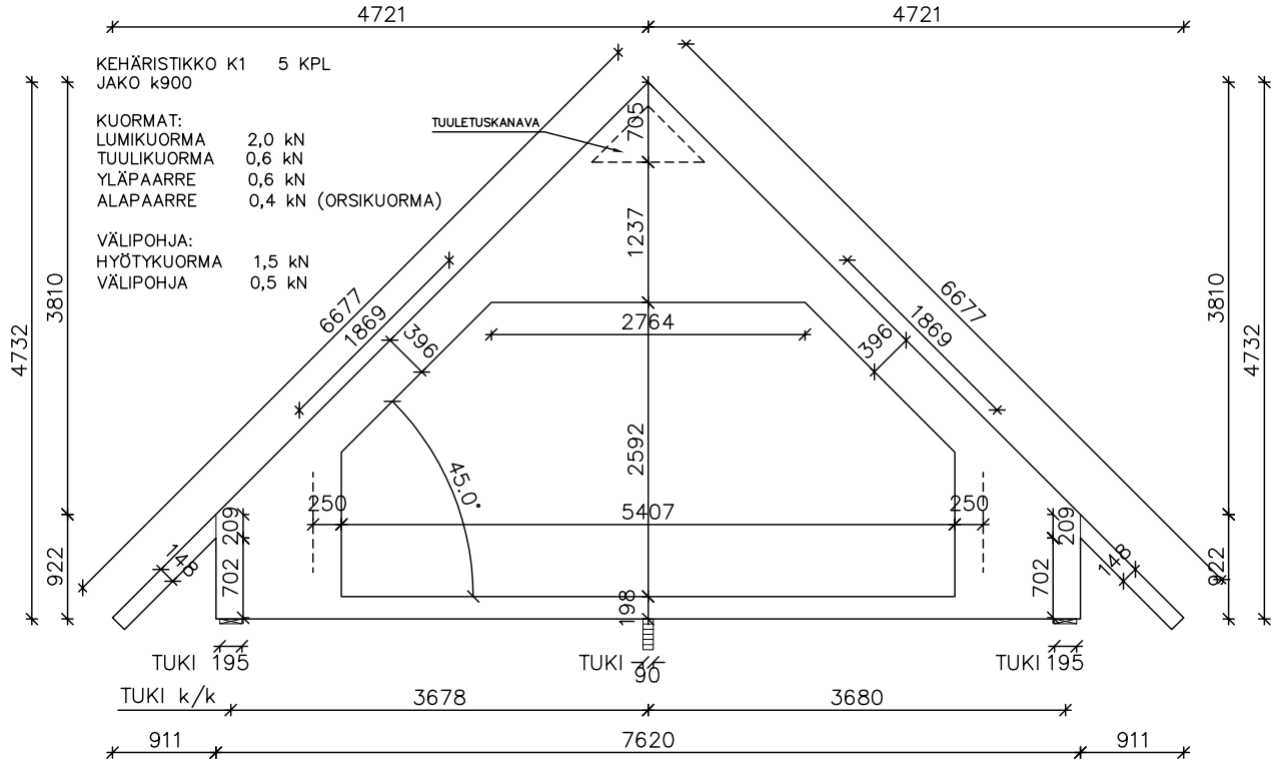
Truss tool parts are:



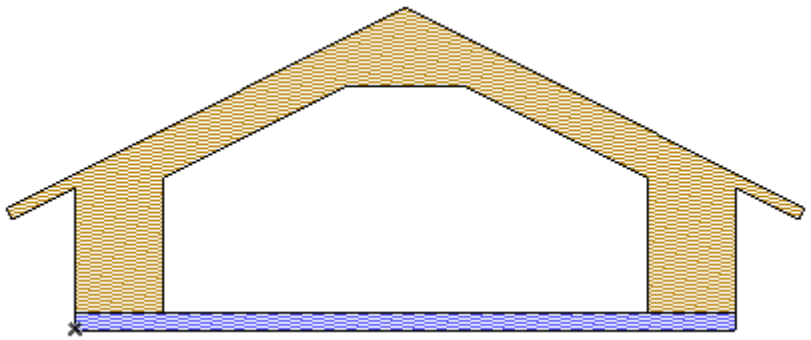
- *Thickness*, entire truss has the same thickness.
- *Level*, the level of lowest point of the truss.
- *Level relative to*, level anchor point.
- *Floor plan*, the floor plan display.
- *3D View*, it may be useful to show part of the truss as solid and the rest as wire frame.
- *To fill*, creates a fill to be used to edit the truss shape.
- *Pick from fill*, picks the shape from selected fill(s) or placed truss. Picked shape is used to create new trusses or it can be set to existing ones.
- *Add* or *Set shape*, adds new truss to given place or sets shape for selected trusses to shape picked earlier.
- *Give IDs*, gives similar trusses same ID. Can be issued with or without selection.

18.1 From truss dimension drawing to fill

The dimension drawing is possibly easiest to draw with the line or polyline tool. For example, from the illustration below, the outline is picked to ArchiCAD fill by selecting the fill tool and clicking any of the contour edges having space bar pressed. The hole is added afterwards:



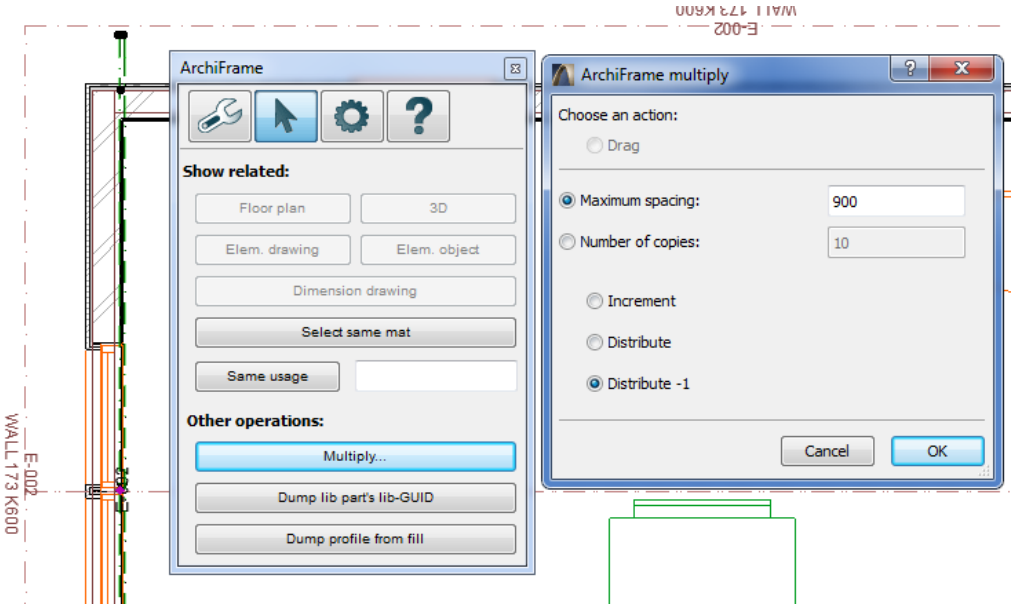
In the following example the lower chord is modelled with its own fill so it is only visible in the element elevations. The selection order for the fills is important – the lower chord part is selected last:



Shape is picked with truss tool's *Pick from fill*–operation. Last selected fill is the part that is visible in element elevations. If the shape is picked from just one fill, whole truss will be projected to elevation.

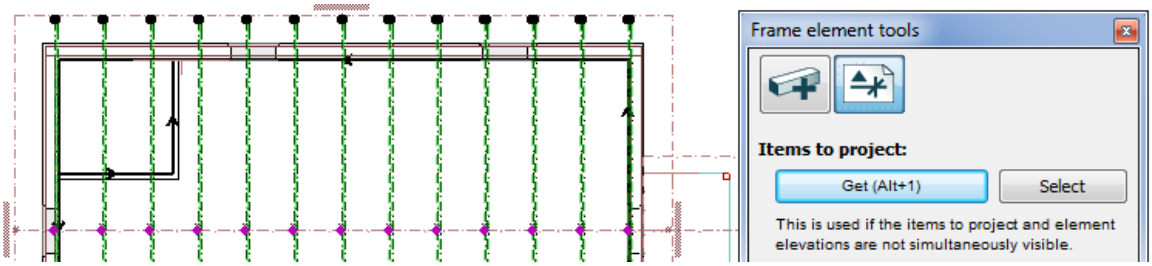
18.2 Placing trusses

In this example, the shape is first picked and then two trusses are placed next to the gable elements. Use the ArchiFrame *Multiply*-tool to ensure there is maximum 900 mm spacing and that the trusses are placed evenly. Select the first truss and open *Multiply* settings:

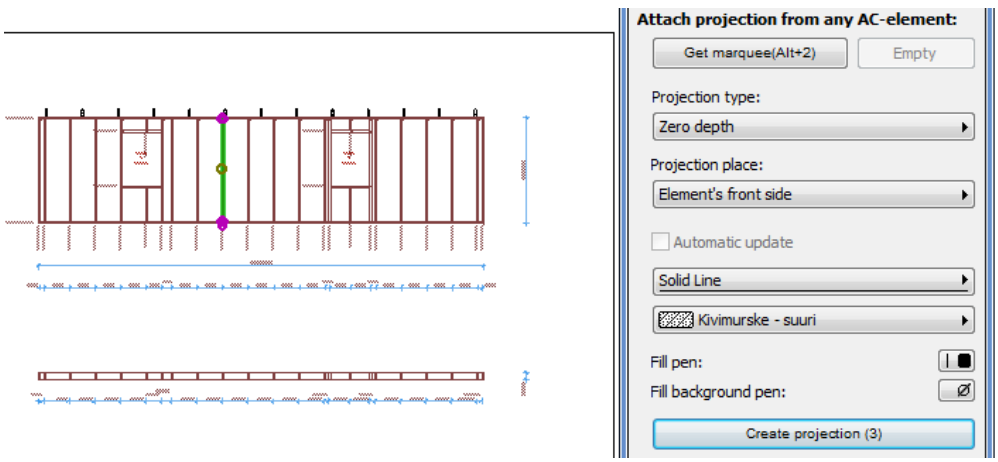


Draw a line from the first truss middle line hotspot, to the other side similar hotspot. After creating every truss, *Give IDs* is used to assign IDs.

The lower chords are attached to the element elevations using the element projection tool. First select the trusses and pick them as *Items to project*:



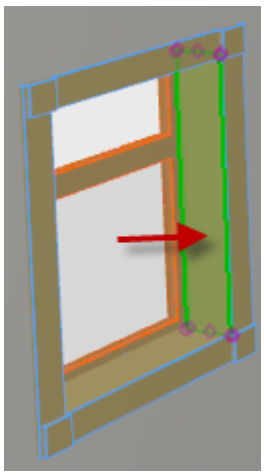
Then select one plank from destination projection and add the trusses with *Create projection*–operation:



Note! At the moment the projections are not updated if the truss is changed and vice versa.

19 Weatherboards

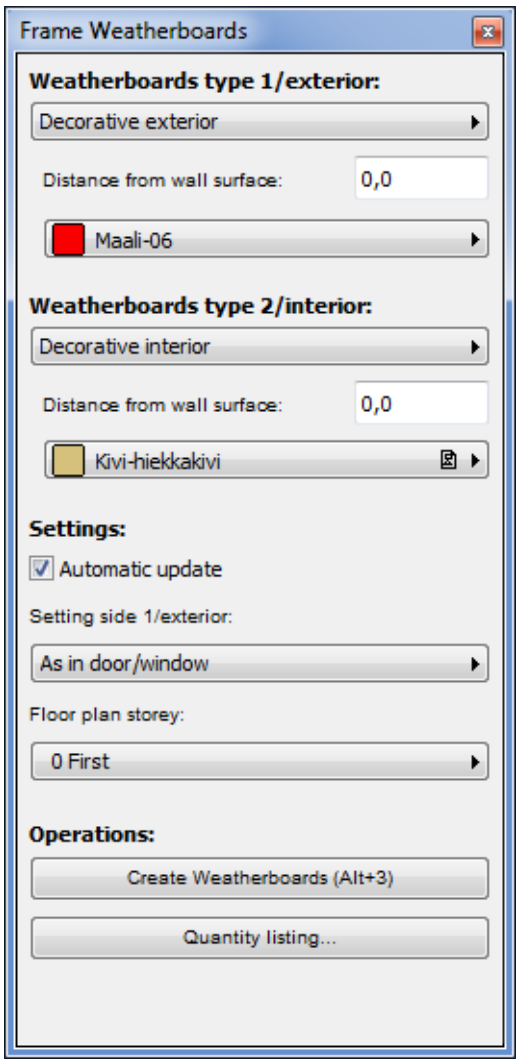
Weatherboards are separate *ArchiFramePlank*-objects and they may be used with any door or window. ArchiFrame recognises changes to walls and openings and updates the weatherboards automatically when they are changed. Weatherboards may contain also window jambs:



ArchiFrame produces cut list and even cnc-file for the weatherboards. An example of complicated but fully automatic exterior weatherboarding:

When using ArchiFrame weatherboards the openings' own similar parts are switched off from the door/window settings.

