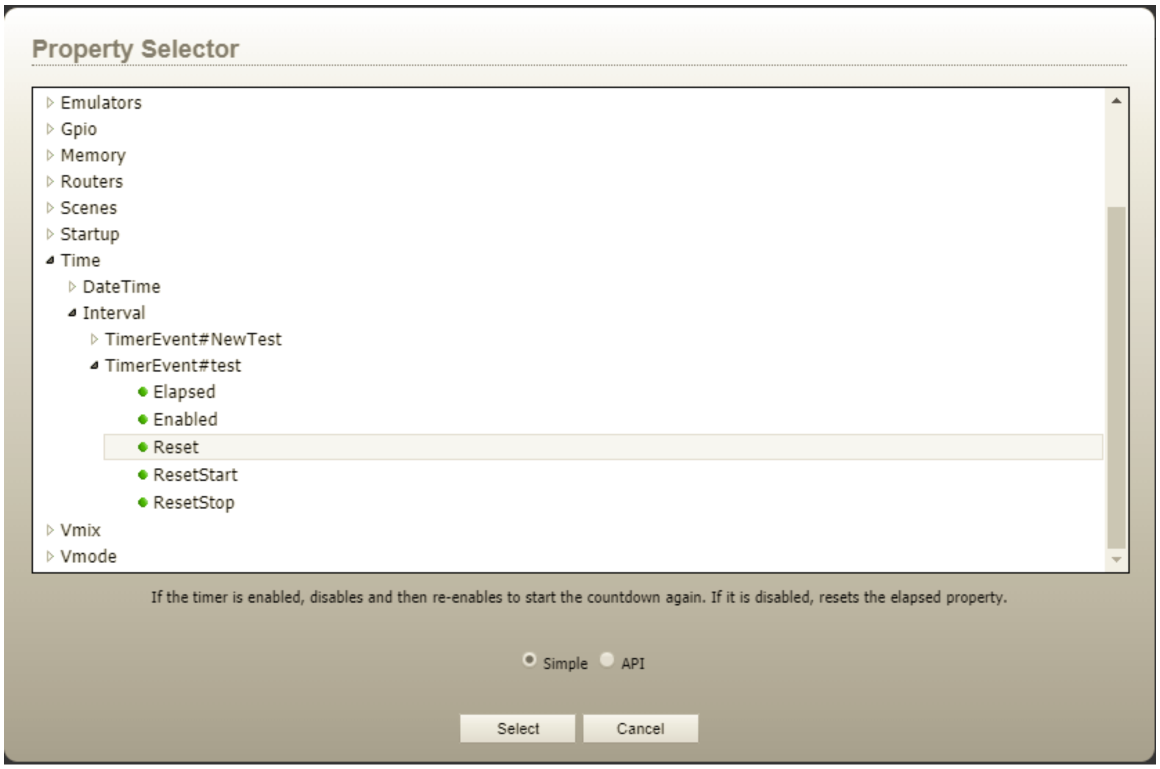
Version 1.3.5.01 Changes

Interval Timers

Interval timers now have three additional write only properties available. These are Reset, ResetStart, and ResetStop. These are available in the Logic Flows simple tree when an interval timer endpoint is selected in the logic flow. Since these are write only properties they will not be shown when interval timer start points are selected – only endpoints.



If the timer is enabled, disables and then re-enables to start the countdown again. If it is disabled, resets the elapsed property.

All three properties accept true/false values and setting any of them to true will initiate the reset. They do not need to be set to false again afterwards as passing the true value is just a trigger to initiate the reset. The value is not retained. These properties can be thought of more as actions than traditional properties.

The description of each of these properties is as follows:

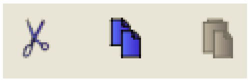
- Reset: If the timer is enabled, this property will disable and then re-enable the timer to start the countdown over. If the timer is disabled, this will set the elapsed property to false.
- ResetStart: Stops the timer (if it is running) and then starts the timer to reset the countdown.

- ResetStop: Disables the timer and changes the elapsed state to false.

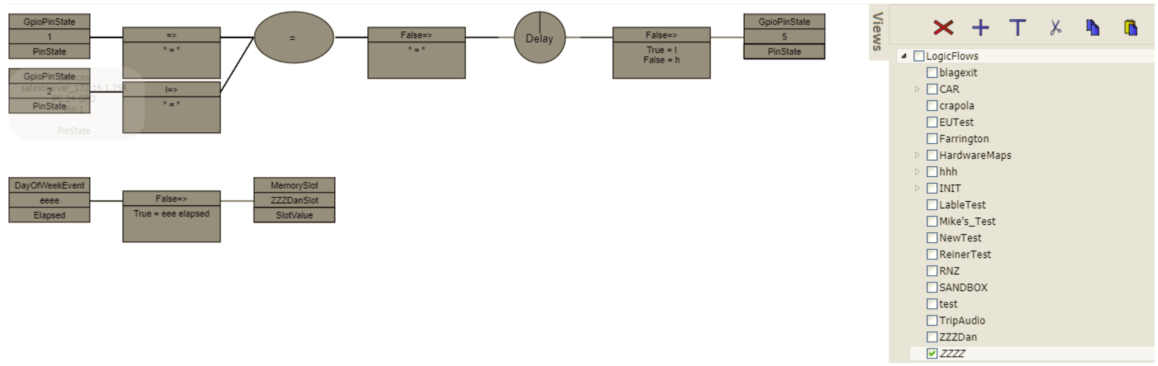
Logic Flow Changes

View Cut, Copy, and Paste

The views tab in Logic Flows now includes additional tools for cut, copy, and paste of entire views.



These tools will only be enabled when a single view is selected. To copy a view, select the view folder and deselect any other views and click the copy icon. Once a view is on the clipboard, the paste icon will be enabled. Select a new parent view folder, deselect all other folders, and click paste. The system will ask for a new name for the new view to be pasted, and it will then paste the copied view using the new name. You cannot select multiple views for copy and paste operations though if you select a view with child views those child views are a part of the copy operation.



In the example above ZZZDan was copied to the clipboard. Then we selected the LogicFlows view and pasted using the name ZZZZ. It is important to note that all pasted logic flows will be in a disabled state. After selecting the ZZZ folder you will notice the pasted flows are disabled.

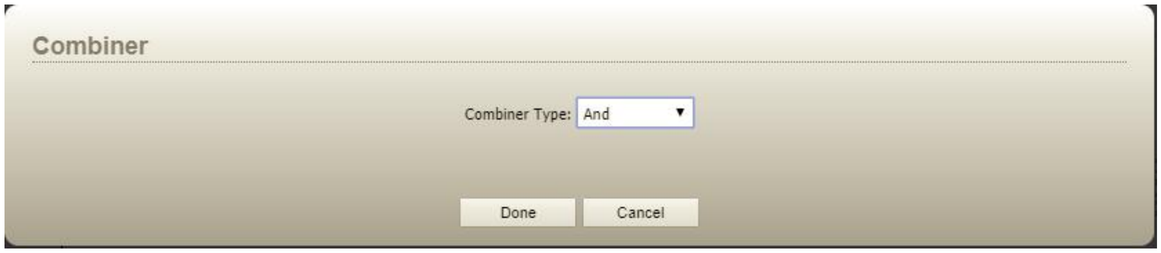
It is assumed that after completing a copy and paste you will want to modify the start and end points of the flow to reproduce behavior using different endpoints. It is also important to note that view Copy and Paste operations are completed by the Core PRO appliance rather than the browser and so do not require an apply.

Note: Under the hood the browser sets a write only CopyTo property on the copied view folder object via a SapV2 set message. The value used in the set message is the path to which the view should be copied. The PathfinderCore PRO logic flow engine then handles the data cloning.

Cut works the same way as copy except that after pasting, the original view is deleted. It is important to note that the paste creates disabled flows whether the operation started as a cut or a copy. Therefore, if you are cutting and pasting you will still need to reenable the pasted flows.

Combiner Editing

Double clicking on a combiner no longer rotates through the combiner types. Instead it will open a dialog for selecting the type of combiner and filling in any parameters that the combiner requires.



Combiner

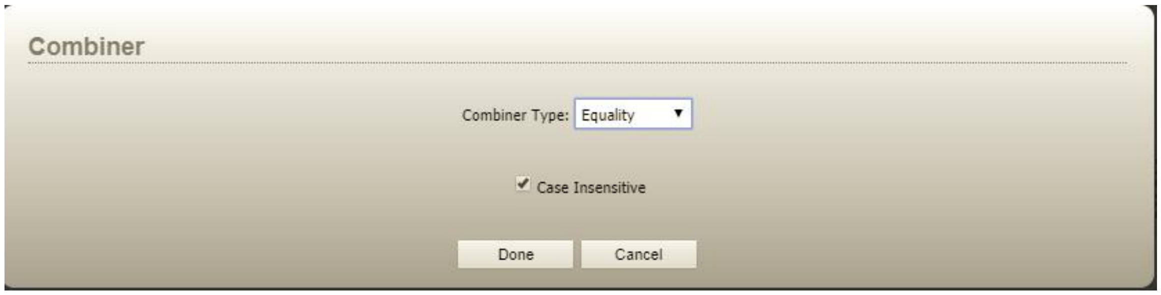
Combiner Type: And

Use the drop down to select the combiner type. Only combiners that accept the number of inputs that are currently assigned to the combiner will be present in the dropdown. For example, a single input will yield a dropdown list that includes Not, PassThru, and Delay. Delay is new and described below. Whereas two inputs will result in a dropdown list that includes And, Or, Nor, etc. The list will express options that match the number of inputs to the combiner.

In some cases, selecting a combiner type will display additional configuration options. Examples are presented below with the new Equality and Delay combiners.

Equality Combiner

The equality combiner takes multiple inputs and will result in an output value of True or False depending on whether all input values match. To create an equality combiner, add a combiner to the flow and make sure the combiner has at least two inputs. Then double click on the combiner and select equality from the dropdown.

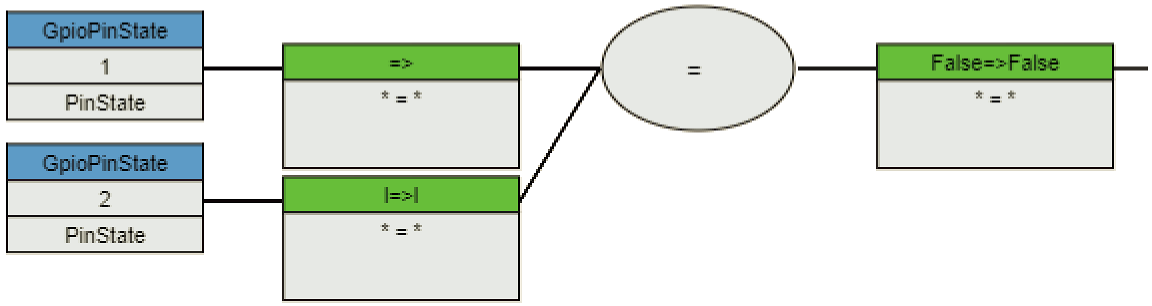


Combiner

Combiner Type: Equality

✓ Case Insensitive

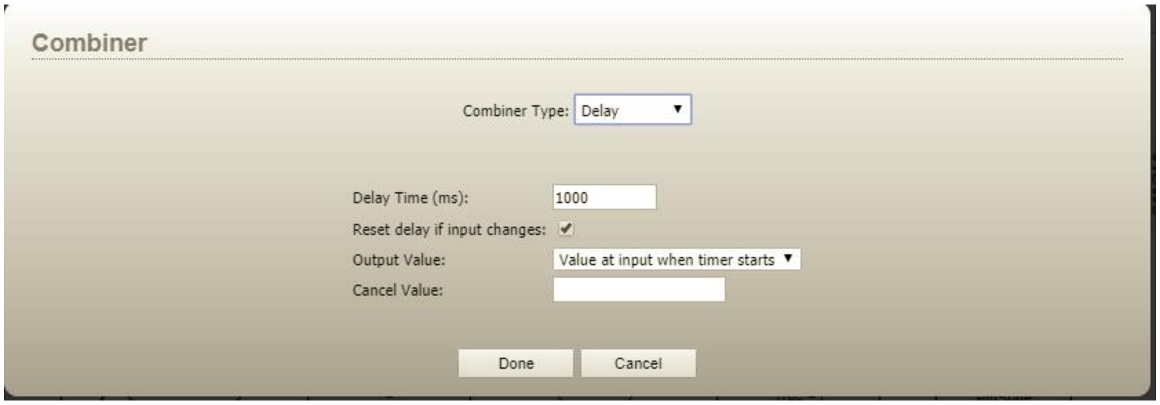
The equality combiner also has a case insensitive checkbox which defines whether comparisons are done in a case sensitive or insensitive way. Click Done once configured properly.



This combiner can be used in situations where the primary concern is whether the property states are equivalent or not.

Delay Combiner

The delay combiner takes a single input and introduces delay directly into the logic flow. It works like a passthru combiner in that it takes the input value and passes it through to the output but only after a configured number of milliseconds. There are also some additional parameters that can affect how this combiner functions. To create a delay combiner, create a new combiner with a single input. Double click on it and select the delay combiner type. Several configuration options will appear.


Combiner

Combiner Type: Delay

Delay Time (ms): 1000
Reset delay if input changes: ✓
Output Value: Value at input when timer starts
Cancel Value:

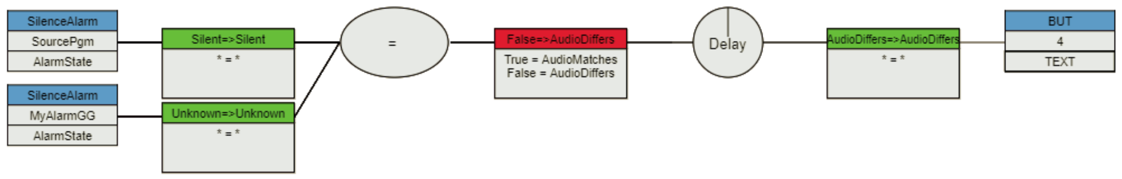
- Delay Time: Defines the number of milliseconds to delay.
- Reset delay if input changes: The delay countdown starts every time the input changes its value. If this option is selected the countdown will be reset whenever the input value changes. If you have a parameter that is fluttering and you only want an action to happen if it settles down to a fixed value for more than x milliseconds, this option should be checked. If you want to value to happen x milliseconds after a change even if additional changes happen during the countdown (delay time) then this option should be unchecked.

- Output Value: Defines whether the value that is passed through is the input value at the start of the delay countdown or the end of the countdown. This option accounts for the possibility that the input value could change again during the delay countdown and allows you to define which value gets passed through. The options are:
    - Value at input when timer starts
    - Value at input when timer ends
- Cancel Value: Allows you to define an input value that will cancel the timer and not make any change to the combiner output (not pass any different value through).

Delay combiners can be used in many situations where you need to introduce some delay into a logic flow. Previously this required using an interval timer. Therefore, there will be many situations where an interval timer is no longer necessary to accomplish the task as the delay can be built directly into the flow. This also reduces licensing requirements as the delay combiner does not require an intermediary timer endpoint.

Interval timers are still useful in situations where a single delay needs to be stopped, started, reset, or manipulated by different flows.

Delay/Equality Flow Example

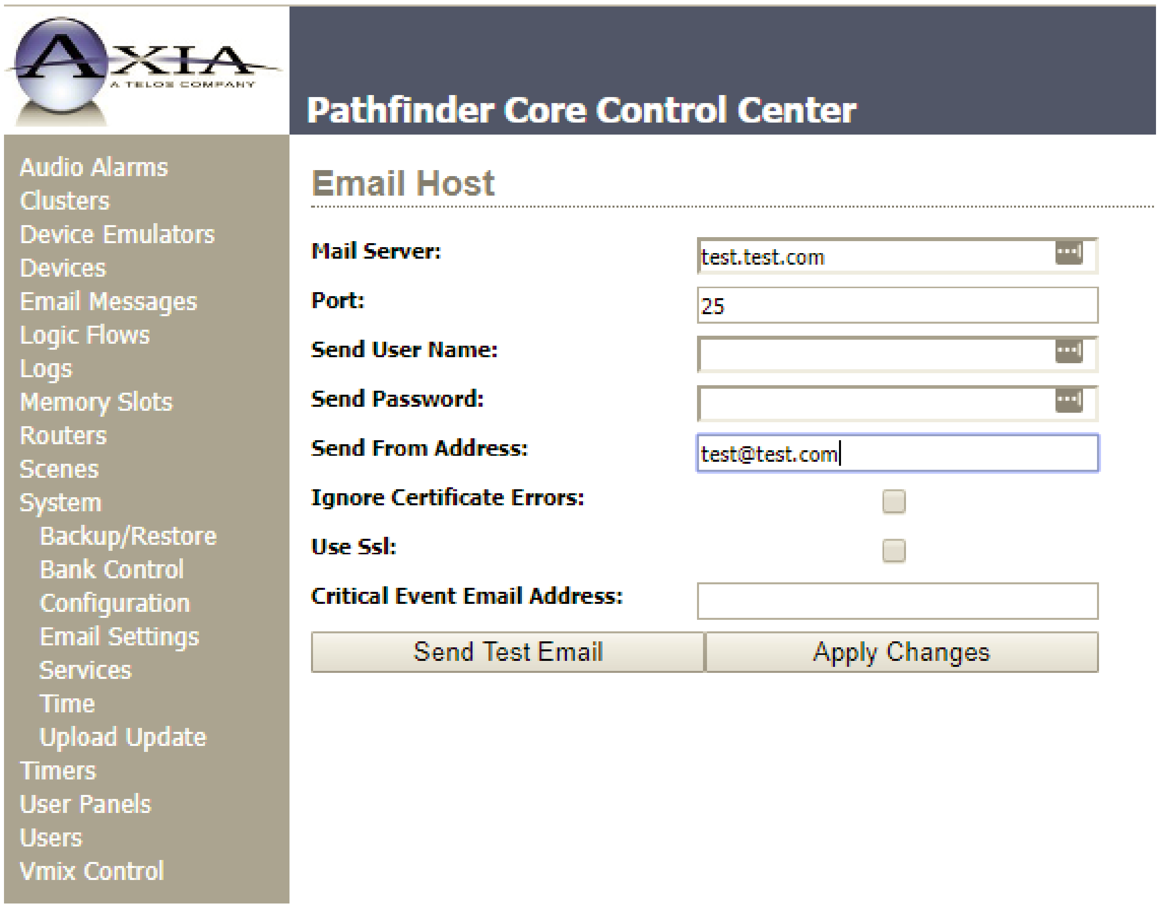


This is an example that uses the new combiners available in 1.3.5.01. In this example, the equality combiner outputs true or false depending on whether two silence alarms are in the same state. This can be useful if two audio channels should have the same audio. Silence is expected in certain situations but we are trying to make sure that if audio is present on one, it is present on both and if it is silent on one then it is silent on both. True or False is translated into a message of AudioMatches or AudioDiffers. A delay is introduced to expunge short variations of alarm states and only pass on definitive states. This is one example of how these new combiners might be used.

Version 1.3.6.02 Changes

Email Host

When configuring the email host settings, using a blank user name and password will now cause PathfinderCore PRO to skip including any credentialing in the email send. This should allow PathfinderCore PRO to send emails to email servers that are protected by source IP address rather than user credentials.



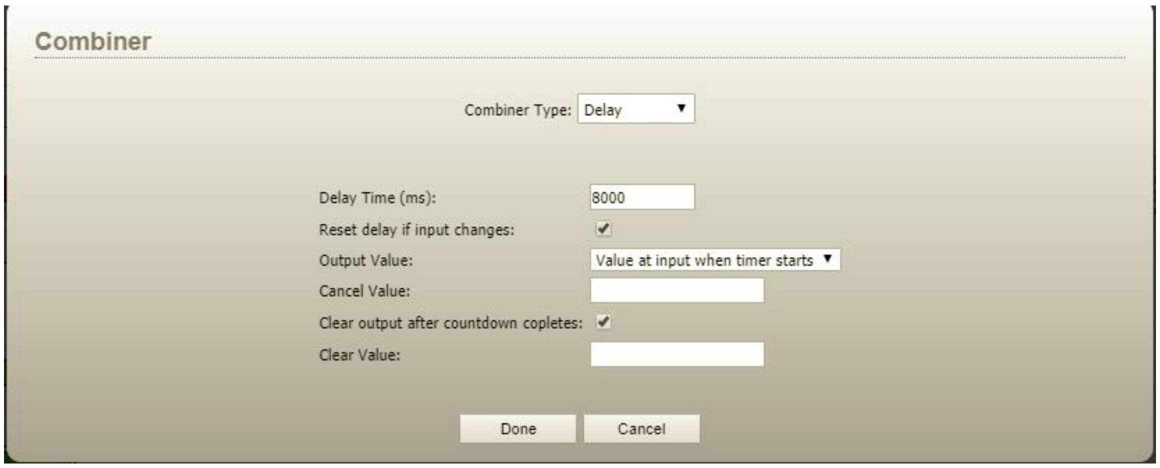
Email Host

Mail Server:	test.test.com
Port:	25
Send User Name:	
Send Password:	
Send From Address:	test@test.com
Ignore Certificate Errors:	☐
Use Ssl:	☐
Critical Event Email Address:	

Please note that after applying a blank password, the password field will likely show a series of stars. This does not mean the blank password has not been applied. In addition to masking the characters of a password, password fields also mask the length of the password with a constant number of stars. So even a blank password will display with some stars after it has been applied to the system.

Delay Combiner

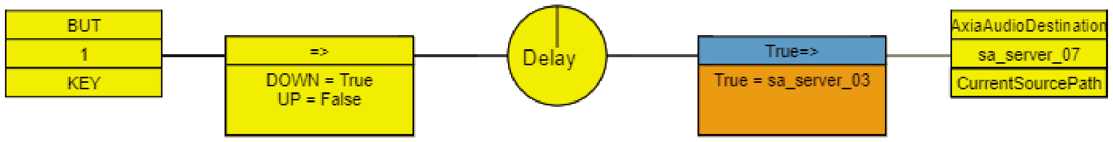
Two additional properties have been added to Delay combiners which allow the delay combiner to operate in a momentary fashion.


Combiner

Combiner Type: Delay

Delay Time (ms):	8000
Reset delay if input changes:	✓
Output Value:	Value at input when timer starts
Cancel Value:	
Clear output after countdown copletes:	✓
Clear Value:	

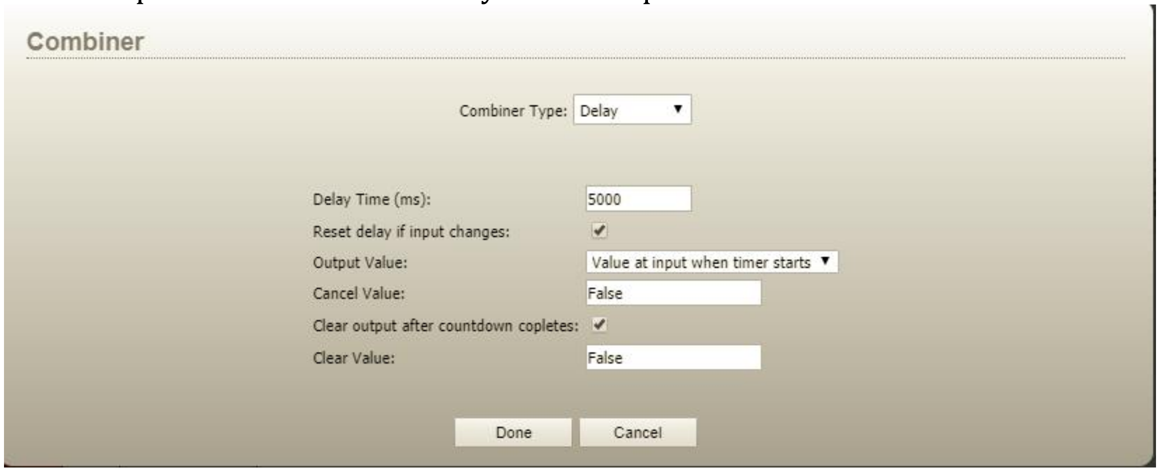
In many cases you may want the delay to pass a value through after the delay but then reset the output for the next change. For example, if we want a flow that requires a user to hold a button down for a length of time before making a change, the flow might look something like:



Our goal with this flow is the following:

- If the button is pressed, start a 5 second timer
- If the button is released cancel the timer
- If the button is held for 5 seconds make the route change

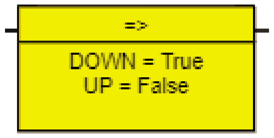
The Delay combiner has an input value and an output value. Changes only get passed to the output translator when the combiner's output value changes. And the delay is only analyzed for countdown when the combiner's input value changes. For this example we would set the delay combiner parameters like:


Combiner

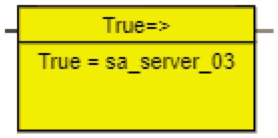
Combiner Type: Delay

Delay Time (ms):	5000
Reset delay if input changes:	✓
Output Value:	Value at input when timer starts
Cancel Value:	False
Clear output after countdown copletes:	✓
Clear Value:	False

- If the user presses the button the input value gets set to True based on the inbound translation:



- The timer starts counting.
- If the user releases the button before the timer elapses, the input value of the combiner gets set to False. Since this matches the cancel value the delay timer stops counting.
- If the user does not release the button for 5 seconds, then the true value gets passed to the combiner output which is being monitored by the output translator:



- A route change is made.
- Since the "Clear Value" and "Clear Output after countdown completes" options are set, the output value of the combiner is then set to False again so it is ready for the next button press.
- When the User releases the button, the combiner input is set to False but since that is the Cancel field no change to the combiner output is made.

