X-KeyPad Version 1.5.1

User's Guide

6/13/2023
Stick and Rudder Studios

Table of Contents

Release Notes 8
1.5.1 8
Overview 8
Upgrading from X-KeyPad 1.5 8
Introduction 9
Elgato Stream Deck Devices 9
P.I. Engineering Programmable Keyboards 10
Virtual Devices on Regular or Touch Sensitive Monitors 10
X-Touch Mini MIDI Controller 11
Other KeyBoards 12
Licensing 12
Copyright 12
Registration 13
Keyboard Only 13
P.I. Engineering Keyboards and Virtual Devices 13
X-Touch Mini 13
Stream Deck 13
Registered Mode 13
Purchasing from the Stick and Rudder Studios Website 13
Purchasing from the X-Plane.org Store 13
Entering the key 14
Installation 14
Linux 15
X-Touch Mini 16
Platform Support 16
Recommended Hardware 16
Physical Devices 16
Elgato Stream Deck Devices 16
X-Keys Devices 16
X-Touch Mini 17
Virtual Devices 17

Concepts 18
Configuration Editors and Files 18
X-Plane Commands and Datarefs 19
Commands 19
Datarefs 20
Logic Test 21
Expressions and Formulas 21
Expressive Commands 22
Logic and Numeric Datarefs 22
Finders 22
Command Finder 23
Dataref Finder 24
Number Buffer 26
Colors 27
Key Labels 27
Units 28
Shift State 29
Shared Datarefs 30
FlyWithLua 30
Configuring X-Keys Devices 30
Overview 30
Key Numbering and Layout 31
File Operations 32
Append 33
Open 33
New 33
Close 33
Save as Default Configuration 33
Active Configuration 34
Includes / Templates 34
Samples 36

Tutorial Videos 36
Configuring Virtual Devices 36
Copy and Paste 39
Defining Basic Keys 39
Conditional Key Color 41
Back Ground Images 43
Color Selection Using Red / Blue Leds 44
Conditional Commands 45
Multiple Commands and Speech 46
Using Numeric Datarefs 48
Complex Key Labels 52
Using Formulas and Expressions 62
Virtual Device Layouts 67
Configuring P.I. Engineer Keyboards 69
Unit Numbers 70
X-Touch Mini 70
Overview 70
Samples 71
Configuration 72
Commands and Datarefs 72
Sample X-Touch Mini Configuration 73
Using the Configuration UI 75
Using the Clipboard 76
Configuring Encoders 76
Dual / Single Encoders 76
Encoder Configurations 76
Panel Brightness Example 79
Multi-Encoder Example 80
Controlling the Encoder Ring Leds 81
Configuring Buttons 82
Beacon Light Example 82

Configuring Sliders 84
Clear Button 84
Using Multiple X-Touch Mini's 84
Saving a Default Configuration 86
X-Touch Mini with FlyWithLua 87
Keyboard Only Support 88
X-KeyPad.csv File Format 88
Loading the Numeric Buffer 89
Command Execution 90
Piggyback Commands 90
Advanced Dataref Manipulation 91
Making Changes to the File 91
Settings 91
Working With Multiple X-Key Devices 93
Auto Save 95
Stream Deck Devices 97
Integration Overview 97
Limitations 98
Comparison to X-Touch Mini and Virtual Devices 98
Installing the Stream Deck X-KeyPad Plugin 99
Stream Deck Sample 99
Append vs Selective Append 100
Binding Keys to Functions 100
Creating a New Key 100
Encoders 100
Encoder Layouts 101
Expressive Commands 102
Sharing Configurations 103
Tips and Other References 104
Commands vs Datarefs 104
Dataref Tool 104

Bitwise AND Operations 104
FlyWithLua 104
XSaitekPanels 104
Hardware vs. Software Mode 104
Background Images 105
Running System Commands 105
Support 107
Appendix 108
Defined Commands 108
Defined Datarefs 109
Font Maps 110
Arrow Font 110
Combined Symbol Font 111
7-Segment Font 111
14-Segment Font 112
X-Touch Mini Encoder Led Ring Values 112
Single Mode 114
Fan Mode 114
Trim Mode 114
Spread Mode 115
Index 116

Change Log

Version	Changes
1.5.1	• Added support for the Elgato Stream Deck Devices • Added ability to shift units individually using command and/or a dataref array as outlined in the appendix • Added optional SPD-SAY command line arguments on Linux • Moved some settings to the global settings file • Added line bar graph support • Added command speech for the X-Touch Mini led enabled buttons • Added Expressive Commands • Added a case sensitive sort on the image names when using the image picker • Fixed a bug with the X-Touch Mini C172 sample heading and ADF encoder where it was getting stuck around 0 • Added support for flashing images on both Stream Deck devices and Virtual Devices • Added the dataref: SRS/X-KeyPad/LedBlinkState • Added copy and paste for expressions • Added round(val) to the expression / formula math functions • Added support for nested image folders • Made the image preview on a selected image honor the aspect ratio of the image • Fixed a bug where configurations were getting reloaded using the “Reload Configuration Files” menu item even if you said No in the confirmation dialog • Added Issue #11, Support for selectable key background images • Fixed a bug where you could not select the Roboto Bold 12 font • Added Issue #2, Support an optional setting on the VD layout to use large unit name buttons • Added Issue #28, Support remembering the maximized state of a Virtual Device on Windows OS • Added an option to disable includes when preparing a configuration for sharing on X-Plane.org • Added Issue #35, auto save • Added Issue #20, Confirmation on revert • Added Issue #4, Inheritance (Template Includes) • Added Issue 17, support for multiple commands on a key • Added Issue 16, support for more than two encoders on a Dual encoder • Added encoder reset timeout for a dual encoder • Updated X-Touch Mini C172 sample to use new multi-encoder feature and reset time out on the Layer B transponder encoder • Fixed issue #29 - Hidden Blank commands can give an error • Added issue #25 – Support Numeric and Command speech natively on Linux using spd-say • Added issue 21 - Toggle visibility commands for both the X-Keys and X-Touch configuration editors

	• Added issue #15 – Ability to execute OS level commands • Added 7-segment and 14-segment fonts • Changed the operation of the X-Touch Mini encoder long press to send a BeginCommand and then and EndCommand after 70MS to simulate a true button click. Some aircraft were not responding to the single CommandOnce function which was previously used to for Long Press commands • Implemented a better way to set and clear Mackie mode so it is not reliant on the Global Channel being set to 1. • Enhanced the substring format to deal with variable length string arrays • Added support for Device ID's when using more than one X-Touch Mini
1.5.0	• Added support for X-Touch Mini • Added GUI for doing all configurations • Added support for formulas and expressions to reduce the need for FlyWithLua scripts • Multiple enhancements for creating dynamic content on Virtual Device key labels

Table 1

Release Notes

1.5.1

Overview

The major change in 1.5.1 is support for the Elgato Stream Deck devices. See the Stream Deck section of this user guide for more detail on how to use X-KeyPad with the Stream Deck. You can also watch the tutorial video: https://youtu.be/hiofS9Xg-xI

There are also a fair number of enhancements for both the X-Touch Mini and Virtual Devices.

- Ability to shift virtual device units individually
- Added integration with SPD-SAY for Linux users
- Line Bar Graph support for virtual devices
- Added command speech for X-Touch Mini led enabled buttons
- Added Expressive Commands for use on Virtual Devices and Stream Decks
- Added support for selectable background images
- Added a round(val) function to expressions and formulas
- Added Auto-Save
- Added support for more than one encoder definition on an X-Touch Mini encoder
- Added encoder reset timeout
- Added ability to execute OS level commands
- Added support for device ID's when using more than one X-Touch Mini
- Improved samples to demonstrate some of the above enhancements

Upgrading from X-KeyPad 1.5

1. Make backup of your existing X-Plane\Resources\plugins\X-KeyPad folder. **DO NOT** delete the original X-KeyPad folder as it contains your license key.
2. Unzip the X-KeyPad 1.5.1 archive to a temporary folder
3. Copy the resulting X-KeyPad folder from step 2 and paste it into the X-Plane\Resources\plugins folder allowing the copy utility to replace any files with the same name. Make sure you do not inadvertently nest the X-KeyPad folder from step 2 under the existing plugins\X-KeyPad folder
4. Start X-Plane and verify that the plugins/X-KeyPad/About menu indicates that you are running 1.5.1

If you plan to use X-KeyPad 1.5.1 with a Stream Deck device follow the instructions for installing the X-KeyPad Stream Deck plugin.

Introduction

X-KeyPad is a plugin for X-Plane 11 and above that allows you to use the following devices to control the simulator:

- Elgato Stream Deck Devices including tablets and phones
- P.I. Engineering X-Keys programmable keyboards
- Behringer X-Touch Mini MIDI controller
- Touch enabled monitor
- Monitor using just the mouse
- iPhone, iPad, or android tablet using Duet Display or other software the make the device look like a touch enabled monitor

X-KeyPad also has an innovative feature that allows you enter numeric data on the keys and then write that data to simulator variables when a key is pressed. This gives you the ability to set many values in the simulator that normally would require a rotary encoder:

- Radio Frequencies
- Altimeter
- Directional Gyro
- OBS values
- Auto Pilot values such as heading, altitude, VS, and air speed
- Transponder Code
- Percent flaps, panel light brightness
- Etc.

Elgato Stream Deck Devices

X-KeyPad supports all of the Elgato Stream Deck devices.

P.I. Engineering Programmable Keyboards

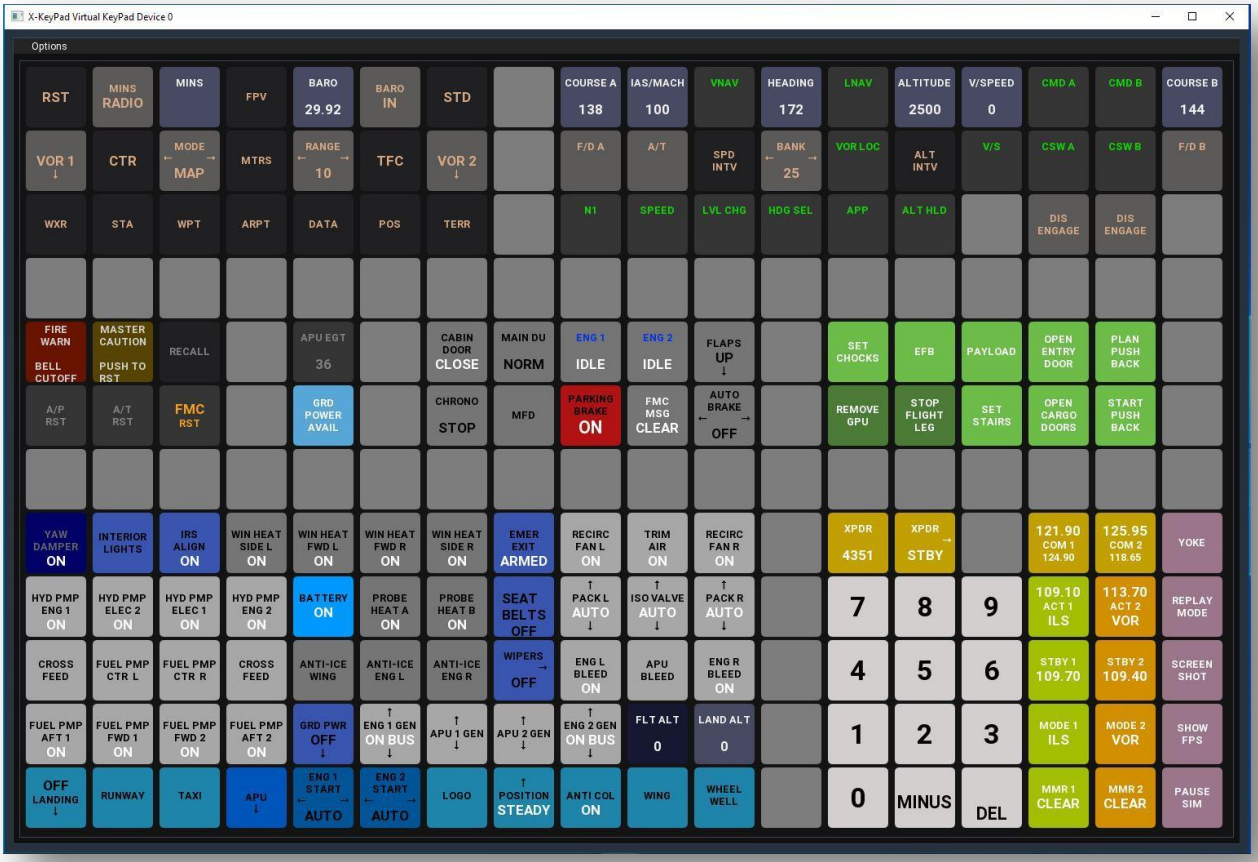
X-KeyPad can be used with PI Engineering X-Key devices to control the red and blue backlighting of individual keys based on dataref values or by lua scripts.