19.1 Weatherboard tools



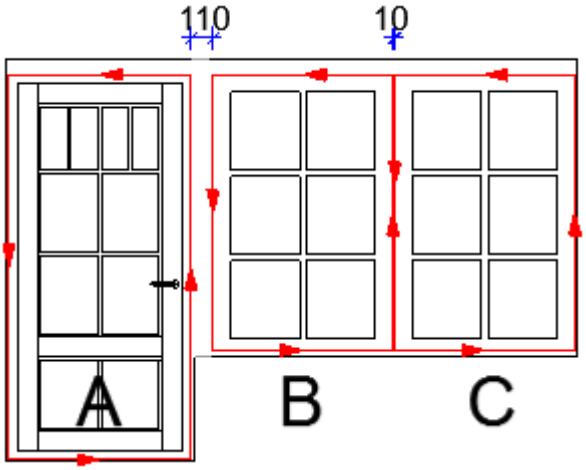
- *Weatherboard type 1/exterior* defines in exterior walls the exterior type and the type for the other side of the partition walls.
- *Distance from wall surface* is the distance between the weatherboards and the wall. This can be used to make bigger jambs. Negative value places weatherboards inside the wall.
- *Automatic update* allows ArchiFrame to update the weatherboards whenever the wall or the opening changes.
- *Setting side 1/exterior* defines how to detect the exterior side. If the same types are used on both sides then this setting does not matter.
- *Floor plan storey* is the home storey for the weatherboards. Even if the weatherboards are not visible in the floor plan ArchiCAD will use the plank hotspots. This may cause the mouse click to attach to the wrong invisible hotspot when adding dimension lines. It is recommended to create a new helper storey to place all the weatherboards. With default settings ArchiFrame will place the weatherboards to storey having text *constructional storey 1* in its name.
- *Create weatherboards* creates the planks. Selection can be for a single opening, group of openings, wall or multiple walls. ArchiFrame automatically combines together openings which are close enough to each other. This distance is defined in the xml-settings.

Tool palette can also be used to edit existing weatherboards. Selection can contain a plank, opening or one or more walls. Home storey cannot be changed here after creating the weatherboards.

19.2 Weatherboards setting file ArchiFrameCoveringBoards.xml

Because the weatherboards are quite complicated, the rules to create them are not simple. However, everyone who has done GDL-objects should be able to add and edit the rules.

The weatherboards are based on the contour lines made from the frames. For example:



Contour lines are originally oriented counter clockwise. Line between door A and window B is attached to door A's right hand side because it is longer than the window edge. At the same time window B's horizontal edges are extended to the door's vertical line. Lines between windows B and C are combined to form a single line which is centred between the windows. There will also be a special cover plank defined in the settings.

ArchiFrame detects standard rectangular windows. To form the contour line for an angled opening there is a special Lua script. Some of the angled openings in ArchiCAD standard library are already supported by the script, but other openings and libraries require editing the script. Examples and instructions to define the frame edges can be found from the xml-settings under tag <openings_script>.

Weatherboard definitions include rules to handle frame edges <linegroup>, operations between the planks <operations> and *Lua*-scripts <scriptplanks> for example to add supporting planks for fascia board.

19.2.1 Variables in expressions

It is possible to use variables instead of fixed values to allow changing materials without editing the rules to place them.

Variable	Description
jamb_depth	Jamb width, in other words the distance from frame to wall surface.
mat_thickness	Current material size.
mat_height	Current material size.
wall_distance	Distance from wall surface (comes from tool palette).
cosangle	Absolute value of cover plank angle, horizontal=1.0, vertical=0.0.
sinangle	Absolute value of cover plank angle, horizontal=0.0, vertical=1.0.
[parentid]	Can be used to set textual parameter for a plank: Owner opening's ID. For example setting the weatherboard ID: <objparam name="#id">WB-[parentid]</objparam>

19.2.2 Settings <settings>

19.2.2.1 Common settings <plank>

```
<plank>
  <misc minlen="0.050" layer="Cover boards" floor="*constructional storey 1*"></misc>
</plank>
```

XIm-attribute	Description
minlen	Minimum length for a plank. Shorter will not be created.
layer	Forced layer. If not given, the weatherboards are created onto wall's layer. Layer is created and shown automatically but locked layer is not automatically unlocked.
floor	Default home storey for weatherboards. Name can contain wild cards * and ?. See Weatherboard tools.

19.2.2.2 Combining adjacent openings <lines>

```
<lines>
  <combinelines id="cover_out">
    <combine maxdist="0.020" settype="combine20">
      <material id="L990" zoff="jamb_depth-mat_thickness*0.5" linexoff="-mat_height*0.5"
rotangle="90"></material>
    </combine>
    <combine maxdist="0.111">
      <!-- Use type's default material -->
    </combine>
    <combine maxdist="0.136">
      <material id="L149" zoff="-mat_thickness*0.5" rotangle="90"></material>
    </combine>
  </combinelines>

  <!-- The same for inner cover planks -->
  <combinelines id="cover_in">
    <combine maxdist="0.020" settype="combine20">
      <material id="L989" zoff="jamb_depth-mat_thickness*0.5" linexoff="-mat_height*0.5"
rotangle="90"></material>
    </combine>
    <combine maxdist="0.111">
      <!-- Use type's default material -->
    </combine>
    <combine maxdist="0.136">
      <material id="L124" zoff="-mat_thickness*0.5" rotangle="90"></material>
    </combine>
```

```
  </combinelines>
</lines>
```

Defines rules for combining edges and special cover planks for narrow, normal and wide gap. In the example, the narrow covering plank is used up to 20 mm gap, weatherboard material up to 111 mm and special wide material from 111 to 136 mm gap. Material settings are given as in element module, see Material settings <material>.

```
<combinelines id="cover_out">
```

Gives the settings ID "cover_out" that is used to refer to these settings in many following weatherboard type definitions.

```
<combine maxdist="0.020" settype="combine20">
  <material id="L990" zoff="jamb_depth-mat_thickness*0.5" linexoff="-mat_height*0.5"
rotangle="90"></material>
</combine>
```

Xlm-attribute	Description
maxdist	Maximum distance for edges to be combined.
settype	Gives the edge a geometry ID that will be used instead of ID coming from geometry rules.

19.2.2.3 Default settings for new plank <newplank>

See XML-settings for an ArchiCAD element.

19.2.3 Weatherboardtypes <covertypes> / <covertype>

In this part we will go through weatherboard type Decorative exterior.

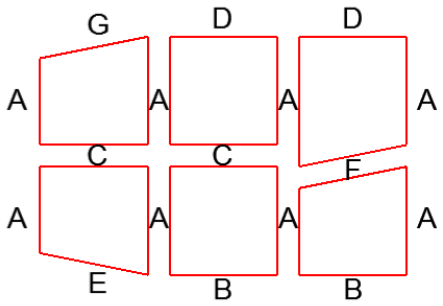
19.2.3.1 Settings for the type <settings>

```
<settings>
  <newplank>
  </newplank>

  <combinelines ref="cover_out">
  </combinelines>
</settings>
```

<*newplank*>-tag is used to override settings given at the beginning of the file. Tag <combinelines ref = "cover_out"> refers to settings defined earlier.

19.2.3.2 Weatherboard group <create>/<linegroup>

The <linegroup> defines handling for frame edges. Edges will get geometry ID based on the position:



Letter	Geometry ID
A	vertical, middle ones verticalmid
B	horizontalbot
C	horizontal
D	horizontaltop
E	inclinedbot
F	inclined
G	inclinedtop

In addition to these, the geometry ID can come from Combining adjacent openings <lines>-section, for example <combine maxdist = "0.020" settype = "combine20"> gives geometry ID combine20.

<*linegroup*>-tag may contain attribute type = "jamb". In this case, the definitions are for jambs and the operations will change:

- Frame edges will not be combined but instead every adjacent frame will get the jambs.
- Jamb is created only if there is a weatherboard on top of it or if the weatherboard type defines only jambs (the only <*linegroup*>-rule is for jambs).

19.2.3.3 Weatherboards group planks <create>/<linegroup>/<planks>

<linegroup> contains <*planks*>-tags that define the actual planks.

```
<planks door="0" group="vertical" type="vertical*" lineoff="-0.010" extendtop="0.120"
extendbot="0.195" extendt="0">
  <material id="block" thickness="0.022" height="0.120" zoff="-mat_thickness*0.5"
rotangle="90">
    <objparam name="iUsageId">VUORIL-LEV</objparam>
    <objparam name="#id">VLLEV-[parentid]</objparam>
  </material>
  <script>
  </script>
</planks>
```

<*planks*>-tag may contain attributes:

Xlm-attribute	Description
door="0/1"	Will this <*planks*>-tag be handled with doors. Default is one, will be handled.
window="0/1"	As previous but for windows.
group="name"	Group name for the planks to be used in later <*operations*>-rules that adjusts the planks together.
type="name"	The kind of lines which will be handled here. Name may contain wild cards, for example, horizontal*. Different values can be given separated with vertical bar, for example, horizontaltop|horizontal|inclined*.
lineoff="meters"	Distance to move the edge right/away from the opening. Negative moves inwards. For example, if the weatherboard should cover 10 mm of the frame, the value is -0.010.
extendtop="m"	How much the edge is extended at upper end.
extendbot="m"	How much the edge is extended at lower end.
extendt="0"	Can be used in conjuction with extendtop and extendbot: If edge ends to middle of another line (T-joint), it will not be extended.
extend="m"	How much the edge is extended from both ends.

extendfree="m"	Extend only if the end of the edge is not inside the opening and does not end at the middle of the other edge (T-joint). Corner is extended.

<*planks*>-tag has <*material*>-tag that defines the material type. If <*material*>-tag is missing, no plank will be created for the edge, but the <*plank*>-definition may be necessary to adjust other edges.

Material settings are given as in element module, see Material settings <material>. In addition, it is possible to specify XML-settings for an ArchiCAD element.

Group may have also *Lua*-script. An example of the script is included in weatherboard type Decorative exterior. Script has a global variable gPlank having following fields (note that changing these fields will not change the plank - the plank must be edited with ac_objectset()-function):

Field	Description
bDoor	true = door edge, false = window edge.
x1,y1,x2,y2	Edge coordinates in local 2D-co-ordinate system.
gx1,gy1,gz1, gx2,gy2,gz2	Plank coordinates in global model coordinates.
len	Plank length.
begConn	Begin joint type: • 0, No connection. • 1, Connected to other edge's end. • 2, Connected to middle of another edge (T-joint) or end is inside the opening.
endConn	End joint type.
bExtR	true = exterior to right, false = plank direction is swapped and exterior to left.
group	Edge geometry ID name, for example "vertical".
sideout	Only when creating planks with script: Side to be outside from the wall. In scriptplanks-section values 11-16 may be used. Those values make the plank's direction to be swapped without swapping the machinings.

Script can set global variable gnSideOut to value 1…6 which defines which side of the specified plank must be outside from the wall (see About the planks). This is required to allow placing planks with asymmetric profile. In addition the script can set globals gtblCutBeg and gtblCutEnd to define the cutting plane for the plank. These variables may contain fields:

Field	Description
x,y,z	3D-coordinate on the plane.
dx,dy,dz	Plane normal vector.

19.2.3.4 Operations between the planks <create>/<operations>

As in element module, see Operations between the planks <operations>. With weatherboards it is possible to give additional attributes targetex and operatorex, which refer to weatherboard place:

```
<jointo target="vertical*" targetex="verticalmid"  operator="inclined*">
```

19.2.3.5 Script defined planks <create>/<scriptplanks>

First there is a script to define the planks to be handled:

```
<findgroup resulttbl="gtblTop">
  <base name="topbox"></base >
</findgroup>
```

```
<findcross resulttbl="gtblWeather">
  <base name="weatherboard"></base >
  <cross name="vertical*" maxdist="0.150" linepos="mid"></cross>
</findcross>
```

This example creates global *Lua*-variables gtblTop and gtblWeather. Table gtblTop will contain planks with matching group = "topbox"–definition. <findcross> will identify all the fascia boards (weatherboard) intersecting with the vertical planks (vertical*). <*base*>- and <*cross*>-tags may have attributes:

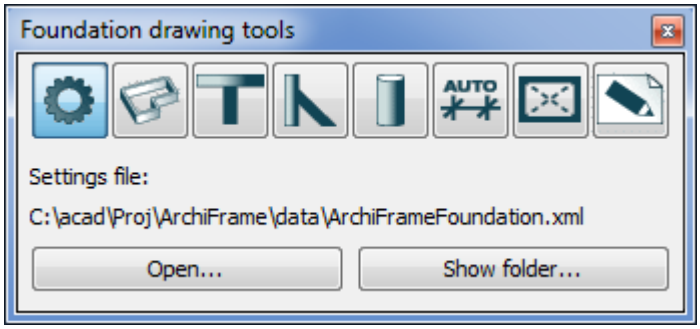
Xlm-attribute	Description
Name = "xxx"	Identifies every plank that has matching group name <planks group = "xxx"…>.
Maxdist = "m"	How much current line is extended when checking its intersection with other lines.
Linepos = "mid"	Use middle lines instead of plank reference lines.

