MACRO & VARIABLE LEVELS

Why do we have different levels of macros and variables in APEX?

Using tags, different macro and variable levels allow you to affect different parts of the interface when making changes or incorporating functionality and feedback.

What are tags and why do we use them?

Bitmaps that serve as buttons as well as elements that provide feedback have identifiers called tags. APEX creates user interfaces when you add a device, and each element is assigned a tag. For example, a cable interface play button will have a play tag in the same way a Blu-Ray player play button will have the same tag. Tags can also be created by the end user. By combining these identifiers with macro and variable levels, a programmer can decide how to affect changes in a system file. He can do so globally, on a source level, on an entire room level, or on a room level but only specify the change to a single controller. This provides efficiency and other advantages that were not possible in Integration Designer 9.

What are the advantages of using tags combined with different levels of macros and variables?

- *Auto-programming macros and variables when adding devices to APEX*
- *Performing corrections and maintenance to a custom interface, particularly when there is a lot of product.*
- *Swapping devices – Apex allows you to swap IR devices with drivers, drivers with IR, RS-232 with drivers, etc.*
- *Room Proxy – Using the room proxy, a programing feature which allows APEX to track the room selected, dynamically change "Room" macros and variables and base if/then logic and functionality based on that selection.*
- *Building efficient templates – If you are using third-party graphics or wish to introduce your own brand of graphics, by tagging buttons and variable placeholders you can insure device interfaces are auto-programmed regardless of how many different devices you sell. Most devices with similar functions share the same tag names – guaranteeing you quick-turnaround by populating all your device IR, RS-232, or driver codes and feedback.*

GLOBAL MACROS & VARIABLES

Common example of macro/variable: (Door Lock / Door Lock Status)

What it will Affect: All Rooms, All Sources, All Controllers

Priority: Lowest

Usage: If I create a tag called "ABC" on a button or 2-way element and program a macro or variable on that object on a "global" level, anything I tag "ABC" on the entire interface (any room, any source device, on any controller) will reflect that functionality.

Hint: Great for a single button that arms the security or reports the arm/disarm status. You can tag that button and attach the functionality, and you can use that tag anywhere without programming it again! But be careful with your tag names, as the global level might "reach" the furthest, but it has the lowest priority. If you come across any other buttons or feedback elements that have a higher-level macro or variable that share the same tag, the higher level will reflect.

SOURCE MACROS & VARIABLES

Common example of macro/variable: (Play Command / Now Playing Artist)

What it will affect: Single source device in a single room, on all controllers.

Priority: Second Lowest

Usage: If I create a tag called "ABC" on a button or 2-way element and program a macro or variable on that object on a "source" level, that tag would maintain the functionality attached to it in the room and source it was created on and reflect on all controllers.

Hint: This is the most common level and allow you to make a change on a source device in a specific room without having to do it on every controller! If you want to add a lighting command to the "pause" button on a Blu-Ray or report the track of the disc you are watching, you don't need to do it on every controller!

ROOM MACROS & VARIABLES

Common example of a macro/variable: (AVR Volume / AVR Volume Level)

What it will affect: All sources in a single room, on all controllers.

Usage: If I create a tag called "ABC" on a button or 2-way element and program a macro or variable on that object on a "room" level, that tag would maintain the functionality attached to it in the room you created it on, on every single source in that room, and on every controller.

Hint: When you add a cable box to Apex, it will automatically populate that components volume commands on a "source" level. However, once you add a receiver and specify you want that AVR to control the volume in that entire room, tags like "Volume Up, Volume Down & Mute" will add a room macro that re-directs to that AVR's volume, taking precedence over global and source macros.

CONTROLLER MACROS AND VARIABLES

Common example of a macro/variable: (Standalone command / Virtual Panel Connection Status)

What it will affect: All sources in a single room, on a single controller.

Priority Level: Highest Priority

Usage: Sharing similarities with the "Room" level, If I create a tag called "ABC" on a button or 2-way element and program a macro or variable on that object on a "controller" level, that tag would only maintain the functionality attached to it in the room you created it on, on every single source in that room, but only on the controller you created it on. If there are other controllers in that room, it will not be reflected.

Hint: When you have the need to put a button in standalone mode, that is, you want to bypass the processor and send a command directly from a controller, controller level macros are your only option. From a feedback standpoint, you might place a variable on an iPad which reports its connection status. Considering you would not want that to reflect on the other iPad, which uses a different variable, you would utilize a controller variable.

	SOURCES		***ROOMS***		***CONTROLLERS***	
	Single Source	***All Sources***	***Single Room***	***All Rooms***	***Single Controller***	***All Controllers***
GLOBAL		✓		✓		✓
SOURCE	✓		✓			✓
ROOM		✓	✓			✓
CONTROLLER		✓	✓		✓	

This table shows you how far each macro or variable level "reaches". Note: A tagged button or element may contain different macro and variable levels, but the one with the highest priority will run when the user presses a button, and in displaying feedback.

A global macro or variable has the lowest priority type, while the controller macro will have the highest, superseding the other three types.