Without the clear value and clear output after countdown options, the output of the combiner would remain true and so the next press would not change the output and therefore would not trigger another route. If we cleared the Cancel value and did not use the new clear options, then releasing the button would pass false to the output but 5 seconds after the button was released.

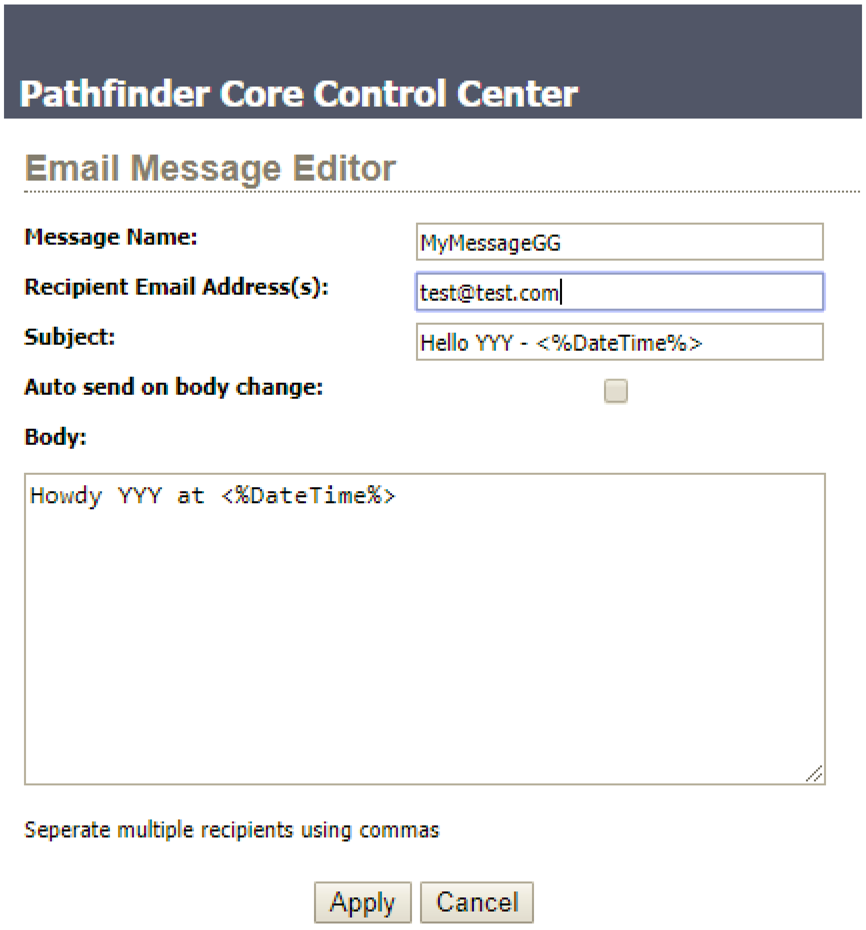
It is important to note that a cancel value if set will cancel the countdown but the cancel value does not get passed through.

These parameters will allow you to produce differing effects depending on the required goal.

Version 1.3.7.03 Changes

Email Messages

Beginning with this version you can now use <%DateTime%> in the subject or body of an email message. This will get replaced when the email is sent with the current date and time. For example:



Pathfinder Core Control Center

Email Message Editor

Message Name:	MyMessageGG
Recipient Email Address(s):	test@test.com
Subject:	Hello YYY - <%DateTime%>
Auto send on body change:	☐
Body:	

```
Howdy YYY at <%DateTime%>
```

Seperate multiple recipients using commas

In this case when the email message gets sent, the subject will look like:

Hello YYY - 2017-12-14T13:51:28.033-05:00

And the body will look like:

Howdy YYY at 2017-12-14T13:51:28.033-05:00

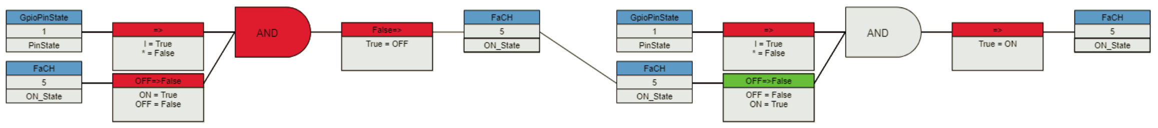
Use this feature to insert date and time values into the email message content.

Version 1.3.10.09 Changes

Swap Values

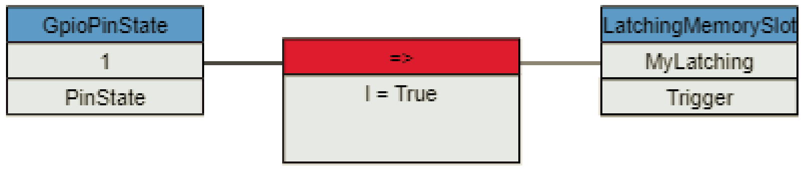
Beginning with this version properties which are binary (have two possible values) will now have a third option called swap. For example, a GPO pin state will now accept low, high, and swap instead of just low and high. This makes certain kinds of latching functionality much simpler to create. Let's look at some examples to see how this new option simplifies certain kinds of tasks.

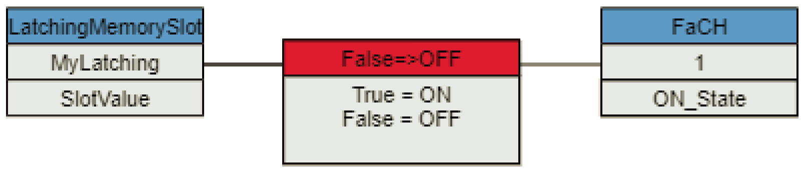
Many novice Pathfinder Core PRO users may try to create a flow that looks like this:
Warning: Do not use this flow. This is an example of what not to do.



The goal of this flow is that each time a GPI goes low, the user wants to toggle the on state of the fader back and forth between on and off. The user has tried to create a flow where if the GPI is low and the button is on then turn it off and if the GPI is low and the button is off then turn it on. The problem with this flow is that it creates and endless loop for as long as the GPI is low. Setting the GPI to low changes the state of the console channel which in turn causes the second half of the flow to change it again and so on until the GPI is returned to high.

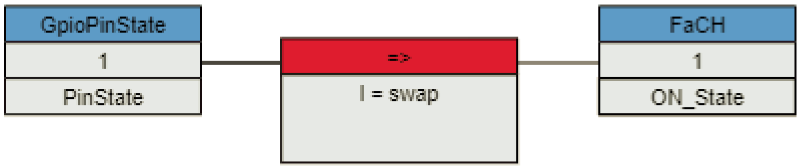
With Pathfinder Core PRO this has historically been solved by using a latching memory slot.





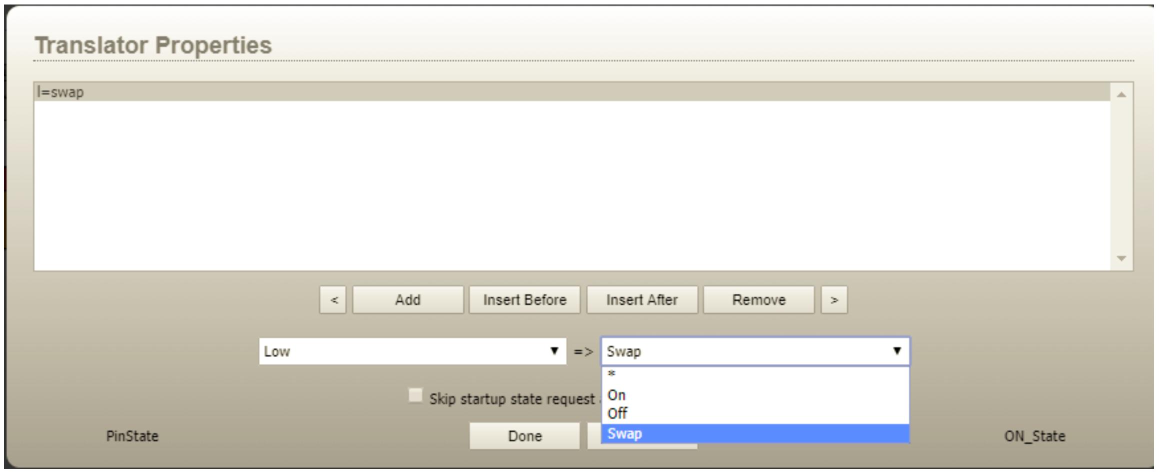
In this case the Trigger property causes the latching memory slot to swap between true and false each time the GPI goes low and then that true or false is translated to the on/off state of the fader. The problem with this flow is that the fader can also be turned on and off with the actual console button which causes the latching memory slot to get out of sync. This could be solved with a third flow if we allowed you to force the state of the memory slot each time the fader state changed. But this requires numerous flows and the creation of a memory slot for every instance of this functionality. It feels complicated to do what on the surface feels like it should be simple.

With the new swap value this becomes much simpler:



That is all you need to solve the problem. Each time the GPI goes low a value of swap is sent to the fader. The device manager picks up this request, checks to see whether the fader is currently on or off and sends the opposite state as a command to the equipment. The problem with the earlier flows is that the flow logic had to encapsulate all viable options of the existing state and specifically request what state to move to accordingly. The swap property removes that complexity from the flow logic and allows the system to handle it for you.

It is important to note that the swap value is an action and so will only appear in the translation options for an endpoint. It has no meaning for a start point because it is not a real actual state and so it will not appear in the options on a start point. Below you will find a list of all the properties to which this value option has been added. In depth knowledge of this list is not necessary as the logic flow user interface will present the swap option in the translation when it is available. For example:



Note: Axia LCD button and user button indicators represent a special case in that they have more than two options including on, off, flash, wink, and a variety of other flashing states. However, we chose to add a swap value to this property because on and off are the states that are most often used. If a button indicator if off and the swap value is sent, it will turn the indicator on. If it is in any other state when the swap value is sent, the indicator will be turned off.

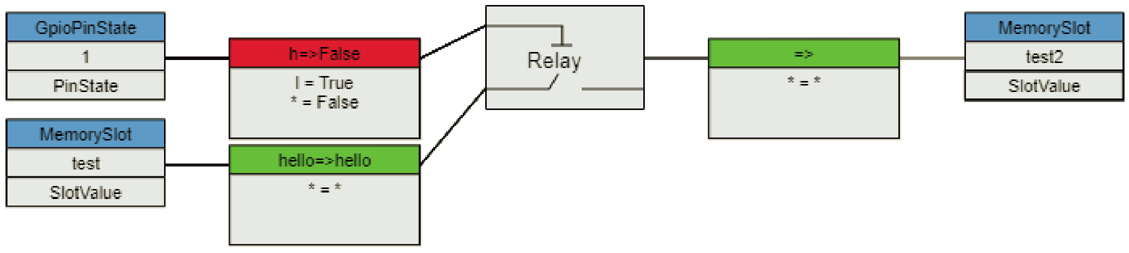
Full List of properties with the new swap value

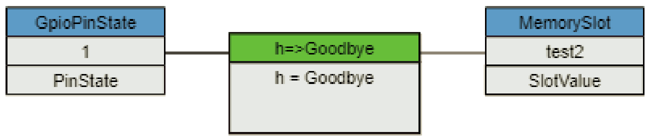
- GPIOPinstate – low, high, swap
- VMixer IN State – ON, OFF, swap
- LCD Button IND – (See note above)
- Fusion User Button IND – (See note above)
- Console Fader
    - On_State - ON, OFF, swap
    - Mute_State – MUTED, NORMAL, swap
    - ASG_PGM1 - ON, OFF, swap
    - ASG_PGM2 - ON, OFF, swap
    - ASG_PGM3 - ON, OFF, swap
    - ASG_PGM4 - ON, OFF, swap
    - ASG_PREV - ON, OFF, swap
    - Talkback - ON, OFF, swap
    - tt_cr - ON, OFF, swap
    - tt_prev - ON, OFF, swap
    - tt_st – ON, OFF, swap
- Console Monitor Section
    - cr_mute – MUTED, NORMAL, swap
    - cr_dim – DIMMED, NORMAL, swap
    - st_mute – MUTED, NORMAL, swap
    - st_dim – DIMMED, NORMAL, swap
- XNode MixInput InputActive – False, True, swap
- Axia Device Inp PhantomPower – 0(Off), 1(On), swap
- Legacy Panels
    - Flash – ON, OFF, swap
    - MeterFaderBuss
        - VisibleFader – False, True, swap
        - VisibleOnOff – False, True, swap
        - Enabled – False, True, swap
        - IsOn – False, True, swap
    - Button
        - Enabled – False, True, swap
        - State – On, Off, swap
        - Latching – On, Off swap
        - IsDown – False, True, swap
    - MeterFader
        - False, True, swap
- Timers
    - Enabled – False, True, swap
    - Elapsed – False, True, swap
    - AutoReset – False, True, swap

Version 1.3.11.10/11 Changes

Logic Flows

Prior to this version, logic flow steps each required a change to take place to advance to the next step. For example:





In this example Memory Slot test2 would be set to Goodbye when the GPI pin is high. When it goes low the value from memory slot test will be passed through the relay to test2. Then when we release the GPI to high again, test2 will return to Goodbye. The problem comes the second time the GPI goes low. At that point the state between the relay and the translator is already set to hello even though the endpoint is at good bye. Therefore, no change would happen on the endpoint because no change happens in the second translation. This is an unexpected outcome. Starting in this version the slot value will be passed through to the endpoint regardless of whether it is different from the last sent state. And this applies to all translators and combiners.

It is important to note that start points (which are inherently not on the inside of a flow) are generally still based on changes. For example, this flow will still only ever get triggered and/or analyzed when a message comes in to the logic flow system that the GPI pin has changed or that the memory slot test has changed. There are some exceptions to this rule. These include:

- System Startup
- Flow creation/edit change
- Enable/Disable

In each of these situations a get message is sent by logic flows to establish the initial state of the start point. And the return from that get message will cause the flow to be acted upon. If you wish to avoid that action and wait for the next change then use the “Skip startup state request and wait for next change option.”

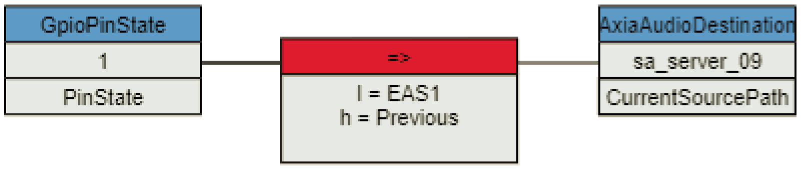
Please note that this change could be a breaking change to some flows. We have spent time analyzing numerous customer flows and believe the impact will be minimal or non-existent but it is a behavioral change and therefore could cause a flow to work differently than it has in the past. If you find a flow that is not working correctly after this update please revert to an earlier version and then report the broken flow to us. We will then analyze to determine if we need to add additional options to the software to cover the case or if there is a better way to design the flow under the revised structure.

Console Channel Subscription

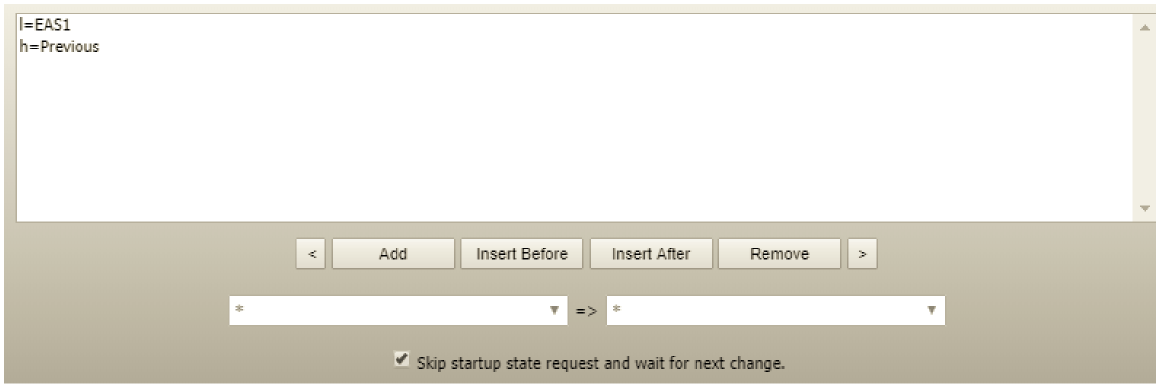
With newer versions of console software, a command has been added that allows us to subscribe to the movements of the console faders. Prior to these software updates we had to poll at regular intervals to determine the current level of the fader. To decrease load on the system when often a fader was going to be in a static position for extended periods of time, our poll rate was set at a reasonably low rate. This caused things like meter faders in user panels to jump every few seconds when a physical fader was moved instead of smoothly following the move. The addition of this subscription mechanism in the console code has allowed us to subscribe to those changes and track the physical moves much more closely. This new subscription command has been utilized in this version of PathfinderCore PRO. We left polling in place as well to continue to support console software revisions prior to the ones that supports this new subscription command. This new feature should exist in Element/Fusion firmware version 3.2 and later and Qor 2.3.0.59 and later.

Previous Route Source

While there has not been an actual change to the functionality of the previous route source, user cases have arisen that require clearer recommendations. For example, this is a pretty common flow we might see for EAS.



First if you are going to use a flow like this, it is critically important that you also turn on the “Skip startup state request and wait for next change” option.

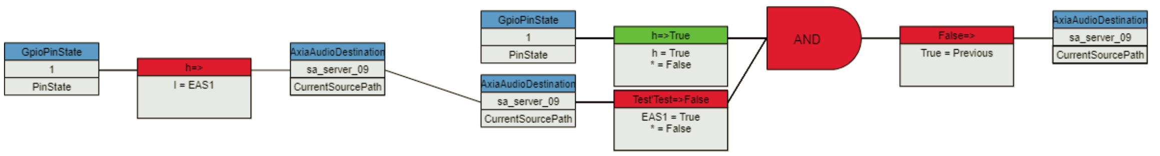


```
l=EAS1
h=Previous
```

☑ Skip startup state request and wait for next change.

Without this option a restart of the system will request the current GPI state which is high and will therefore toggle the previous route. In many cases this will route nothing to the air chain because no previous state exists yet.

However, even with that option enabled this flow may produce unforeseen results. For example, what happens if the GPI goes low and someone manually changes the route before it goes high again? At that point EAS is now the previous source when it does go high. This can be solved by using a slightly more complex flow with previous.



In this example, a route to previous will only happen if the GPI is high and we are currently on the EAS source. This will function in a much more reliable manner.

While these examples have been EAS related the same rules apply to other uses of previous. More importantly, if you can avoid the use of previous and design the flows to be more specific the outcome will also be more reliably specific.

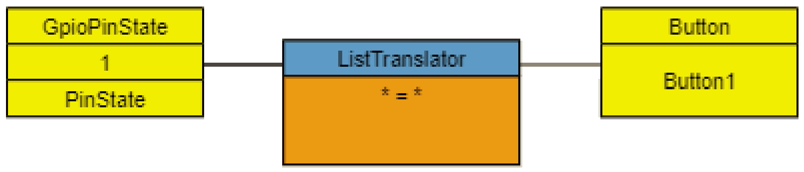
Watch the PathfinderPC.com site for a future article that delves into some of these other options.

Version 1.3.12.13 Changes

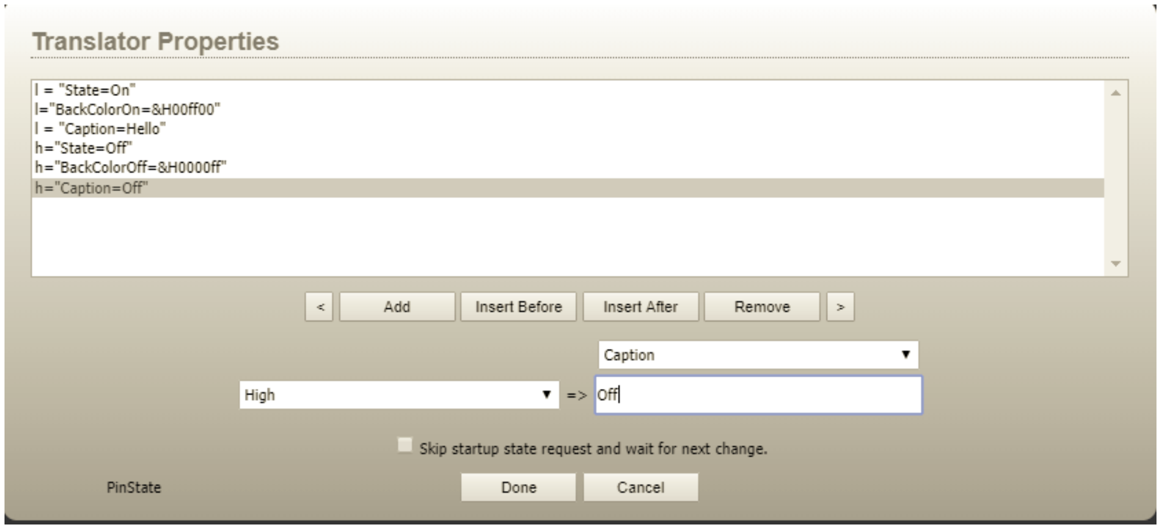
Object Translators

Object translators differ from property translators in that an Object translator can affect multiple properties of a given object whereas a property translator only affects a single property. As described in the full Pathfinder Core PRO documentation, object translators are often used for mirroring states of full vmix channels or vmixers with a single translation. Object translators are created by selecting an object end point from the property list rather than a property. Please review the Pathfinder Core PRO manual for more details about object translators.

However, object translators prior to version 1.3.12.13 had a restriction based on their inheritance from property translators that the input side of the conversion list had to be unique. So for example:



When you edit the translation list for this translation which has a GPIO pin state as an input and a button object for an output, you can select multiple properties on the right hand side of the translation. But you cannot repeat the same value on the left hand side. So the example below would not work:

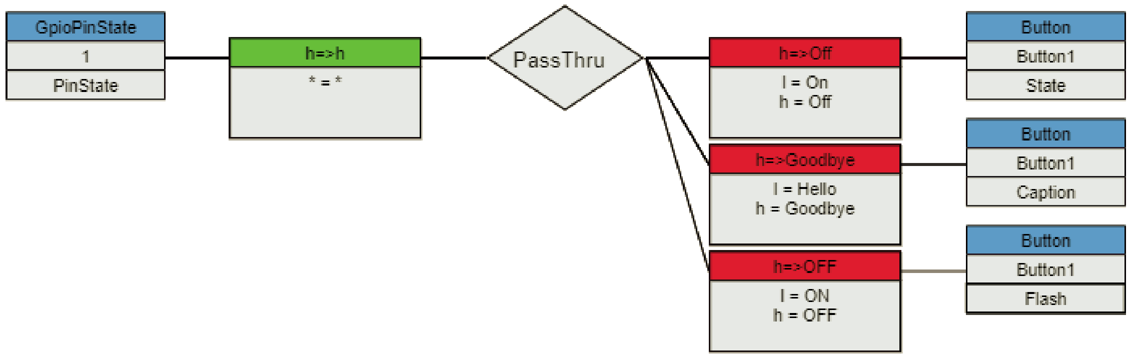

Translator Properties

```
l = "State=On"
l="BackColorOn=&H00ff00"
l = "Caption=Hello"
h="State=Off"
h="BackColorOff=&H0000ff"
h="Caption=Off"
```

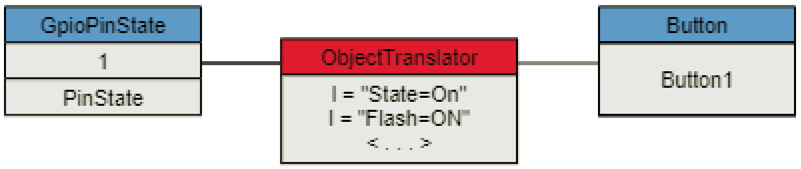
< Add Insert Before Insert After Remove >
Caption
High => Off
Skip startup state request and wait for next change.
PinState
Done Cancel

This translation list would get stripped down after clicking Done to only the first of the output property state changes for low and the first of the output property state changes for high on the output side. Starting with version 1.3.12.13, that is no longer the case and the example above will work. This allows multiple properties to be set at once in a single translation block to a given object.

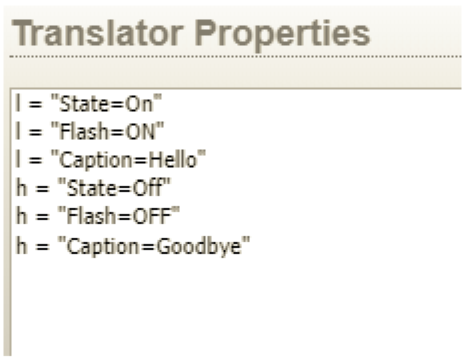
So for example the following flow:



Can be reduced to this flow instead:



Where the translation looks like:



Translator Properties

```
l = "State=On"
l = "Flash=ON"
l = "Caption=Hello"
h = "State=Off"
h = "Flash=OFF"
h = "Caption=Goodbye"
```

This can greatly reduce both complexity and license utilization when trying to set multiple properties on an object such as a button or fader at the same time.