Every plank has same fields as in the plank specific script, see Weatherboards group planks <create>/<linegroup>/<planks>. However, fields' begConn and endConn are always zero here. There is also the field tblCross that contains all crossing planks. It contains fields:

Field	Description
x,y	Intersection point.
tblPlankx	Information about crossing plank (contains all the same fields as the base plank).

20 Foundation drawing

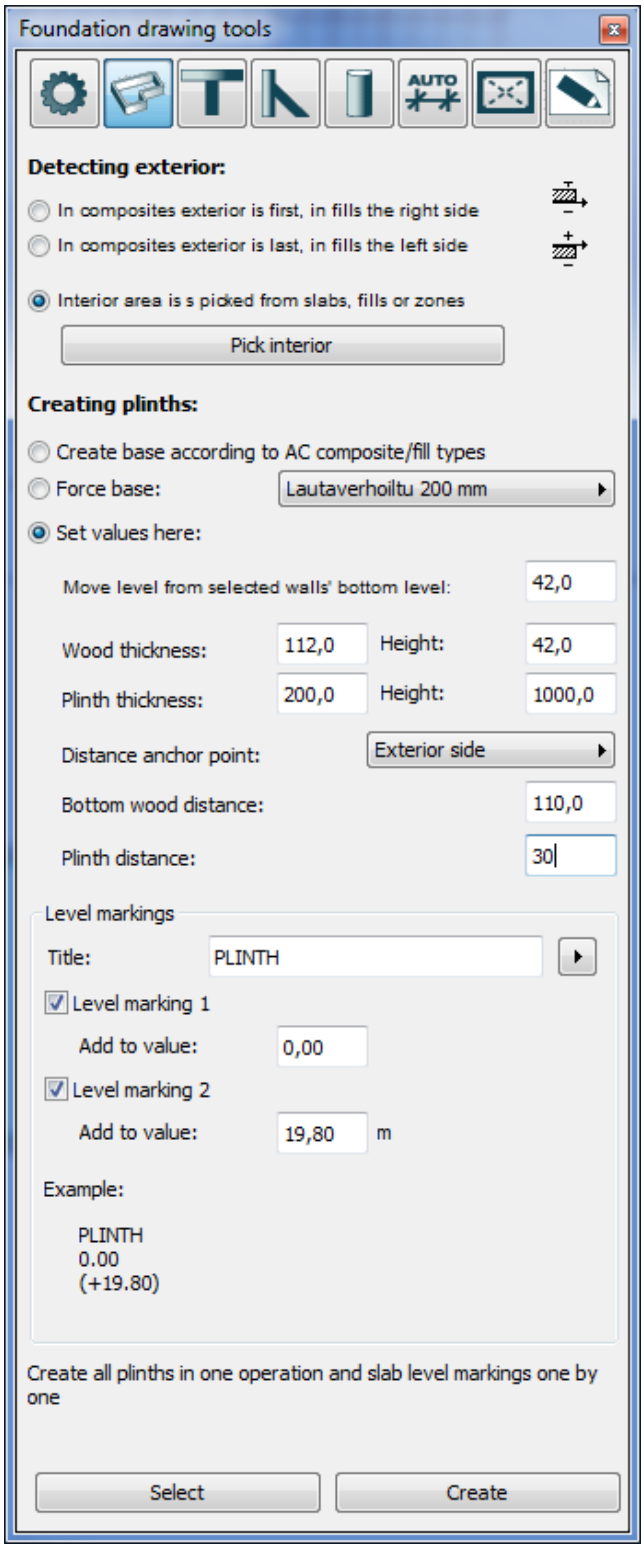
Foundation drawing part is used to produce 2D drawing from the 3D model.



All editable settings are placed into xml setting file. Before editing, the stock file is copied and renamed before changes are made to it. Instructions are included inside the xml-file. For automatic dimensions, the plinth, sole plate, columns, load bearing and stiffing walls must be placed onto separate layers.

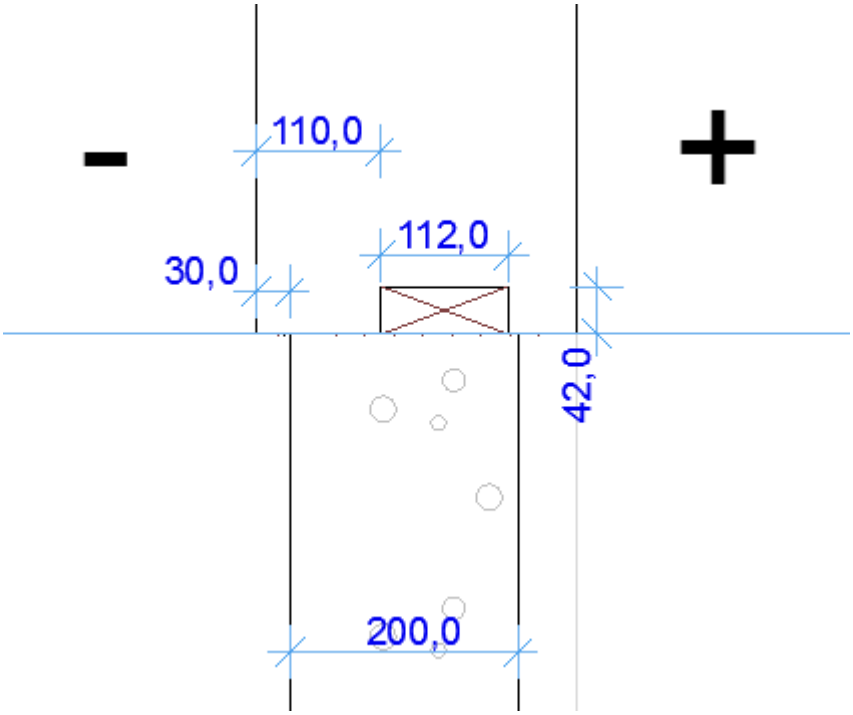
20.1 Plinths and slabs

First select the way to detect wall's exterior side. With composite structures the structure defines the exterior side but for walls made with just fills, the interior area is defined by picking slabs, zones or fills that cover an entire interior area. Then ArchiCAD *wall*-tool settings are adjusted to fit the plinth and sole plate before ArchiFrame-settings are edited:



Plinth and sole plate are created by the following settings:

- *Create base according to AC composite/fill type,* which means that the setting file contains the standard composite types and work according to the values entered there.
- *Force base*, use the selected composite type rules for every wall.
- *Set values here,* enter values in the tool palettes. For example, the values entered above is like this (the sole plate will be placed inside the original wall because level offset is equal to the sole plate thickness):



- *Level markings* will be like the example in the tool palette.
- *Select* selects all elements defined in the setting file (need to edit the settings to get meaningful results). Elements can be selected using standard *ArchiCAD*-tools.

To create plinths, select walls that are on top of the plinths. If *Select*-button does not select suitable walls, the walls are selected manually.

If necessary, the plinth is done and finished manually first. After that, the level markings for the porch, terrace and floors etc. are modelled with ArchiCAD *slab*-tool. ArchiFrame extends the slab markings close enough to the plinth to touch the plinth.

20.2 Load bearing and stiffing walls

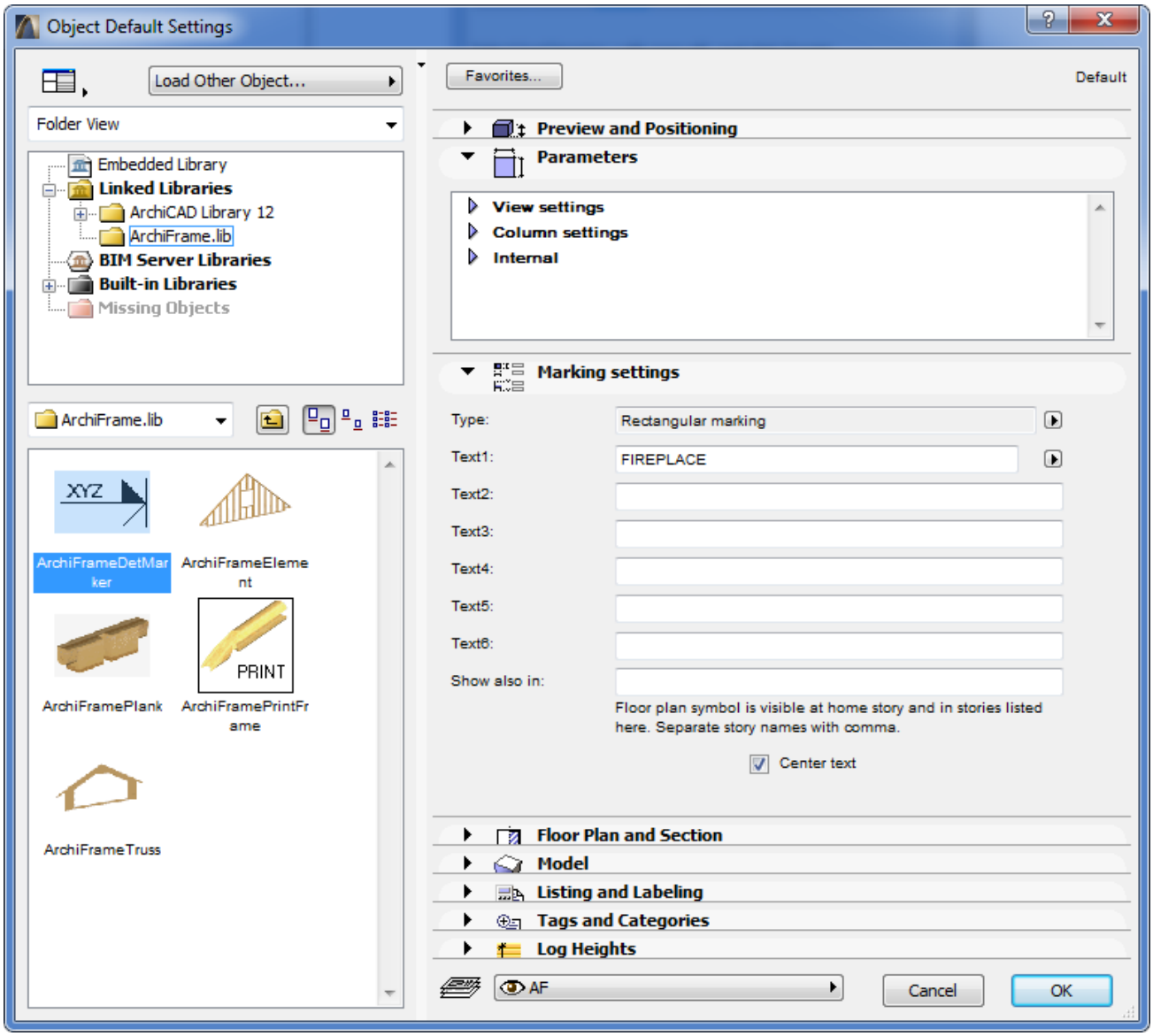
First the walls in question are selected and *Create* will create the markings.

20.3 Columns

This operation adds column markings with *ArchiFrameDetMarker*-object. Columns modeled with ArchiCAD objects may require programming into the settings file.

20.4 Adding other parts manually

For example, fireplaces are added manually using object ArchiFrameDetMarker before automatic dimensions are applied as shown below:



Marking is added to the floor plan and stretched, then moved to the correct place.

Other kinds of marking are added in the same way.

20.5 Automatic dimensions

After the foundation drawing has been completed and any additional items are placed, the dimension lines are added with the *Create*-button. Automatic dimensions require that different items like plinth and load bearing walls are on separate layers. Another requirement is that there must be a plinth to anchor other items too.