Virtual Devices on Regular or Touch Sensitive Monitors

X-KeyPad also supports Virtual Devices which are pop-out windows that can be created and dragged to other monitors on your simulator computer. These Virtual Device windows emulate a programmable keyboard that can be activated with a mouse click or typically touch gestures when the window is run on a touch enabled monitor or on an iPad or Android device using a monitor emulator like Duet Display.

X-KeyPad gives you a high degree of control over information being displayed on the keys on these virtual devices:

- Key background colors
- Key font colors
- Label text
- Etc..



X-Touch Mini MIDI Controller

X-KeyPad also supports the X-Touch Mini MIDI controller. These devices have eight rotary encoders that have a center push button as well as 16 additional push buttons with LED lighting that can be controlled using X-KeyPad. The device also has a slider that can be used to control a simulator function such as flap / elevator trim position.

The X-Touch Mini has two layers which will multiply the number of inputs giving you 16 encoders, 32 LED enabled push buttons and two sliders.

The encoders can emulate a dual concentric rotary encoder which is very common in many aircraft.

Other KeyBoards

X-KeyPad can work with any keyboard or game controller to set numeric simulator variables. A PI Engineering X-Key device is just one example. Cherry POS keyboards or Genovation ControlPads are just two examples of devices that may prove to be useful with X-KeyPad due to the fact that many of these units have keyboard keys that can have their legends customized.

Programmable backlighting support is only available with PI Engineering X-Key devices , Virtual Device windows, or the X-Touch Mini.

Licensing

Copyright

X-KeyPad is copyrighted software.

- You have the right to use this software for personal, non-commercial use in a flight simulator on a single PC.
- You may not redistribute the software in any way including
  - Posting it on the internet
  - Embedding it in another product
  - Transmitting or sharing the files with a third party in any way
- You may not share or publish your license key if you are running the registered version.
- You may make a backup copy of the software for archival purposes
- You may share any of the configurations you create but make sure you are honoring any copyrights for things like images that may be included in your configuration.

Registration

X-KeyPad operates in an unregistered and registered mode. The unregistered mode is free to use and does not require a license key.

Keyboard Only

With the unregistered version you can define up to 10 numeric datarefs that can be set with numeric buffer data.

P.I. Engineering Keyboards and Virtual Devices

You can program the backlighting of keys 0 – 31 of a PI Engineering X-Keys device or Virtual Device. The unregistered version will support up to two units.

X-Touch Mini

The unregistered version will operate the first four encoders and the first 16 buttons on an X-Touch Mini. The registered version supports all encoders, buttons, and sliders.

Stream Deck

The unregistered version will allow you to save and load the 8 key definitions and 2 encoder definitions.

Registered Mode

The registered mode requires the purchase of a license key. Registered mode allows you to define up to 50 numeric datarefs as well as allowing you program keys 0 – 127 on an X-Key device and / or keys 0 – 255 on a Virtual Device. You can also define up to eight units.

All the encoders, buttons, and slider on the X-Touch Mini can be used with the registered version. Up to eight units are supported with the registered version.

Stream Deck configurations can have up to 512 key definitions and up to 64 Multi Encoder definitions with the registered version.

Purchasing from the Stick and Rudder Studios Website

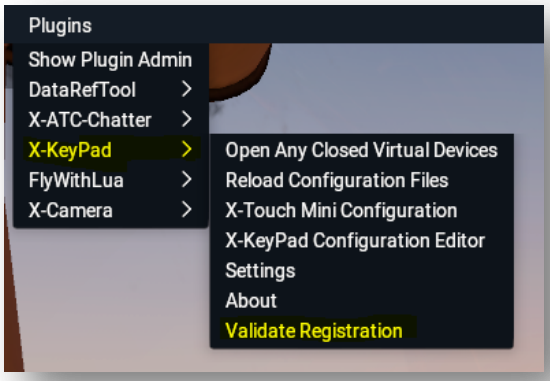
When you purchase your key directly from Stick and Rudder Studios you will get an email containing your key.

Purchasing from the X-Plane.org Store

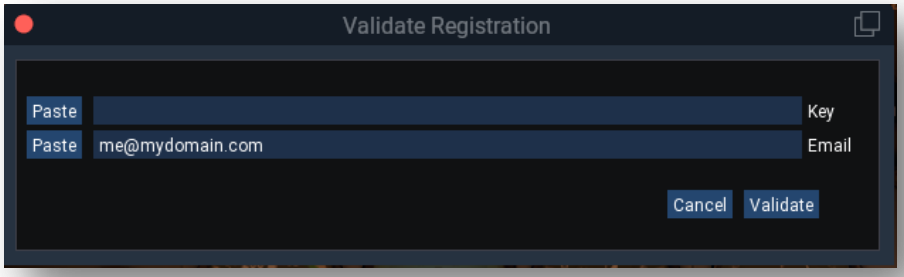
If you purchase X-KeyPad from the X-Plane.org store a key (serial number) will be provided during the checkout process.

Entering the key

You can enter the key using the “Validate Registration” menu item. Go to the X-Plane plugins/X-KeyPad/Validate Registration menu.



Copy your key to the clipboard using the appropriate method for your OS. Then paste the key into the validation dialog and enter the email you used when you purchased the product.



Once this is entered it will be stored in the license.txt file so the key can be validated when X-Plane is started in the future.

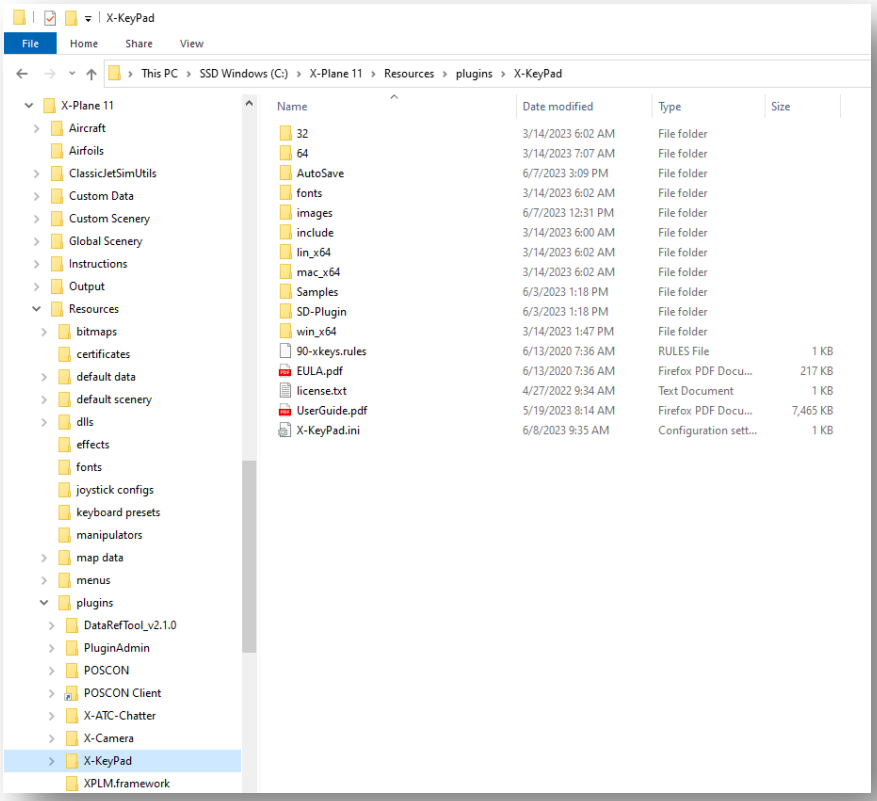
Installation

X-KeyPad is distributed as a ZIP archive. Unzip the X-KeyPad archive to a temporary directory and then copy the resulting X-KeyPad folder to your X-Plane plugins folder:

X-Plane 11\Resources\plugins

Note: Do not change the name of the X-KeyPad folder. X-Plane is very fussy in that it expects the XPL file name to match the plugin folder name. In addition X-KeyPad attempts to load supporting files from that folder name so if you change it X-KeyPad will not work properly.

The folder structure should look like this when you are done:



Linux

If you are running on Linux and using a physical PI Engineering X-Keys device, you will need to copy 90-xkeys.rules to /lib/udev/rules.d. To do this, open a shell and cd to the X-KeyPad folder:

1. cd X-Plane 11/Resources/plugins/X-KeyPad
2. sudo cp 90-xkeys.rules /lib/udev/rules.d
3. reboot

You will also need to make sure you have freeglut3 installed for both physical as well as virtual devices. If you do not have it installed you can do that with the command:

- sudo apt install freeglut3

Portions of the X-KeyPad UI support clipboard operations. In order for this to work on Linux you must install Xclip.

- sudo apt update
- sudo apt install xclip

X-Touch Mini

If you plan to use one or more X-Touch Minis with X-Pad you need to make sure that the Global CH on the device is set to CH 1.

You can verify / edit the device using the X-Touch editor from Behringer:

https://www.behringer.com/product.html?modelCode=P0B3M

Platform Support

X-KeyPad will run on X-Plane 11 and 12 on all three OS platforms.

Recommended Hardware

Physical Devices

Elgato Stream Deck Devices

https://www.elgato.com/us/en/p/stream-deck-plus-black

X-Keys Devices

PI Engineering offers a number of programmable keyboards where the key legends can be customized. These devices are very well made and have great tactile feedback.

https://xkeys.com/utilization/xkeypad.html

The downside to these devices is that the key legends cannot change with different aircraft you may be flying. They also tend to have a higher cost per key than a virtual device.

X-Touch Mini

This hardware MIDI controller sports eight rotary encoders each with a center push button as well as 16 led enabled push buttons and a slider controller.

https://www.behringer.com/product.html?modelCode=P0B3M

Virtual Devices

Virtual Devices are simply pop-out windows created by X-Keypad that have a grid of keys that you can operate with the mouse or a touch enabled monitor or emulator. When used with a touch monitor they don't have same tactile feedback you would get with a physical keyboard but they do offer a lot of additional features like more keys per device and configuration control over key and legend colors and text. This allows virtual devices to easily be configured for different aircraft.

Although virtual devices work fine with just a spare monitor and a mouse they offer a better cockpit experience when used with a touch monitor. There are many touch monitors available for purchase. I used the Planar Helium PCT2235 for all of the X-KeyPad development and was very impressed with its performance.

https://www.planar.com/products/desktop-touch-screen-monitors/20-inch/pct2235/

There are also touch screen emulators available will turn your iPad or Android tablet into a touch sensitive monitor. Duet Display is the one we tested with X-KeyPad:

https://www.duetdisplay.com/

https://www.spacedesk.net/

It is important to keep in mind that any virtual devices you create that will be used on a tablet cannot have the large number of keys that would be available on a 22 or 27 inch touch monitor. The tablet's smaller screen area would make each key's touch area small if you try to configure too many keys.

Concepts

Configuring X-KeyPad requires an understanding of certain concepts. The concepts apply to both X-Keys type devices as well as the X-Touch Mini and Stream Deck devices.

Configuration Editors and Files

There are three configuration editors, Virtual Devices and the P.I. Engineering keyboards, X-Touch Mini devices. And the Stream Deck devices. Configurations are created for a specific aircraft as in many cases those aircraft have unique commands and datarefs that will be used in the configuration to operate the plane. For this reason configurations are saved to the X-Plane aircraft folder for the currently loaded plane.

Configurations are stored in a JSON format. This is text file with a format uniquely designed to allow for saving complex structured configuration data. Although they can be edited directly using a text editor like NotePad we recommend that you use the UI based configuration editors.

Configuration files have a naming convention similar to the .ACF file that defines the aircraft. This naming convention allows you to have multiple definitions in the same aircraft folder for different variants of the plane. As an example, the default Cessna 172 has two variants in the same aircraft folder, one for the steam gauge version and one for the G1000 glass panel.

- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ Cessna_172SP.acf
- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ Cessna_172SP_G1000.acf

The corresponding X-KeyPad configurations will follow a similar naming convention:

- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ X-Keys_Cessna_172SP.json
- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ X-Keys_Cessna_172SP_G1000.json
- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ X-Touch-0_Cessna_172SP.json
- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ X-Touch-0_Cessna_172SP_G1000.json
- X-Plane 11\Aircraft\Laminar Research\Cessna 172SP\ X-KeyPad-SD_Cessna_172SP.json

X-KeyPad will look for these files when the aircraft is loaded and if they are found the configuration will be loaded.

X-KeyPad also has the ability to open a default or generic configuration. If your configuration is only supporting standard X-Plane commands and datarefs that are common to many aircraft then you can save a default configuration to the X-KeyPad folder.

- X-Keys.json
- X-Touch-0.json
- X-KeyPad-SD.json

If X-KeyPad does not find an aircraft specific configuration it will then look for the above mentioned files in the X-KeyPad folder and load them if they are present.