Version 1.3.13.20 Changes

Important Notes and Known Issues

This build includes a preview version of the new HTML panels and panel designer. It also includes a graphical refresh of Logic Flows. There are number of known issues with these new features that need to be mentioned.

- The new panels are not subject to cluster synchronization in this build. Therefore, user panels built on one node of the cluster using the new html panels will not appear on the other node without a manual sync. Clustering support will be coming soon.
- Because of the number of changes in this build, cached files in the browser may cause incorrect operation.
    - Clearing your browser cache will fix this problem.
    - Alternatively, if something is not working correctly or something is missing, or a graphic does not look correct, try holding the Ctrl and Shift keys and clicking the refresh icon in Chrome: ⟳. This combination forces a hard refresh where chrome re-downloads the files rather than relying on the cache. This may have to be done on different pages of the application if that page is not working correctly. Once the new code is loaded, this should generally not be necessary. We are continuing to work on better ways to manage how and when browser caching occurs.
- The new panels have been designed using Chrome and are not fully tested with other browsers at this point in time.
- If you save a user panel and the saved changes are not displayed, try forcefully refreshing the browser as described above.

Web Page Connection State

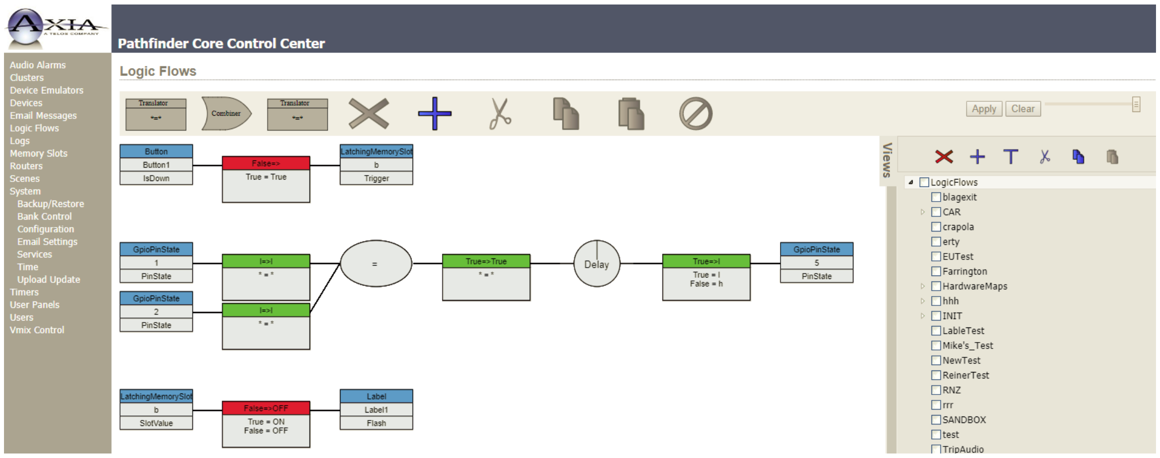
Pathfinder Core PRO does most of its dynamic data and configuration via a web socket. When the web page opens, it downloads the web pages and javascript and then opens a web socket to port 8001 which is extended to the API socket for actual real time communications and changes. In previous versions it was unclear as to whether this websocket connection was successful. This meant that if there was a loss of connection between the web browser and PathfinderCore PRO there was no visual indication and the user would think that that the configuration had disappeared or was not functioning. Another example is that during a system restart the web socket may not be available for a period of time as the system boots and loads its configuration. And sometimes after a restart the user might need to refresh the web page to get it to reconnect. This version now adds an icon on each web page that uses the web socket. This icon shows the status of the web socket connection:

If this icon is red that means the web socket connection has been disconnected or is not established. Blue is the color that shows that the socket is properly connected. It is perfectly normal for this icon to briefly flash red and then settle on blue when the page is first loaded. Additionally this version will now generate an error message if configuration changes are saved when the socket is in a disconnected state.

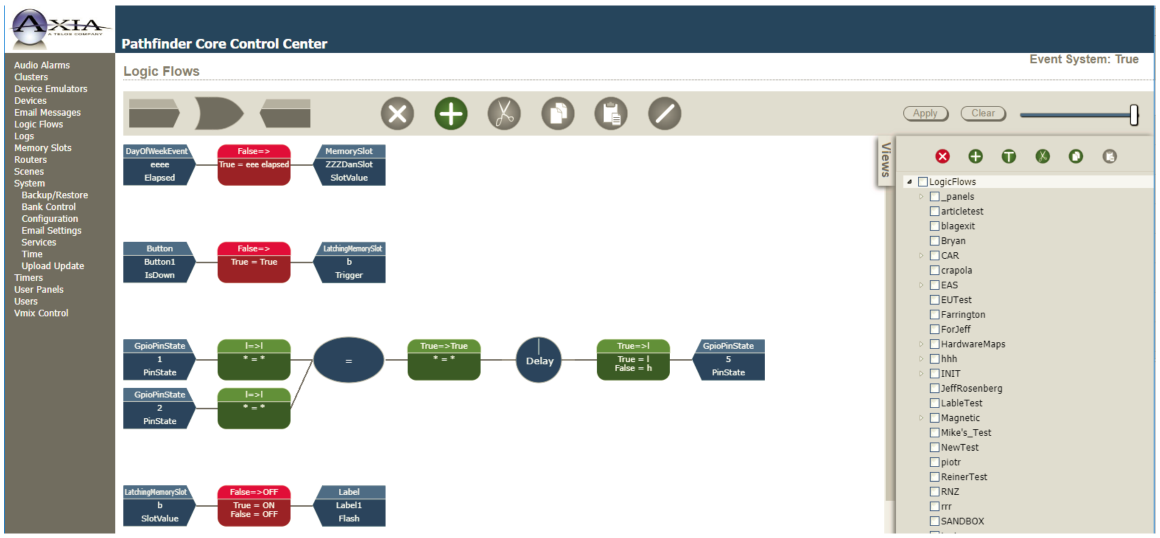
Logic Flows

While logic flows function in the same manner in this version as the previous beta, the graphics have been updated. Both the tollbar icons and flow shapes have been improved. The following before and after pictures show the differences:

Before:



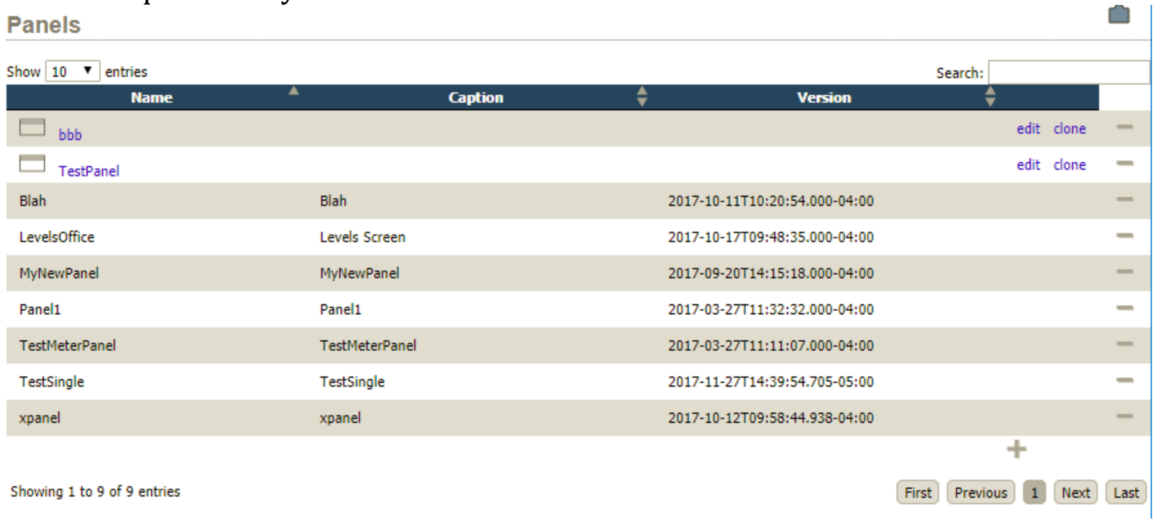
After:



User Panels

User panels may now be created from the web interface. It is important to note that the new user panels are not compatible with the old legacy panel designer and/or PathfinderClient/Mini. The new user panels are completely Html5 and therefore may be both designed and used with just a web browser. In this version Chrome is the recommended browser and testing/implementation has not been completed for other browsers yet.

Html user panels may be used and edited from the User Panels menu item.



Panels

Show 10 entries Search:

Name	Caption	Version		
bbb			edit clone	—
TestPanel			edit clone	—
Blah	Blah	2017-10-11T10:20:54.000-04:00		—
LevelsOffice	Levels Screen	2017-10-17T09:48:35.000-04:00		—
MyNewPanel	MyNewPanel	2017-09-20T14:15:18.000-04:00		—
Panel1	Panel1	2017-03-27T11:32:32.000-04:00		—
TestMeterPanel	TestMeterPanel	2017-03-27T11:11:07.000-04:00		—
TestSingle	TestSingle	2017-11-27T14:39:54.705-05:00		—
xpanel	xpanel	2017-10-12T09:58:44.938-04:00		—

Showing 1 to 9 of 9 entries

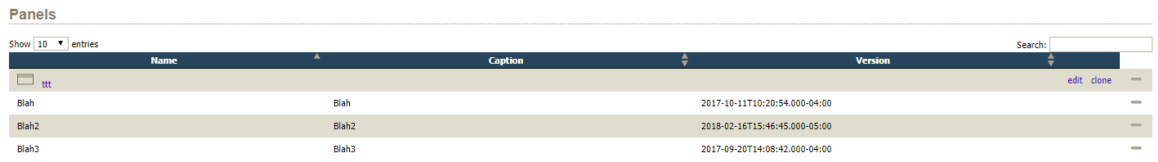
First Previous 1 Next Last

This list will show both the new Html5 panels and the original legacy panels created with the legacy panel designer. Html5 panels will have their name displayed as a link. Clicking that link will display the panel within the context of the

PathfinderCore PRO web pages. Clicking the icon will open the panel in its own window without most of the browser menu systems. Finally, the html5 panels will have edit and clone links. The edit link will open the panel in the html panel designer and the clone link will make a duplicate of the panel as it exists under a new panel name. The icon will delete an existing panel. The icon will open the html panel designer for creating a new panel.

Creating a new panel

To create and use the new panels, select the User Panels menu item.

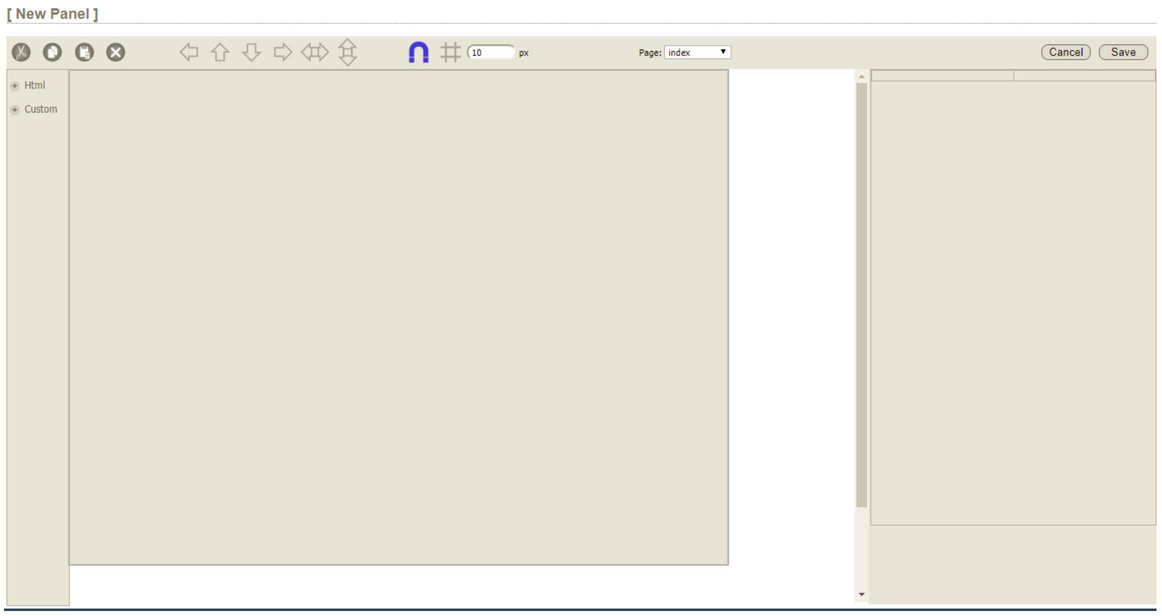


Panels

Show 10 entries   Search:

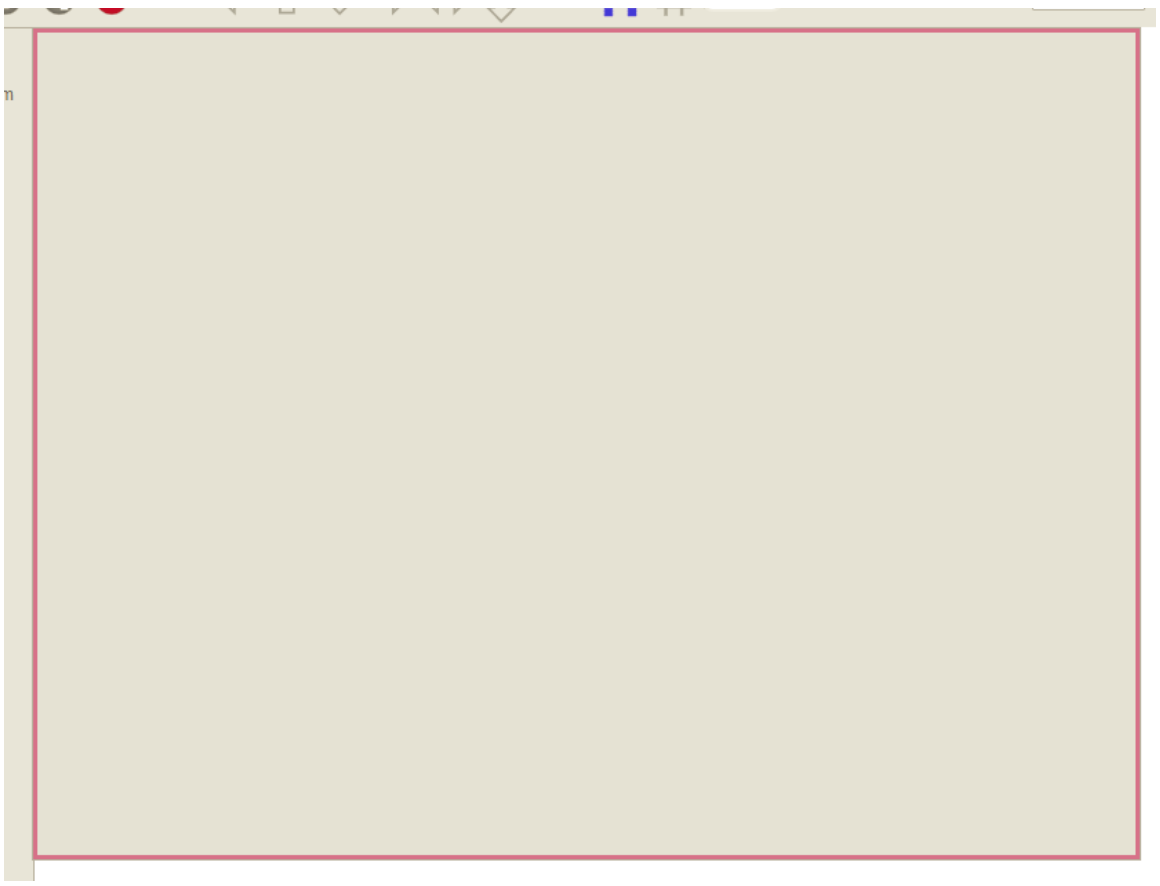
Name	Caption	Version		
ttt			edit clone	—
Blah	Blah	2017-10-11T10:20:54.000-04:00		—
Blah2	Blah2	2018-02-16T15:46:45.000-05:00		—
Blah3	Blah3	2017-09-20T14:08:42.000-04:00		—

Click the icon to create a new panel.

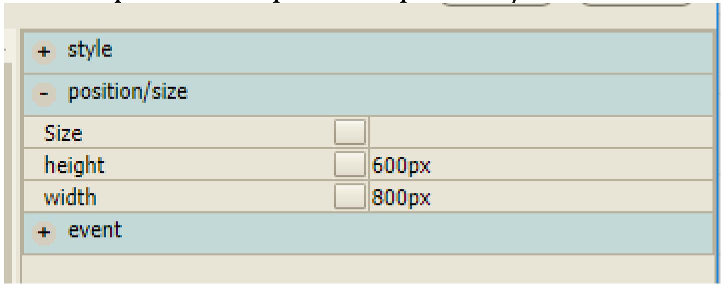


This will open a new panel in the panel designer. We recommend clicking the Save button next in order to give your panel a name and instantiate it before you begin adding components. After saving and giving the panel a name, the name of the panel should appear in the upper corner:

Next set the size of the panel. Click in the main panel so that the panel gets highlighted.



On the right side, the property window will fill with sections of properties that may be manipulated. Expand the position/size section.

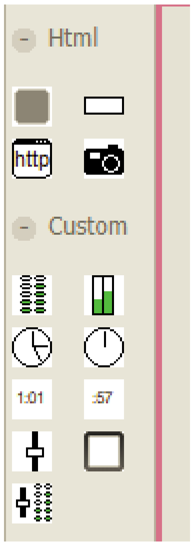


Size	
height	600px
width	800px

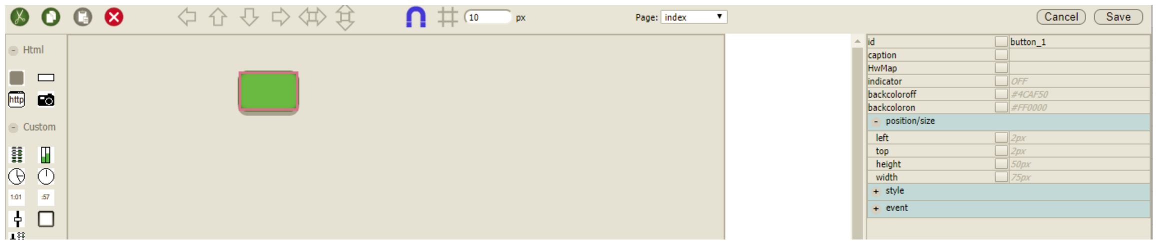
Change the size of the panel by clicking in the size field and selecting from a pre-determined size or by altering the height and width properties. The size of the panel area should change accordingly.

Adding Components

To add components to your panel, expand the Html and/or custom sections in the left toolbar.



Hovering over any tool will show a hover balloon with the name of the component. To explain the tools for the components and how to use them this document will start with a simple html button. Later we will discuss the other components that may be used. Click and drag the top left html button (hover balloon says Button) from the tool bar into the panel. When you let go of the button at the end of the drag operation a new button should appear on the panel and it should be highlighted with the red box indicating it is the selected component:



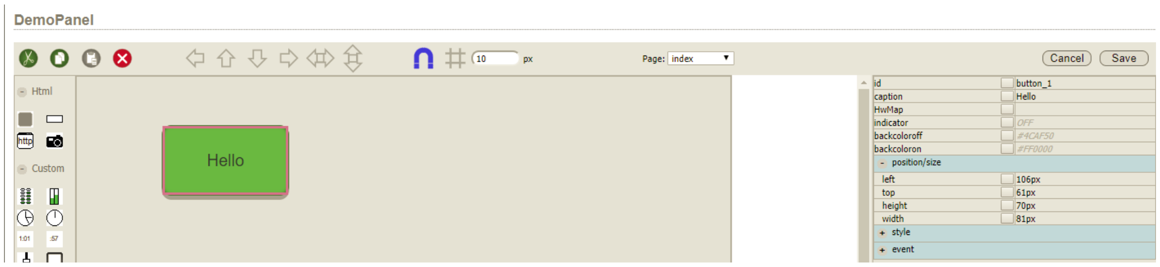
id	button_1
caption	
HwMap	
indicator	OFF
backcoloroff	#4CAF50
backcoloron	#FF0000
- position/size	
left	2px
top	2px
height	50px
width	75px
+ style	
+ event	

Dragging the edges of the button will resize it whereas dragging from the center of the button will move it. You can also use the arrow keys on your keyboard to nudge the component by small amounts in any direction. It will move 1 pixel at a time unless the grid is enabled in which case it will move by the grid amount. We will discuss the grid more shortly. The selection handle displays just inside the actual edges of the object by design so when aligning objects, you can still see the actual edge of the object.

Note: Some components may by default resize both height and width when one or the other is dragged. These are generally more complex components such as the console fader component. This is because non-scaled resizing causes the component to look skewed and stretched so both sizes are changed to maintain aspect. Holding the shift key while resizing overrides this behavior and allows you to skew the component if desired.

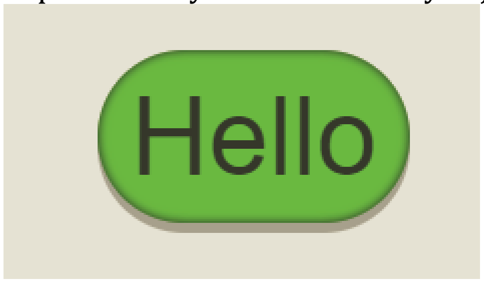
Adjusting Properties

It is also important to note that the property list on the right will have updated to present the properties that may be changed for the currently selected component (our new button). Clicking in the field and changing one of the properties will make the corresponding changes to the button. For example, click in the caption field and add a caption:



id	button_1
caption	Hello
HwMap	
indicator	OFF
backcoloroff	#4CAF50
backcoloron	#FF0000
- position/size	
left	106px
top	61px
height	70px
width	81px
+ style	
+ event	

Expand the style section and try adjusting the border-radius or the font-size.



As you can see there is a high degree of power to achieve exactly the desired design using the properties in the property grid.

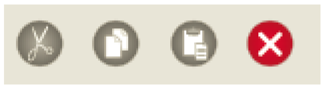
Clicking cancel will return the panel to its previous save state and clicking save will save the changes.

Note: In this version there is no undo or redo so frequent saves as you are working are recommended.

Tool Bar

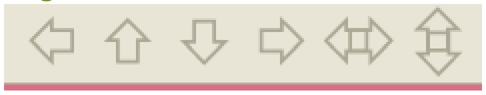
Cut, Copy, Paste, Delete

The top tool bar has several tools to help with the layout of components:

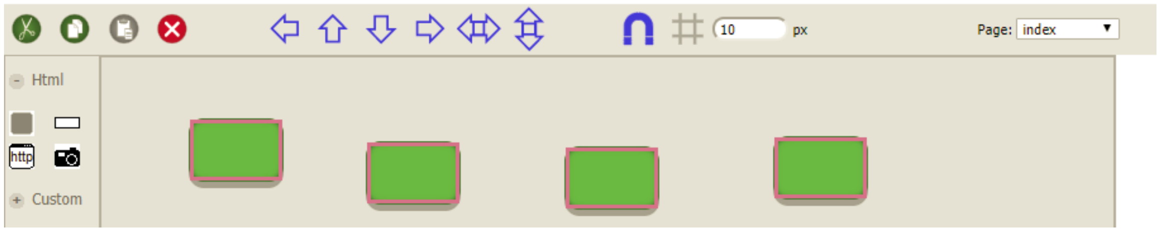


These are the standard Cut, Copy, Paste, and Delete tools. They can be used with any components currently selected on the panel. You can select multiple components by holding the shift key while you click

Alignment Tools



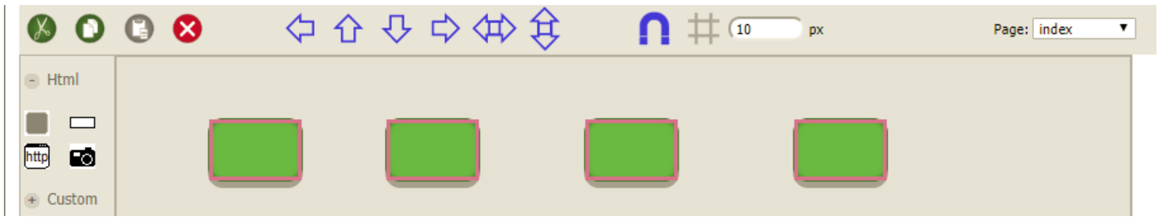
These tools are alignment tools and will only be available for use if you have multiple components selected in the panel. For example, drag three or four buttons into the panel. Then while holding the Shift or Control key click on each component until the red select box is around each of them:



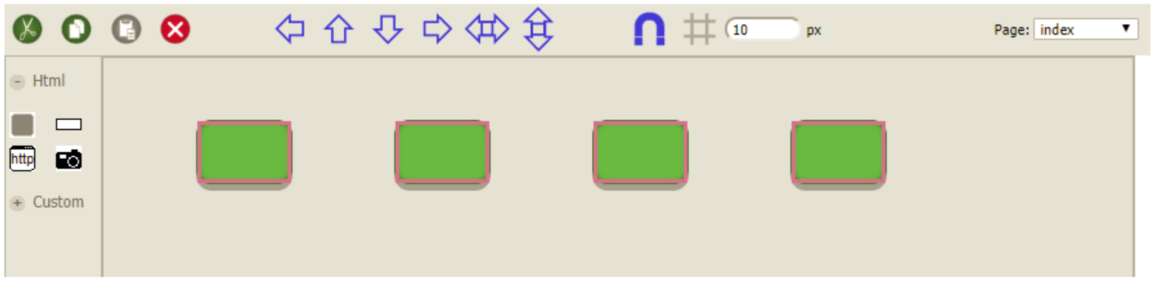
Once more than one component are selected, the align tools will become enabled. These tools are:

- Align Left
- Align Top
- Align Bottom
- Align Right
- Spread Horizontally
- Spread Vertically

In each case the system will look for the most extreme edge and align to that. For example, with the 4 buttons selected in the example above, clicking the align top will find the selected button that is closest to the top of the panel and align all of the buttons so that their tops match that button.