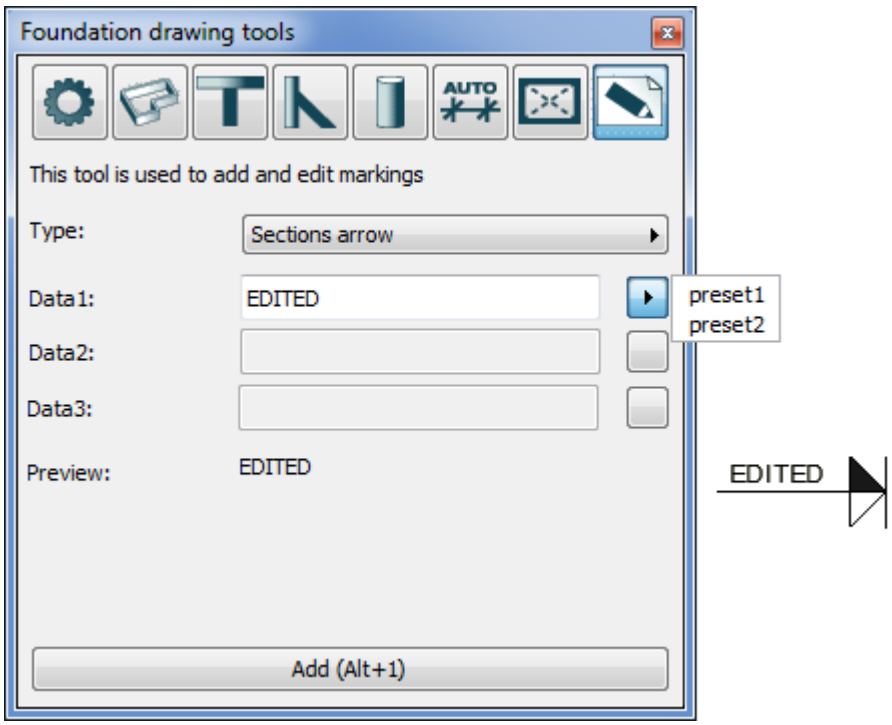
Without any selection, this tool adds dimension lines to every element. For example, if a fire place is added after creating the dimension lines, the tool will only create a dimension line for that item.

20.6 Cross dimensions

This tool adds dimension lines anchored to plinth and slab marking corners. ArchiFrame searches for the closest corner at the clicked place. This makes clicking easier.

20.7 Markings

This tool is used for example to add references to external detail drawings. The preset lists are defined in the settings file and it is very useful to add frequently used detail names there. For example below, the text EDITED is written manually to the Data1 input:



21 XML-settings for an ArchiCAD element

The settings are usually given the same way as in corresponding ArchiCAD setting dialog. All dimensions are given in meters using a full stop as the decimal separator (1.234) and angles as degrees. Sometimes these settings are used to filter elements but usually the settings are for new elements.

ArchiCAD attributes (line type, fill etc) are referenced with number, name or partial name which may contain wild cards?*. When using names it is possible to have many names separated with vertical bar |. This may be useful to support many languages in single setting file. For example, concrete in English and in Finnish: *concrete*|*betoni*.

21.1 <floor>storey</floor >

Storey is numeric or textual as attributes.

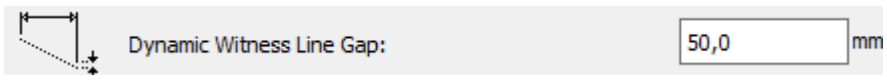
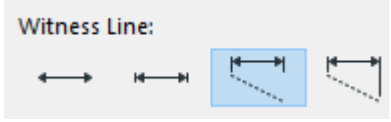
21.2 <layer>layer</layer>

Numeric or textual.

21.3 <elemtype>number</elemtype>

Used when searching elements. The numbers for different element types are found with Google search *API_ElemTypeID*.

21.4 <elemparam name="xxx">value</elemparam>

Name/xxx	Description
fontname	Font name.
fontsize	Font size.
fontstyle	Font style, list using comma as separator that may contain: bold, italic, underline.
just	Justification: left, center, right, full.
pen	Default pen.
linetype	Line type.
dimensiontype	Dimension line type, possible values: • linear • cumulative
dimlinepen	Pen for dimension lines.
markersize	Marker size for dimension line.
markerpen	Marker pen for dimension line.
markertype	Marker type for dimension line: APIMark_CrossLine, 0 APIMark_EmptyCirc, 1 APIMark_SlashLine, 2 APIMark_OpenArrow30, 3 APIMark_ClosArrow30, 4 APIMark_FullArrow30, ... APIMark_SlashLine45, APIMark_CrossCirc, APIMark_OpenArrow90, APIMark_ClosArrow90, APIMark_FullArrow90, APIMark_FullCirc, APIMark_PepitaCirc, APIMark_BandArrow
witness	Dimension witness line type, numbers 0-3: APIWtn_None = 0. APIWtn_Small = 1. APIWtn_Large = 2. APIWtn_Fix = 3.
dimwitnessgap	From dimension tool settings, if witness = 2:   If witness=3: 

	
radius	Angle dimension arc radius.
level	Level for wall or object.
height	Height for wall.
thickness	Thickness for wall.
fill	Fill, for walls building material starting from AC17.
cutfill	From Cut surfaces settings.
cutlinepen	From Cut surfaces settings.
cutfillpen	From Cut surfaces settings.
cutfillbgpen	From Cut surfaces settings.
material	3D material.
materialrgb	Material in RGB-format. The color is between 0…1. For example 100% green is 0, 1, 0.
libname	Libpart name for searching.
angle	Angle for object in degrees.

<objparam name="xxx">value</elemparam>.

Sets library part's parameter. Attributes can be given as text. It is possible to set the attribute type when setting textual value for example to *integer*-type parameter. For example setting fill to *integer*-parameter: <objparam name = "int_par" type = "fill">*concrete*</elemparam>. Attribute type must have one of following values: "linetype", "material", "materialrgb", "fill" or "composite".

These special values can be used as objparam-tag’s name attribute:

- #id, object’s ID
- #useobjlinetype, set value of [ ] Use object’s line type setting

For bit fields attribute bitmask="x" is given. For example, with value 8 only bit 3 (0-based) will be set to target. Value 0 means clear bit and any non-zero value will set the bit.

It is possible to set numeric tables with following syntax, array will be automatically expanded:

```
<objparam name="iMc">
  <row><col>101.00000000</col><col>167.63750758</col><col>90.00000000</col><col>-
0.05560000</col></row>
  <row><col>305.00000000</col><col>1.00000000</col><col>0</col><col>0</col><col>-
0.15370000</col><col>0.02150000</col></row>
  <row><col>305.00000000</col><col>3.00000000</col><col>0</col><col>0</col><col>-
0.15370000</col><col>0.02150000</col></row>
  <row><col>201.00000000</col><col>167.63750758</col><col>90.00000000</col><col>-
0.05560000</col></row>
</objparam>
```

22 Lua scripts

Lua is an open source scripting language that is used for various automations. Information about the language is available at: http://www.lua.org.

ArchiCAD 21 and later: Any ArchiFrame *Lua*-script is using western 8-bit character set and strings are converted to/from utf-8. An exception to this are scripts loaded from xml-files – these scripts are using utf8 always internally. Script may change this default behaviour by setting global gScriptUtf8 to values:

- Missing/nil, default.
- 0, script uses western locale.
- 1, script is fully utf8.

Useful code pieces:

```
-- Dumps given variable into string
-- To use: ac_msgbox(dump(v))
function dump(o)
  if type(o) == 'table' then
    local s = '{ '
    for k,v in pairs(o) do
      if type(k) ~= 'number' then k = '"'..k..'"' end
      s = s .. '['..k..'] = ' .. dump(v) .. ','
    end
    return s .. '} '
  else
    return tostring(o)
  end
end
```

22.1 Built-in support functions

22.1.1 ac_elemget (strGuid [,strOptions])

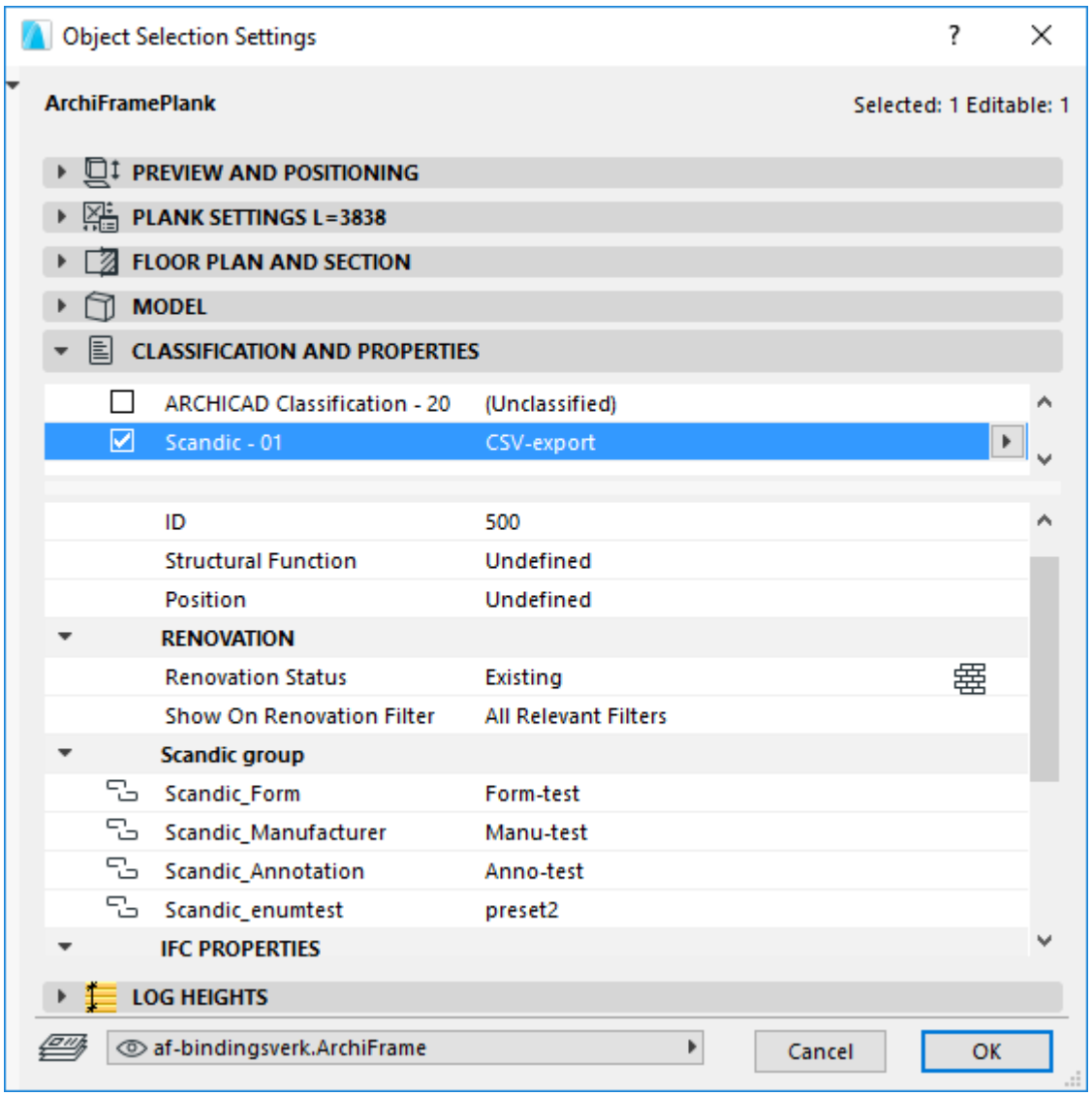
strOptions, specify to get more data. Multiple values can be used, and possible values are:

- props=1 Give all set properties in returned table field props (works starting from ArchiCAD 22)

Returns table of given element having fields:

- header.typeID, element type number, 6 = object.
- header.floor, element floor number.
- header.layer, element's layer number.
- header.libGuid, library part guid if any.
- If requested, field prop which is array of ArchiCAD properties set for the element. Property name is the array key, for example value=prop["propname"].strvalue. ArchiFrame sets property name to lowercase. The fields are:
    - numvalue, set if it is a number
    - strvalue, set if it is a string value
    - Value may be missing if property value type is currently unsupported (like value list)

For example, following properties:



Are returned from expression elem=ac_elemget (elemguid, "props=1") as:

```
elem.props["scandic_form"].strvalue="Form-test"
elem.props["scandic_manufacturer"].strvalue="Manu-test" …
```

22.1.2 ac_findobj (strObjName [, nUseSelection])

Fills table of selected or all object in active storey. Table contains object guids.

- nUseSelection, 0 = get all; 1 = get selected or all if no selection; 2 = just give selected objects.

Returns: The table.

22.1.3 ac_getobjparam (strObjGuid, strParamName [, nRowNum, nColNum])

Gives requested parameter from the object, door or window. Parameters nRowNum and nColNum indexing starts from 1. Value 0 is ignored and -1 returns the table dimension. For example to get number of rows, set nRowNum to -1 and nColNum to 0.