The File Operations section of this user guide as well as the X-Touch section will describe how to save a default configuration.

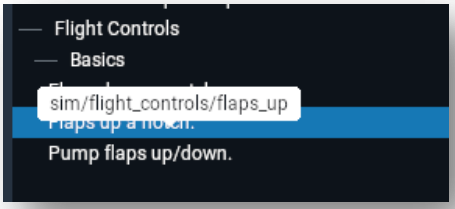
X-Plane Commands and Datarefs

Commands

Commands are actions that can be activated in the simulator. Most things you can do in the 3D cockpit usually have an underlying command behind them. Examples of commands are:

- Turn on a light such as the Taxi, Landing, Strobe lights
- Raise or lower the flaps
- Raise or lower the landing gear
- Tune a radio

When you map joystick buttons or a keyboard key in X-Plane you are essentially binding that action to an X-Plane command. Commands are identified by a unique string. You can see that unique string when you hover over the action in the X-Planning mapping UI.



X-KeyPad key activity, presses, clicks, etc., will use a command to activate the underlying simulator / aircraft operation.

Command Types

When using a command on most X-KeyPad configurations you will see a dropdown that lets you pick the command type: Single, Continuous, Repeat, Expressive, or System.

Single Commands begin and end in one flight loop. They normally are triggered as soon as you press a key associated with them or when the long press time has passed in the case of those keys that are configured for long press mode. Some aircraft and some simulator functions will not respond with a command that begins and ends in one flight loop. If you find that the command is not responsive in Single mode try using Continuous.

Continuous Commands begin when the key is pressed or the long press time period has expired. They end when the key is released. Because they are tied to the physical operation of the key they will start in one flight loop and end in a different flight loop. Some commands like the heading bug up and down command will repeat automatically as long as the key is pressed.

Repeat Commands are a way to emulate the command repeating while the key is pressed when the command being used does not normally support the function.

Expressive Commands allow you to use and expression to calculate a value for a writable dataref. They are particularly useful in the situations where the aircraft designer did not define a command to operate a 3D cockpit function but they did define a writable dataref that can be used to operate the function. Increments modes on Numeric Datarefs can also be used to operate functions where a command is not available.

System Commands execute an operating system command when a key is pressed. This can be useful to open up browser sessions or run other flight sim software such as LittleNavMap and Pilot2ATC. Visit the "Running System Commands" section of this user guide for more detail.

Datarefs

Datarefs (Data Reference) are references to simulator data. Examples of simulator data are:

- Flap position
- State of a light
- Fuel quantity
- Altitude
- Airspeed
- Etc.

Datarefs also have a unique identifying string and take the same form as commands. Here are some example datarefs:

Reference	Data
sim/flightmodel/controls/flaprat	Flap deployment ratio
sim/cockpit/autopilot/altitude	Autopilot altitude
sim/cockpit2/fuel/fuel_quantity	Fuel quantity

Data is represented in different formats:

- Integers (whole numbers)
- Float (numbers that can have a decimal point)
- Double (high precision floating point number)
- Arrays of any of the above
- Byte Data (arrays of ASCII characters or just raw 8 bit data)

X-KeyPad uses datarefs to make decisions on what commands to operate, what colors to display on a key, what text should displayed on a key, and to determine the state of an LED on a hardware device.

Datarefs can be read-only or they can be read-write. In the case of a writable dataref X-KeyPad has the ability to manipulate the data in the dataref to control simulator behavior. Examples of this are:

- Setting radio frequencies

- Setting autopilot parameters (altitude hold, course, heading, etc.)
- Fuel quantities
- Transponder mode
- Etc.

Logic Test

A logic test is used to make decisions concerning operation and visual appearance of the supported X-Keys devices:

- Key color
- Font Color
- Font
- Which command to execute
- Which Led to light
- What data to display
- Etc.

The logic test takes the value of a dataref as input and then compares the value using a configurable expression to a constant value:

- > 1
- < 5
- = 0
- Range 4 through 7
- >= 90
- Etc.

The result of a logic test is either TRUE or FLASE. The result is used to make a selection or light a Led.

Logic Tests are often arranged in a set of tests. In this case X-KeyPad evaluates the tests in the order they appear in the set. As soon as a logic test evaluates to TRUE X-KeyPad will ignore all other tests in the set.

Expressions and Formulas

An expression has a configurable set of variables where each variable value is taken from a dataref. The variable set is further combined with a mathematical expression that returns a new result. This allows you to combine multiple datarefs mathematically to produce a result that can be used in a Logic Test or to display new data.

A formula is nothing more than a global expression. You can have up to 100 formulas. The result of the formula expression is available as a dataref so its value can be used by any key or led definition and it can also be used by a Lua script.

Expressive Commands

An expressive command is similar to a formula except that it is evaluated when triggered by a key on an X-Keys device, Virtual Device, or Stream Deck. It can be used to manipulate a writable dataref when a key on those devices is pressed. The Taxi Light definition in the Stream Deck generic configuration uses an expressive command to demonstrate the concept.

Logic and Numeric Datarefs

Key definitions can have a logic dataref, a numeric dataref, or both.

The logic dataref is the default data source for all key logic tests. It can be overridden at each logic test but in most cases one piece of simulator data is all that is needed to fully configure the key's operation and visual state. Logic datarefs can be either read only or writable although X-KeyPad never modifies the value of a logic dataref. A Logic Dataref can also be defined as an expression in cases where you need to make decisions based on a number of datarefs.

Numeric Datarefs are datarefs that can be written to and modified by X-KeyPad. The simplest examples of this are the radio standby and active frequencies and the transponder code. In both these examples X-KeyPad is able to write data to the numeric dataref on a key press. Typically this data comes from the Number Buffer discussed below.

Numeric Datarefs can also by manipulate with key activity:

- Increment on a keypress
- Adjusted when the key is dragged with the mouse
- Set to a specified value when a key is pressed
- Etc.

Finders

One of the greatest challenges in creating an X-KeyPad configuration is determining what commands and datarefs represent which aircraft functions. The default commands and datarefs used by X-Plane can be found in two text files:

- X-Plane 11\Resources\plugins\Commands.txt
- X-Plane 11\Resources\plugins\ DataRefs.txt

However, many aircraft define their own custom commands and datarefs for aircraft functions unique to that plane. To complicate matters the aircraft designers are not always good at documenting those commands and datarefs.

The best tool for finding and trying commands and datarefs is the Dataref Tool by Lee Baker. You can download it from:

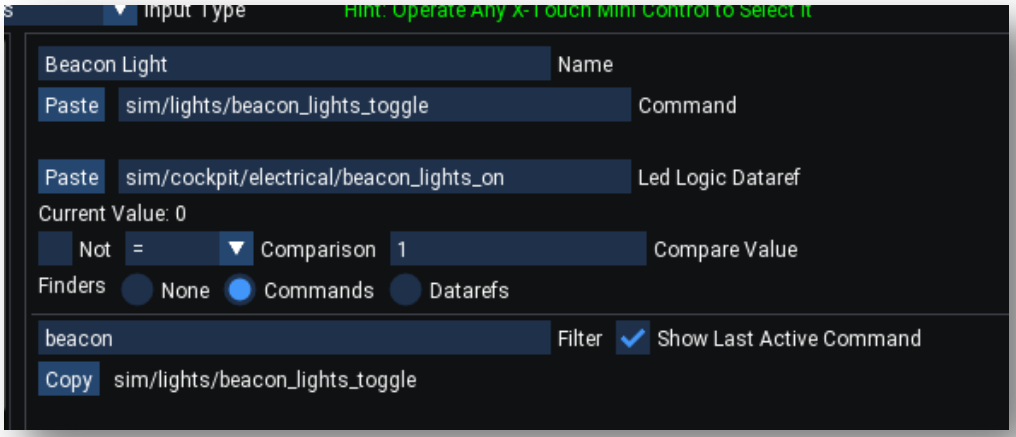
https://datareftool.com/

The tool can be used to search for commands and datarefs by name and it even has the ability to show which ones are changing as you operate corresponding controls in the 3D cockpit. Using the tool helps you to find the command or dataref you need and then copy its name (reference) to the clipboard where it can then be pasted into the appropriate field in the X-KeyPad configuration editor.

X-KeyPad also has a set of finders build right into the configuration UIs that feature integration with the Dataref tool to get the most complete list of items. The X-KeyPad finders do have search capabilities and can detect the last command executed but they are not as powerful as what the dataref tool can provide.

Command Finder

The command finder has two modes. The first mode is to show just the name of the last active command:

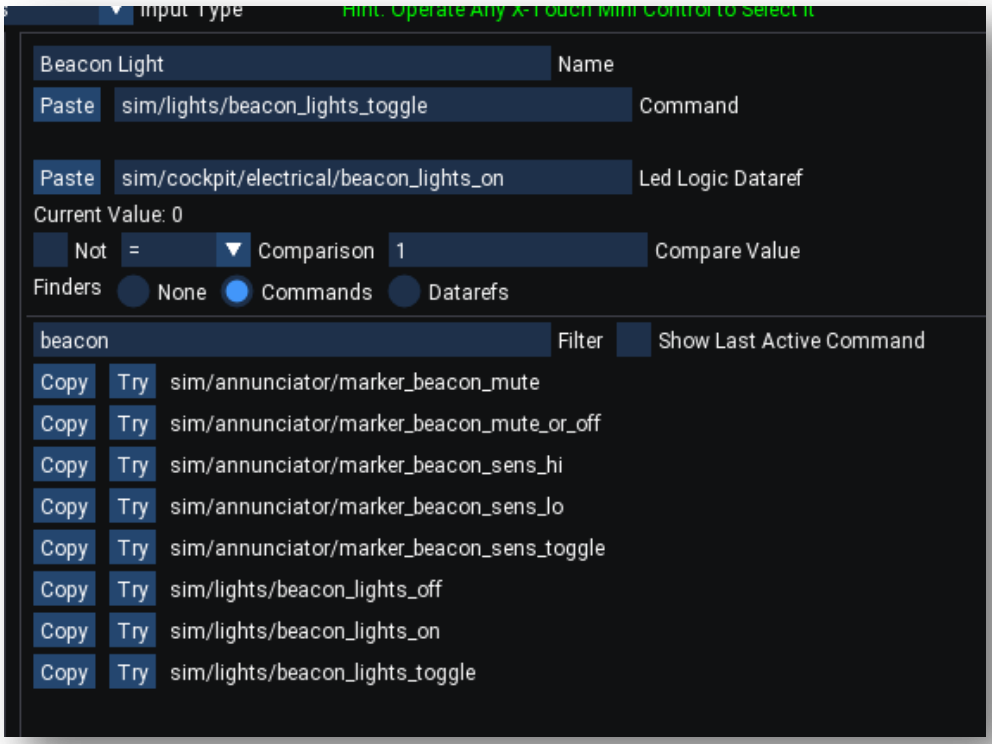


In this mode you can simply activate controls in the 3D cockpit and the command associated with that control will show up. You use the Copy button to copy the command name to the clipboard and then use Paste to paste it to the appropriate field.

In many cases the command you want may not be modeled in the 3D cockpit. A good example of this is the Beacon light switch. The two commands that can be activated with the 3D cockpit beacon switch will be:

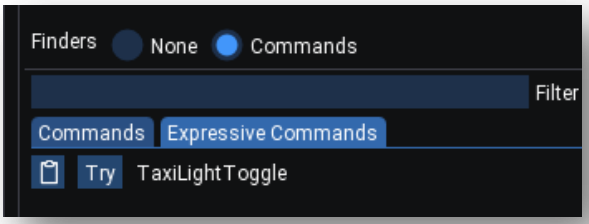
- sim/lights/beacon_lights_off
- sim/lights/beacon_lights_on

However we want to have just one button that will toggle the beacon light on and off. In this case we will want to use a filter to try and find and try the appropriate command.