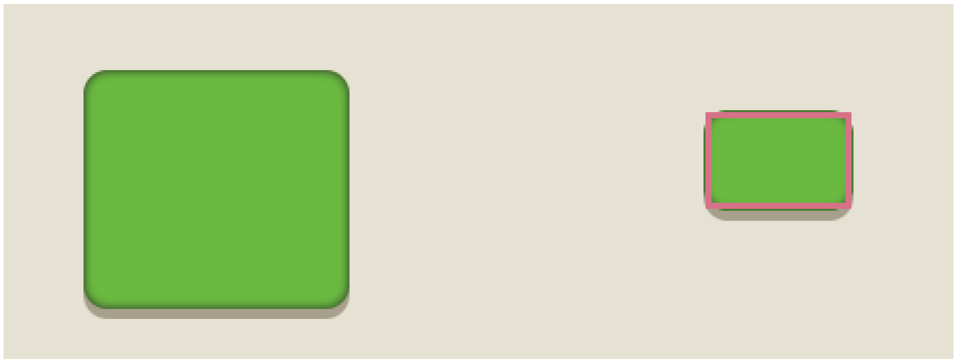


Clicking the Spread Horizontally tool will make them spread evenly between the left and right button

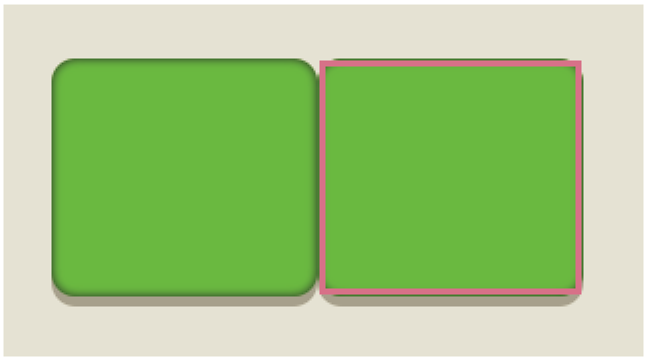


Magnet

The magnet tool is enabled by default but may be disabled by clicking on the tool. This tool helps to align and resize items of like kind edge to edge. For example, create a new button and resize it. And then add a second button and leave it at the default size.

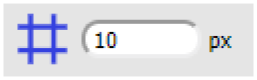


If you then drag the small button so that its edge meets the large button it will immediately snap to the size of the large button and align itself to the large button's edge.



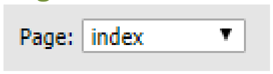
This is extremely useful when trying to build a panel with many same sized components lined up edge to edge. However, it can also be disconcerting when you want the components to be different sizes and/or not to line up. So always be aware of whether the magnet tool is on or off when trying to align components.

Grid

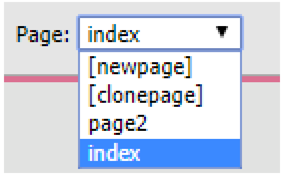


The grid tool if enabled will align component's location and size according to a grid of a specified pixel density. For example, if you enabled the grid with a pixel density of 10 pixels, you will notice that dragging components will jump by 10 pixels. This is useful when trying to evenly align components.

Page



The page tool allows you to create additional pages for a panel. Buttons may be created to switch between the pages. The method to make a panel switch pages will be described in more detail later. However, to design a new page, click on the down arrow and select the [newpage] option.

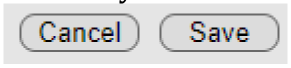


After clicking the new page option click save to give the new page a name. Then you can design a new page as if it was any other panel. Each panel has an index page which is the default page that will be loaded when the panel is first displayed. You can select any page to edit by clicking the page in the drop down list. You can clone your work to a new page by selecting the page to be cloned and then selecting the clone page drop down item. The system will ask you for a new page name for the cloned page.

You can delete a page by selecting the page background and clicking the delete icon. The system will then ask you if you want to delete the page. You cannot delete the default index page.

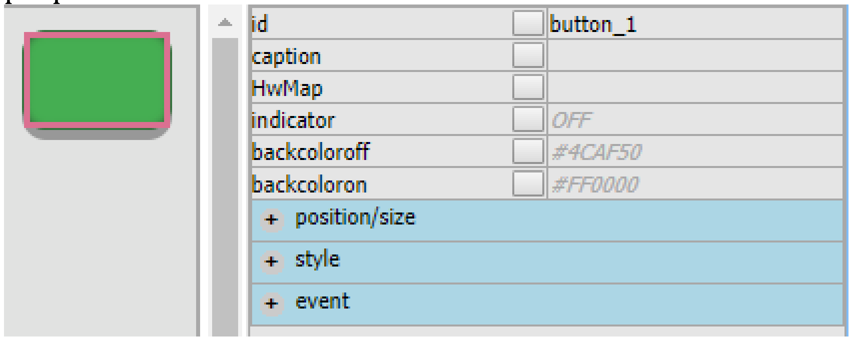
Cancel/Save

The cancel and save buttons may be used to save your work or cancel pending changes and reload from the last save point. Since this version does not have an undo/redo button and this functionality is still very much in beta testing, frequent saves of your work are recommended.



Property Grid

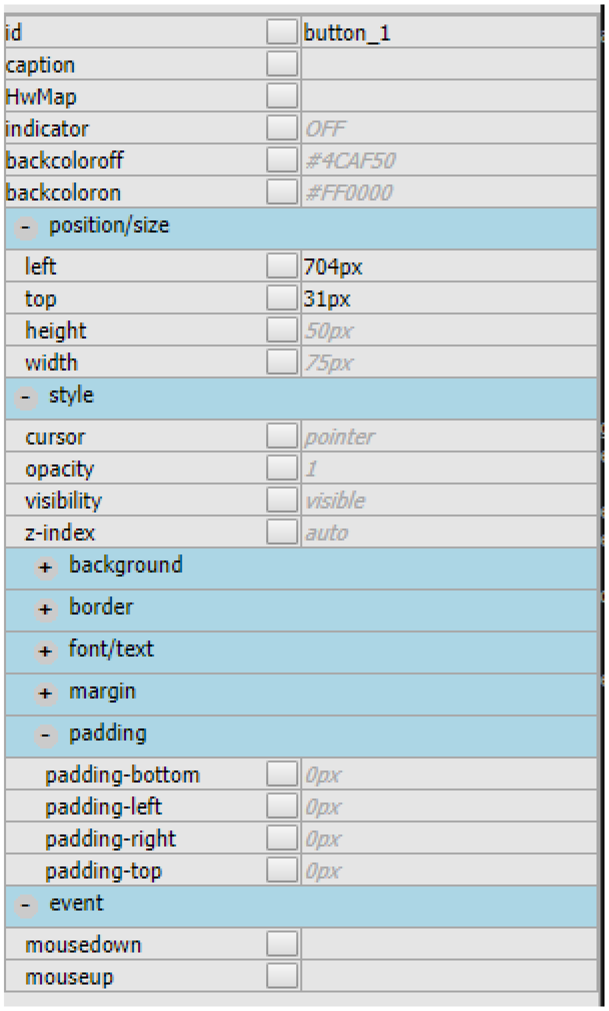
You were already introduced to the property grid in the section on adjusting properties above. This section will go into some additional detail. Drag a button onto your panel and select it so that the property grid displays the available properties of the button.



id	☐	button_1
caption	☐	
HwMap	☐	
indicator	☐	OFF
backcoloroff	☐	#4CAF50
backcoloron	☐	#FF0000

+ position/size
+ style
+ event

Different components may have different property sections and sub sections as well as properties that are specific to that component but this is an example of the property sections you will see. Expanding the sections will display addition sections and properties.



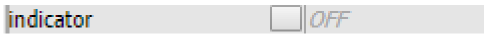
id	☐	button_1
caption	☐	
HwMap	☐	
indicator	☐	*OFF*
backcoloroff	☐	*#4CAF50*
backcoloron	☐	*#FF0000*
- position/size		
left	☐	704px
top	☐	31px
height	☐	*50px*
width	☐	*75px*
- style		
cursor	☐	*pointer*
opacity	☐	*1*
visibility	☐	*visible*
z-index	☐	*auto*
+ background		
+ border		
+ font/text		
+ margin		
- padding		
padding-bottom	☐	*0px*
padding-left	☐	*0px*
padding-right	☐	*0px*
padding-top	☐	*0px*
- event		
mousedown	☐	
mouseup	☐	

Because the property list is still changing as we progress through the beta stages of Html5 panels we will not list the meaning of every property in this document. We will mention ones that have specific meaning unique to that object. However, many of these properties are standard css style properties used by any web page designer. One of the best references we have found for css properties is w3schools.com: https://www.w3schools.com/cssref/. This link will provide information about all of the css properties exposed in the property grid along with their meaning and usage information.

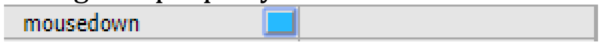
There are also some properties that you will not find in the css reference above because they are custom to our usage of that component. For example, in the case of the button component, caption, HwMap, and indicator are all properties that are not standard css properties. We will describe their usage more in the examples below.

Bind Button

Each property in the property grid has a button between the property name and the property value at the end of the property name side of the grid. This is called a bind button.



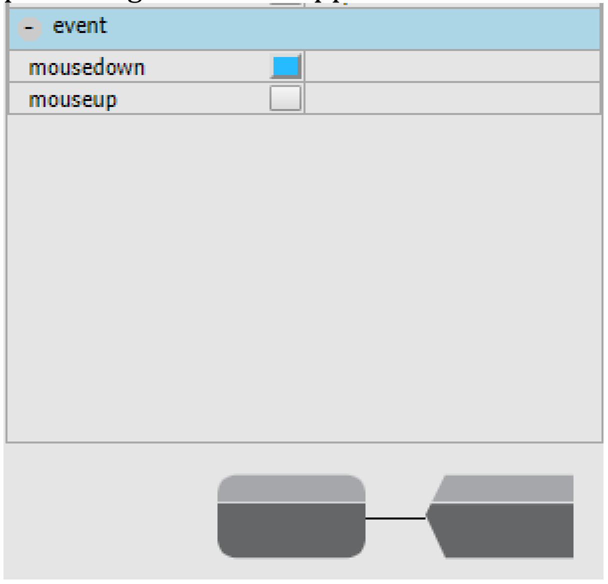
The bind button defines the properties that should be exposed to PathfinderCore PRO for use in logic flows. In some cases, there may be hundreds of properties for a given component, but there are only a few that you will want to dynamically change while the panel is running. For example, once you position a button on a panel and size it to the size you desire, it is unlikely that you will want that position to change while your end user is using the panel. Therefore, there is really no need for the left and top properties to be cluttering up the logic flow tree. Clicking the bind button for a given property will turn the button blue.



Saving the panel will then identify to PathfinderCore PRO that this is a property that we expect to dynamically manipulate with logic flows and so should be tracked by PathfinderCore PRO and made available to Logic Flows.

Binding Flows

You may also notice that after enabling a property for binding that an image of a partial logic flow will appear at the bottom of the property grid:

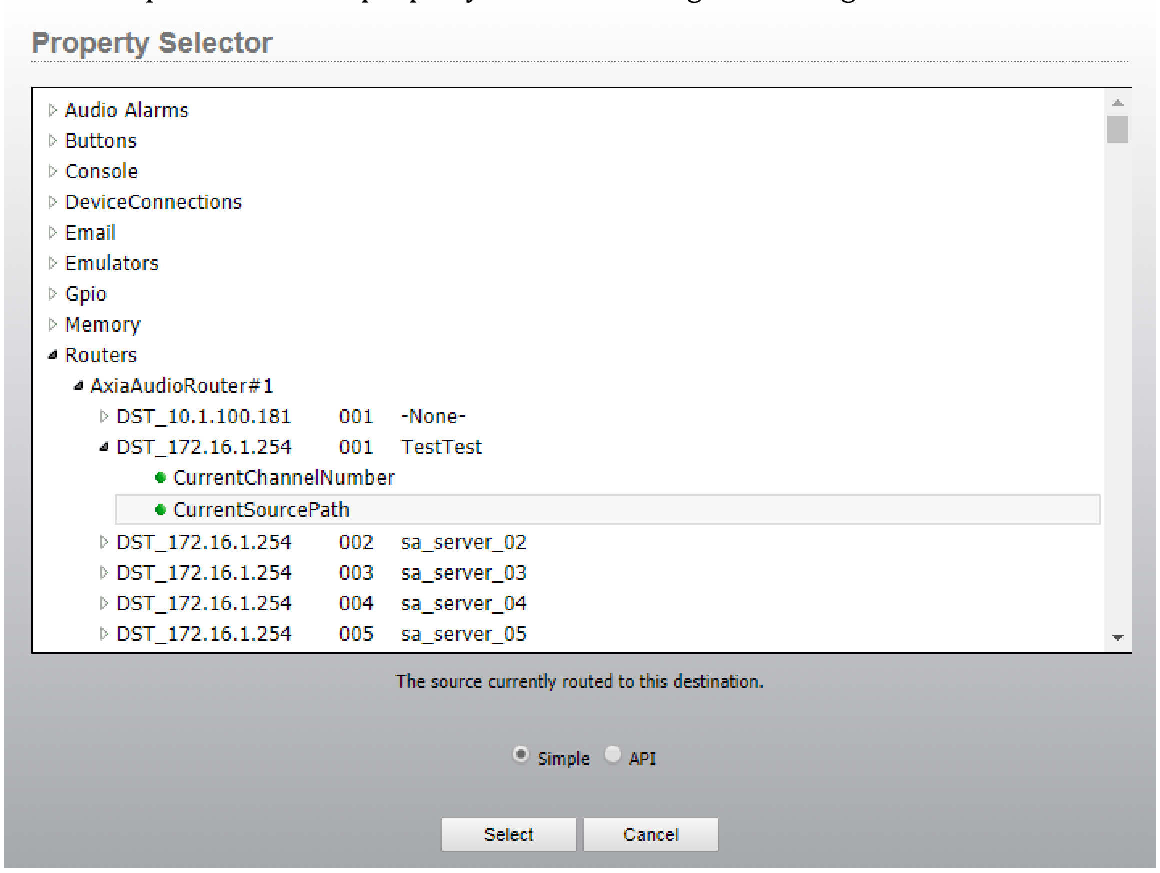


This is a simple shortcut that allows you to generate a simple flow to bind values to the property without having to switch over to the logic flow designer. In addition, since these flows simply bind system states to panel properties, the flows generated by this method do not count against your license count. It is an easy way to quickly add simple functionality. But it will be easier to understand with an example.

Let's say we want the button we have dragged onto the panel to trigger a route change. Select the button and enable the binding button on the mousedown event.

We are defining what we want to have happen when the button is pressed. Then double click on the endpoint in the flow image.

This will open the normal property selection dialog used in logic flows:



Property Selector

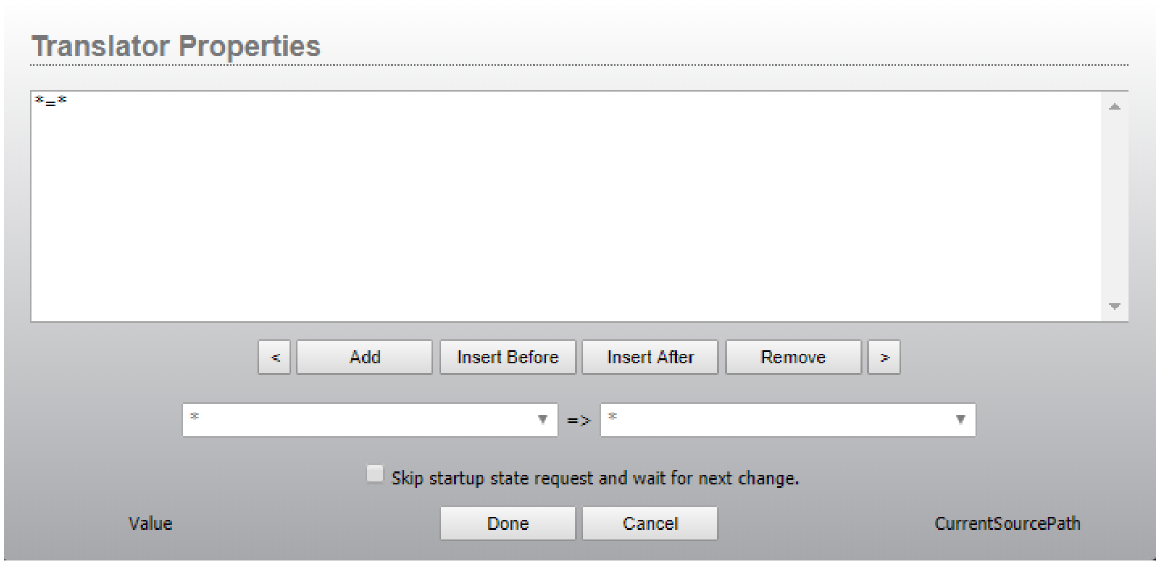
```
▷ Audio Alarms
▷ Buttons
▷ Console
▷ DeviceConnections
▷ Email
▷ Emulators
▷ Gpio
▷ Memory
◢ Routers
  ◢ AxiaAudioRouter#1
    ▷ DST_10.1.100.181    001   -None-
    ◢ DST_172.16.1.254    001   TestTest
        ● CurrentChannelNumber
        ● CurrentSourcePath
    ▷ DST_172.16.1.254    002   sa_server_02
    ▷ DST_172.16.1.254    003   sa_server_03
    ▷ DST_172.16.1.254    004   sa_server_04
    ▷ DST_172.16.1.254    005   sa_server_05
```

The source currently routed to this destination.

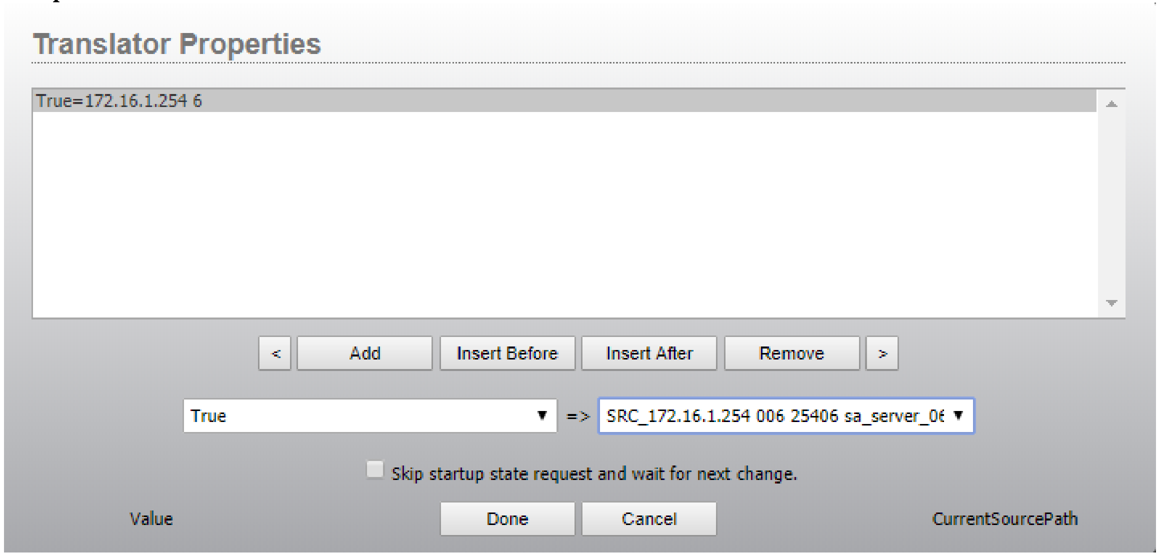
[✓ Simple] API

Expand the Routers section, expand a router, and expand the destination you want to change when the button is pushed, and then click on the CurrentSourcePath Property. Then click select.

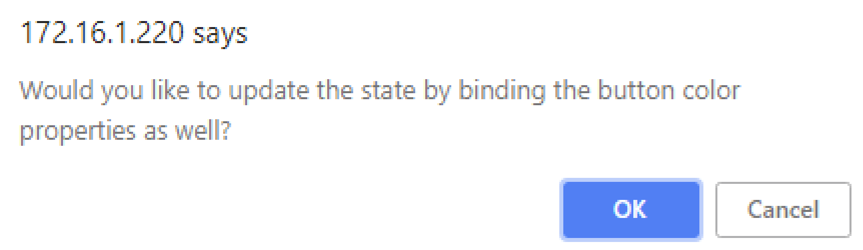
The system will automatically move to the translation dialog.



Click on the *=* item in the list and then select the True item in the left hand drop down and the source you want to route to the selected destination in the right hand drop down.



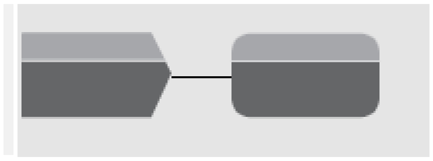
We have just defined that if the mousedown event is true, the sa_server_06 source will get routed to the TestTest destination. However, when you click Done you will get a pop up message.


172.16.1.220 says

Would you like to update the state by binding the button color properties as well?

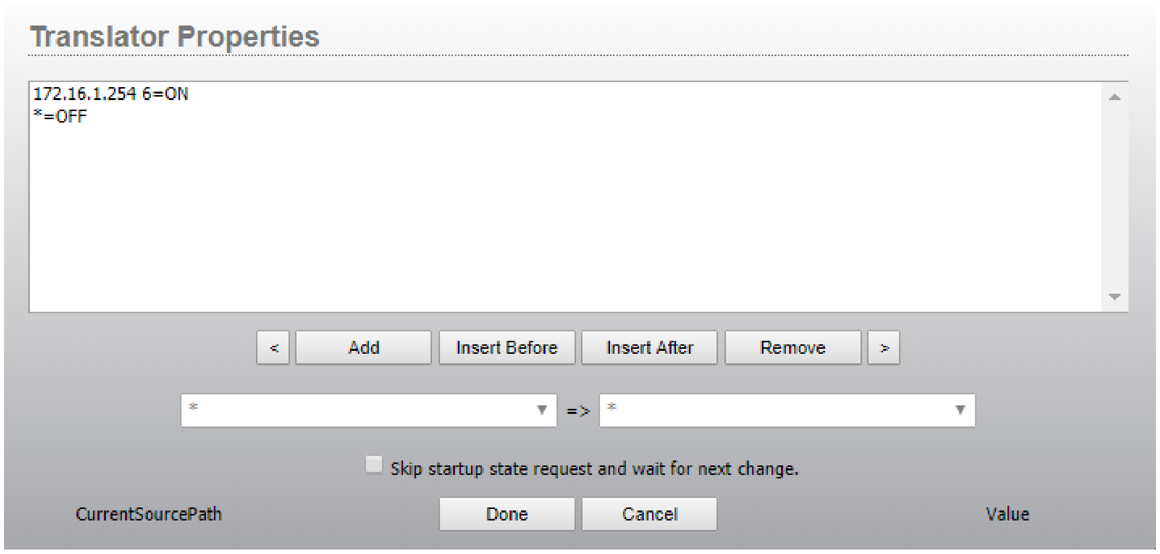
This message will only appear if you are generating flows on the mousedown or indicator properties of a button. In this case it knows that since we are defining what we want the button to do, we probably also want some indication on the button that the requested action has been done. If we click OK, it will automatically turn the binding button on for the indicator property and open the flow definition for the indicator. In this case it is smart enough to fill things in for us.

It is important to notice that the flow for the indicator property looks different than the one for the event.



The system is smart enough to know which direction these flows should go. For example, with an event the start point is not displayed in the flow because the event we have selected is the start point and the end point is what we are going to change. On the other hand, standard properties like the indicator are changed based on things that are changing in the system. So, you select what property in the system is causing the indicator to change. In that case the partial flow shows the start point and the translation and the endpoint is the property of the panel component we have selected. The rule of thumb is that events will display partial flows with an endpoint and other properties will display a flow with a start point. The missing part of the flow is the event or the property itself.