Special *strParamName*-values:

- #id, returns the element ID.
- #guid, returns element's internal GUID.

- #floor, returns the storey number.
- #pos.x, returns x-co-ordinate of an object or middle pos for opening.
- #pos.y, returns y-co-ordinate of an object or lower pos for opening.
- #level, level relative to object's home storey.
- #levelglob, added also floor zero level.
- #angle, returns object angle in radians or reflected state for opening (0/PI).
- #mirroring, returns 0/1 reflected state for objects and doors & windows.
- #libname, returns library part name.
- #libguid, the same as GUID.
- #pen, default pen.
- #linetype, line type.
- #useobjlinetype, use object's line types.
- #mat, 3D material index
- #matname, 3D material name

For doors and windows:

- #wido_sill, sill value as GDL global WIDO_SILL.
- #oside, true if mirrored to x axis.
- #refside, true if mirrored to x axis same as oSide.

For ArchiFrame elements:

Value	Description
"#af_elemdata"	Gives information of the plank related to the element. The table fields are: • x1, y1, z1, x2, y2, z2 Plank's base line coordinates relative to the element surface. • bx1, bx2, bz1, by1, by2, bz2 Plank's bounding box in parent elemen's coordinates. Z axis is element's watching direction and is calculated ignoring any cuts in the plank unlike x- and y-coordinates. • Poly, plank's polygon. • index Plank's 1-based index. • rotangle, rotation angle in degrees from element's watching direction • ptr, see af_request/plankinfo. If used from projection scripts (dimension collector or *DoProjSettings()*-function), ptr refers to projection plank's master/3D plank. Also, then following extra fields are available: • typetag, related element layer's type: core, intstud etc.

Also, all values described <elemparam name="xxx">value</elemparam> are supported. Character # must be given before the element's parameter name, for example #cutfill. Attribute values in this case are returned as two values: numeric and attribute name. Exceptions:

- #materialrgb returns single string. It is the material's colour in format "RGB" where all color values are between 0…1. Decimal separator is dot, for example "0.5, 0, 0". If the material is GENERAL, the return value is "0".

Returns: The parameter numeric/string or nil = parameter missing.

22.1.4 ac_getattrinfo (strAttrGuid) or ac_getattrinfo (nTypeNum, nAttrIndex)

Gives information of an attribute:

- Type, "fill"|"composite"|"type_as_number".
- Name, attribute name.
- Thickness, for composite the total thickness and for composite layers, the layer thickness.

  For composites only:
  - num_layers, number of layers.
  - Layers, table of layers each having similar fields as requested for fill, except thickness also set.

Returns: The table

22.1.5 ac_objectopen (strGuid)

In special cases strGuid can be nil and the function will open the default object depending on the special usage case. Opens the object for further processing. The guid may also be a C pointer obtained from ArchiFrame. The C pointer start with @ and is followed by 8 (32 bit) or 16 (64 bit) characters. Use pointers with care – misuse will crash the program.

Parameter strGuid may begin with character >. For example, if used in plank observer script, it will work with last saved version skipping ArchiFrame's object cache buffer.

Returns: Nothing, in case of error, Lua error is raised.

22.1.6 ac_objectget (strParamName [, nRowNum, nColNum])

Works as ac_getobjparam but for object opened with call to ac_objectopen.

22.1.7 ac_objectset (strParamName, value [, nRowNum, nColNum])

Sets the parameter only if changed.

Returns: Nothing, in case of error, Lua error is raised.

22.1.8 ac_objectclose ()

Saves changes if any to opened object.

22.1.9 ac_environment (strSel [, additional parameters])

Selector can be one of the following:

ac_environment ("lang").
Returns language of theArchiCAD: "eng", "fin" or "swe".

ac_environment ("ntos", nVal, strType, strPrefs)
Converts number to string using current ArchiCAD-settings. Parameter strType can be "length"| "angle" | "area" | "volume". strPrefs is either "work", "dim"or "calc" and it reflects to ArchiCAD working, dimension lines and calculation settings. Parameter nVal is the numeric value. Returns string value.

ac_environment ("units", strType, strPrefs).
Returns unit type as number and second ret value unit string. Parameters are as "ntos" but no nVal parameter is not given. First return value numbers are:

```
APIUnit_Metric=0,
APIUnit_Centimetric,
APIUnit_Millimetric,
APIUnit_FootInch,
APIUnit_FootDecInch,
APIUnit_DecFoot,
APIUnit_Inch,
APIUnit_DecInch,
APIUnit_Decimetric,
APIUnit_DecYard,
APIUnit_Gallon
```

Please note that from AC22 on the values are (decimetre added so check the unit string for imperial check):

```
Meter,
Decimeter,
Centimeter,
Millimeter,
FootFracInch,
FootDecInch,
DecFoot,
FracInch,
DecInch
```

ac_environment ("userorigin").
Returns X, Y, Z of the user origin.

ac_environment ("reflevels").
Returns two numeric reference levels defined in AC.

ac_environment ("parsetext", strIn).
Parses texts having ArchiCAD automatic texts like <ARCHITECT>. Returns parsed string.

ac_environment ("createguid").
Creates new GUID in form {4CABE438-C7F5-4B43-A6DF-B07114D6BDDF}.

ac_environment ("strreplace", strSource, strFind, strReplace)
Does find & replace without any character being pattern.

ac_environment ("layer", strName, nShow, nSetMask).
Shows (1) or hides (0) given layer(s). Name may contain wild characters * and ?, for example, “structural*”. If parameter nShow is -1, it will return the state of the layer (if wild card, the state of first matching layer). State is bit combination:

- 1, bit0, is visible: 0 = hidden, 1 = visible.
- 2, bit1, is locked.
- 4, bit2, is mine in TeamWork.
- 16, bit4, is forced to wireframe.
- 32, bit5, the layer belongs to an Xref.

ac_environment ("rebuild", nRegenerate).

Rebuilds display (for example after showing/hiding layers). With nRegenerate value 1 rebuild is done deeper.

ac_environment ("tolog", strText).
Puts given text to a log window (not a dialog).

ac_environment ("bittest", value, bitnum).
Returns value 0/1 for given bit in the value. Parameter bitnum must be between 0…31.

ac_environment ("getsel").
Returns table of selected element's guids. Processes all element types. Return value nil means no selection.

ac_environment ("setsel", tblGuids).
Sets selection to given guid tbl which must start indexing from [1]. Nil = remove selection.

ac_environment ("suspendgroups", bSusped).
Sets suspending the groups to given bool value (false = nope, others = suspend). Returns previuous state of suspending: false = not suspended, true = yes.

ac_environment ("getcurrwindow").
Returns type of current window:

- 1, floor plan (Google for text APIWind_FloorPlanID to get all values).
- 2, section.
- 3, detail.
- 4, 3D window.

ac_environment ("getall", nFilterBits, nElemType1, nElemType2,…, nElemTypeN).
Returns table of all element's guids (never nil but table can be empty).

NFilterBits as in API-calls, for example APIFilt_OnVisLayer|APIFilt_OnActFloor is 2+4 = 6. -1 = default: APIFilt_OnVisLayer|APIFilt_OnActFloor if current window is floor plan; APIFilt_OnVisLayer if current layer is anything else. ArchiFrame sets global gnListingAll to value 1 if selected (*) whole model even if in floor plan.

ac_environment ("filedialog", bSaveAs, strTitle, strFilter, strFileExt).
Runs open or save as file dialog if bSaveAs is true. Parameter strTitle is the title of the dialog box, strFilter is in format "Text files (*.txt)|*.txt|All (*.*)|*.*|", strFileExt is the default file extension without the period. Returns nil if cancelled.

ac_environment ("findattr", nAttrType, strAttrName).
Find given attribute by name, strAttrName may contain wildcards and multiple values may be given with vertical bar. Returns 0 if not found.

```
ac_objectset("iMc", ac_environment("findattr", 6, "*tre-osb*|Tre-Mahogny"), add, 10)
```

nAttrType must be one of following (number increases by one for each line).

```
API_PenID = API_1,
API_LayerID,
API_LinetypeID,
API_FilltypeID,
API_CompWallID,
```

```
API_MaterialID,
API_CityID,
API_LayerCombID,
API_ZoneCatID,
API_FontID,
API_ProfileID,
API_PenTableID,
API_DimStandID,
API_ModelViewOptionsID,
API_MEPSystemID,
API_OperationProfileID,
API_GraphicOverrideID,
API_BuildingMaterialID,
```

ac_environment ("luacomchars", nCodePage)
Sets LuaCOM-interface's global character set. Default is ANSI (0) and other possible value is UTF8 (65001). Returns previous setting that must be restored so that other sripts are not affected.

ac_environment ("acver")
Returns current ArchiCAD-version as number: 19=1900, 20=2000, 21=2100 etc.

22.1.10 ac_msgbox (strText)

Shows message box. Useful for debugging.

22.1.11 ac_mbstoutf8 (strIn)

Converts given text encoding into UTF8. Returns converted string.

22.1.12 ac_geo (strSel [, additional parameters])

Selector can be one of following:

ac_geo ("linex", x1first, y1first, x2first, y2first, x1second, y1second, x2second, y2second).
Gives intersection point of two lines. Lines are infinite length. Returns: No value = no intersection pt, (X, Y) = the intersection point.

ac_geo ("linedist", x1, y1, x2, y2, x, y).
Calculates point distance from line. Return three values: Distance from the line (negative to the left and positive at right hand side), distance from beginning point towards end (negative before line starting point) and length of given line.

ac_geo ("calctranworld3", origfrom, vecxfrom, vecyfrom, veczfrom, origto, vecxto, vecyto, veczto).
Calculates transformation matrix [1...12] for given coordinate worlds. Returns table [12]. To transform a point, use:

Xnew = x*tblTran [1]+ y*tblTran [2]+ z*tblTran [3]+tblTran [4].
Ynew = x*tblTran [5]+ y*tblTran [6]+ z*tblTran [7]+tblTran [8].
Znew = x*tblTran [9]+ y*tblTran [10]+ z*tblTran [11]+tblTran [12].

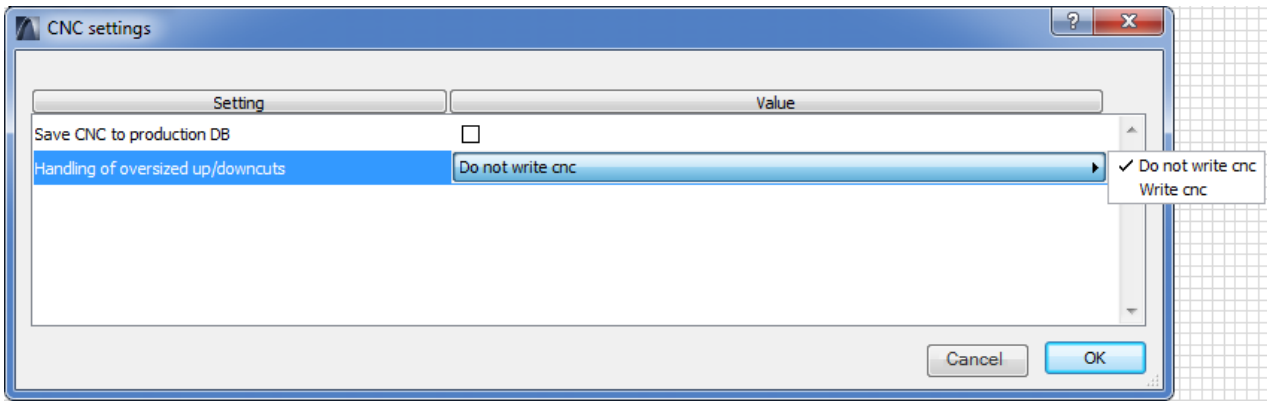
ac_geo ("matmul", mat1, mat2).
Multiplies matrix1 with matrix2. Matrices are in format calctranworld3 uses, col 4 is just added.

ac_geo ("linepolyx", x1, y1, x2, y2, tblPolys).
Calculates intersection of given line and table of polygons (format as in af_request ("getpoly",…)). TblPolys is 1-based table of separate polygons (to be able to handle split surfaces).

Returns 1-based table of line segments each segment having fields x1, y1, x2, y2.