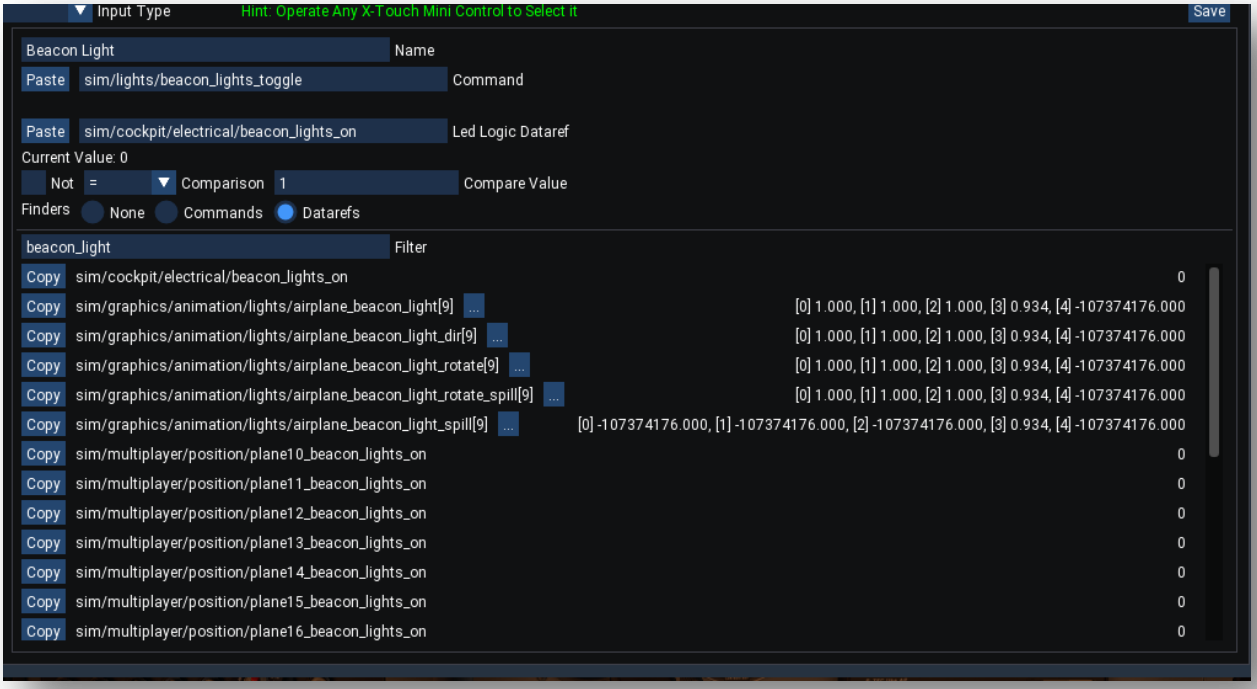
We can enter the word beacon in the filter and all the commands that have the word beacon in them will be displayed. We can then use the Try button to activate a possible command to see if it gives us the behavior we want.

The command finder in the X-Key and Stream Deck configuration editors also has the ability to find predefined Expressive Commands:

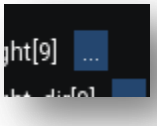


Dataref Finder

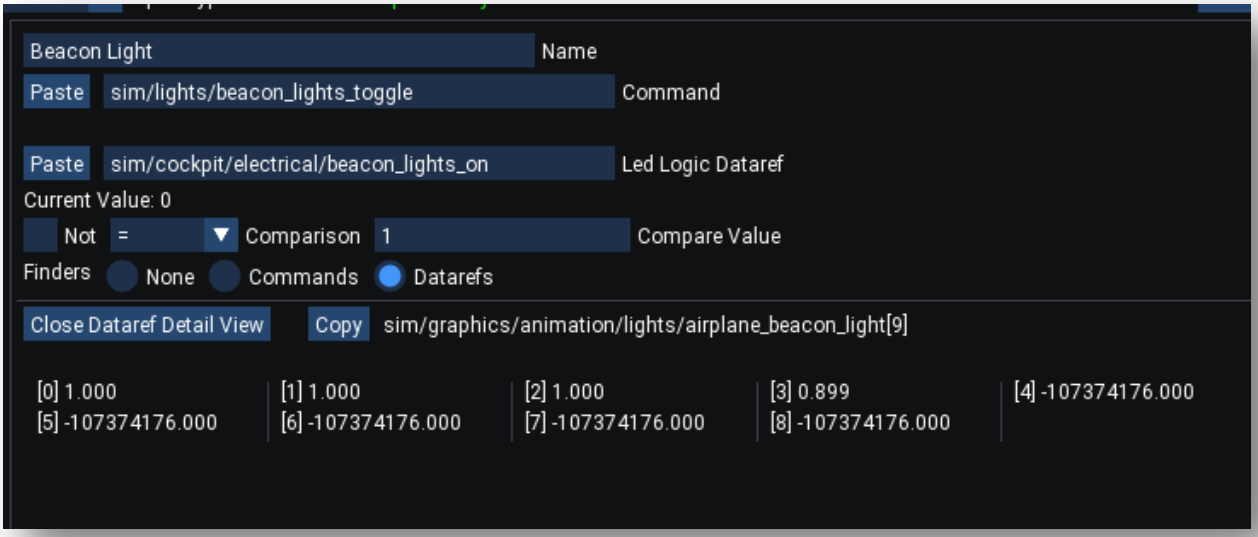
The dataref finder also has a filter option to help you locate the possible datarefs associated with a function. The finder will also who you the current value of the dataref.



In the case of an array dataref the finder will show you the first 5 indexes. If the array is larger than 5 then a detail button will be displayed:



Clicking the detail button will bring up a display of values for just that dataref:



Use the “Close Dataref Detail View” button to return to the filtered dataref finder view.

Number Buffer

X-KeyPad has a concept of a number buffer that can be used to set a numeric dataref to a value represented by the contents of the buffer. The number buffer can be loaded with numerals when certain X-KeyPad commands are executed. These commands are typically mapped to number pad keys on the device but they could also be mapped to numpad keys on a keyboard.

The number buffer is used with a numeric dataref to set data in the simulator. A common example of this is entering the desired radio frequency on the number pad and then pressing the COM1 key. This will cause X-Pad to transfer the data just entered directly into the numeric dataref that represents the active or standby COM1 frequency.

The following table shows the commands and the data that would be appended to the number buffer.

Command	Action
SRS/X-KeyPad/Keypad_0	Append a 0 to the number buffer
SRS/X-KeyPad/Keypad_1	Append a 1 to the number buffer
SRS/X-KeyPad/Keypad_2	Append a 2 to the number buffer
SRS/X-KeyPad/Keypad_3	Append a 3 to the number buffer
SRS/X-KeyPad/Keypad_4	Append a 4 to the number buffer
SRS/X-KeyPad/Keypad_5	Append a 5 to the number buffer
SRS/X-KeyPad/Keypad_6	Append a 6 to the number buffer
SRS/X-KeyPad/Keypad_7	Append a 7 to the number buffer
SRS/X-KeyPad/Keypad_8	Append a 8 to the number buffer
SRS/X-KeyPad/Keypad_9	Append a 9 to the number buffer
SRS/X-KeyPad/Keypad_Decimal	Append a decimal point to the number buffer
SRS/X-KeyPad/Keypad_Minus	Append a minus sign to the number buffer
SRS/X-KeyPad/Keypad_Clear	Clear the current contents of the number buffer
SRS/X-KeyPad/Keypad_BackSpace	Remove the last character in the number buffer

Colors

X-KeyPad virtual devices allow you to choose key background colors and key font colors. It is also possible to define a color set (dual color) that flashes by alternating between the two colors.

The colors for a given key can be dynamically changed by using a set of logic tests with a dataref, expression, or formula.

Physical devices can also have illuminated keys but the color choices are much more limited.

The P.I. Engineering keyboards have a red and blue led under each key. The possible led states are: Off, On, or Flashing. With both the blue and red leds illuminated the key can have a purple glow to it. It is important to remember that Virtual Devices have the ability to emulate a P.I. Engineering keyboard so it is possible to define Virtual Device keys just using the primary red and blue led concept. However, if you do specify a key label color that will take precedence over any primary led configurations.

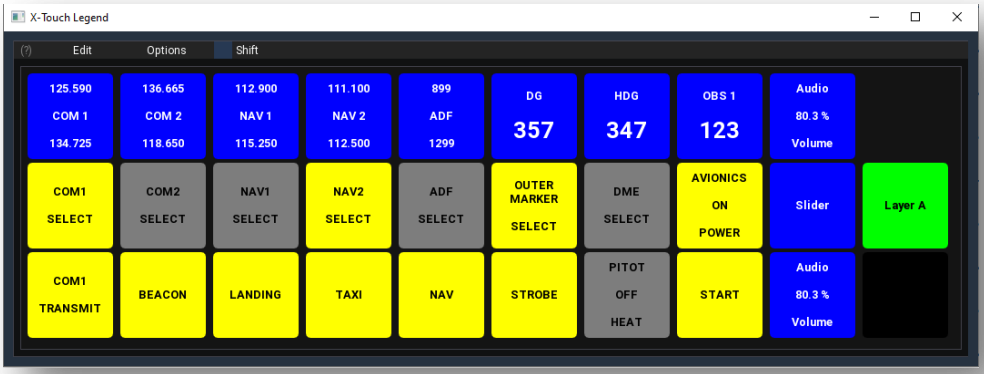
The X-Touch Mini has a white led underneath the push buttons which can be either Off, On, or Flashing. The encoders have red ring leds that can be illuminated in various patterns.

Key Labels

The physical device labels are obviously static. In the case of the P.I. Engineering keyboards the clear key cap can be removed and you can insert the label underneath. One method is to print your labels on clear vellum or transparency using a laser printer and then cut out the label with scissors and place them under the keycaps.

The X-Touch devices would need to be labeled using a label maker or some other method. One approach to creating a dynamic legend for the X-Touch Mini is to use a Virtual Device that can be placed

on a small inexpensive monitor right behind the X-Touch Mini. The X-Touch Mini C172 sample has one of these Virtual Devices defined that you can experiment with:



Virtual Devices have a great deal of flexibility on the label as the label contents can be generated at render time and you can use conditions to choose the label contents you want. You also have the ability to choose label and font colors and also format numeric data from the simulator for inclusion on the label.

Virtual Device keys have a structure to the label. A key can have multiple label definitions where one would be selected based on a condition. Each conditional label can then have multiple lines and each line can have multiple line segments. The key can even have different background images.

Having a structure like this allows lines to use different fonts and even override the label font color. Segments allow you to mix static text with numeric data and even special symbols to build a more complex line.

Units

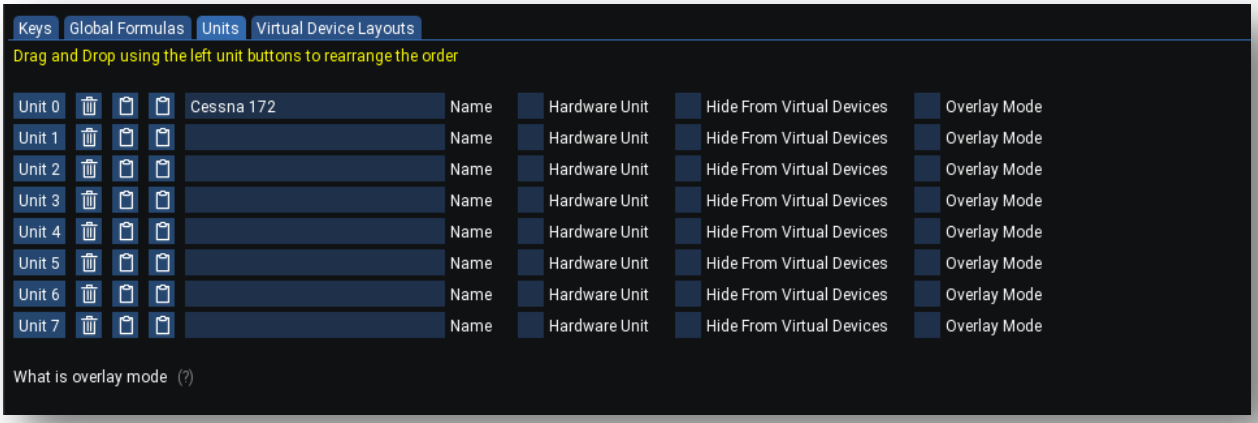
Both X-Key devices and X-Touch Mini devices have a concept of units. X-KeyPad supports up to eight units. In the case of physical devices like the P.E. Engineering keyboards and the X-Touch Mini there is a direct correlation to the unit number and the physical device in the order they were found on the USB controller.

Virtual Devices can also have units and their units can work in harmony with a P.I. Engineering keyboard unit. That means you can have a mix of physical devices and virtual devices.

Units on Virtual Devices are considerably more flexible. As an example, you can define multiple units for a Virtual Device and then have the ability to very easily switch between then on one Virtual Device window. This reduces screen real estate but still allowing for an incredible number of key definitions.

It is also possible to create separate Virtual Device windows where each one is associated with a different unit definition.

Units for the X-Keys P.I. Engineering or Virtual Devices can be configured using the Units tab:



Since Hardware units like the P.I. Engineering keyboards don't have the ability to define fonts, font colors, dynamic key labels, and key background colors beyond the primary red and blue leds, the Hardware Unit checkbox can be set to indicate that the unit is a hardware unit. This will hide some of the tabs on the configuration editor since they have no bearing on the behavior of a hardware unit.

You can also indicate that a unit should not be selectable on a Virtual Device. This allows you to use both physical and virtual devices with the same configuration. Keep in mind that if you have a Virtual Device open and it is displaying the same unit number as a physical device then the Virtual Device will take precedence and the hardware device with that same unit number will be temporarily disabled. As soon the VD is closed or a different unit is selected the hardware device will be enabled again.