When we click OK to the message above you will notice the system will skip picking the start point. This is a special case for buttons where you are configuring the mouse down and indicator properties. Since we just defined what we want to change when mouse down is pressed, the pop up message is asking whether we want the successful change of that route to be reflected in the indicator. So, if we click OK, the system automatically turns on the binding for indicator and fills in the start point with the destination selected, and then displays the translation settings.


```
172.16.1.254 6=ON
*=OFF
```

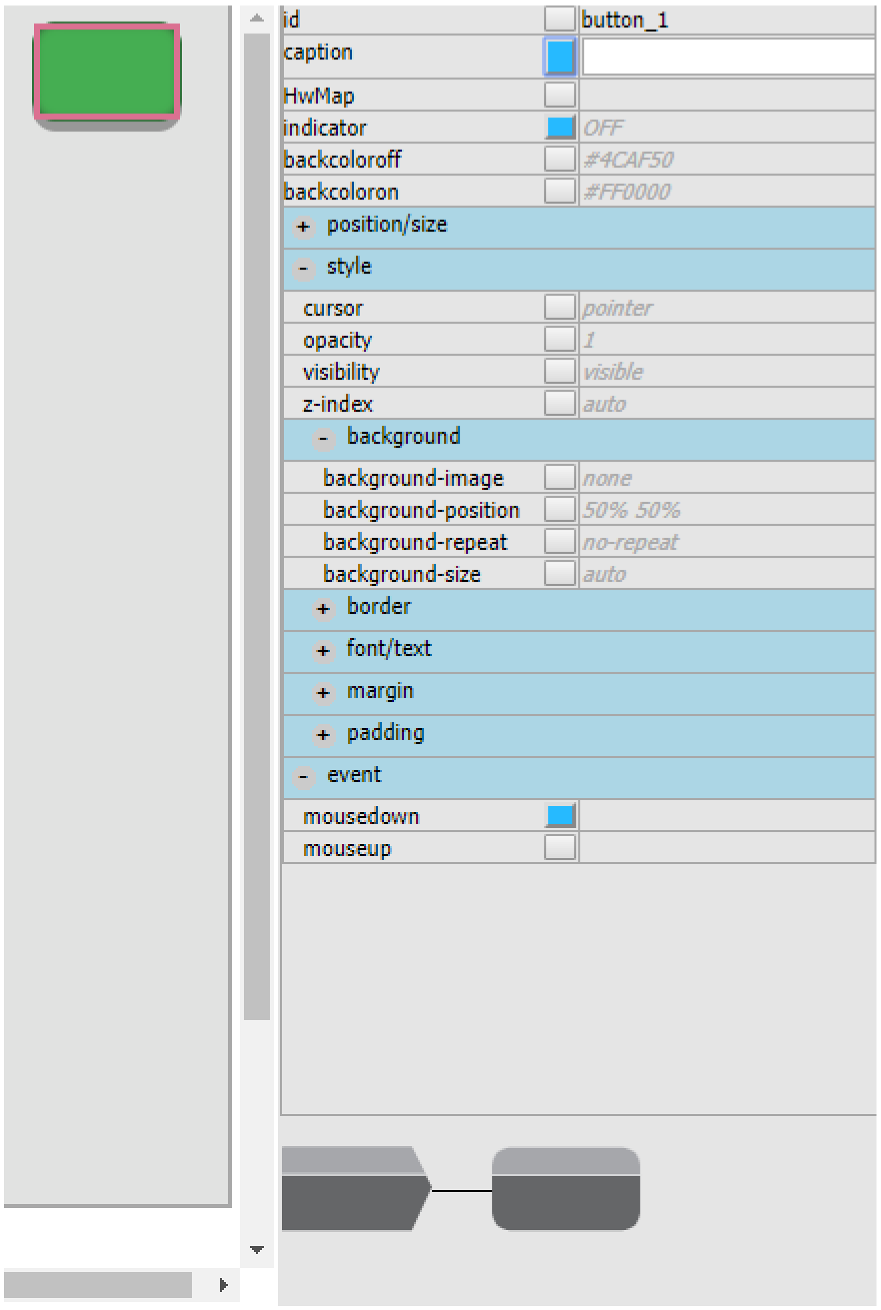
You will also notice that the system is assuming you will want the indicator to be on if the selected source is routed to the destination and off if it is not.

Click Done.

You will notice that the flows are no longer gray and have turned blue to indicate they have been defined. Saving the panel will cause the flows to be created and start working in logic flows. Flows created in this manner will be generated in a special folder in logic flows called _panels. The flows in this folder may be monitored for troubleshooting purposes but they cannot be changed from within logic flows. They are only edited through the panel designer.

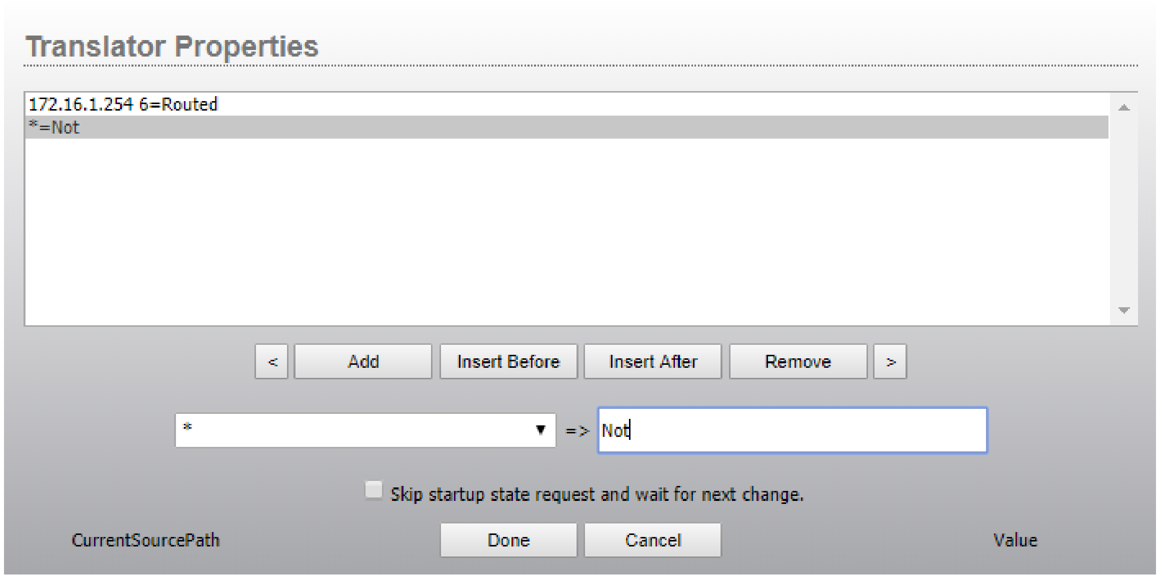
Note: To see these flows working you need to go back to user panels and open the panel for usage by clicking on the panel link rather than the edit link. It is sometimes useful to have this open in a separate browser tab while you are working. Then after you save changes in the designer you can switch over to the tab with the running panel, refresh the page and see your changes in action.

To extend this example, turn on the binding for caption as well.



id		button_1
caption		
HwMap		
indicator		OFF
backcoloroff		#4CAF50
backcoloron		#FF0000
+ position/size		
- style		
cursor		pointer
opacity		1
visibility		visible
z-index		auto
- background		
background-image		none
background-position		50% 50%
background-repeat		no-repeat
background-size		auto
+ border		
+ font/text		
+ margin		
+ padding		
- event		
mousedown		
mouseup		

Now double click on the start point and select the same CurrentSourcePath property of the same destination. Now in the translation select what you want the button to say when the source is routed and what you want it to say when it is not.



Translator Properties

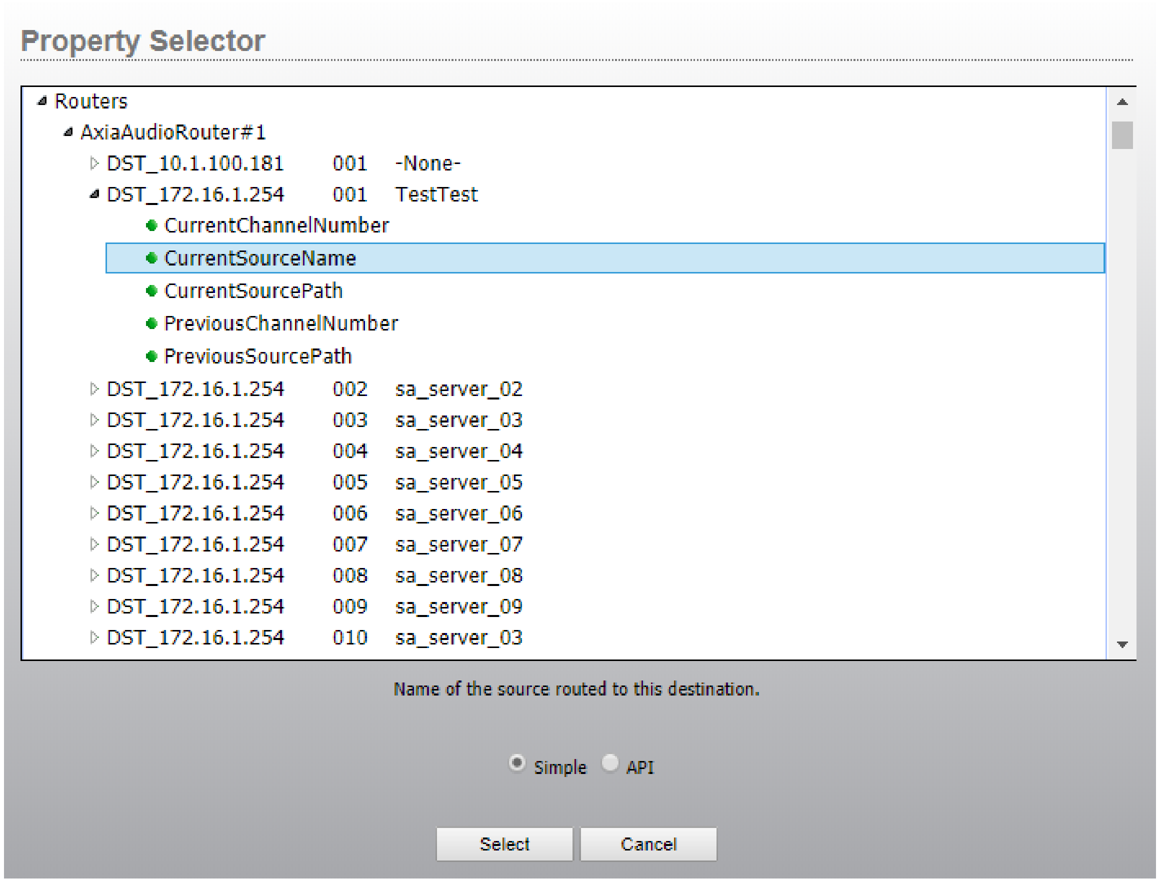
172.16.1.254 6=Routed
*=Not

* => Not

☐ Skip startup state request and wait for next change.

Click Done and Save to save your changes. Executing the panel should now display Routed or Not in the buttons caption depending on whether the selected source is routed to the destination.

A more useful change you can make with the caption property is to select the CurrentSourceName property of the destination in the logic flow property selector and use a *=* translation. You can change this by double clicking the start point while the caption property is selected.



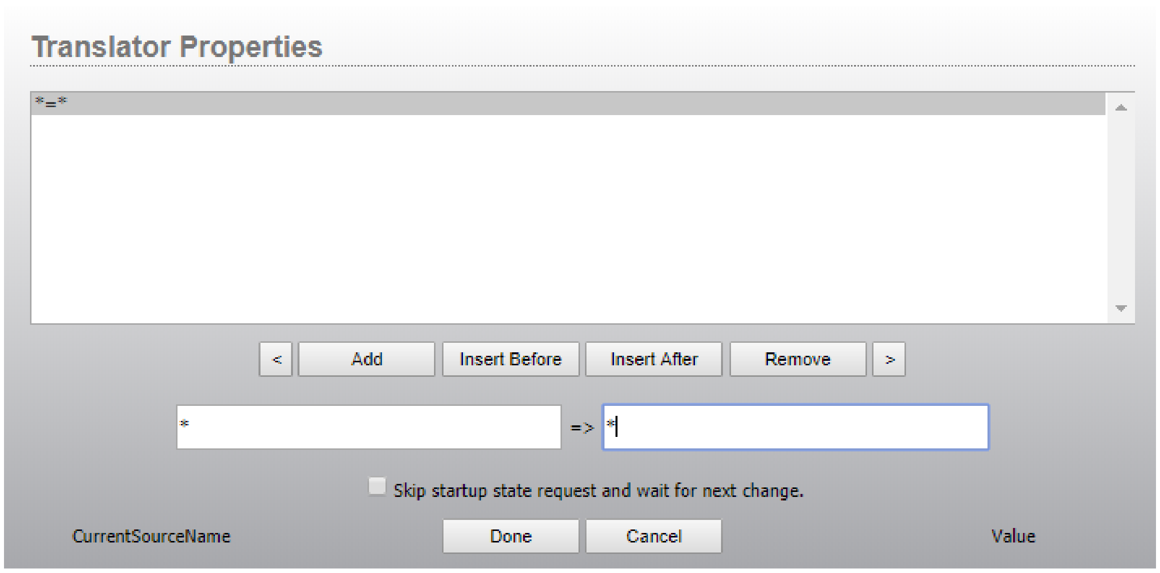
Property Selector

- Routers
  - AxiaAudioRouter#1
    - DST_10.1.100.181 001 -None-
    - DST_172.16.1.254 001 TestTest
      - ● CurrentChannelNumber
      - ● CurrentSourceName
      - ● CurrentSourcePath
      - ● PreviousChannelNumber
      - ● PreviousSourcePath
    - DST_172.16.1.254 002 sa_server_02
    - DST_172.16.1.254 003 sa_server_03
    - DST_172.16.1.254 004 sa_server_04
    - DST_172.16.1.254 005 sa_server_05
    - DST_172.16.1.254 006 sa_server_06
    - DST_172.16.1.254 007 sa_server_07
    - DST_172.16.1.254 008 sa_server_08
    - DST_172.16.1.254 009 sa_server_09
    - DST_172.16.1.254 010 sa_server_03

Name of the source routed to this destination.

[✓ Simple] API

Now pick the currentSourceName Property instead of the CurrentSourcePath property and click select.



Translator Properties

=

* => *

Skip startup state request and wait for next change.

CurrentSourceName

Value

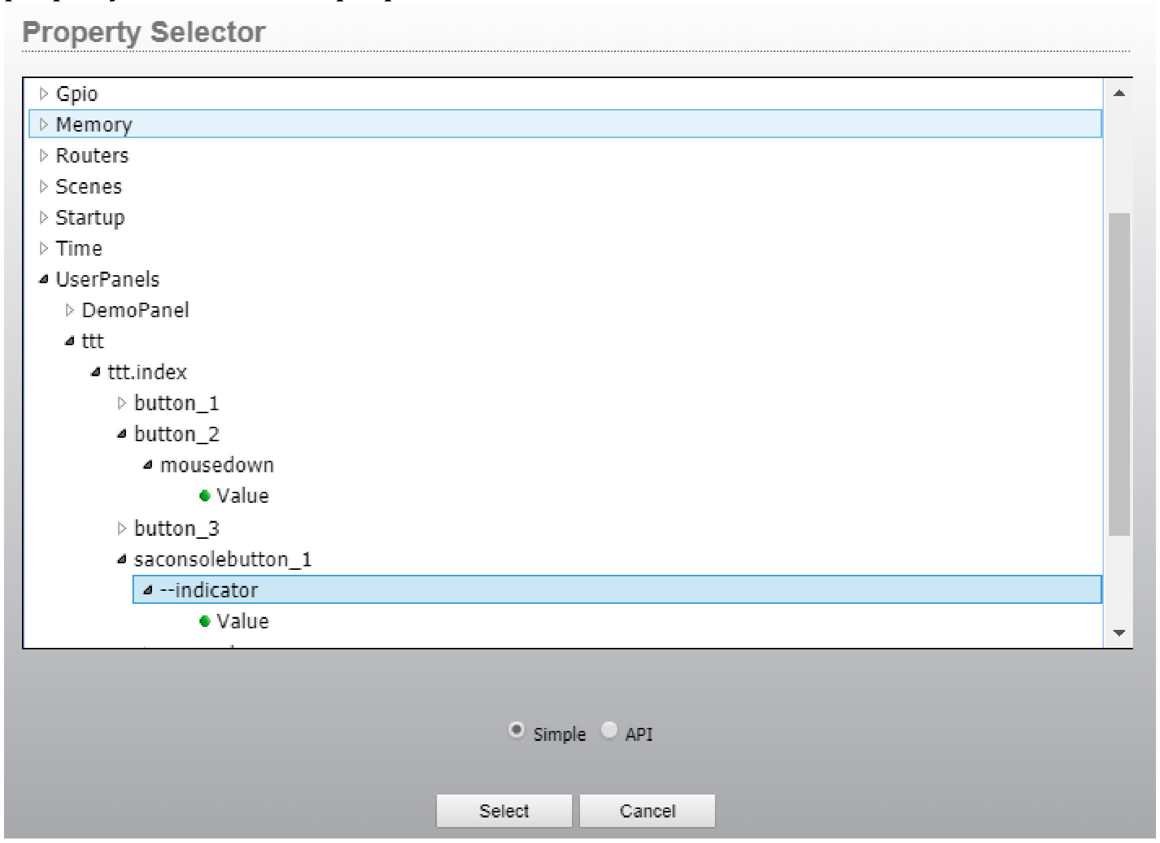
Change the translation to be *=*. Then click Done. Now the button's caption will be tied to the name of whatever source is currently routed to the destination.

After saving the panel and opening it up for use you should find that pressing the button will make the route change, the indicator will light or unlight according to the back-color properties depending on whether the route is made, and the caption should display the name of the source that is routed to the destination.

By using these techniques, you can edit functionality into the panel components in very easy and extremely powerful ways.

Complex panel flows

In many cases you may wish to create more complex flows than described in the examples above. For example, you may want your indicator state on a button to be the product of numerous conditions in the system. Those kinds of flows can still easily be created, but they must be created within the logic flows designer. Simply enable the binding button for the properties these flows need to manipulate without generating a flow in designer. Save the panel and then from within the logic flows property selector, these properties will be available for use.


Property Selector

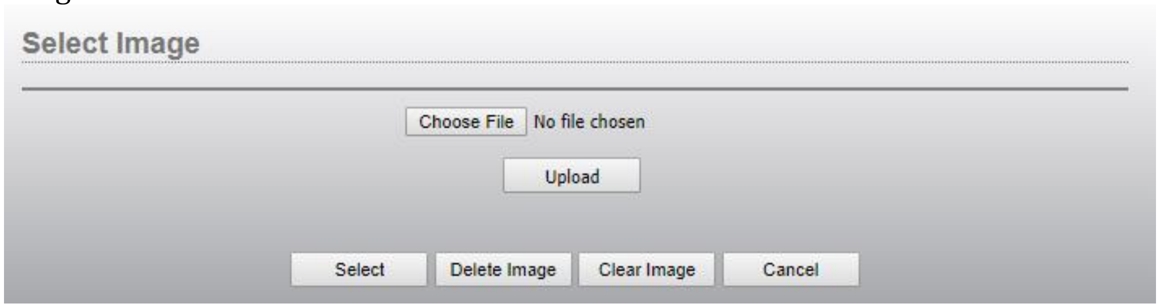
```
▷ Gpio
▷ Memory
▷ Routers
▷ Scenes
▷ Startup
▷ Time
◢ UserPanels
   ▷ DemoPanel
   ◢ ttt
      ◢ ttt.index
         ▷ button_1
         ◢ button_2
            ◢ mousedown
                ● Value
         ▷ button_3
         ◢ saconsolebutton_1
            ◢ --indicator
                ● Value
```

[● Simple] API

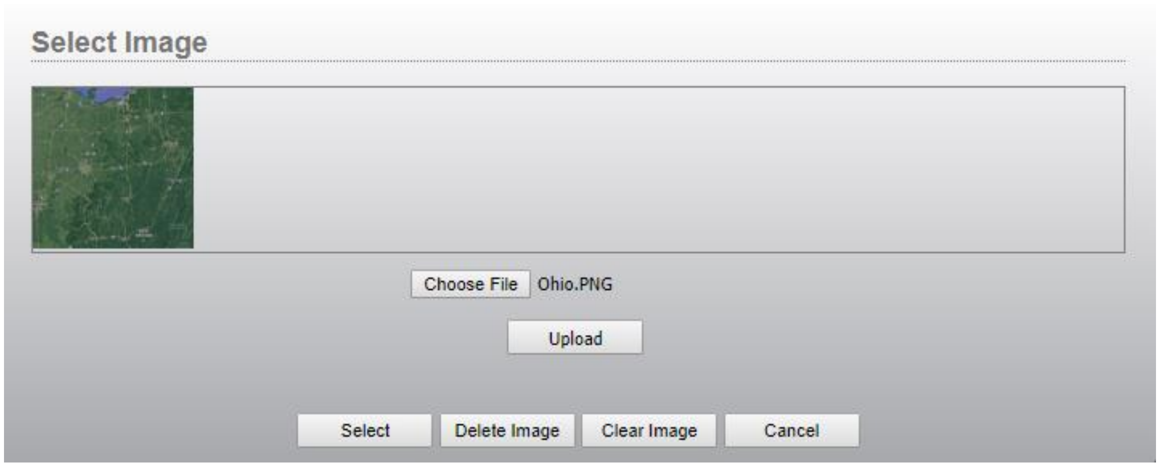
Changing Pages

The toolbars section above talked about multiple pages within a panel. This section will use the information learned above to create a panel that switches between multiple pages.

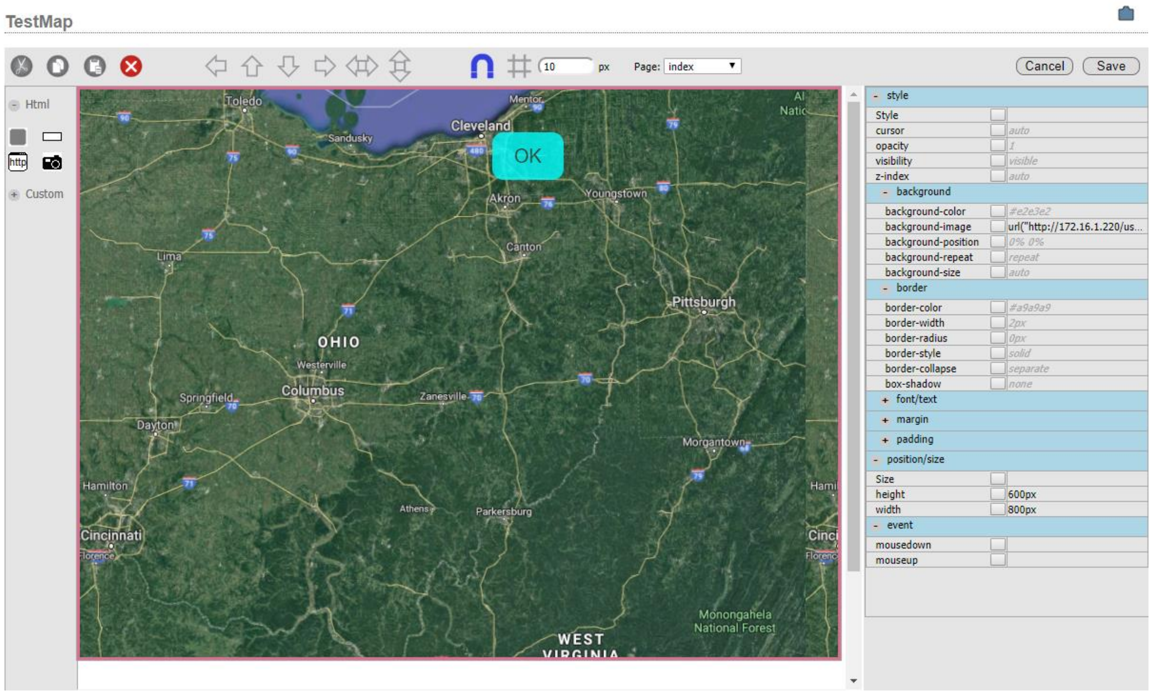
First create a new panel and Save it as TestMap. Click on the panel and then in the property grid, expand the style and background sections. Click on the value for the background-image property. A dialog will appear for selecting and uploading images.



In this case I am going to upload an image of a map of Ohio. Click Choose File and select your stored image and then click Upload. The image will then appear in the selection list.