22.1.13 ac_optiondlg (strID, strTitle, tblSettings)

Runs setting dialog and saves settings application a specific way – usually to user specific configuration file and to the current pln-file. Dialog looks like:



And code to run this is:

```
gbAskOptions=true
gbCnc-ToDb=true

-- CNC- writing settings, raises error if canceled
function GetCnc-Options()
            local       tblSettings, bRes, sErr, s, n

            -- Ask only once
            if gbAskOptions==false then
                        return true
            end

            tblSettings={}
            tblSettings             [1]         ={}
            tblSettings[1].cfgonly  =1
            tblSettings  [1].type   =2
            tblSettings[1].prompt   ="Save CNC- to production DB"
            tblSettings  [1].key    ="cnc-db"
            tblSettings[1].defvalue =0

            tblSettings             [2]         ={}
            tblSettings[2].cfgonly  =1
            tblSettings  [2].type   =1
            tblSettings[2].prompt   ="Handling of oversized up/downcuts"
            tblSettings  [2].key    ="cnc-cut"
            tblSettings[2].valuelist="\"1:Do not write cnc-\",\"2:Write cnc-\""
            tblSettings[2].defvalue =1

            bRes,sErr=ac_optiondlg("LDCN", "CNC- settings", tblSettings)
            if not bRes then
                        if sErr~=nil then
                                    RaiseError(sErr)
                        else
                                    RaiseError("CANCELLED")
                        end
            else
                        if tblSettings[1].value==0 then
                                    gbCnc-ToDb=false
                        else
                                    gbCnc-ToDb=true
                        end
            end

            gbAskOptions=false
```

```
end
```

Parameters:

Param	Description
strID	Four letter long ID of the settings. Always start ID with LD not to mix it with application's own settings. For example LDCN.
strTitle	Dialog title, set to nil if you just want to get the saved settings into the table tblSettings. No dialog is shown in this case.
tblSettings	Table of the settings items.

Return values: Boolean bRes, string strErr: bRes is true if dialog was run ok and closed with OK, false in case of internal error or dialog cancelled. In case of internal error strErr is set.

Settings table has fields:

Field	Description
cfgonly	Optional number: • 0 or missing, save option to cfg and pln-file. • 1, save setting only to cfg-file (global to user).
type	Numeric type: • 1, value list/combo box. • 2, check box. • 3, length edit. • 4, any text.
prompt	To setting-column in the dialog.
key	Unique key for the setting inside this setting dialog.
defvalue	Default value to be used if no previously saved settings.
value	If given for input: Value is not loaded from settings but instead this value is used. On output: The actual setting.
valuelist	Given if type is 1. Contains items in format: "Number: Item text". Number is used as unique ID for the item. Items are separated with comma. Note Lua syntax for adding quotation marks in the string: "\"1:Do not write cnc-\", \"2:Write cnc-\"".

22.2 ArchiFrame-specific functions

22.2.1 af_request (strSelector [, additional parameters])

Parameter strSelector defines what is requested:

Value	Description
"singlemat"	TblMat = af_request ("singlemat", "id"), returns table for single material or nil if material does not exist. Fields are: • id, material id. • name, name. • shape, "block" | "plane" | "round".

	• thickness, height, in meters. • height, in meters. • m3factor, volume factor.
"matlist"	Returns table of all materials in material list.

"plankinfo"	tblRes = af_request ("plankinfo", strPlankGuid, plankInfoProj), returns information about given plank. If strPlankGuid is missing, uses currently opened object/plank. If plankInfoProj is given, it contains projection information. More info later.
	
	It is possible to get geometry information for element objects or even ArchiCAD beams, walls and columns too by giving strPlankGuid. Table has fields:
	• ptr, plank's or board's internal C-pointer (memory address) starting with character @ and followed by 8 or 16 digit hex address. Can be used as plank identifier in plank operations and in ac_objectopen. Ptr is valid only during current operation – it may not be saved for later use and misusing ptr will cause whole program to crash.
	• guid, plank's guid (nil for new pieces not yet created into ArchiCAD).
	• type, 1 = plank, 2 = log object, 4 = board.
	• begc, start point in model coordinates beg. X, Y, Z.
	• endc, end point in model coordinates end. X, Y, Z.
	• len, plank length.
	• width, width.
	• height, height.
	• typename, plank/board type name for listings.
	• vecx, vecy, vecz, plank's geometry axes.
	• masterelemguid, guid of the floor plan/3D piece if current part is projection piece
	• ownerelemguid, if belonging to an element, the owning element's guid. Missing if not part of an element. You can use for example "elem_quantities" with this guid.
	• ownerelemid, if belonging to an element, the owning element ID string. Missing if not part of an element.
	• del, true = marked for delete, false = nope.
	• begoff, for log object cnc-writers: Offset from X-coordinate zero point to actual plank start. Positive extends, negative shortens (opposite to plank dir).
	• tblWeather, set if the plank is part of weather boards. Has fields:
	  ○ pa, pb, pc, pd, plane equation – pa, pb, pc is the normal vector towards the wall.
	  ○ vecy, vertical vector X, Y, Z.
	  ○ vecx, horizontal vector always pointing right looked to the surface from normal vector's opposite direction.
	• tblSides, table of geometry for each side [1]…[6] each having fields:
	  ○ origc, 3D origin.

	○ vecx, vecy, Direction vectors for surface axes. ○ vecz, normal vector pointing outside of the piece. ○ dx, dy, surface size in X- and Y-directions. If plankInfoProj is given, the plank is projected to given plane. Result is saved to field elemdata and has the same members as with #af_elemdata. Parameter plankInfoProj must contain fields (plankinfo queried for an element is fine): • begc, plane's origin. • vecx, vecy, vecz, 3D axes. Returns nil if failed.
"getpoly"	tblRes = af_request ("getpoly", tblSettings, strPlankGuid), returns polygon information for given strGuid elem (plank, board, element). If strPlankGuid is missing, uses currently opened object/plank. tblSettings for boards may be nil. For the planks contains fields: • side, 1 = top, 2 = front, 3 = bottom, 4 = back. Which side to process. • polynum, if surface is split by groove(s), there may be multiple polygons. Number of the polygon to return 1…N. If missing 1st one will be returned. • tooldia, milling tool diameter for overbegin and overendin, default = 10 mm. • holes, value: ○ Missing, give full polygon with holes. ○ 0, throw holes away. ○ 1, will return list of holes (openings) in the element. If given result field poly is 1-based array of the holes polygons. • expand, if given, the value to expand the polygons, positive gives bigger polygons (also if holes requested), negative smaller. • fromboard, with value 1 gives polygon list from the boards belonging to the element (same format as for holes). May contain many polygons if surface is split. In owning element's rotated coordinate world. With value 2 the polygon extra info is taken from board object's parameter iPolygon. With value 2 parameter strPlankGuid must be a board guid. Result fields overbegin, overendin and cutfromobj come from object instead of calculated values. Use value 2 only if board object's parameter iSawlines is nonzero. NOTE! If holes are requested, they will be returned clockwise instead of standard ccw for contour line. Value 2 implies also givelist = 1. • givelist, with value 1 returns always [1]-based polygon list. Otherwise single polygon unless holes or fromboard is given. • calcopening, with value 1 calculates openingtype-field using the related element object's links to ArchiCAD-walls/openings. • camera, if given contains projection plane as given from "plankinfo"-request: ○ begc, plane's origin.

<table>
<tr><td></td><td>
vecx, vecy, vecz, 3D axes.

Returns table having following fields or nil = not supported type:

numpolys, number of polygons (in case of split surface).
x1, y1, x2, y2, the bounding box of the polygon. Currently polygon is always normalized to have x1, y1 at 0, 0. Not valid if holes or expand given.
xoff, yoff, currently always 0, 0. The offset from the lower left corner of the raw board material. Not valid if holes or expand given.
area, area of the polygon(s) with holes reduced.
poly, [1]-based array of polygon points from the front surface. Contour end points are not duplicated to be same as contour begin. Each point has members:

x,y, the point position.
isbeghole, set only if point is begin of a hole.
isendcontour, set only if point is end of a contour.
isendpolygon, set only if point is last point of the polygon, always with isendcontour.
overbegin, length of line inside the polygon extended by tooldia from next point to current point. Use to check for any overcut in cnc.
overendin, length of line inside the polygon extended by tooldia from previous point to current point. Use to check for any overcut in cnc.
cutfromobj, if board object had parameter iSawlines set, this tells if the cut from current point to next point is set by user: 0 = no cut, 1 = do the cut.
openingtype, if edge from current pt to next pt is related to an opening (set only for element objects):

0, not related to an opening or related to an empty opening.
11 window top.
12 window right.
13 window bottom.
14 window left.
15 window unknown.
21-25 door similar.
111-115 empty window similar.
121-125 empty door similar.

err, error message if there was an error in extend operation. nil =ok.

</td></tr>
<tr><td>“delplank”</td><td>Marks currently opened object/ac_objectopen() to be deleted. Deletes all related projections etc.</td></tr>
<tr><td>"getselplanks"</td><td>Returns guid table of selected plank objects. If selection contains projections, its master 3D plank will be returned instead. For historical reasons indexing of this starts from [0] and you need to use Lua pairs instead of ipairs:
<code>for i,v in pairs(tblSel) do</code></td></tr>
</table>

	xxx end To get only boards, give second parameter “board”. With any other value only planks are collected. tblBoards = af_request ("getselplanks", "board")

"getsellisting"	Returns table to be used as source elements for listing/cnc-. If anything selected, the lists are limited to the selection. If no selection, the result contains:
	• All visible elements in current floor if in floor plan.
	• All visible elements in the model if in 3D.
	Returns 1-based array of tables containing fields:
	• guid.
	• ptr.
	• id.
	• type, element type as:
	  ○ 1 = plank.
	  ○ 2 = log.
	  ○ 4 = board.
	  ○ 8 = ArchiCAD wall, beam or column.
	If no elements, returns nil.

“getbegendplane”	TblPlane = af_request ("getbegendplane", "beg", nOptions). Returns global cutting plane for current plank’s begin or end (replace “beg” with “end”). Parameter tblPlane has fields pa, pb, pc, pd defining the plane equation. With numeric values any side can be queried, replace “beg” with values (see About the planks): • “1”, top. • “2”, front. • “3”, bottom. • “4”, back. • “5”, begin. • “6”, end. If parameter nOptions is missing or is 0, first cut for begin or end will be returned. With value 1, the result is always straight relative to the plank.
“adjustbegend”	af_request ("adjustbegend", "beg", tblPlane). Adjusts currently opened plank’s end (“beg” or “end”) to given global plane. Parameter tblPlane has fields pa, pb, pc, pd defining the plane equation.
"cutplank"	af_request ("cutplank", tblPlane). Splits currently opened plank into two pieces. Parameter tblPlane defines the cutting plane in global coordinates. Returns values: • None → no valid cut – plank untouched. • strPtr → Return value is the guid/ptr string of the new plank.
"joinplanks"	af_request ("joinplanks", strGuid1, strGuid2). Joins given planks together. The requirements to join are: • Planks must be parallel.

	Returns: true = ok and strGuid2 marked to be deleted at the end of the undo-scope, nil = nothing done.

"adjusttoside"	af_request ("adjusttoside", tblSettings, strGuidTarget, tblGuidOperators).
	Adjust to side operation as with adjust to side tool palette. Parameter tblSettings may have fields.
	• doadjust, 0 = no adjusting, 1 = adjust (default).
	• gap, gap in the joint, default = 0.
	• cutop, how to cut:
	  ○ 0 = just cut (default).
	  ○ 1 = remove upwards.
	  ○ 2 = remove downwards.
	  ○ 3 = keep the part closer to set keep origin.
	• endshape, the end shape:
	  ○ 0 = angled, extend and keep existing cuts (default).
	  ○ 1 = as 0 but do not extend.
	  ○ 2 = as 0 but delete any existing cuts.
	  ○ 3 = straight not intersecting with operator piece delete existing cuts.
	  ○ 4 = straight extend to touch operator plane fully delete existing cuts.
	  ○ 5 = as 4 but do not delete existing cuts.
	• planepa, planepb, planepc, planepd, use this plane instead of operators.
	• keeporig, table xyz of origin to define which part to keep.
	• cuttoendplane, adjust to last intersecting plane instead of the first: 0 = no (default), 1 = yes.
	• expandopfind, if given, the operator is expanded with given value to test if there is an intersection between the planks. Useful to handle cases where angled planks touch just with a corner.
	• skipopsides, bit mask to skip checking operator sides. bit0 = top, bit1 = front etc. Bit 1 means skip checking this operator side. Default = 0, do not skip any side.
	• domark, 0 = no marking (default), 1 = markings.
	• marklines, 0 = to sides, 1 = to center.
	• marktext, text for marking, tags as in the tool palette.
	• markposx, text position 0 = left, 1 = center, 2 = right.
	• markposy, text position 0 = bottom, 1 = center, 2 = top.
	• markfontsize, in centimeters, 0 = default.
	• markdir, text in line's direction.
	
	Returns: nil = nothing done, tblNewPlanks = table of edited/new planks from the operation.