When Overlay Mode is enabled on a unit the Shift State function will be modified. When overlay mode is not used any undefined keys on the shifted state will show up as blank nonfunctioning keys. When overlay mode is enabled any keys defined in the non-shifted state that have an undefined key in the same position as the shifted state will be disabled when the unit is placed in the shifted state. Essentially any defined keys in the shifted state will replace (Overlay) the keys in the non-shifted state.

Shift State

Both P.I. Engineering keyboards as well as X-Touch Mini devices have the concept of a shift state. In the shifted state you can have a completely different set of key definitions. In the case of the X-Touch Mini the shift state is named layer A and B but the concept is identical. This shift concept gives you twice the number of keys on a device.

Since Virtual Devices have the ability to emulate a P.I. Engineering keyboard configuration they also support two shift states.

Note: there is a special Overlay Mode that can be enabled on the unit. This is discussed in the previous section.

The P.I. Engineering Keyboards and Virtual Devices can either be shifted globally for all units or each unit can be put into a shifted state separately. Units can be shifted globally or individually by using the special X-KeyPad commands or a writable dataref array outlined in the appendix.

Shared Datarefs

X-KeyPad defines arrays of 50 float and 50 integer datarefs that can be used to exchange data between X-KeyPad and FlyWithLua scripts.

- SRS/X-KeyPad/SharedFloat[50]
- SRS/X-KeyPad/SharedInt[50]

They can also be used to control interaction between multiple key definitions. The HG/HPA and Baro keys are an example that uses the SharedFloat[0] dataref to provide the combined key functionality.

X-keyPad also defines 50 custom string datarefs that each can contain up to 200 characters.

There are also 100 formula datarefs:

- SRS/X-KeyPad/FormulaFloat[100]
- SRS/X-KeyPad/FormulaInt[100]

Any of the 100 possible formula's will store their results in these datarefs allowing you to use the value across multiple keys and devices as well as in FlyWithLua scripts.

Note that the datarefs are all set to zero and the custom strings are emptied any time a new aircraft is loaded.

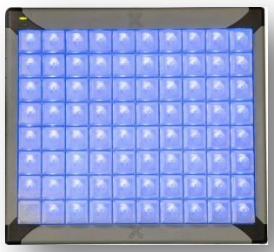
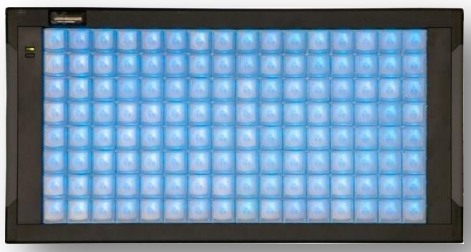
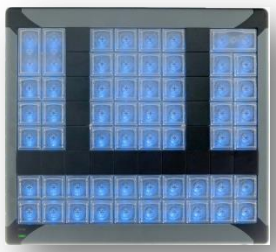
FlyWithLua

In some cases you may need to execute complex logic to create data for display on a key or possibly to execute multiple functions on a key press. Although X-KeyPad gives you a great deal of capability with its built-in functions and formula's / expressions it is not uncommon to find community supplied configurations using Lua scripts to augment X-KeyPad's capabilities.

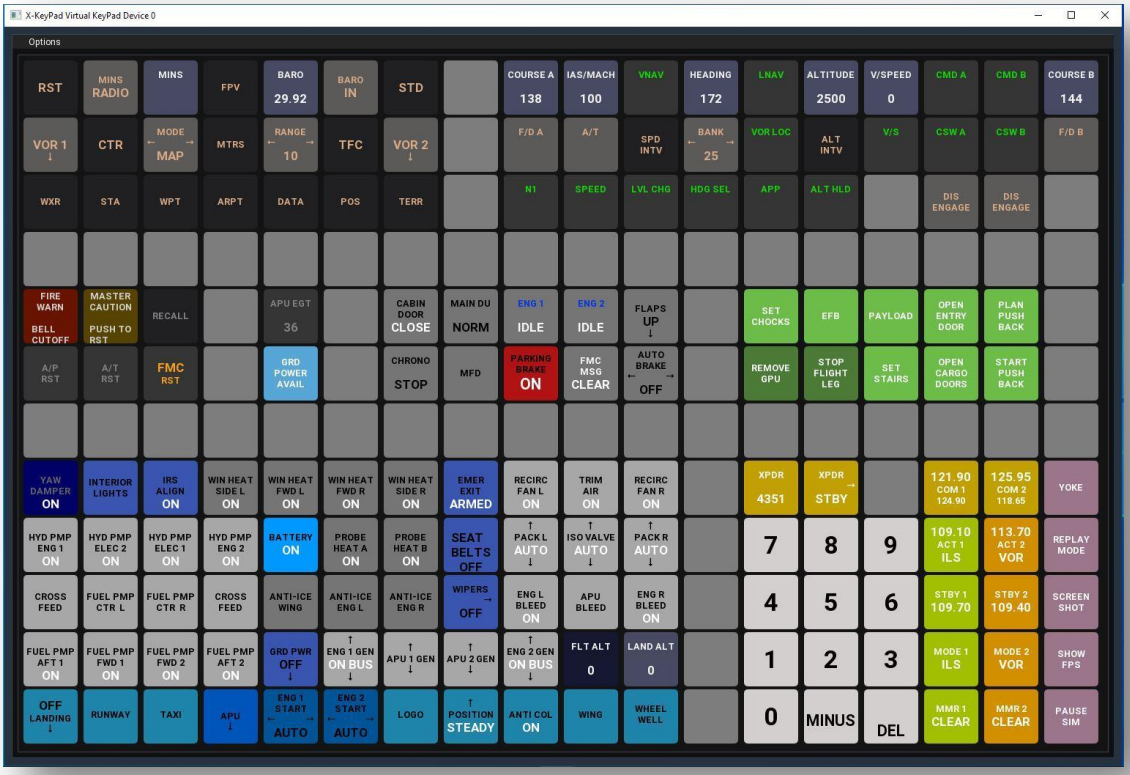
Configuring X-Keys Devices

Overview

X-Key devices were originally just the P.I. Engineering programmable keyboards.

However, X-KeyPad has also supported a Virtual Device since version 1.4. A Virtual Device is a pop-out window that allows you to create keys in a grid layout that can be manipulated with a mouse or more commonly a touch monitor.



The X-KeyPad configuration editor supports creating and modifying configurations for both physical as well as virtual X-Keys devices. You can access the X-KeyPad configuration editor by using the X-KeyPad plugins menu.

Key Numbering and Layout

One concept that is important to understand is the key number and how it maps to a Virtual Device or physical device key. X-KeyPad supports up to 256 keys (0 – 255) on each of two shift states giving you 512 keys per unit. On top of that you can have up to 8 units. Needless to say, that is a lot of keys.

Both the P.I. Engineering keyboards and the Virtual Devices map the key numbers starting in the upper left corner and then incrementing through all the rows in column 1 and then starting at the top of column 2 and going through all the rows.

What this means is that if you initially create a configuration with 8 rows and 10 columns that same configuration is not going to layout the same on a different row and column layout. Generally if you keep the row count the same you can increase the number of columns and things will continue to layout

the same way and you will have a bunch of new blank keys on the right. However, if you change the row count the layout will be very different.

One trick to get around that is to create a new unit and a new Virtual Device that matches what you want. You can then try to copy keys from the old VD and unit and paste them on the new VD and unit. The origin of the copy and paste is always the most left and highest up key. X-KeyPad will attempt to paste the block of keys in the same grid layout. This of course only works when the height and width of the paste are is large enough to paste the group. It definitely takes a little practice.

It is also fairly easy to just the use the drag and drop swap mode to move the keys to their new positions.

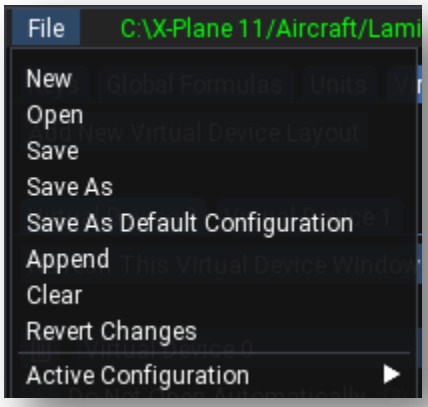
File Operations

As soon as you load an aircraft X-KeyPad create a blank configuration. Since no keys or virtual devices are defined you will not see and Virtual Device windows now will any physical keyboards work. However you can immediately open the X-KeyPad configuration and start configuring keys. If you plan to use the configuration with a Virtual Device it is generally a good practice to first go to the Virtual Device tab and create one with the layout you want.

Once you have keys defined do a SAVE to create a JSON file in the current aircraft folder. From that point on X-KeyPad will read that JSON file and create any Virtual Device window whenever that plane is loaded.

Note: Remember to save your changes often.

The following image shows you the available File Operations from the File Menu:



We will discuss them below in the most likely order you might use them.

Append

When you first start out you might like to import (Append) and exiting configuration. This can be either from an existing configuration you created, one of the X-KeyPad samples, or possibly one of the X-KeyPad community configurations:

https://forums.x-plane.org/index.php?/files/category/187-x-keypad-configurations/

You can use File-Append to locate the JSON or older style CSV file to append the configuration to the one you are currently working. Keep in mind that if the configuration you are appending has key definitions on units that you already have created definitions for, the append operation will replace them. If you are starting with a completely blank configuration that should not be a concern. If you want to just snag some other key definitions then a File-Open operation may be a better option.

Remember to do a SAVE after you have appended and retained any key definitions you want. If you don't like it you can do a REVERT.

Open

The File-Open operation is useful when you want to look at another configuration without affecting the one you are currently working on. Once open to can try the configuration and then either Save it as the aircraft configuration, or more commonly, you might copy the keys you want and then switch back to your primary configuration and paste them where you want them

New

This creates a new empty configuration. You can use this as a scratch pad to experiment with different keys and then copy them to your main configuration. You can also to a Save As to put it in a separate folder where you can manually open later. Lastly, you can save it as your current aircraft configuration.

Close

If the active configuration is one that you Opened or had created with New menu item then you can use the Close item to close it and select the aircraft configuration as the active one.

Save as Default Configuration

X-KeyPad also has the ability to open a default or generic configuration. If your configuration is only supporting standard X-Plane commands and datarefs that are common to many aircraft then you can save a default configuration to the X-KeyPad folder.

- X-Keys.json

If X-KeyPad does not find an aircraft specific configuration it will then look for the above mentioned files in the X-KeyPad folder and load them if they are present.

The Save as Default menu option will save the current configuration to the X-KeyPad folder as X-Keys.json

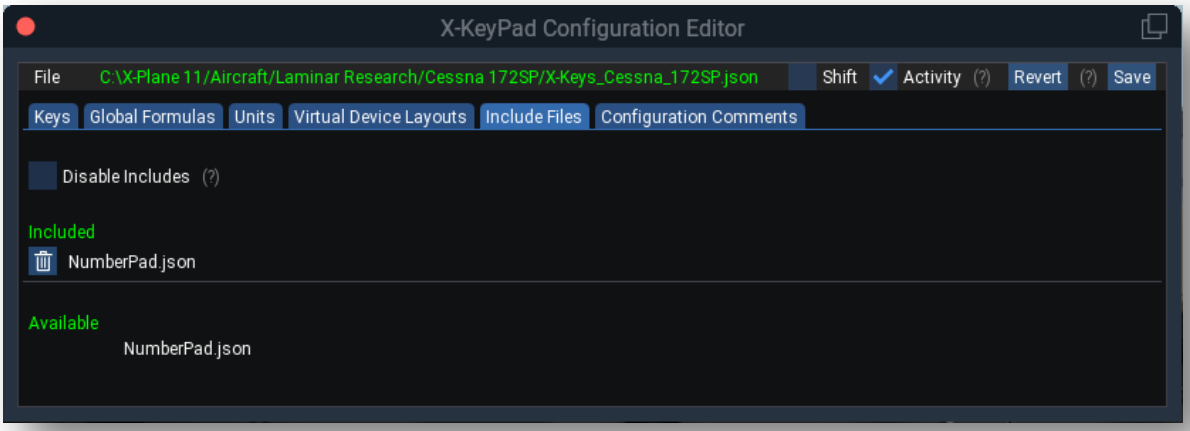
Active Configuration

This will show you a list of all the open configurations. Simply choose the one you want to work on.

Includes / Templates

The X-Keys configuration has the concept of inheritance or templates through the use of include files. An include file is a json configuration that is saved to the X-KeyPad/include folder. The include configuration file can contain common key definitions and positons that are likely to be common across many different aircraft, numeric keypad, radios, lights, transponder, X-Camera / X-Plane views, etc.