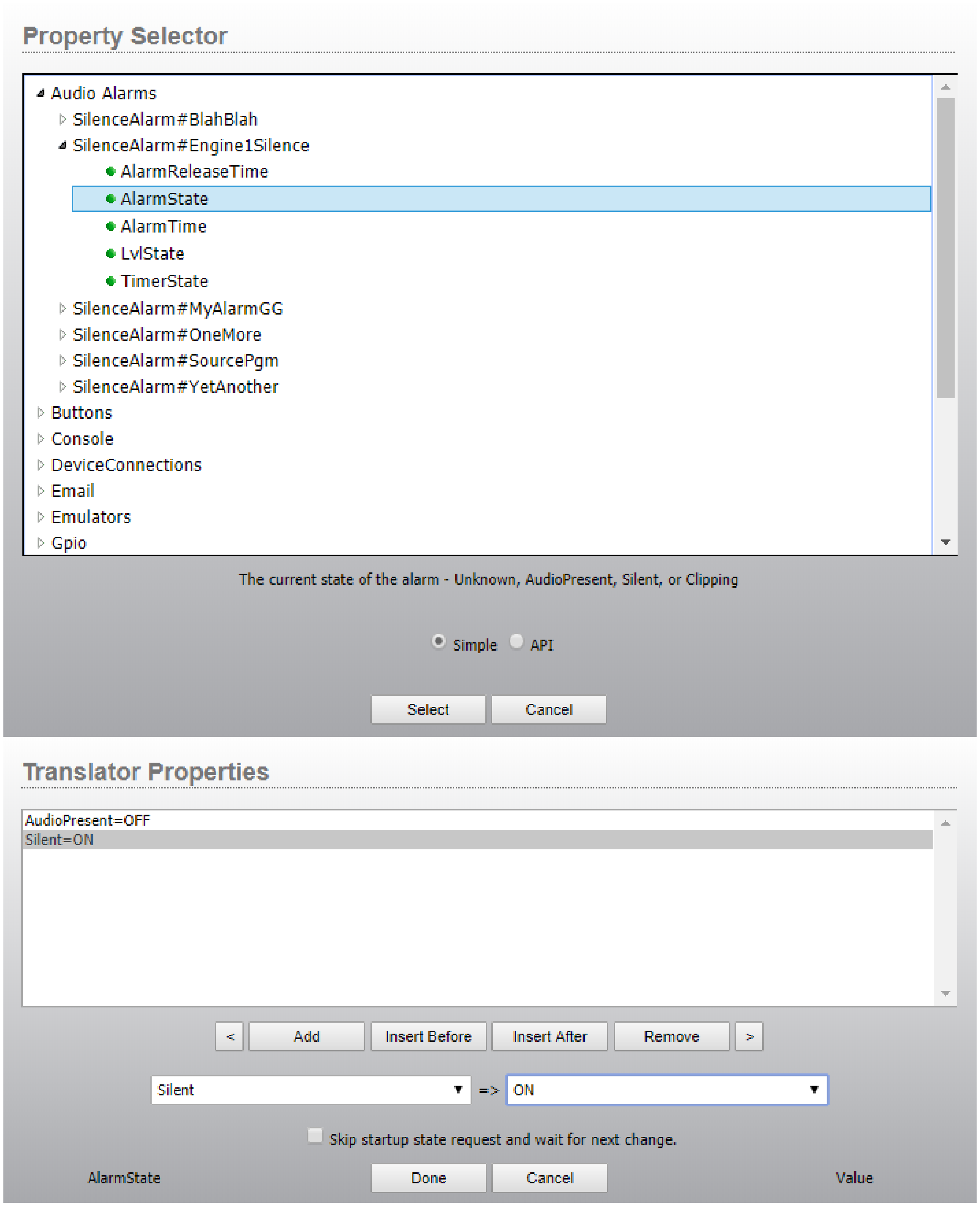


Click on the image and click Select. The background of the panel will now display the image of Ohio. Drag a button on and position it next to Cleveland and then update the caption and color properties of the button to read Ok and select cyan for the backcoloroff. Under the border section, set the box shadow to none. Then set the opacity property to .8. Your panel should now look something like:



- style		
Style	☐	
cursor	☐	auto
opacity	☐	1
visibility	☐	visible
z-index	☐	auto
- background		
background-color	☐	#e2e3e2
background-image	☐	url("http://172.16.1.220/us...
background-position	☐	0% 0%
background-repeat	☐	repeat
background-size	☐	auto
- border		
border-color	☐	#a9a9a9
border-width	☐	2px
border-radius	☐	0px
border-style	☐	solid
border-collapse	☐	separate
box-shadow	☐	none
+ font/text		
+ margin		
+ padding		
- position/size		
Size	☐	
height	☐	600px
width	☐	800px
- event		
mousedown	☐	
mouseup	☐	

Next enable the binding for the indicator state of the button. Then click on the start point and select a silence alarm state as the start point. For the translation select audio presence as indicator off and silence for on.


Property Selector

- Audio Alarms
  - SilenceAlarm#BlahBlah
  - SilenceAlarm#Engine1Silence
    - AlarmReleaseTime
    - AlarmState
    - AlarmTime
    - LvlState
    - TimerState
  - SilenceAlarm#MyAlarmGG
  - SilenceAlarm#OneMore
  - SilenceAlarm#SourcePgm
  - SilenceAlarm#YetAnother
- Buttons
- Console
- DeviceConnections
- Email
- Emulators
- Gpio

The current state of the alarm - Unknown, AudioPresent, Silent, or Clipping

[✓ Simple] API

Translator Properties

AudioPresent=OFF
Silent=ON

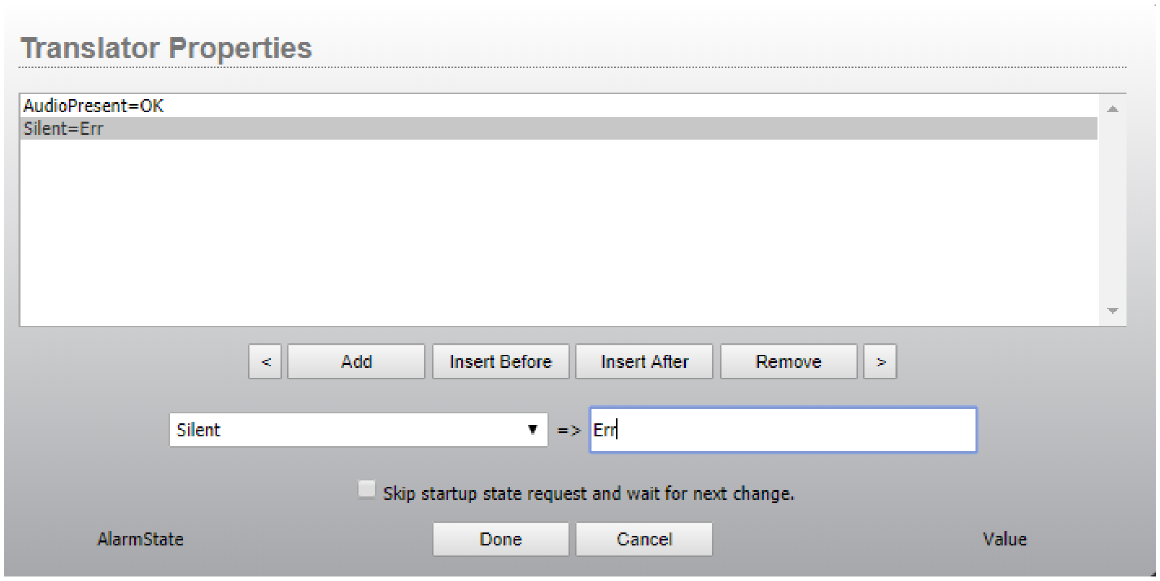
Silent => ON

Skip startup state request and wait for next change.

AlarmState — Value

After clicking Done, click cancel when it asks about the mouse down action. In this case the mouse down is going to change pages and has nothing to do with the property we are picking for the indicator so we will set it separately.

Next turn the binding for the caption property on and double click the start point of that flow. Select the same Alarm State property, but for the translation make silence convert to Err and Audio Present to OK.


Translator Properties

AudioPresent=OK
Silent=Err

Silent => Err

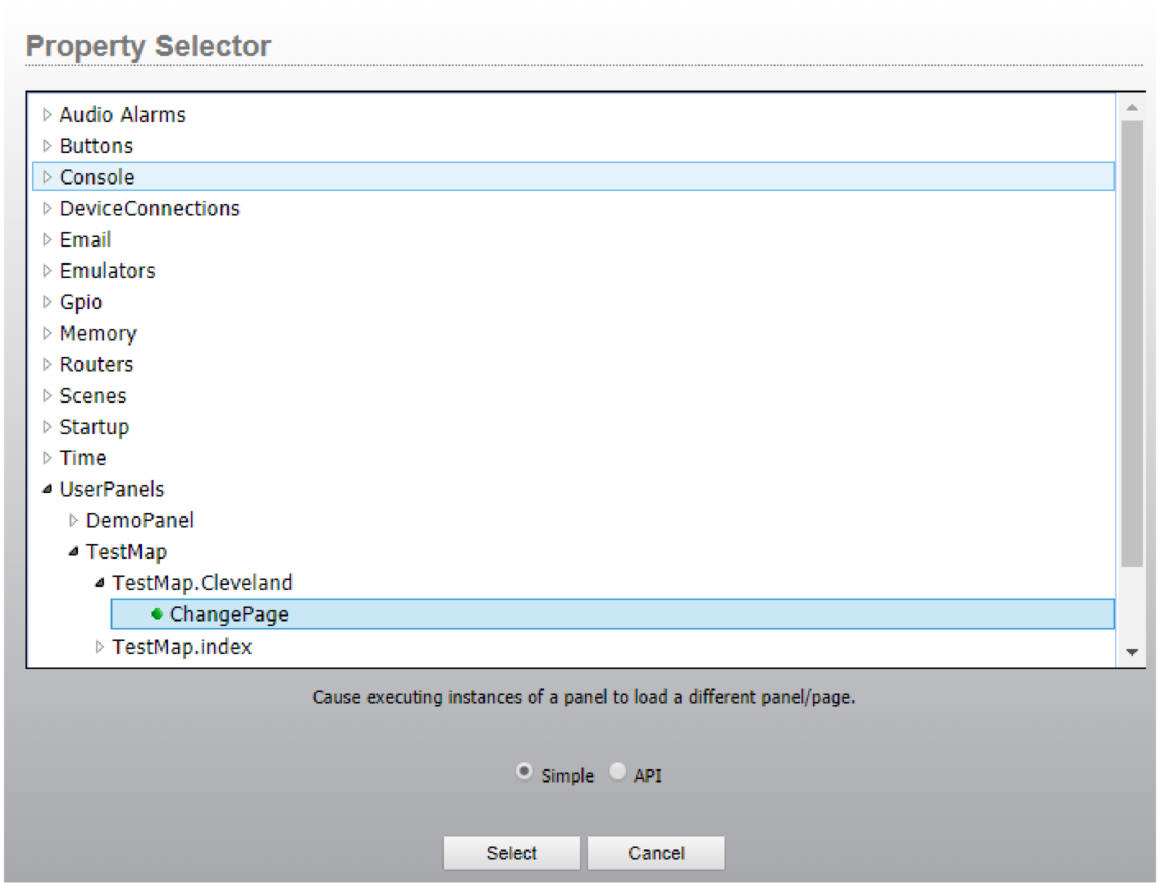
Skip startup state request and wait for next change.

Click Done and the save the panel.

At this point if we were to execute this panel, the button next to Cleveland should show cyan and OK if the silence alarm has audio presence and Err and Red if it is silent.

Next from the page drop down select [NewPage]. Save the New page with the name Cleveland. Drag a new button onto the new page and set its caption to return. Turn on the binding for mouse down on this button and then click the end point. Expand the UserPanels section of the logic flow property selector and the TestMap.Cleveland section and select the ChangePage property.

Note: If you do not find TestMap.Cleveland in the logic flow property selector it means you have not saved the new page yet.



Cause executing instances of a panel to load a different panel/page.

In the translation set True=TestMap.index. Click Done and click cancel for updating the indicator property. Then save the Cleveland page again. We have just defined that when we click the return button on the Cleveland page it will set the ChangePage property of the Cleveland page to TestMap.index effectively returning to the index page.

Now use the page drop down to select the index page again. Click the OK button. Turn the binding for the mouse down event on, click the endpoint, select the TestMap.index ChangePage property and then in the translation set true=TestMap.Cleveland. Click Cancel for the binding indicator question and save the panel.

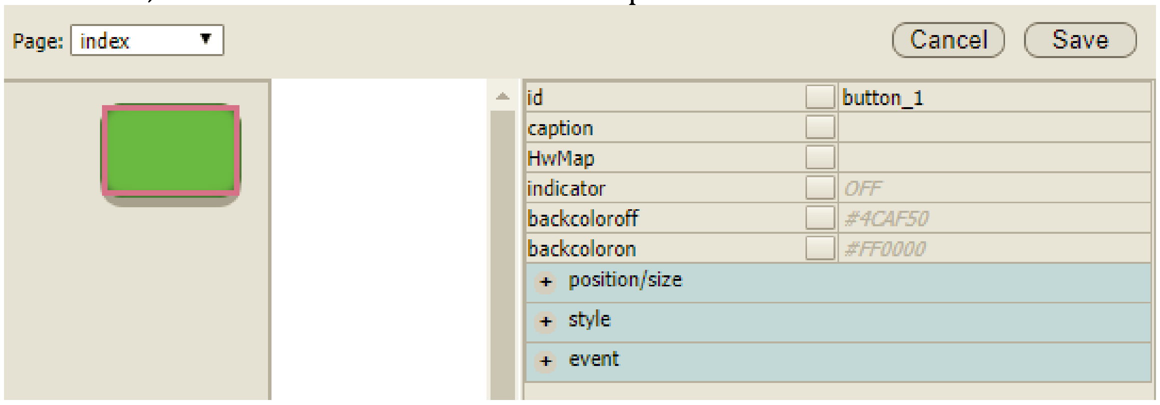
At this point, the button on the main map should switch caption and indicator color based on the silence alarm state and clicking this button should take you to a new panel called Cleveland. Clicking the return button should take you back to the main map index page.

It is important to note that you are always selecting the changepage property of the current page you are on and setting its value through the translation dialog to the page you want to move to. Another interesting point is that the change page property is also available to normal logic flows. So, for example rather than binding this to the mouse down event, we could create a flow in logic flows that automatically switches the panel page to the Cleveland panel whenever a silence alarm occurs and back again when audio presence is restored. Additionally, the change page property is not limited to pages within the same panel. You could switch to a completely different panel.

Obviously in a real use case we would fill the Cleveland page with more information than just a return button. For example, we might create a system flow chart page with meters and buttons and silence alarm states of various parts of the chain to more easily determine where the failure occurred.

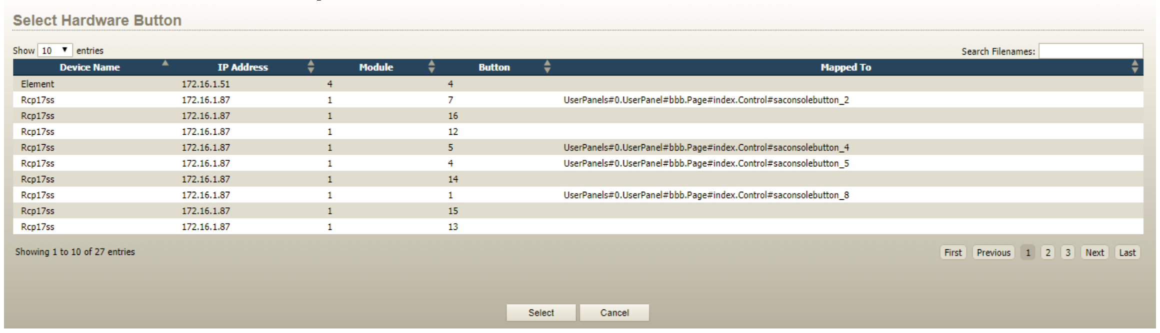
Hardware mapping buttons

Buttons created in a user panel (html or console) may also be hardware mapped to physical LCD buttons in the console or rack mount button panels. Hardware mapping makes the physical button mirror the behavior of the software button. To define this, click on a button created in a user panel.



id	button_1
caption	
HwMap	
indicator	OFF
backcoloroff	#4CAF50
backcoloron	#FF0000

Then click on the HwMap field. A dialog will appear from which you can select known buttons in the system.



Device Name	IP Address	Module	Button	Mapped To
Element	172.16.1.51	4	4	
Rcp17ss	172.16.1.87	1	7	UserPanels#0.UserPanel#bbb.Page#index.Control#saconsolebutton_2
Rcp17ss	172.16.1.87	1	16	
Rcp17ss	172.16.1.87	1	12	
Rcp17ss	172.16.1.87	1	5	UserPanels#0.UserPanel#bbb.Page#index.Control#saconsolebutton_4
Rcp17ss	172.16.1.87	1	4	UserPanels#0.UserPanel#bbb.Page#index.Control#saconsolebutton_5
Rcp17ss	172.16.1.87	1	14	
Rcp17ss	172.16.1.87	1	1	UserPanels#0.UserPanel#bbb.Page#index.Control#saconsolebutton_8
Rcp17ss	172.16.1.87	1	15	
Rcp17ss	172.16.1.87	1	13	

Showing 1 to 10 of 27 entries

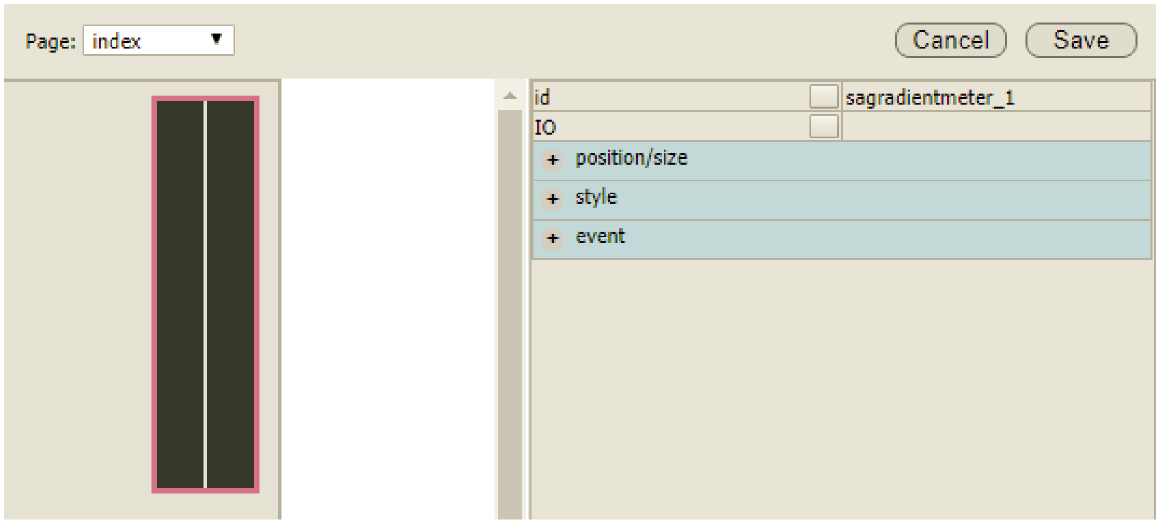
The mapped to field will display if the button has already been hardware mapped as each hardware button can only be mapped to a single software button. Physical buttons do not have to be hardware mapped. They can also just be used directly with logic flows. If the button you want is not shown, make sure the Lcd panel or console is in the devices list. If not, it needs to be discovered. See the PathfinderCore PRO manual for more details on discovering devices. Note that Lcd panels must be manually discovered in devices using the add and Lwcp discovery. If the device exists in the system and it is a console but it is not showing the lcd buttons, try pressing a few of the lcd buttons and then refreshing the web page.

Select the button you want to hardware map and click Select.

Once the panel is saved hardware mapping in html panels takes place natively in the application and does not require hardware map logic flows like the legacy panels did.

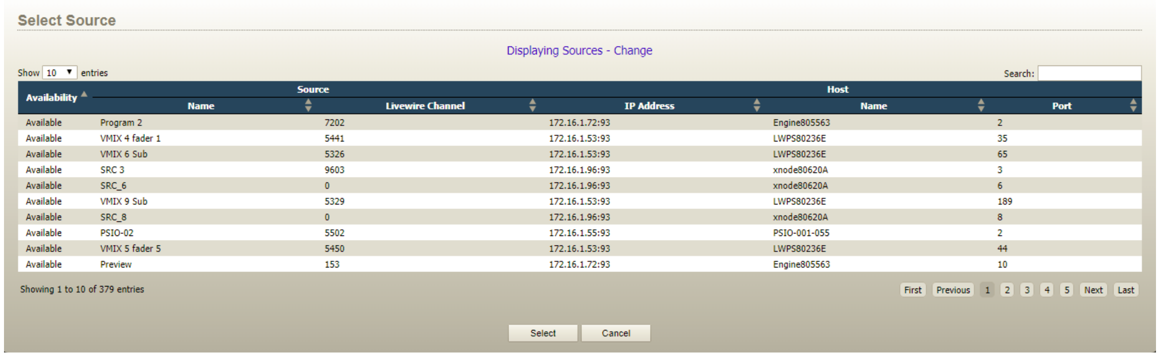
IO selection for meters and faders

Under the custom components there are two different kinds of meters and two different kinds of faders. The components section below will provide details on these components. But for all of these components the method for selecting the IO which the component will display is the IO property. For example, drag a gradient meter onto the palette and resize it to an appropriate size.



id	sagradientmeter_1
IO	

Clicking in the IO property field will open an IO selection dialog box.


Select Source
Displaying Sources - Change
Show 10 entries
Search:
First Previous 1 2 3 4 5 Next Last
Select
Cancel

Availability	Source		Host		
	Name	Livewire Channel	IP Address	Name	Port
Available	Program 2	7202	172.16.1.72:93	Engine805563	2
Available	VMIX 4 fader 1	5441	172.16.1.53:93	LWPS80236E	35
Available	VMIX 6 Sub	5326	172.16.1.53:93	LWPS80236E	65
Available	SRC 3	9603	172.16.1.96:93	xnode80620A	3
Available	SRC_6	0	172.16.1.96:93	xnode80620A	6
Available	VMIX 9 Sub	5329	172.16.1.53:93	LWPS80236E	189
Available	SRC_8	0	172.16.1.96:93	xnode80620A	8
Available	PSIO-02	5502	172.16.1.55:93	PSIO-001-055	2
Available	VMIX 5 fader 5	5450	172.16.1.53:93	LWPS80236E	44
Available	Preview	153	172.16.1.72:93	Engine805563	10

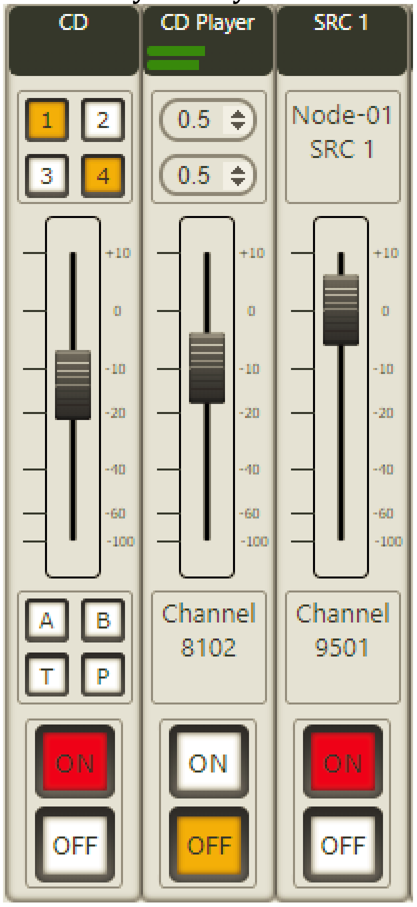
Showing 1 to 10 of 379 entries

The link at the top of the page will display whether sources or destinations are currently being displayed and clicking the link will change between the two. You can use the search box to narrow down the list within either sources or destinations. Click an IO whose metering you want to be tied to the meter and click select. The IO field will fill with the path of the selected IO. Since we are in the designer, no metering will be displayed. It will only display when the panel is executed.

It is also important to note that turning the binding button on for the IO parameter will make it available to logic flows such that you can dynamically change the IO assigned to the meter or fader.

ConsoleFader

Some special attention should be given to the ConsoleFader component. This component will dynamically adjust what it displays depending on the type of IO it is associated with. Those changes will only be shown when the panel is executed. That is when the capabilities of the assigned IO are analyzed. For example, after dragging a console fader onto the panel and assigning IOs, when the panel is executed you may see controls that look like:



In this case the first fader is tied to the channel input of an element console. The second fader is tied to a vmix channel input and so the component changes to show the time up and down parameters. And the third shows an XNode source.

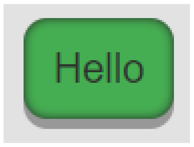
Components

Each component has a whole variety of css properties and component events that are available. As mentioned previously in this document many of these can be understood by googling web page css descriptions. This section will describe each component as well as custom properties of that component that are beyond the standard css style properties. While we have included images of the components below it is important to note that the css properties can be used to make the components look much different than the default we are showing below. Feel free to adjust border, color, and shadowing properties to achieve the design desired.

Important Note: Each component has an ID property. This allows you to define a name for the component. This is useful in differentiating components in logic flows and debugging so it is a good habit to name components as you create them. Ids must be unique within the page.

Important Note: Since this is still beta some properties may appear or disappear as we continue to work on the system. In some situations, complex components have inherited css style properties that are not applicable and so should be hidden. We will continue to fine tune these parameters as the beta testing process progresses.

Html – Button



The button component is a button that can be designed with different colors, borders, background pictures, etc and can then be used to control and indicate changes in the system. This component has been covered in detail in the examples above. It is important to note that wherever this button can be used, a console button could also be used. The difference is that the console button has a slightly more elegant look.

Custom properties include:

- caption: updates the inner html text of the button
- hwmap: used to select a hardware Lcd button. This hardware Lcd button will then mirror the behavior of the software button.
- indicator: used to set the button indication to On, Off, or Flash. The colors used for On, Off, and the two flash colors are the backcoloroff and on properties.
- backccoloron: color used when the indicator state is on.
- backccoloroff: color used when the indicator state is off.