"editplank"	af_request ("editplank", tblEdit). Edits single plank, tblEdit has following fields (give only the fields to edit):

	• guid, must be given. • mat, plank material id. • elemdata, geometry in element's coordinate world. If begin or end is edited, give all three XYZ-coordinates.
"cmpplanks"	af_request ("cmpplanks", strGuid1, strGuid2). Compares two planks (not comparing the IDs). Comparison can be used for example in listings to make sure that planks having similar ID are similar. Return value: • nil, could not compare – only planks and boards are supported. • <0, item1 < item2. • 0, item1 = item2. • >0, item1 > item2.
"mc_getnewmcid"	af_request ("mc_getnewmcid"). Generates new machining id for the plank opened by ac_objectopen().
"mc_findtextpos"	af_request ("mc_findtextpos", nSide14, dX1, dTextWidthMeter). Calculates position for plank ID that is as close as possible to dMidPos. Moves the text away from any existing markings and grooves.
"mc_explodepanel"	af_request ("mc_explodepanel", guidBoardElem, tblSettings). Explodes given *ArchiFrameBoard*-object into separate temporary planks that will not be created into the model (to be used for listings and cnc-). • guidBoardElem, guid of the ArchiFrameBoard-object. • tblSettings, currently not supported. Returns 1-based table of plank @pointers. Nil = no planks created.
"mc_nailings"	af_request("mc_nailings", tblTargets, tblGuidsBehind, tblSettings) Makes nailings to given planks or boards. Parameter tblSettings may contain following fields: • spacing_edge, spacing at board edges. Default = 100 mm. • spacing_mid, spacing at middle of board. Default = 200 mm. • mindist_board, minimum distance from the board edge. Default = 10 mm. • maxdist_board, maximum distance from the board edge. Default = 25 mm. • mindist_plank, minimum distance from the plank edge. Default = 10 mm. • mindist_nails, minimum distance between nails in planks. Default = 25 mm. • spacing_plank_cross, spacing for crossing planks, default = 50 mm. • spacing_plank_parallel, spacing for parallel planks, default = 200 mm.
"mc_sawcuts"	af_request ("mc_sawcuts", guidElem, bViewBack). Finds all placed *ArchiFrameSawCut*-objects that are over any projection of boards owned by given element using default orientation having saw blade on the right side. Parameters: • guidElem, GUID of the element whose saw cuts are returned. • bViewBack, false = View from front (default), true = from back.

	Returns 1-based table of saw cuts ordered by lineindex (main contour first) each item having fields: • x1,y1,x2,y2, coordinates on given element's coordinate system looked from front. • angledeg, saw blade angle looking in saw line direction positive tilts to left and neg to right. • depth, cut depth, 0 = through the layer. • overbeg, 0 = no overcut at begin, 1 = overcut at begin. • overend, as previous for the end. • bladeside, -1 = left from the line, 0 = center, 1 = right. • elemguid, saw blade element guid. • ishole, is it for polygon's hole 0/1. • lineindex, order number inside polygon, holes indexes start from 1000000, if polygon is split, the gap between each is 1000. Successive numbers combine single contour line (for polygon contour or poygon hole).
"mc_getcutinfo"	af_request ("mc_getcutinfo", tblInfo) Calculates information if a cut projected to one of the plank's sides. Parameters are given in a table which has fields: • guid, required GUID of the element that is processed • mcindex, required 1-based index of the machining to process • cutlineside, optional 1-based side number where to anchor the cut Returns a table having fields (btl-fields refer to btl cut code 010), angles are in radians: • fullcut, does the cut go cut whole plank: 0=no, 1=yes • side1, 1-based best surface to anchor the cut • x1, y1, x2, y2, cur plane's line on the surface • btl_p06, angle between cut edge and reference edge • btl_p07, inclination between face and reference side • btl_p08, lengthwise angle at cut bottom • btl_p11, depth of the cut • btl_p12, Length of the cut (from x1,y1 → x2,y2)
"aflang"	Language selected into ArchiFrame: eng/fin/swe/nor/ger/ita.
"dim_section"	af_request("dim_section", strLayer) Available only in dimension line scripts. Builds table of section polygon bound boxes. Section is made from the ArchiFrameElement-objects ignoring related planks/boards. Each returned table cell has these fields: • x1, y1, x2, y2 Polygon bounding box • ?_minx, ?_miny, ?_maxx, ?_maxy The closest and furthest polygon point polygon's four edges. Replace ? with left/right/top/bottom, for example point(right_minx,right_miny) is the lowest point on the right hand side of the polygon. If there is just single point at the edge, min- and max-points are the same.
"dim_camera"	af_request("dim_camera", strLayer)

	Available only in dimension line scripts. Gives current projection camera. Returns table having fields: • begc: plane's origin • vecx, vecy: plane's axes • vecz: plane's normal vector, camera's watching direction • pa, pb, pc, pd: plane'd equation

When processing elements, there are these additional requests available:

Value	Description
"elemcorethickness"	Returns the element core thickness.
"elem_getpolyscount"	Returns number of polygons making up the element. Currently always 1.
"elem_getpoly"	af_request ("elem_getpoly ", polyNum). polyNum is currently unused and should be 1. Creates Lua globals that are available at initial create planks phase. Please see Lua-scripts with elements.
"elem_getinfo"	Returns table containing information about current element. Table fields are: • vecx, lengthwise axis. For vertical elements, this has zero z. • vecy, for vertical elements this is 0,0,1. • vecz, the watching direction for the element.
"elem_createplanks"	Issue when script has set up information for new planks. Please see Lua-scripts with elements.
"elem_getstudmat"	Gives ID of random (actually first one sorted by plank's GUID) vertical plank in the element. If no planks yet, gives ID of first material definition of the element.
"elem_marknoplanks"	af_request ("elem_marknoplanks", x1, y1, x2, y2, nDelExisting), marks give rectangle in the element not to get any planks from spacing rules. • nDelExisting, optional parameter to delete existing planks from given polygon. With value 1 deletes plank intersecting the polygon.

"elem_quantities"	af_request ("elem_quantities", strElemGuid, nOpenLayers).
	Gives quantities for single element or layered element. Parameters:
	• strElemGuid, nil = use the parent element given from environment, str = guid.
	• nOpenLayers, 0 = Open just given single element, 1 = Open every layer.
	
	Returns table containing information for separate layers and total quantities.
	• tblelems, the elements.
	• quant, sum of quantities for each layer.
	• geo, origin of master element and 3D bounding box. Fields:
	  ○ orig.x, orig.y, orig.z
	  ○ x1,y1,z1,x2,y2,z2

	• comptype, type name of the composite element type, nil = could not get it.
	• templateid, for own element types the *based on* type id, nil = not own element type.
	• xmlutf8, in this level contains composite xml referencing layers and having settings for the whole composite.
	• guidstamp, guid of the related element stamp object, nil if no stamp
	• tblguidcutlist, 1-based table of the related cut lists, nil if no cut lists
	
	Fields for single element:
	• guid, element's guid.
	• id, element's id.
	• elemtypeid, for example "WALL 173 K600".
	• elemtype, for example "core", "intstud", "boarding_int" etc from type="xxx" xml-attribute.
	• templateid, for own element types the *based on* type id, nil=not own element type.
	• xmlutf8, element's xml-definition in utf8-characters.
	• areanet, element area calculating openings in m2.
	• areagross, openings not reduced in m2.
	• layertypenum:
	  ○ 0 = planks.
	  ○ 1 = boards.
	  ○ 2 = paneling (planks belonging to this layer are paneling/cladding).
	
	• tblplanks, table of planks belonging to this layer containing fields:
	  ○ guid, the plank guid.
	  ○ ptr, temporary pointer to the plank.
	
	• tblboards, table of boards containing fields (nil if no boards):
	  ○ guid, the board element.
	  ○ ptr, temporary pointer to the board.
	  ○ id, the board id.
	  ○ thickness, its thickness.
	  ○ width, height, the size of the board.
	  ○ ispanel, 0 = no, 1 = board contains panel/cladding.
	
	The quantity fields for single element or combined for multilayer element as calculated in weight tools:
	• elemvolume, full volume from polygon area * thickness.
	• elemvolumereduced, the planks reduced from element volume.
	• elemweight, total element weight elemweightreduced + plankweight + boardpanelweight plus related openings weight.

	• elemweightreduced, insulation weight. Doors and windows are also calculated if set so in the weight settings for the layer/composite. • plankvolume. • plankweight. • boardpanelvolume. • boardpanelweight. • openingweight, combined weight of related openings. The geometry information from geo-table: • orig.x, orig.y, orig.z, origin of the core-element. • vecx, vecy, vecz, direction vectors of the element, vecz is the front projection watching direction. • x1, x2, y1,y2, z1, z2, minimum and maximum coordinates of any plank or board in element's axes (see Element object).
"elem_domarkings"	af_request ("elem_domarkings", strXml). Like the xml-definition, may refer to settings inside <*presettings*>-tag: <markings ref = "mark_big"/>.

22.2.1.1 Element layers

af_request ("elem_openparent", guidElem).
GuidElem can be nil to get current element.

Resulting table contains fields:

Field	Description
id	ID of element type from ArchiFrameElements.xml: • <elemtype class="wall" **id="WALL 173+42 VERT"** idstamp="173+42">.
guid	Guid of the core layer/single layer element object.
ptr	Another temporary ID for the master element.
thickness	Total thickness of the layered elements including any empty space between elements.
tblelems	Table of individual elements.

Tblelems is indexed by *1*-based index in the same order as defined in *ArchiFrameElements.xml*.
The table contains these fields:

Field	Description
id	ID of the referenced element or same as parent element if single layer element: • <layer **ref="WALL 173 K600"** anchorname1="Core ext"…
type	Type of the element layer (core for single layer elements): <layer ref="WALL 42x42 VERT" … **type="intstud"**>.
guid	Guid of the element object.
ptr	Another temporary ID for the layer element.
zoff	Offset in watching direction to first surface.
xsize	Maximum width of the element. Lower left corner has always coordinates 0, 0. For walls this is the element length in floor plan.

ysize	Maximum height of the element. For walls this is the height.
thickness	Layer's thickness, logically zsize.
x1off	Offset from core layer's left-hand side: positive extends over core.
x2off	Offset from core layer's right-hand side: positive extends over core.
y1off	Offset from core layer's lower side: positive extends over core.
y2off	Offset from core layer's upper side: positive extends over core.

af_request ("elem_closeparent", nSave).
Closes the current multilayer element.

Parameter	Description
nSave	1 = Save changes, with any other value all changes are discarded.

af_request ("elem_openlayer", guidElem), af_request ("elem_closelayer").
Opens the given layer to edit planks and element polygon shape. Close does not save the changes, it just clears internal information. The changes are finally saved with elem_closeparent.

af_request ("elem_moveside", nCorner, offset).
Moves side of an element layer.

Parameter	Description
nCorner	1 = left, 2 = right, 3 = bottom, 4 = top.
offset	Positive extends, negative makes smaller.

22.2.1.2 Element relations

af_request ("elem_getrelations", guidElem [, xtolerance}).
Gets the relations for single element layer. GuidElem can be nil to process current element layer.

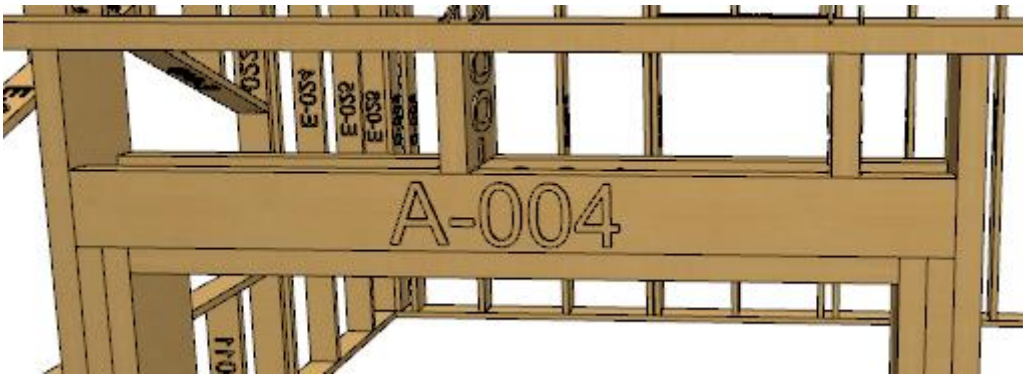
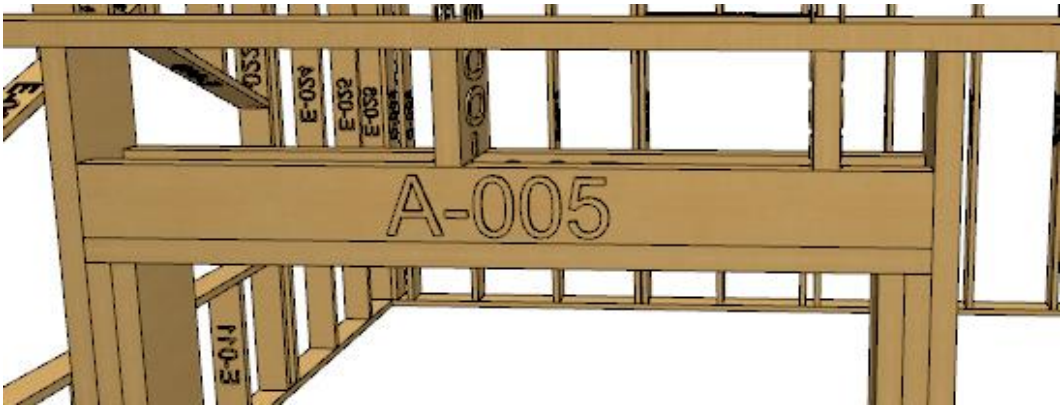
Parameter	Description
GuidElem	Element layer, nil = use current layer.
xtolerance	How far from the intersection the element's end may be, default = 1 mm. Use if core is shortened over that in the corner type.