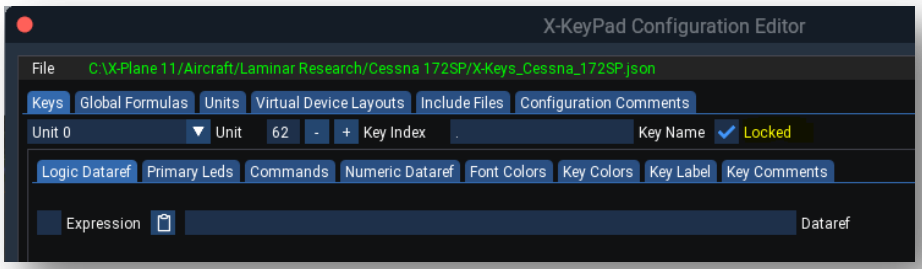
You can create common templates and then do a save-as to save them to the X-KeyPad/include folder. Once they are saved they can be automatically included in any configuration on the includes tab:



As soon as you add an include file X-KeyPad will effectively append that file to your configuration. It will also auto append any includes when the configuration is loaded.

What this allows you to do is maintain any changes to the common functions by editing just the configurations in the X-KeyPad/include folder. Any configuration that references that include will always pickup those changes when the configuration is loaded.

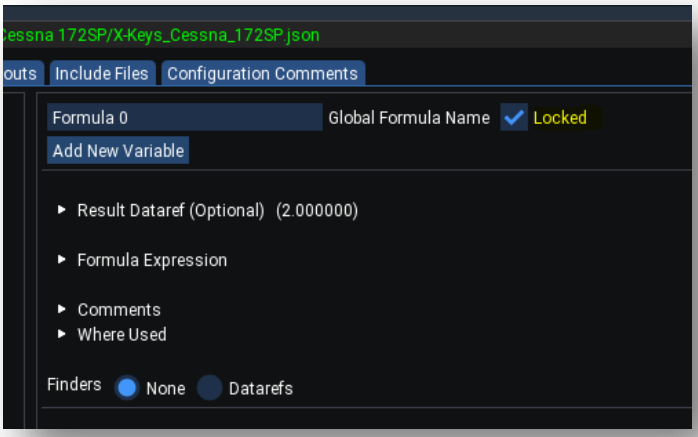
Sometimes you may want to change the way an included key behaves for a specific aircraft. In order to prevent your changes to the key from getting over-written by the included file you need to lock the key.



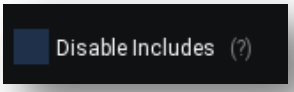
When a key is locked it will not get replaced by the same key in any of the include files. This allows you to over-ride the included key's definition.

If your include file contains global formulas it is important to give the global formula a unique name. This allows X-KeyPad to match formulas in the saved configuration by their name so that it will only replace a formula definition when the names match.

As with locking a key it is also possible to lock a formula so you can over-ride any included formula's definition.



The Disable Includes checkbox is used to stop X-KeyPad from automatically including the listed include files.



This would be used when you want to save a version of the configuration that you can share in the community configurations forum on X-Plane.org:

https://forums.x-plane.org/index.php?/files/category/187-x-keypad-configurations/

After a configuration is opened and all the include files have been appended the configuration in memory actually has the final definition of all keys and formulas. Saving it with the checkbox set allows you to publish the configuration as fully functional in the forum so that when users download it and use it they will not see any errors because of missing include files.

Samples

The configuration concepts discussed in this section of the guide will regularly refer to the C172 Virtual Device sample. We recommend installing this sample so you can follow along. The sample file is located in:

X-Plane 11\Resources\plugins\X-KeyPad\Samples\Virtual Devices\C172\ X-Keys_Cessna_172SP.json

Copy the file above to the folder: X-Plane 11\Aircraft\Laminar Research\Cessna 172SP

Tutorial Videos

Although this User Guide will attempt to cover all the aspects of the configuration UI some concepts are better covered using a video. There is a full set of tutorial videos that also use the C172 samples mentioned above:

https://www.stickandrudderstudios.com/x-keypad-tutorials/

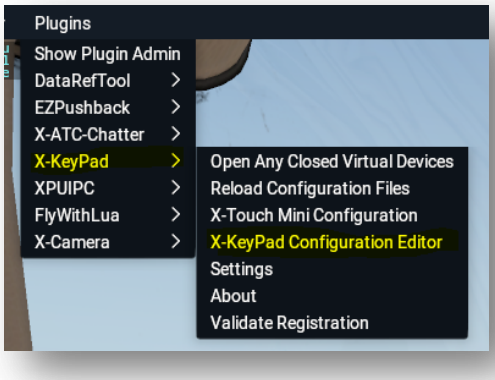
Configuring Virtual Devices

As a reminder a virtual device is a pop-out window that shows your key configuration in a grid. If you had installed the C172 configuration X-KeyPad should create a window that looks like the following when the C172 steam gauge aircraft is loaded:

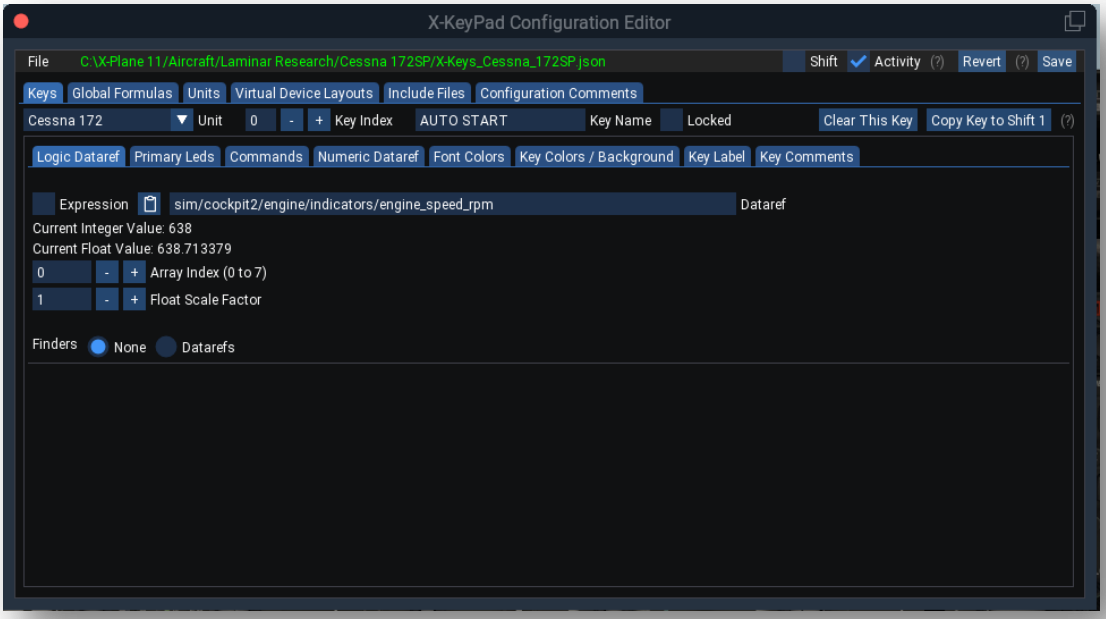
You can resize the window and the keys will adjust their size to fit the window. Note: if you make it too small the font may not fit in the keys. The sample C172 configuration was designed to work with 1920 X 1080 monitor.

You can interact with the Virtual Device using your mouse but it is more common to use it with a touch enabled monitor or a tablet that emulates a touch monitor using software like Duet Display or something similar.

To explore the key definitions for the C172 sample you will want to open the X-Keys configuration editor. Use the X-KeyPad menu to open the editor:

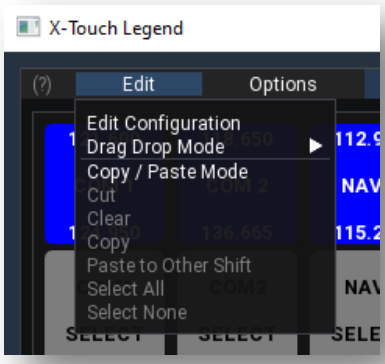


Once opened the editor UI looks like this:



Copy and Paste

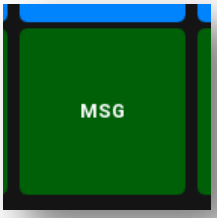
The Virtual Device windows have a unique ability to copy / paste keys from one unit or configuration to another. It also has a Drag/Drop mode that allows you to move keys around using the mouse.



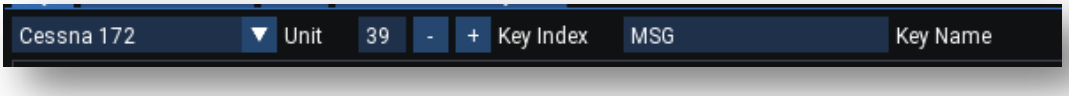
You have to place the VD in one of the two modes and then exit the mode when you are done.

Defining Basic Keys

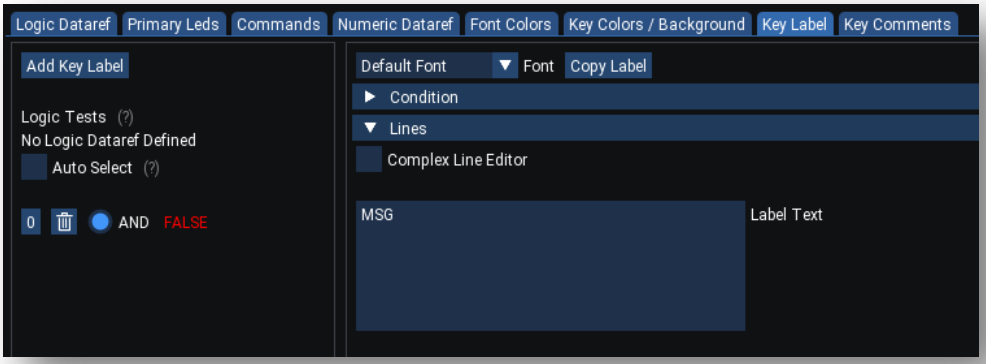
An example of a very basic key is the Garmin 530 MSG key:



This key has just one command associated with it and a static green background color. To get this key selected in the editor simply click on the MSG key in the Virtual Device and it will be automatically selected in the editor:



The key label is defined on the Key Label tab:

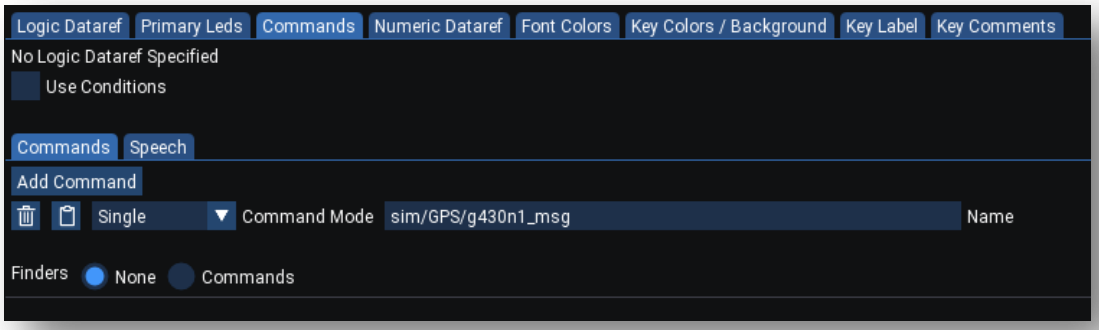


This key does not have a Logic Dataref nor a Logic Test associated with it therefore it does not have any conditions. That is because the key never changes. It is always forest green with a white font and the text is static, MSG.

When the key is pressed it will actuate the Garmin 430 msg command:

sim/GPS/g430n1_msg

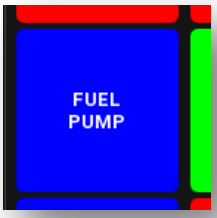
The command definition is on the Commands Tab:



Similar to the label there is no condition. That one command will be executed any time the key is pressed. It is also using the single command mode. That means the command will only be sent once even if you keep the key pressed. The other command modes of Continuous and Repeat will be demonstrated on some other sample keys.