Html - Label

Labels can be used to generate textual label information in the system. The caption of the label may be dynamically updated by flows by binding the textContent property.

Custom properties include:

- textContent: The textual information displayed by the label.

Html - Web page

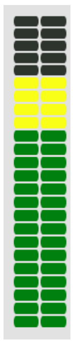
The web page component allows you to embed a web page from another site into the panel. This component is an html iframe. It is important to note especially during testing that some sites (such as google) prevent their content from being displayed in an iframe. This component can be used to display video streams and other web page content. Use the src property to enter the url for the page to be displayed.

If you intend to use this as a background component with other components on top, you may need to manually adjust the z-index property of this component or the overlayed components to get them to display properly.

Html – Image

This allows you to embed an image in a page. However, most elements also support the background-image property. For example, you can put an image on the panel itself without using an image component and you can put background images on buttons and labels as well. This component can also be used to create background borders around a set of components. It is recommended to use the background-image property rather than the src property to assign an image to this component.

Custom – Led Meter

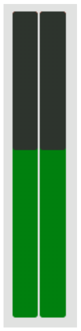


This component can be tied to audio ios that support metering in the system. Use the IO property to display a list of sources or destinations which can be assigned to the meter.

Custom properties include:

- IO: Used to select the audio io this meter will display.
- orientation: Used to select whether the meter will display horizontally or vertically.
- metercale: Used to select the scale of the meter. Options include standard, linear, and british.
- metrics: Used to define whether the numerical values for the meter are displayed next to the meter or not. Options include none, lefttop, middle, rightbottom.
- autosizefont: Used to define whether the metrics font will scale automatically as the size of the meter is adjusted.
- Led\color: properties used to adjust the on and off colors used for each of three sections of the meter.
- Led\border: used to adjust the border settings of the individual led blocks.

Custom – Gradient Meter

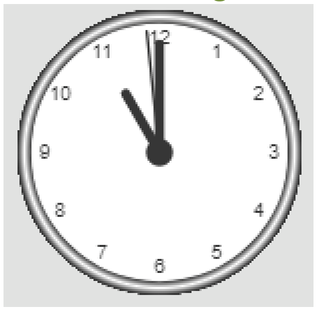


This component can be tied to audio ios that support metering in the system. Use the IO property to display a list of sources or destinations which can be assigned to the meter.

Custom properties include:

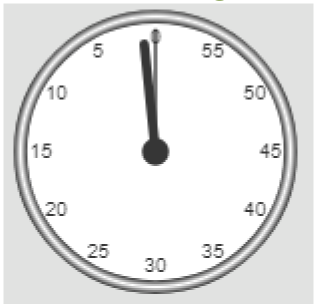
- IO: Used to select the audio io this meter will display.
- orientation: Used to select whether the meter will display horizontally or vertically.
- metercale: Used to select the scale of the meter. Options include standard, linear, and british.
- metrics: Used to define whether the numerical values for the meter are displayed next to the meter or not. Options include none, lefttop, middle, rightbottom.
- style\optimum percent: defines the optimum meter percentage.
- autosizefont: Used to define whether the metrics font will scale automatically as the size of the meter is adjusted.
- Led\color: properties used to adjust the on and off colors used for each of three sections of the meter.

Custom – Analog Clock



This component displays the current time using an analog clock style.

Custom – Analog Coutdown

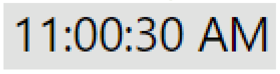


This component allows you to define a trigger a countdown. The clock will display the countdown value.

Custom properties include:

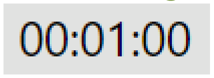
- countdownlength: time in seconds for the countdown.
- countdownstart: generally exposed via bindings for a logic flow to trigger the start of a countdown. Options are true or false.

Custom – Digital Clock



This component displays the current time using a digital clock style.

Custom – Digital Coutdown

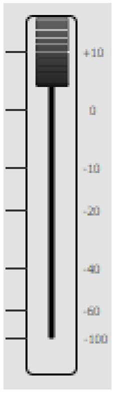


This component allows you to define and trigger a countdown. The clock will display the countdown value.

Custom properties include:

- countdownlength: time in seconds for the countdown.
- countdownstart: generally exposed via bindings for a logic flow to trigger the start of a countdown. Options are true or false.

Custom – Fader



This component can be tied to audio ios whose gain may be manipulated. Use the IO property to select the controllable source or destination.

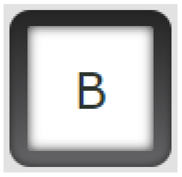
Custom properties include:

- IO: used to elect the IO whose gain you wish to control.
- metrics: how the numbering for the fader is displayed. Options include none and lefttop.
- slider-height: percentage of the overall component height used for the slider height.
- slidebarwidth: width in pixels of the bar on which the slider rides.
- metricoffset: percent of width used for the metrics.
- metriclinecolor: color for the lines drawn for each metric line.
- slidebarcolor: color of the bar on which the fader rides.
- slidebarradius: radius in pixels
- slidebardisplay: whether the slider bar is displayed.

- faderimage: rather than using the default designed slider, an image may be used.
- slider-margin-left: margin offset for the slider.
- slider-border-style: border style for the slider.
- slider-border-width: border width for the slider.
- slider-border-radius: border radius for the slider.
- slider-border-color: border color for the slider.
- optimum: gain level which is designed to be optimum or unity.
- type: whether this is an audio slider or a slider for controlling other numeric values. Options are audio and linear. Note linear option and the ability to control other numeric values is for future use.
- autosizefont: whether the metric font automatically scales as the size of the fader is adjusted.

Important Note: Currently Qor faders are not supported by this component in the first beta release. Additionally it currently uses the fach object for fusion faders. Support for Qor and Lwch will be added in the near future.

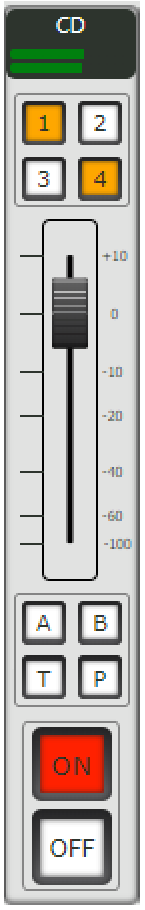
Custom – Console Button



This button works the same as the html button but has a more interesting look.

Custom properties include:
- saconsolebutton-caption: updates the text displayed on the button
- hwmap: used to select a hardware Lcd button. This hardware Lcd button will then mirror the behavior of the software button.
- indicator: used to set the button indication to On, Off, or Flash. The colors used for On, Off, and the two flash colors are the backcoloroff and on properties.
- backccoloron: color used when the indicator state is on.
- backccoloroff: color used when the indicator state is off.
- saconsolebutton-image: used instead of the standard css style background image to update an image inside the button. Because this component is built from several embedded html objects, this makes sure the correct inner component displays the image.

Custom – Console Fader



This fader is a smart fader that displays a variety of things depending on the type of io assigned to it. In the example above, this has been assigned to a Fusion console input. When executed the component understands the type of IO to which it has been assigned and updates the controls accordingly. In this case it shows the name of the source profile, a meter obtained from the input stage of the fader, the program buss assignments, a fader which maps to the fusion fader, A and B user buttons, talk and preview buttons, and On and Off. The Io can be defined using the IO property.

Custom properties include:

- IO: used to elect the IO whose gain you wish to control.

Important Note: Currently Qor faders are not supported by this component in the first beta release. Additionally it currently uses the fach object for fusion faders. Support for Qor and Lwch will be added in the near future.

Launching the panel from a Desktop icon (windows)

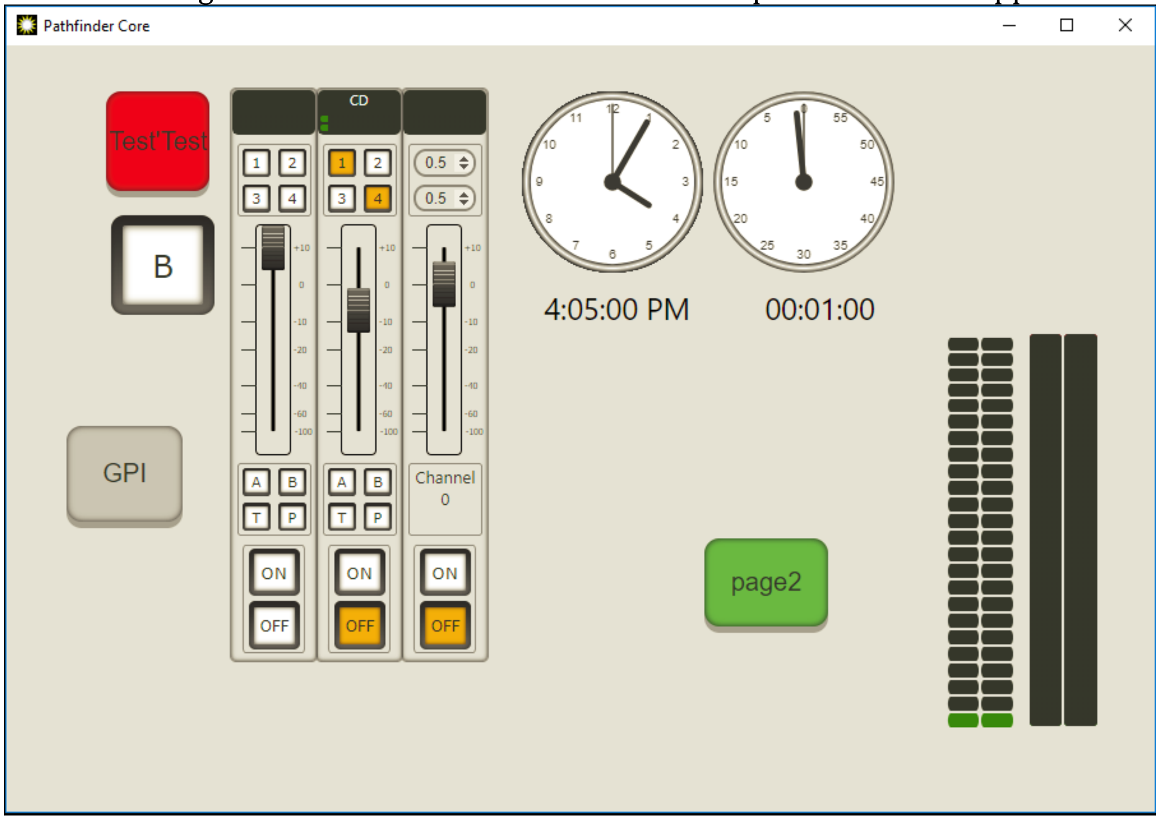
If you are using chrome, there are some command line options that will allow you to launch a PathfinderCore PRO panel as if it was an application. Copy and paste the following into a notepad or text editor:

```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" --
app="data:text/html,<html><body><script>window.location='http://Admin:Admin
@172.16.1.220/userpanelframemin.php?panel=ttt&page=index';</script></body>
</html>"
```

When copied, this should be one line in the editor. The quotes at the beginning and end are part of the text so do not remove them. After the word "location" there is an http link. Change the Username:Password to be one that reflects your system. Also change the ip address (172.16.1.220) to match the ip address of your Pathfinder Core PRO. Finally, after panel= change the name of the panel you want to launch from ttt to the name of your panel. If you want to target a page other than the default index page, change that as well. Once you have the edits made, select the entire text again and copy it to your clipboard.

Now, right click on your desktop and select new shortcut. Paste the text into the location of the item field. Click next and input a name into the shortcut and click Finish.

Double clicking on the shortcut should now launch the panel as its own application.



If you always want the panel to launch in the same place on the screen you can add another option: window.moveTo(580,240). For example:

```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe” --
app="data:text/html,<html><body><script> window.moveTo(580,240);
window.location='http://Admin:Admin@172.16.1.220/userpanelframemin.php?panel=ttt&page=index';<
/script></body></html>"
```

In the future we will investigate a way to generate the shortcut (or at least the shortcut text) automatically.

Version 1.3.13.21 Changes

Disk Space Object

This version adds a disk space object in the API and exposes some of those properties in Logic Flows:

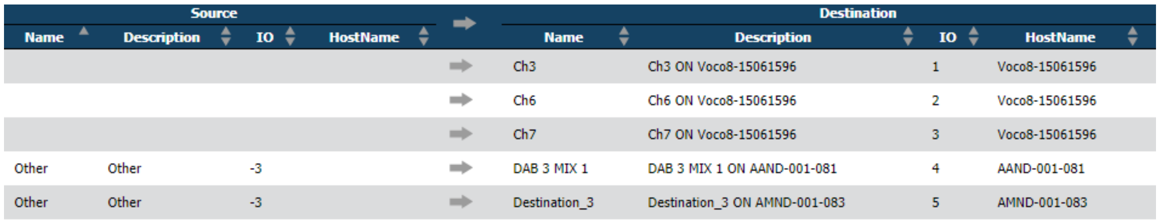
```
get System#0.DiskInfo#0

indi System#0.DiskInfo#0 AvailableSpace="262.67", CurrentBankSize="949.51",
PercentUsed="72.34", AlertPercents=95, SpaceAlertState=-1, FriendlyName=DiskInfo
```

The AlertPercents property allows you to define a comma delineated list of disk space percentages. If any of those percentages is crossed, the SpaceAlertState will be updated with that percentage. This can be used in Logic flows to generate alerts if the disk is getting full. By default the AlertPercents is set to 95%. It can be changed using the Advanced Options in the system configuration page. This api object can also be used to obtain the current disk utilization information.

Virtual Routers

IO column

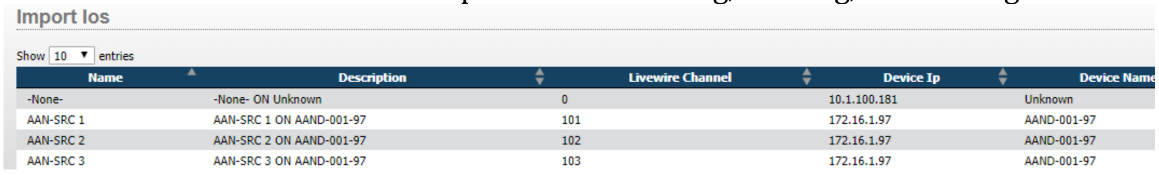
This version adds an IO column to the routes page of Virtual Routers to display the virtual router IO number.



Source					Destination			
Name	Description	IO	HostName		Name	Description	IO	HostName
					Ch3	Ch3 ON Voco8-15061596	1	Voco8-15061596
					Ch6	Ch6 ON Voco8-15061596	2	Voco8-15061596
					Ch7	Ch7 ON Voco8-15061596	3	Voco8-15061596
Other	Other	-3			DAB 3 MIX 1	DAB 3 MIX 1 ON AAND-001-081	4	AAND-001-081
Other	Other	-3			Destination_3	Destination_3 ON AMND-001-083	5	AMND-001-083

Livewire Channel Number in import lists

When importing Livewire Audio router sources into a virtual router, the livewire channel number will now also be present for filtering, viewing, and sorting.



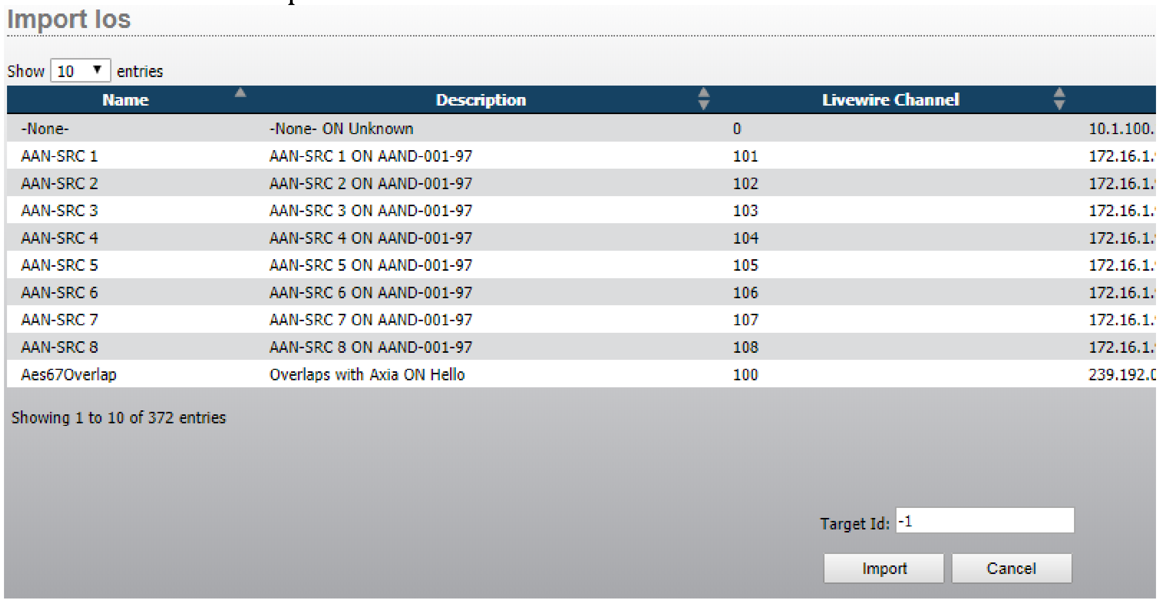
Import Ios

Show 10 entries

Name	Description	Livewire Channel	Device Ip	Device Name
-None-	-None- ON Unknown	0	10.1.100.181	Unknown
AAN-SRC 1	AAN-SRC 1 ON AAND-001-97	101	172.16.1.97	AAND-001-97
AAN-SRC 2	AAN-SRC 2 ON AAND-001-97	102	172.16.1.97	AAND-001-97
AAN-SRC 3	AAN-SRC 3 ON AAND-001-97	103	172.16.1.97	AAND-001-97

Import Target Id

This version adds the ability to enter the id in the virtual router where you want the imported IOs to start when importing IOs. This will fail if there are not enough open numbers at the target id to support the number of selected IOs. Leaving the value at the default -1 will import to the end of the virtual router.

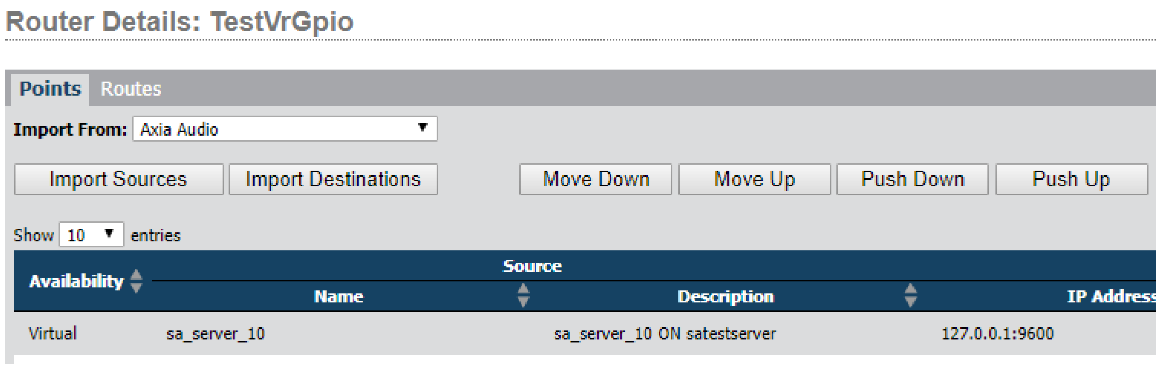


Name	Description	Livewire Channel	
-None-	-None- ON Unknown	0	10.1.100.
AAN-SRC 1	AAN-SRC 1 ON AAND-001-97	101	172.16.1.
AAN-SRC 2	AAN-SRC 2 ON AAND-001-97	102	172.16.1.
AAN-SRC 3	AAN-SRC 3 ON AAND-001-97	103	172.16.1.
AAN-SRC 4	AAN-SRC 4 ON AAND-001-97	104	172.16.1.
AAN-SRC 5	AAN-SRC 5 ON AAND-001-97	105	172.16.1.
AAN-SRC 6	AAN-SRC 6 ON AAND-001-97	106	172.16.1.
AAN-SRC 7	AAN-SRC 7 ON AAND-001-97	107	172.16.1.
AAN-SRC 8	AAN-SRC 8 ON AAND-001-97	108	172.16.1.
Aes67Overlap	Overlaps with Axia ON Hello	100	239.192.C

Showing 1 to 10 of 372 entries

Target Id: -1

Move Up/Down and Push Up/Down



Import From: Axia Audio

Availability	Source		
	Name	Description	IP Address
Virtual	sa_server_10	sa_server_10 ON satestserver	127.0.0.1:9600

This version adds the ability to move IOs in the virtual router around when designing the router. This effectively alters the number of the IO in the virtual router. It is recommended to sort by Port when using this feature as it manipulates up and down based on the port number and not the sort order. This is a design tool to specify a specific numbering for the virtual Ios and is most useful when those IOs will be exposed to emulated routing protocols that expect a number to be assigned to each io.

Move Up – moves the IO up in the router effectively decreasing its IO number and swapping with its lower next door neighbor if necessary.

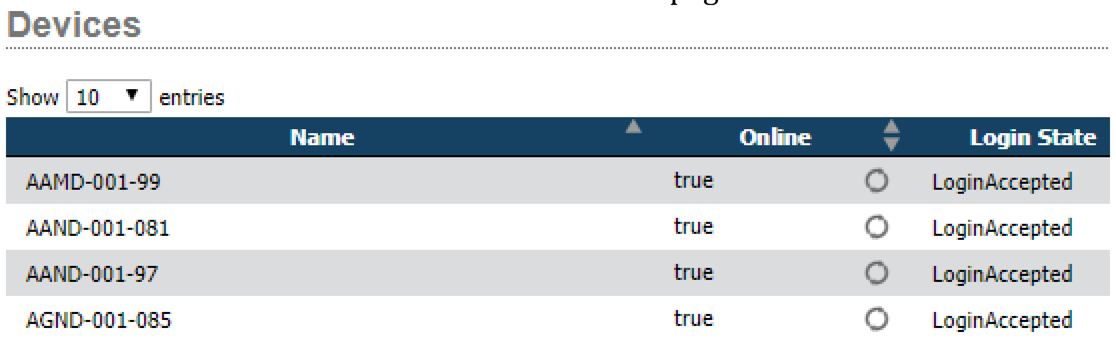
Move Down – moves the IO down in the router effectively increasing its IO number and swapping with its higher next door neighbor if necessary.

Push Up - moves the IO up in the router effectively decreasing its IO number and pushing any lower numbered IOs it runs into up as well decreasing their numbers until it reaches a hole in the numbering or 0.

Push Down - moves the IO down in the router effectively increasing its IO number and pushing any higher numbered IOs it runs into down as well increasing their numbers until it reaches a hole in the numbering or the highest value in the router.

Device Manual Reconnect

This version adds a reconnect icon to the devices page.



Devices

Show 10 entries

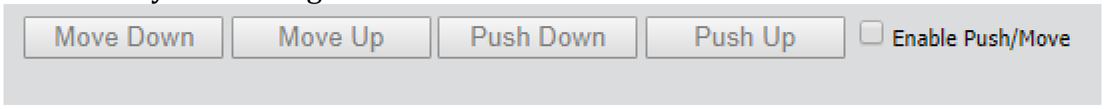
Name	Online		Login State
AAMD-001-99	true	○	LoginAccepted
AAND-001-081	true	○	LoginAccepted
AAND-001-97	true	○	LoginAccepted
AGND-001-085	true	○	LoginAccepted

Clicking on the circular icon for any device will force PathfinderCore PRO to recycle the connection to that device.

Version 1.3.13.22 Changes

Virtual Router Editing Move Up/Down Enable

This version adds a checkbox to enable/disable the move up/down and push up/down buttons added in 1.3.13.21. This is a safety measure to prevent accidentally reordering a virtual router.



Version 1.3.13.22 Changes

Send hex escape characters

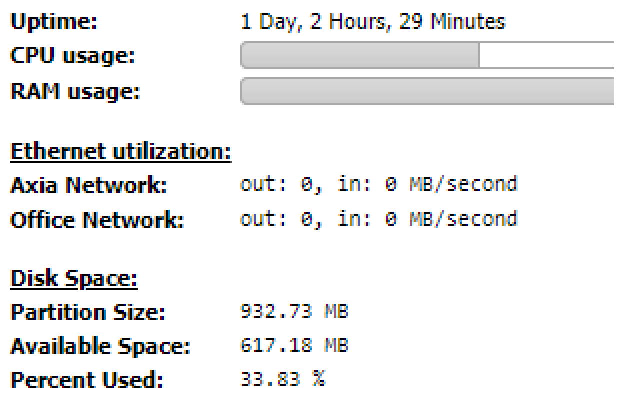
This version adds escape characters that can be used in generic protocol translator ToSend properties as well as in Lwrp and Lwcp toSend properties. Escape values are now:

- \cr – carriage return
- \lf – line feed
- \t – tab
- \%XX – hex asci character where XX is a two digit hex number

Double slash escapes the escape. So for example \%41 sends A but \\%41 sends \%41 and %41 sends %41.

Disk Space System Status

This version adds disk space to the system status page.