Returns 1-based table of relations. Each relation items which are tables also have following fields:

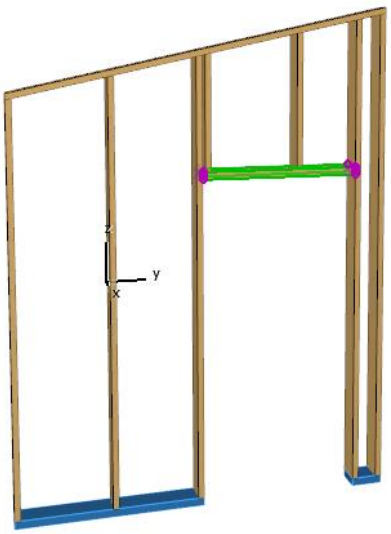
Field	Description
guidother	Guid of the related element layer object.
typeother	Type of the related element layer: <layer ref="WALL 42x42 VERT" … **type="intstud"**>.
reltype	Numeric value: • 1 to begin. • 2 to end. • 3 to middle (or crossing elements).
x1,x2	Intersection coordinates minimum and maximum values in X-axis. Can be outside the element area, negative before left hand side, bigger than length right from right hand side.
x1off,x2off	Maximum distance from min/max point to the intersection of other plank
angledeg	Angle in degrees between current element's X-axis and other's X-axis (which direction may have been swapped). Positive = on the left hand side, negative = on the right hand side.

22.2.1.3 Element lintels

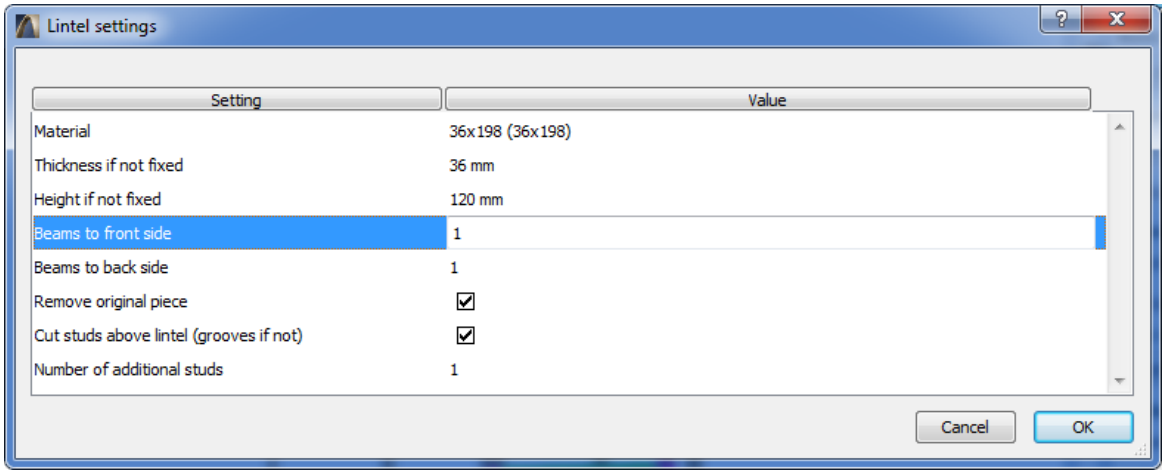
af_request (“elem_lintels”, settings) or af_request (“lintels”, settings).
Creates lintels according to passed setting table. Single lintel settings item contains these fields:

Field	Description
minwidth	Minimum width of the opening to use these rules. Place definition for longest first and continue with shorter ones.
mat	Material ID for the lintel beam.
thickness, height	If material ID is generic without fixed size, these must be given.
front	Number of lintel beams to front side of the element, 0 = none.
back	Number of lintel beams to back side of the element, 0 = none.
delorg	0 = keep source plank and place beams above it.
extorg	0:  1: 
totop	0 = place lintel just above the opening (default). 1 = place the lintel below top sill.
cutabove	0 = extend studs above the lintel beam to bottom of the beam and create grooves, 1 = cut studs above lintel beams.
studs	Number of additional studs, 0 = just create grooves to existing studs.

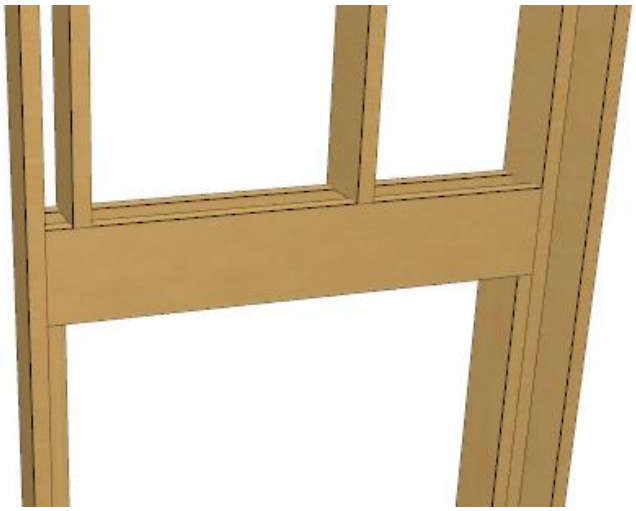
Starting point:



These settings:



The result:



22.3 Lua-scripts with elements

It is possible to create planks to the element with script instead of *xml*-definitions. The script gets information about element contour lines including openings. Contour line is always counter

clockwise for both main contour and openings. For the main contour the left hand side is towards the element and for openings, the right hand side.

Lua globals set by ArchiFrame when handling elements:

Name	Description
gGuidElem	Element object's GUID to be used for ac_objectopen. Core layer or the owning layer for options scripts.
gGuidElemStamp	Set if handling element stamp, nil = unknown.
gElemIdStamp	The element ID to put into stamp: <elemtype class = "wall" id = "WALL 100 C/C 600" **idstamp="50x100"**>.
gAdjustElems	For options and corner scripts: Set to true if processing setting palette, nil if creating element planks.

Edges are in Lua global gtblEdges and one line contains the following fields:

Field	Description
index	Index to identify the edge.
x1,y1,x2,y2	Edge coordinates in element's coordinate system. Lower left corner is (0, 0).
gx1,gy1,gz1 gx2,gy2,gz2	Edge coordinates in global model 3D-coordinates. Missing if there is no plank related to this edge.
holeindex	0 = contour line, other = opening number 1…N.
used	Is there a plank for this place from earlier <planks>-definitions: • 0, no. • 1, yes. • 2, yes from *<planks axis = "unused" …>*-tag.
tblPlankNums	Indexes to the planks created to this edge in creation order, nil = no planks yet.

Planks to be created are placed into table gtblCreate which has fields:

Field	Description
group	Group name for <operations>-section.
id	Material id.
thickness	Material size if not fixed by material id.
height	Material size if not fixed by material id.
zoff	As in xml-definitions.
rotangle	As in xml-definitions.
x1,y1,x2,y2	Plank's reference line on element's surface.
lineside	Defines reference line alignment: -1 = left, 0 = middle (default), 1= right.
fixlen	If given, sets fixed length to the plank.
guidsettings	If given, the plank settings like pens, fills etc. are taken from this existing plank.
copyops	Related to guidsettings: If value = 1 all existing linked machinings (operations) like automatically updating grooves are copied from plank pointed by guidsettings.
xmlsettings	Settings as XML-settings for an ArchiCAD element.
mayturn	If given and set to true, ArchiFrame may turn the plank to be best in the element elevation.

postlua	Lua-script inside <tag> to be run after the plank has been actually created internally. May refer to base script with attribute ref = “name”.
extendtoelem	Possible values: • Missing or 0, no effect. • 1, given line is extended to cover whole element polygon and split if there is an opening intersecting the line. • 2, extend and cut with polygon edges (gable cut).
force	Controls overlapping pieces, possible values: • false/0 = do not add if there is already a plank (default) • true/1 = keep rather this plank than any existing one • 2 = create even if outside the element • 3 = leave old intersection piece(s) under new forced piece

Number of existing planks is in global gnPlanks. New planks are appended according to table gtblCreate. Script may define operations for the planks by creating global variable gxmlOperations. Its format is like section Operations between the planks <operations>. For the script planks it is possible to refer to those using index number instead of group name, using format #number (for example target = "#1"). Many numbers may be used by separating numbers with comma, for example target = “#1, 2, 3, 4”. For example adjusting to successive planks together:

```
-- Adjust these two together
strXmlOp = string.format(
    "%s" ..
    "<joinends target=\"#%d\" operator=\"#%d\">\n" ..
    "    <joinends conntype=\"endtoend\" jointgap=\"0\"></joinends>\n" ..
    "</joinends>\n",
    strXmlOp, gnPlanks+i, gnPlanks+i+1)
```

Please see also ac_getobjparam.

ArchiFrame provides these additional functions to this task:

Function	Description
af_hasedgex (edge, vecx, vecy)	Checks whether an infinite area starting from given edge to given direction, intersects with any other edge. This is used to detect for example top side of the element (vex = 0 and vecy = 1, upwards). Returns. True = intersection found, false = no intersection.

In addition to creating new planks, it is possible to edit planks defined earlier. To do so, ac_objectopen() is called with string format #index, where index is *1*-based index to the plank. For example:

```
<!-- Sets different pen for studs -->
<script id="setcolours">
  <![CDATA[
gnPlankCount=0
gtblCreate = {}

 for i=1,gnPlanks do
  ac_objectopen(string.format("#%d",i))

  -- Vertical (studs) with different pen
  x2=ac_objectget("iEndX")
  y2=ac_objectget("iEndY")
  if x2*x2+y2*y2<0.001 then
```

```
    ac_objectset("#pen", 3)
  end
  ac_objectclose()
 end
]]>
</script>
```

22.3.1 The corner script callback functions

If called from corner tool palette, Lua global gAdjustElems is set to true. Also these globals are present:

- gbCornerFindOther, UI value for check box *Find connecting corner.*
- gbCornerAdjustOther, UI value for check box *Adjust connecting corner.*

Set (sSettings, guidParent, nCorner).
Called to set single corner for given parent element. There are two cases when this function is called:

- When new element is created and corner types are already set.
- When corner type is changed from another type or from default using corner tool palette.

Parameter	Description
sSettings	Previous settings string, “” = default settings.
guidParent	Guid of the master element, use af_request (“elem_getparent”, guidParent) to process.
nCorner	1 = left, 2 = right, 3 = bottom, 4 = top.

OnPlanksCreated (sSettings, guidElem, layerType, nCorner).
Called when planks for element outline and openings are placed but before (stud) spacing rules are applied. This script places the planks specific to the corner type.

Parameter	Description
sSettings	Previous settings string, “”= default settings.
guidElem	Current element’s guid.
layerType	Type of the related element layer: <layer ref = "WALL 42x42 VERT" … **type="intstud"**>.
nCorner	Corner to handle 1 = left, 2 = right.

22.3.2 Lua scripts with element options

Element options have fixed function names:

Function	Description
Has()	Called by ArchiFrame to find out if the option is set for options UI. Return: -1 = Not known/no planks, 0 = nope, 1 = has.
Settings(sSettings)	Called by ArchiFrame when user clicks *Settings*-button in UI. Return: nil = canceled, "" = clear settings/use defaults, str = settings str.
Set(sSettings)	Called to set the option on either from options UI (gAdjustElems == true) or when creating element planks.
Reset()	Called to cancel the option from options UI.

GetName(sSettings, baseName)	Called to make special name for the options, for example, when user has set the settings to non-default. BaseName is string of the option name, add it to the front of the result string. Return: string to show in UI.

22.4 Other Lua extensions

Http-requests interface.

Res = curl_httpget ("http://www.google.com").

Makes http-get request. Returns server reply or causes an error if failed.