Conditional Key Color

A good example of a fairly simplistic key operation that has a conditional key color is the Fuel Pump key:



This key has two colors, blue when the fuel pump is off and red when the fuel pump is on. Pressing the key will execute the command:

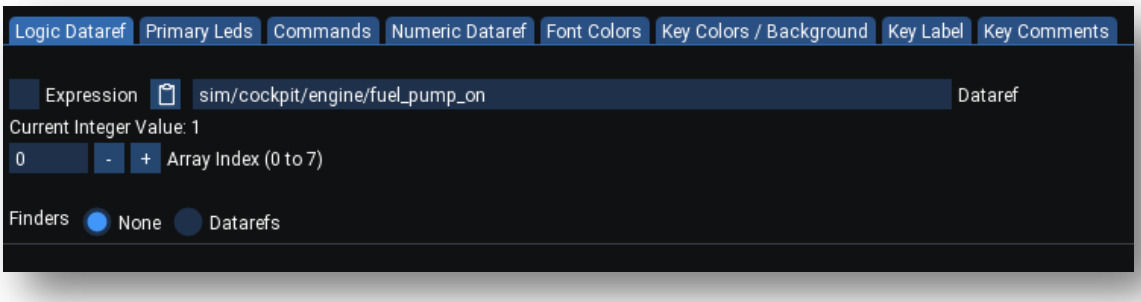
sim/fuel/fuel_pump_1_tog

That command toggles the fuel pump on and off.

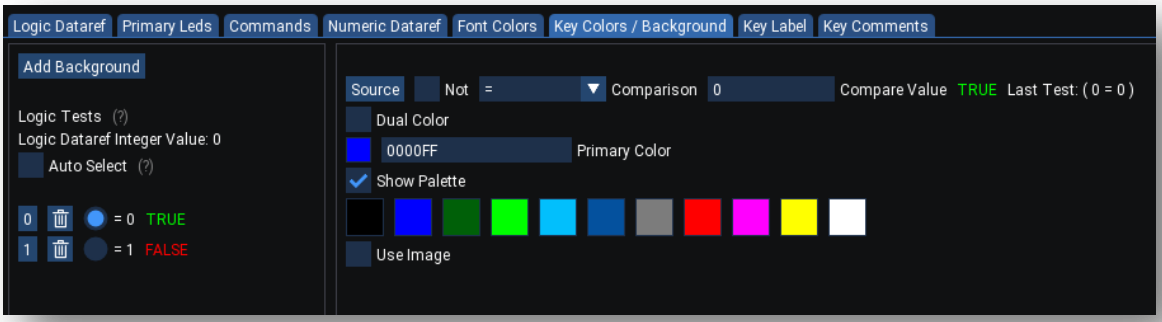
In order to make the color decision on the key we need to use a dataref. In this case we can use the integer dataref that represents the state of the fuel pump switch:

sim/cockpit/engine/fuel_pump_on

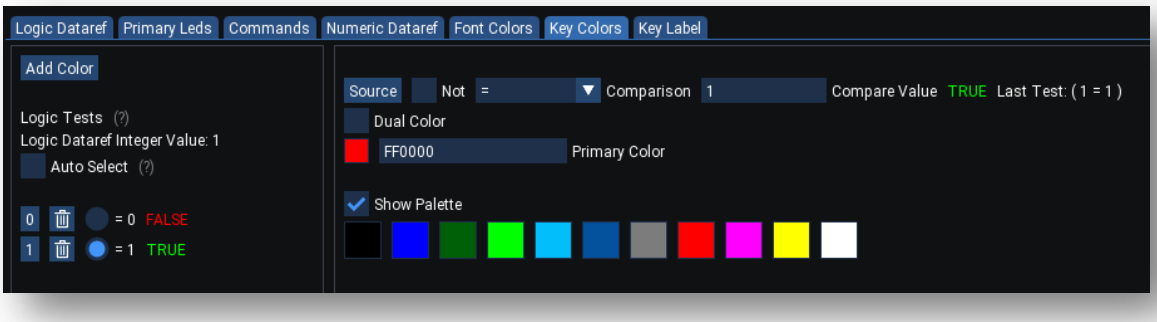
This dataref will have a value of 0 when the pump is off and 1 when the pump is on. We will configure the Logic Dataref to monitor that value:



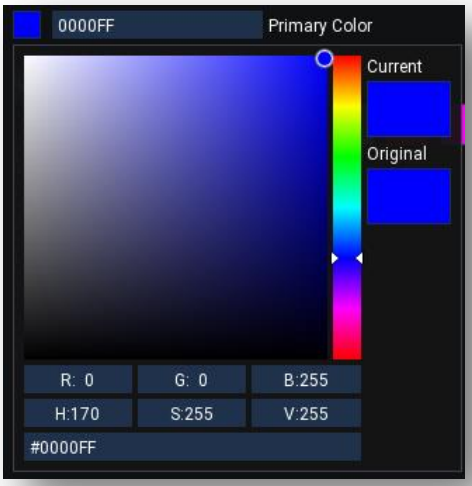
On the Key Colors / Background tab we will define two logic test for the red and blue colors. When the logic dataref is a value of 0 we will select the blue color.



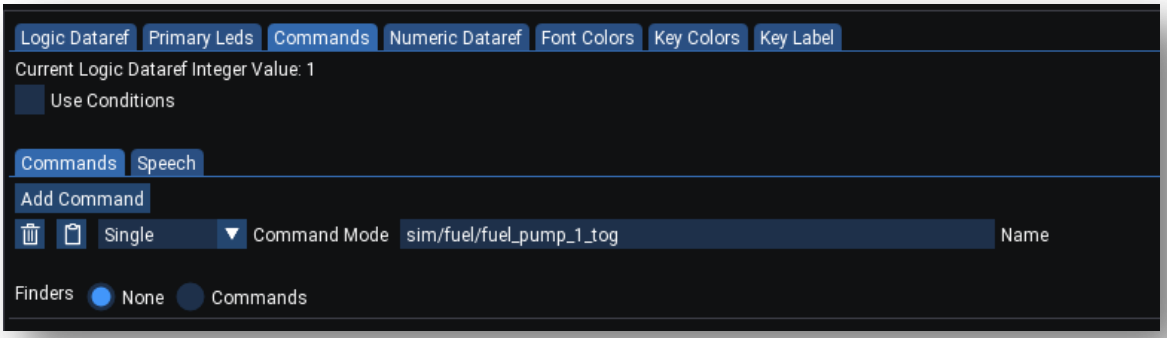
When the logic dataref is 1 we will select the red color.



You can drag and drop colors from the palette to the color selector target. You can also click the color selector target to create a new custom color using the color picker:

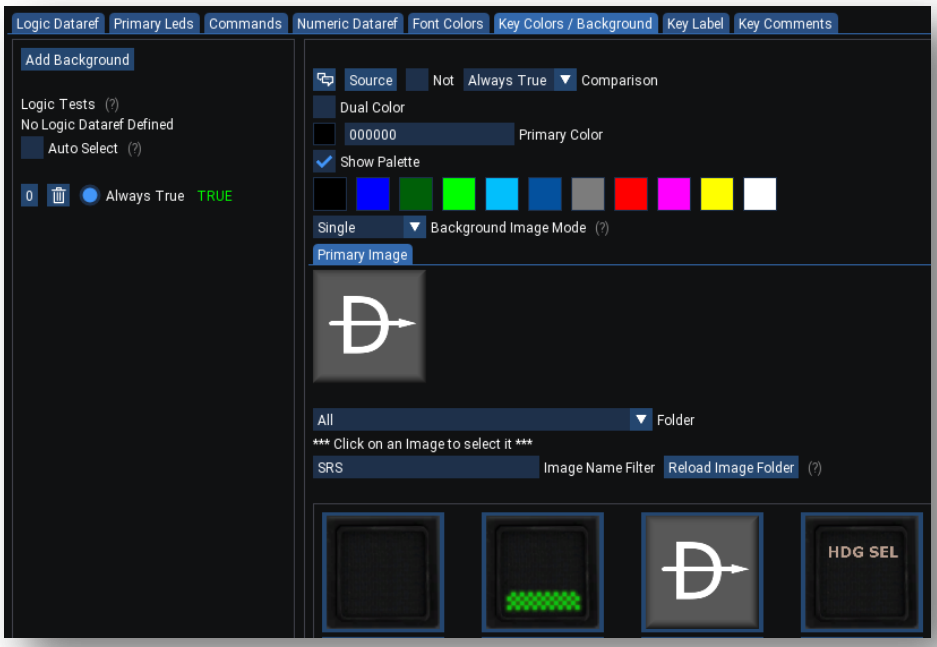


Luckily the fuel pump actuator has a simple toggle command so on each key press we can use an unconditional executing of that command in order to flip the fuel pump switch on or off.



Back Ground Images

In addition to selecting the key color you can also choose to select one or more background image for a key. Look at the definition for the Direct-To key.



On the Key Color / Background tab use the Image Mode dropdown to select either single or dual image mode. If you select dual image mode you will be able to specify both a primary and secondary image. In dual mode X-KeyPad will automatically cycle between the two images at the same rate as a flashing dual color.

A list of images located in the X-Key/images folder will be displayed. Simply click on the image to select it for use as the key background.

You can create your own images and place them in the X-KeyPad images folder. Use the Reload Image Folder button to refresh the last after you have added or removed images. You can use the Image Name Filter entry to limit the list based on a case insensitive sub-string match. As the example shows, the filter text, “SRS”, will only show image files that contain the text SRS in the file name.

Image file must be either PNG or JPG file types. They should be square in shape and we recommend a size of 250 X 250 pixels.

When your configuration is saved the image will be saved with the json file in a base64 format. This lets you share the json configuration on X-Plane.org without the need to package the actual PNG or JPG files.

If you create any of your own images files we suggest naming them a prefix like your initials. This will allow you to potentially share your images with the community and reduces the risk that two people will pick the same file name.

There are also a number of places where you could purchase images specifically designed for flight simulation. Here is an example of one package you could purchase:

https://www.sideshowfx.net/flight-sim-icons

If you do decide to purchase and use an image set make sure you consider any copyright requirements before you share your configuration with the community.

Color Selection Using Red / Blue Leds

You may recall from the concepts section that Virtual Devices can emulate hardware P.I. Engineering keyboards. These keyboards have just two leds under the clear keycap, red and blue. The Pitot Heat key is an example of how you can control the key color on the Primary Leds tab.

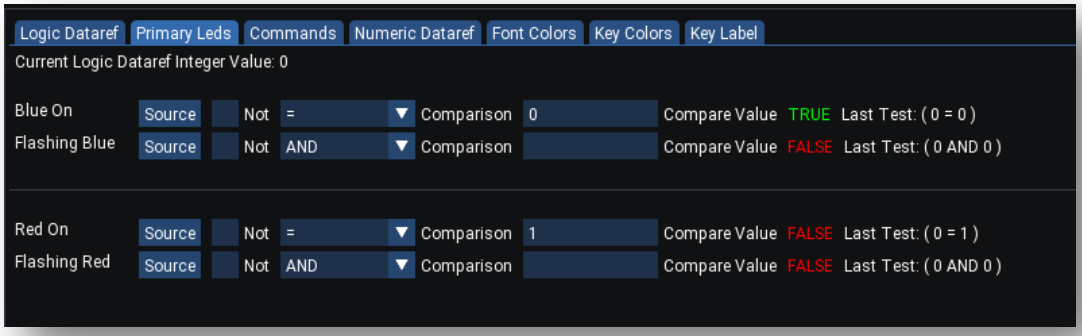
The command structure for the Pitot Heat key if similar to the Fuel Pump key. The difference is the Logic Dataref we will monitor is:

sim/cockpit2/ice/ice_pitot_heat_on_pilot

The command we will execute on a key click is:

sim/ice/pitot_heat0_tog

The Pilot Heat key has no color definition on the Key Colors tab. Rather it has a Primary Led tab definition:



This definition has the blue led on when the dataref = 0 and the red led on when the dataref = 1. The logic tests are group by led. If neither of the two conditions evaluate to true that led will be off. If the On condition evaluates to true then the led will be on and the flash test will not be done. If the On test evaluates to false then the flash test will be done and if true the led will flash on and off.

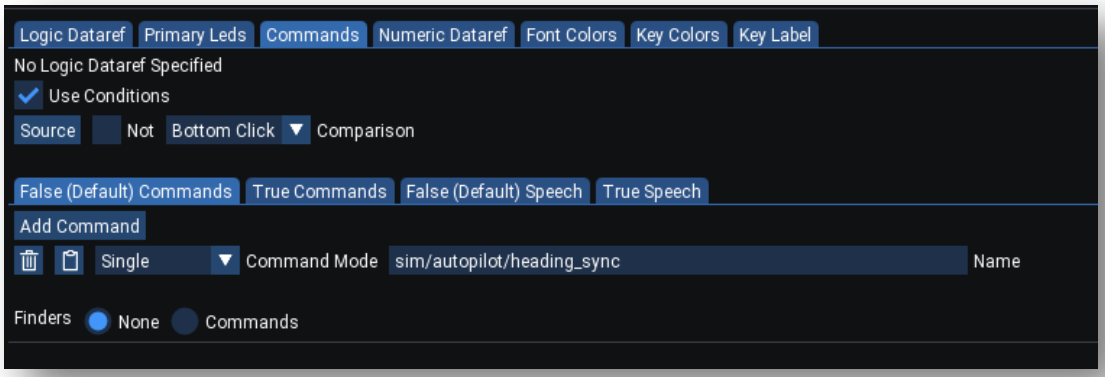
Conditional Commands

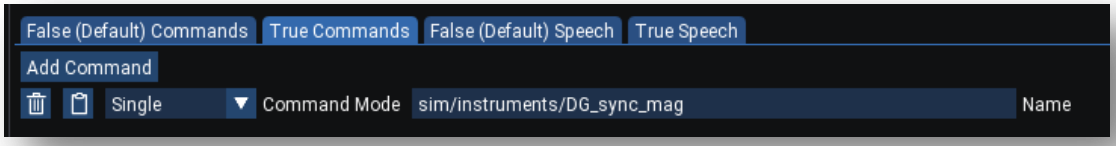
You may need to execute different commands based on the state of dataref or possibly using a click area on a virtual device key. Here are some examples when this is used:

- A simulator function does not have a toggle command available, such as toggling a light on and off but it does have a dataref indicating the current state of the light or function and it does have two separate command for turning the light on and off.
- You want to use one key to operate two separate actions, such as raising or lower flaps, increasing or decreasing the GPS map's range, etc.
- Turning a multi-position knob, like the transponder mode left or right using the same key.