Uptime:	1 Day, 2 Hours, 29 Minutes
CPU usage:	
RAM usage:	

Ethernet utilization:

Axia Network:	out: 0, in: 0 MB/second
Office Network:	out: 0, in: 0 MB/second

Disk Space:

Partition Size:	932.73 MB
Available Space:	617.18 MB
Percent Used:	33.83 %

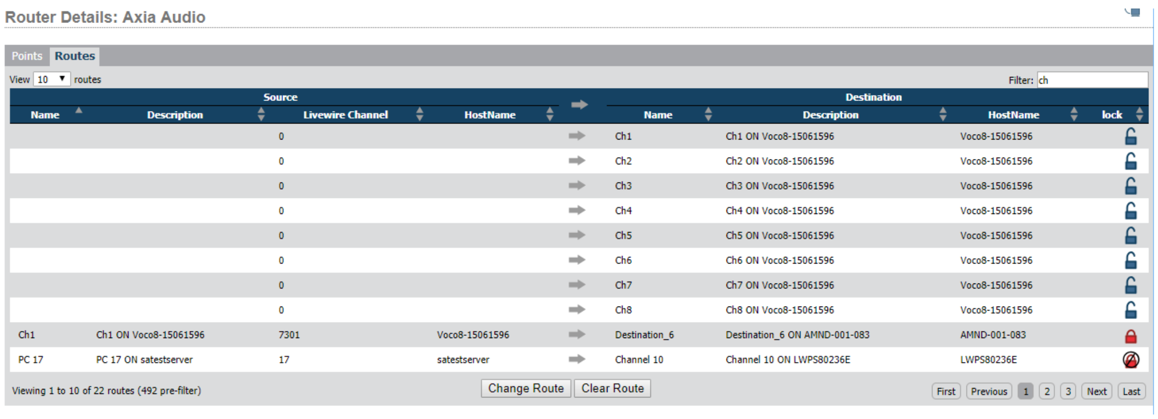
Version 1.3.13.28 Changes

Html Panel Clustering

This version is the first one that supports the clustering of html user panels first introduced in 1.3.13.20. If you have html panels created before this version, you may have to execute a manual sync (review manual and/or contact support) after upgrading both nodes in the cluster in order to initially sync the previously created panels. After that, newly created and edited panels should sync across the cluster properly. Please report any issues you experience with this new functionality.

Route Locking

This version adds route locking. Each route in a router can be Unlocked, Locked, or SystemLocked.



Router Details: Axia Audio

View 10 routes

Filter: ch

Source					Destination			
Name	Description	Livewire Channel	HostName	➡	Name	Description	HostName	lock
		0		➡	Ch1	Ch1 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch2	Ch2 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch3	Ch3 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch4	Ch4 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch5	Ch5 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch6	Ch6 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch7	Ch7 ON Voco8-15061596	Voco8-15061596	🔓
		0		➡	Ch8	Ch8 ON Voco8-15061596	Voco8-15061596	🔓
Ch1	Ch1 ON Voco8-15061596	7301	Voco8-15061596	➡	Destination_6	Destination_6 ON AMND-001-083	AMND-001-083	🔒
PC 17	PC 17 ON satestserver	17	satestserver	➡	Channel 10	Channel 10 ON LWPS80236E	LWPS80236E	🚫

Viewing 1 to 10 of 22 routes (492 pre-filter)

Each lock state is represented by an icon in the Routes table of the router:

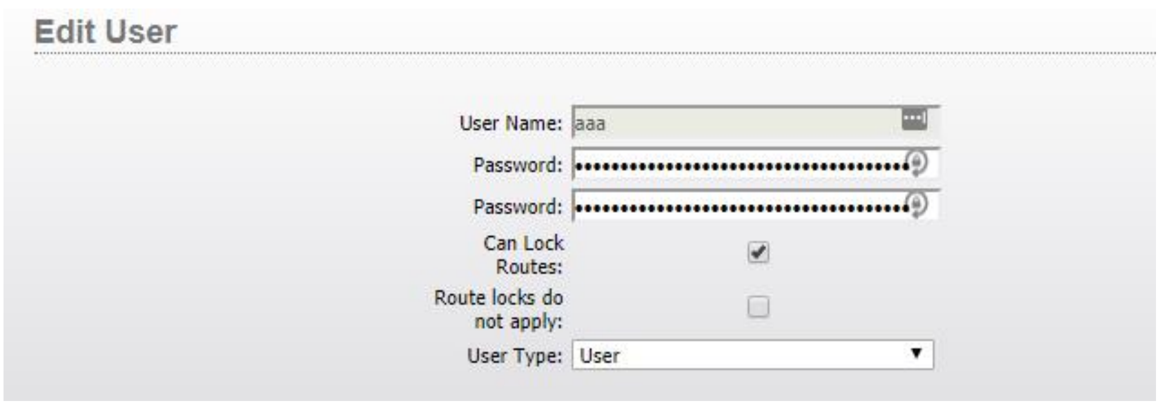
Locked: 🔒

Unlocked: 🔓

System Locked: 🚫

System locked (with the exception of virtual routers described below) indicates that the device itself does not allow route changes to be made on that IO. This is typically seen with console faders where source changes must be accomplished via a source profile load rather than a route change.

It is important to note that other than system locks, locking and unlocking is a Pathfinder state and is not actually changing anything in the equipment as Axia equipment does not have lock parameters. Therefore a locked route in Pathfinder Core PRO could still be changed by the device's web page. Locking or unlocking a route is as simple as clicking on the destinations icon in the route list. Routes that are locked must be unlocked before they can be changed. In addition it is possible to define whether a particular user has the rights to change route locks. When defining a user, a field now exists to configure this:



Edit User

User Name:	aaa
Password:	••••••••••••••••••••••••••••••••••••
Password:	••••••••••••••••••••••••••••••••••••
Can Lock Routes:	✓
Route locks do not apply:	☐
User Type:	User

Note: The Route locks do not apply option is for future use and not fully functional at this point in time.

Enabling the Can Lock Routes checkbox can be used to define if the user has the rights to lock and unlock routes. This allows an Administrator to lock routes and make them unchangeable by normal users.

Some special notes need to be made in relation to virtual routers. A virtual destination is comprised on one or several base destinations from other routers. If you lock a virtual destination, it will not lock the underlying base destination. This is by design and allows an Administrator to create a general user router with locked destinations and an engineering router where those same destinations are unlocked. However, if the base point becomes locked, then the virtual destination will display as system locked and the lock can only be removed by unlocking the lock on the base point. In the case where a virtual destination has multiple base points, the locking of any of the base points results in the entire virtual destination becoming system locked.

Note: The fact that locking a virtual destination does not lock its underlying base point is different from how previous versions of Pathfinder worked but provides better functionality as described above.

Revised Network Icons

The network connectivity icons have been updated for a better look in this version.

Connected:

Disconnected:

Panel Clone Fix

Previous versions of the html panels introduced in 1.3.13.20 exhibited a bug with cloning a panel. The cloned panel would not allow new image files to be uploaded. This was due to an incorrect security setting on the cloned folders. This has been fixed in the clone process, but if you have previously cloned folders that exhibit this problem, they may be fixed with an API command. Open a telnet session to port 9600, login, and send the command. Example:

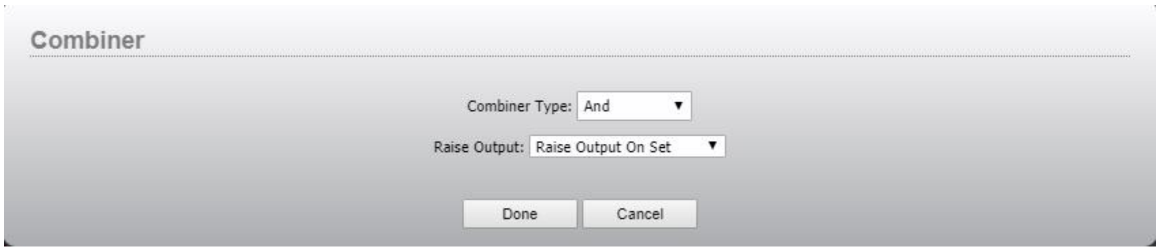
```
Login Admin Admin
SET UserPanels#0 FixPanelSecurity=True
```

This command will reset the permissions on all panel folders and subfolders. This command should only be necessary if you have cloned html panels in versions prior to this version and can't upload pictures to the cloned panels.

Version 1.3.13.30 Changes

Logic Flow Combiners

This version adds a feature to combiners called Raise Output. Double clicking on a combiner will present an additional field for configuring this option:

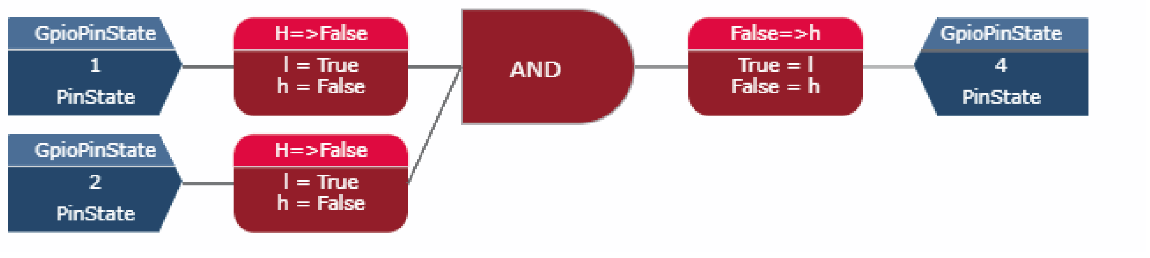


Combiner

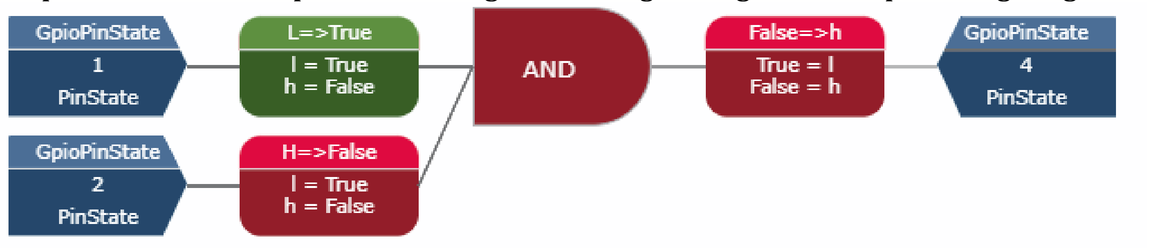
Combiner Type: And

Raise Output: Raise Output On Set

By default, new combiners will be set for raise output on set. This matches the way combiners have worked since version 1.3.11.10. The other option is Raise Output On Change. To understand the difference, let's look at a flow:



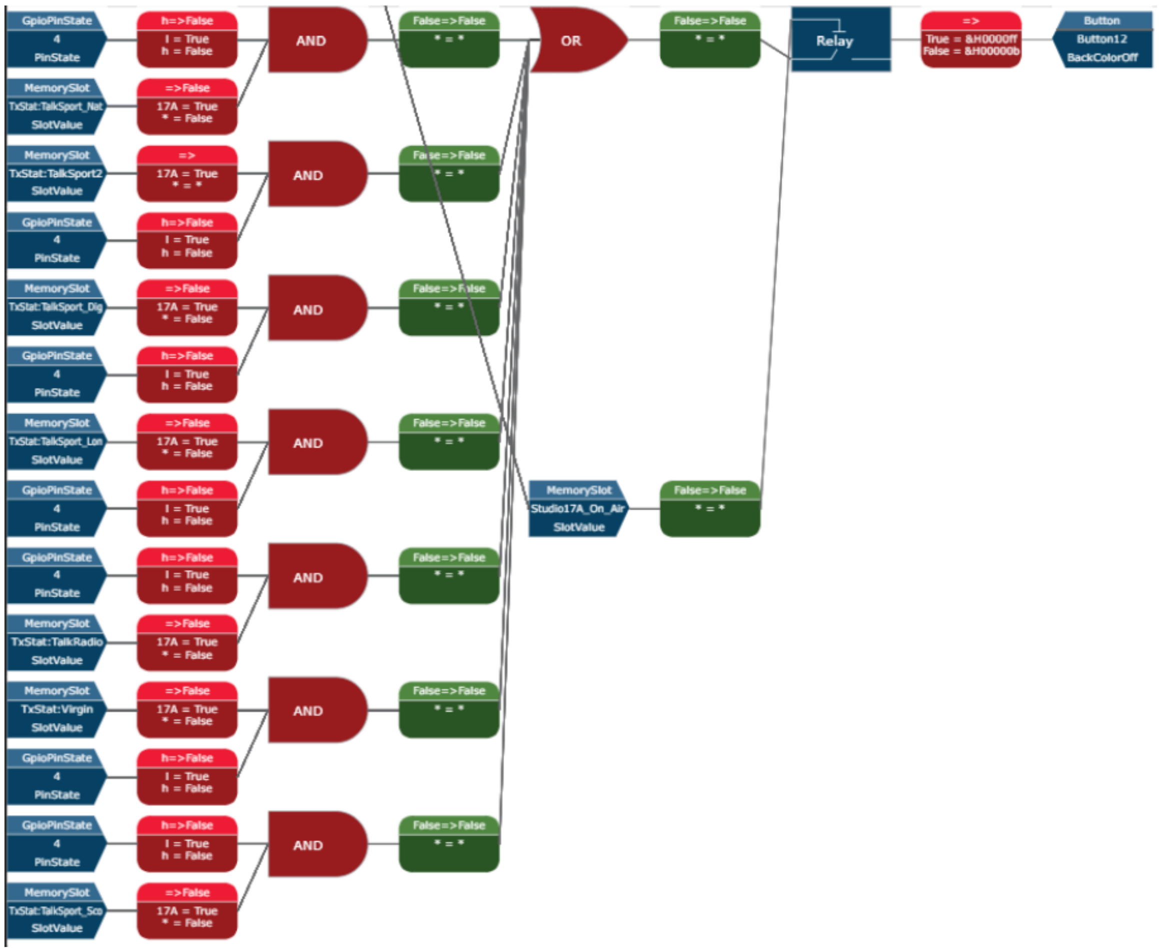
In this flow if both pin 1 and pin 2 are low, then pin 4 will be set low. And if either pin 1 or pin 2 is high, pin 4 will be set high. If the combiner is set to Raise Output On Set, then any time either pin 1 or pin 2 is changed a message will be sent to set pin 4 to the corresponding value. So, if we assume pin 4 is currently high or low and we set pin 1 to low when pin 2 is still high, a message will get sent to pin 4 to go high.



However, if the setting is set to Raise Output on Change, no message would be sent to pin 4. When both pin 1 and 2 were high the resulting output of the And combiner is False. When pin 1 goes low, the resulting output of the And combiner is still False. It's output value has not changed and so no message is sent.

The difference is a bit subtle, but essentially with Raise Output on Set, a change message will be sent to the endpoint any time the input of the combiner changes even if it evaluates to the same output the combiner was at before. Whereas with Raise Output on Change, a change message will only be sent when the combiner's output changes.

Let's look at another example that might be a better example of when Raise Output On Change could be useful:



In this example Gpio pion state 4 appears at each pair of And combiners on the left hand side of the flow. If all the combiners in this flow are set to Raise Output on Set, then the resulting endpoint may get set 7 times when gpio pin 4 changes. This is because, a change to the pin would get fully passed through the flow for each instance where it appears as a start point. However, if the left And combiners are all set to Raise on Change then the output will only get set for combiners where the pin state change actually causes a different state on the output of the combiner. This can greatly reduce load and improve performance in large and complex systems.

It is also important to note that this property is a new variable that must be stored into the backing storage. For backwards compatibility, Raise on Set will store the same way it historically has making that backwards compatible with older software versions. However, any flows that are changed to use Raise On Change will not be loadable by older software versions that do not support this property.

Finally, it is possible to change all of the combiners to a particular setting using the port 9600 API. Each Logic Flow view has a write only property called ChangeAllCombinerRaiseOutput. Using this property all combiners in a view and that view's sub views can be set to the same value. This is useful for changing many combiners at once but should be used with caution. It is recommended that a backup be taken before using this api command. Example:

```
set LogicFlows#0.LogicFlowFolder#yyy ChangeAllCombinerRaiseOutputSettings=
RaiseOutputOnSet
```

```
set LogicFlows#0.LogicFlowFolder#yyy ChangeAllCombinerRaiseOutputSettings=
RaiseOutputOnChange
```

Version 1.3.13.31 Changes

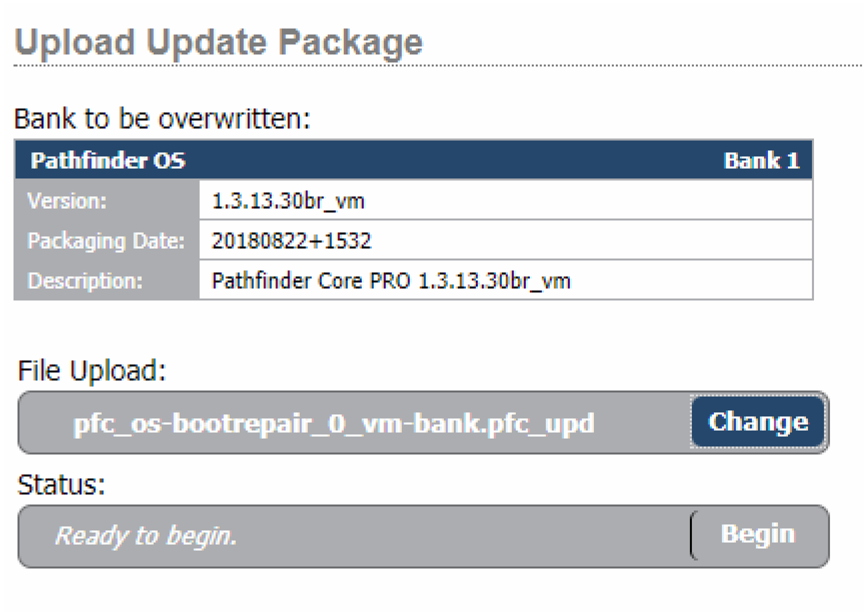
New Operating System

This version is functionally equivalent to 1.3.13.30 but with a new operating system and target framework. Starting with 1.3.13.21 we have had parallel development branches with a new operating system and target framework. This alternate version (referred to as the buildroot version) has been going through internal testing as well as testing with select customers for several months. The difference in this version is that the operating system has been completely reworked to be a purpose-built version of linux rather than being based on a stripped-down version of Debian. This gives us much more control over what exists in the operating system. Additionally, any of the code based on dot net technologies has been reworked and recompiled to target the dotnetcore framework as opposed to the mono framework. We have done this to obtain the following advantages:

- Better control over the operating system and an optimized build process.
- Better control over the codebase.
- Much smaller footprint on the disk (~300MB as opposed to ~600MB). This leaves more space for image files with panels, etc.
- Smaller update packages (~100MB as opposed to ~200MB)
- Significant performance improvements and lower cpu utilization under dotnetcore.
- Improvements in memory consumption and management.
- Subsequent releases will use this new operating system and codebase.

Users of PathfinderCore PRO on the fanless engine platform (as opposed to vm beta testers) will see an initial increase after startup in memory consumption. This is because this build moves from a 32bit OS to a 64bit OS on the Fanless Engine platform. However, we see much better memory management as the system runs.

Additionally, the update package will make certain changes to the boot sector. Upgrading and downgrading of software will continue to work properly in all but one situation. If you upgrade a bank to .31 or later and then switch back to a pre .31 version and try to overwrite the .31 bank with a pre .31 software image, the write will work but booting into that downgraded bank will hang. Power cycling will then boot bank into the previous bank. In this case you can run a special update package that only fixes the boot sector. Simply apply this update package as if you were doing a normal software update. It will not actually change the software in either bank but instead will just run an update script to update the boot loader settings.


Upload Update Package

Bank to be overwritten:

Pathfinder OS	Bank 1
Version:	1.3.13.30br_vm
Packaging Date:	20180822+1532
Description:	Pathfinder Core PRO 1.3.13.30br_vm

File Upload:

pfc_os-bootrepair_0_vm-bank.pfc_upd

Status:

Ready to begin.

Note this only happens when running on a pre .31 bank and trying to downgrade a .31or later bank to a pre .31 software image. Most customers will never experience this. We only expect to see this with systems integrators and support engineers who may need to be switching their software versions regularly to match that of their customers.

Boot Sector Fix Fanless Engine - http://pathfinderpc.com/pfcorepro_downloads/pfc_os-bootrepair_0.pfc_upd

or

Boot Sector Fix VM –

http://pathfinderpc.com/pfcorepro_downloads/pfc_os-bootrepair_0_vm-bank.pfc_upd

Version 1.3.13.32 Changes

Element/Fusion Fader IFB

Each Element and/or Fusion fader has a command for manipulating the Fader's IFB. This requires the fader to be configured in certain ways so review Element/Fusion documentation and/or talk to support to use this feature. There are several properties in PathfinderCore PRO related to this and some nuances around using them properly which are described below. The possible commands that get sent between Pathfinder and the Element/Fusion look like:

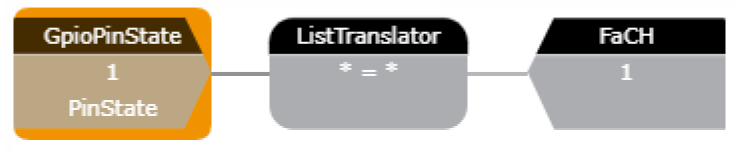
```
=> EVENT FaCH#1 ptt=[DOWN,9505,5000]
<= EVENT FaCH#1 ifb_lwch=9505,ptt=DOWN

=> EVENT FaCH#1 ptt=[UP]
<= EVENT FaCH#1 ifb_lwch=0,ptt=UP
```

Using the LwCH (as opposed to FaCH) to select the fader with a specific livewire channel assigned is also acceptable. The parameters of the command are UP or DOWN for engaging or disengaging the IFB and the livewire channel number to use as well as a timeout value in milliseconds.

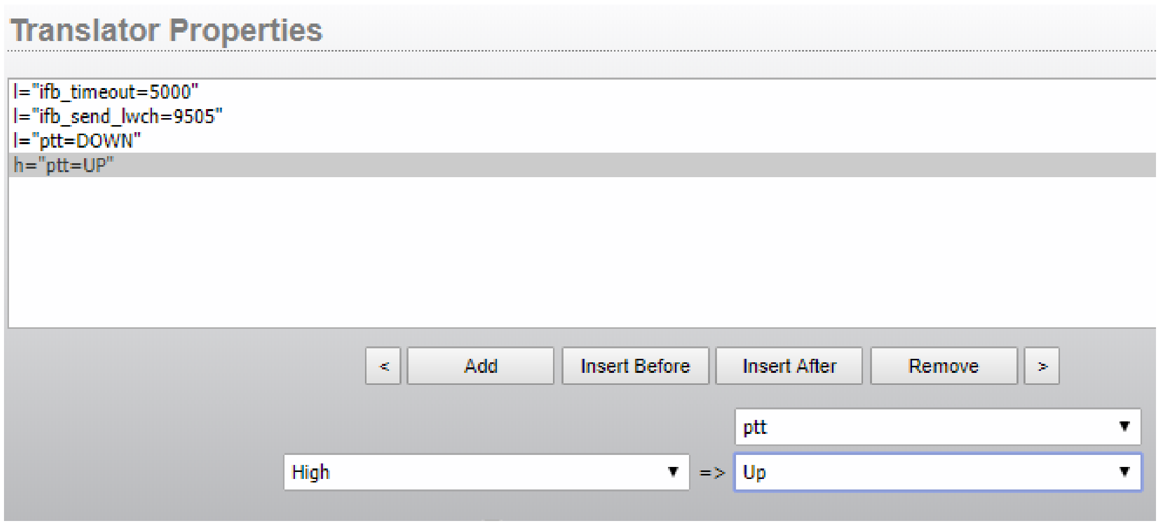
These are represented in PathfinderCore PRO as a set of properties in the FaCH and/or LwCH objects in the logic flows property selection:

- ifb_send_lwch – livewire channel number to send when ptt is down
- ifb_timeout – timeout to send when ptt is set Down
- ifb_lwch – ReadOnly response of the ifb_lwch property when ptt is engaged
- ptt – DOWN/UP

It is important to note some anomalies regarding how these properties work relative to other properties in the system. First, the Element/Fusion only responds to the ptt command on the client that made the request. Other servers connected to the Element/Fusion will not know that an IFB is engaged. Second, there is no way to set in the equipment the ifb channel and timeout to use between ptt actions that gets stored between restarts either in Element/Fusion or PathfinderCore PRO. Therefore, it is important to set all three properties whenever you engage the IFB with a logic flow. This should be accomplished using an Object Translator instead of a property translator. For example, create a flow where the start point is a gpo and the endpoint is an FaCH object. Select the whole FaCH object not an individual property in the object.



Next set the translation list to look like:



```
l="ifb_timeout=5000"
l="ifb_send_lwch=9505"
l="ptt=DOWN"
h="ptt=UP"
```

ptt

High => Up

In this example when the GPO goes low, three properties will be set on this particular Fusion fader, ifb_timeout, ifb_send_lwch, and ptt. The order is important. Always set the ptt after the other two properties in the list. Finally, when the gpo goes high, set ptt up.

This allows the flow to set all necessary properties in one block and do them in an explicit manner that does not rely on storing states.