The Heading / DG Sync key is an example of this technique.

On the commands tab we will check the conditions chechbox which will allow us to enter a condition and true and false commands that will be executed on a key click based on the condition:



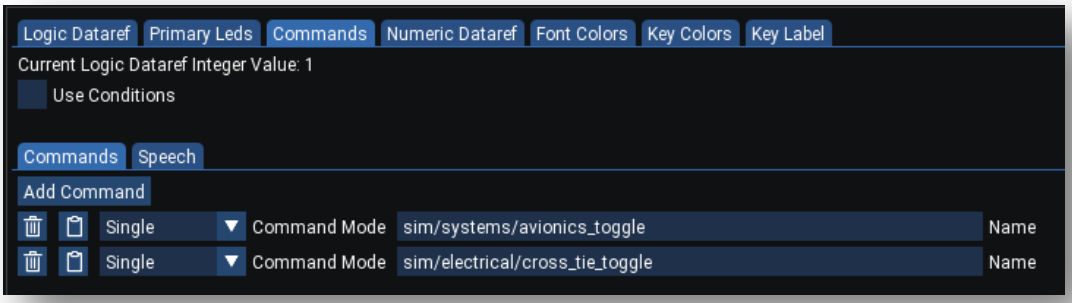


The logic test is using the Bottom Click mode. This means that if you click or touch in the bottom half of the key then the condition will evaluate to true and the true command will be executed. If you click in the top half of the key then the test will evaluate to false since the top half is NOT the bottom half. In this condition the false command will be executed.

Multiple Commands and Speech

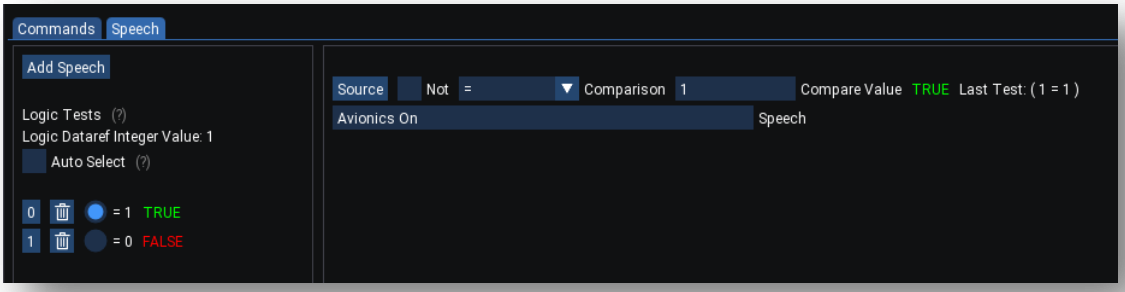
On occasion you may want to operate more than one function with a key click. You also might want to have X-Plane's text to speech function announce the operation. The Avionics Power key is an example of both these options:

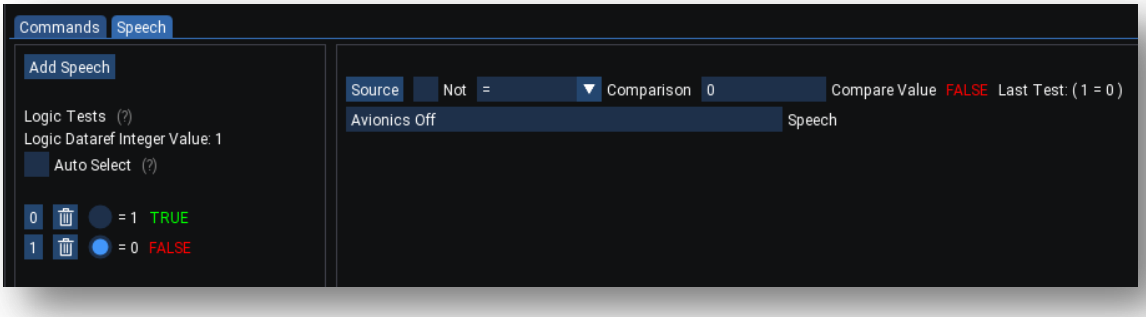
In this example we will monitor the "sim/cockpit2/switches/avionics_power_on" on the Logic Dataref tab. On the commands tab we don't need to use conditions since the two avionics power commands have a toggle command but we will use some conditions on the speech tab as you will see shortly.



Use the Add Command button we added the two toggle commands we want executed when the key is pressed.

On the speech tab we have created two conditions based on the on (1) of off (0) values of the avionics power dataref. When we press the key the appropriate speech string will be sent to the X-Plane's text to speech engine.



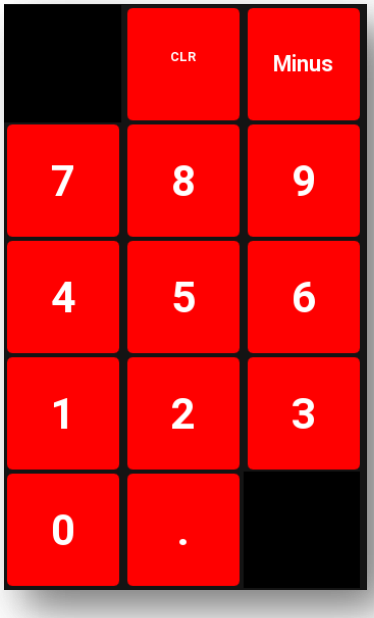


Using Numeric Datarefs

A numeric dataref as a dataref associated with a key that is writable. That means that X-Plane allows you to change that dataref from a plugin like X-KeyPad or FlyWithLua. There are two use cases for this:

1. You want to enter data in the number buffer such as a COM frequency, auto pilot parameter such as altitude, heading, or course, Altimeter Kholsman setting, or something like the transponder squak code.
2. There is an underlying simulator function that may not have a command associated with it or the commands for manipulating the functions are too restrictive but there is a writable dataref that can be manipulated. An example of this scenario is the transponder mode.

You will recall from the concepts section that X-KeyPad has this concept of a number buffer. The number buffer can be filled with digits, a decimal point, or a +- sign by executing a special set of commands. The C172 sample has a set keys associated with the special commands that will add digits to the number buffer when you press them.



Each key is associated with these commands:

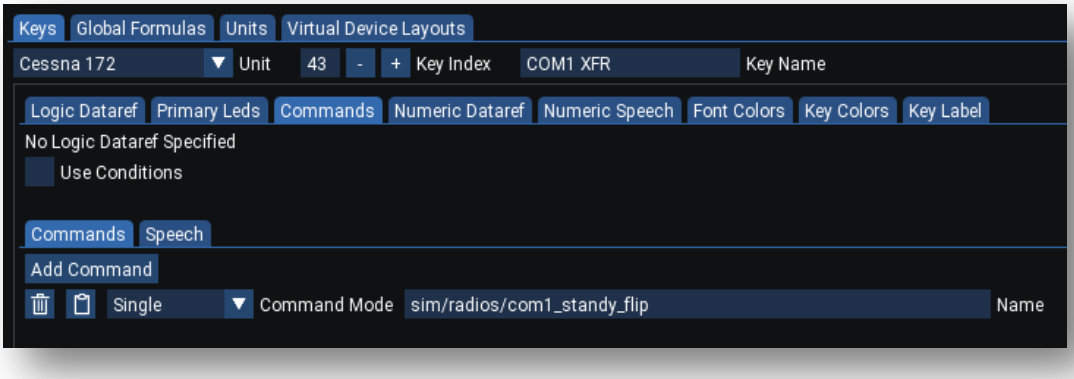
Command	Action
SRS/X-KeyPad/Keypad_0	Append a 0 to the number buffer
SRS/X-KeyPad/Keypad_1	Append a 1 to the number buffer
SRS/X-KeyPad/Keypad_2	Append a 2 to the number buffer
SRS/X-KeyPad/Keypad_3	Append a 3 to the number buffer
SRS/X-KeyPad/Keypad_4	Append a 4 to the number buffer
SRS/X-KeyPad/Keypad_5	Append a 5 to the number buffer
SRS/X-KeyPad/Keypad_6	Append a 6 to the number buffer
SRS/X-KeyPad/Keypad_7	Append a 7 to the number buffer
SRS/X-KeyPad/Keypad_8	Append a 8 to the number buffer
SRS/X-KeyPad/Keypad_9	Append a 9 to the number buffer
SRS/X-KeyPad/Keypad_Decimal	Append a decimal point to the number buffer
SRS/X-KeyPad/Keypad_Minus	Append a minus sign to the number buffer
SRS/X-KeyPad/Keypad_Clear	Clear the current contents of the number buffer
SRS/X-KeyPad/Keypad_BackSpace	Remove the last character in the number buffer

Once the number buffer is filled with the desired value when you click on a key that has a numeric dataref associated with it then the data in the number buffer will be written to the numeric dataref. You can even have the key perform a separate command function when it is clicked and there is no text in the number buffer.

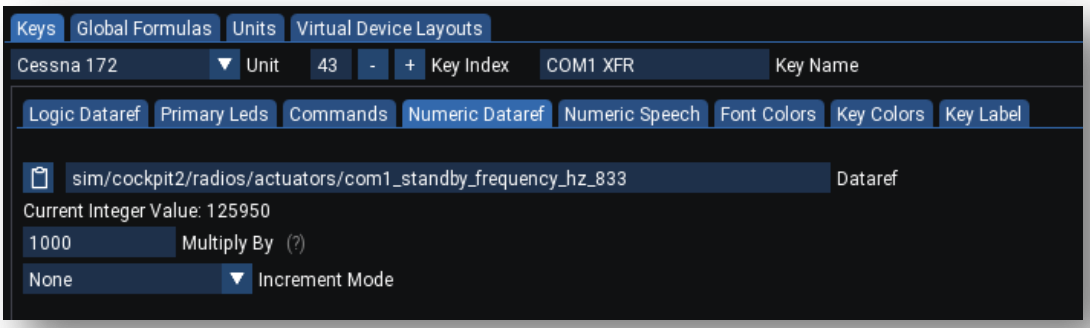
Data Entry Example

Let's take a look at the COM1 standby / swap key as an example:

First, the command associated with this key is the Com 1 swap which will swap the active frequency on COM1 with the standby frequency. This is the command that will be executed when the key is clicked and no data is in the number buffer:



The Numeric Dataref associated with this key is the Com1 standby frequency which is in KHz:

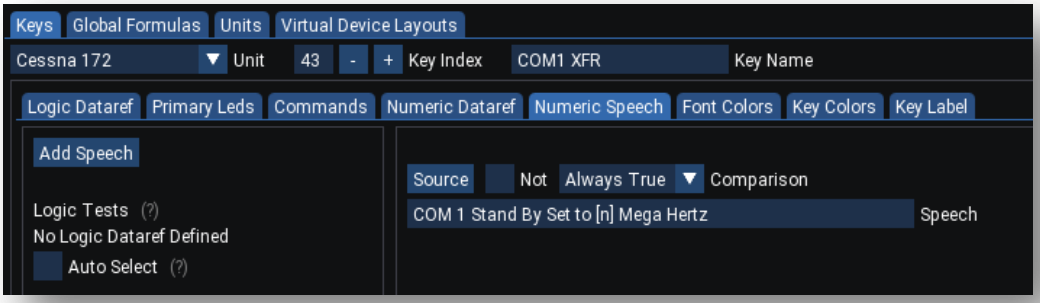


The Multiply By field can be used to multiple the value of the number buffer by this value prior to writing it to the numeric dataref when the key is clicked and there is number buffer data. You will notice that we have this set to 1,000. The reason for that is the numeric dataref wants KHz but we would prefer to enter it in megahertz. As an example 125.95 MHz needs to be 125,950 KHz.

The increment mode is set to none as we just want to execute the command when there is no data in the number buffer.

Optional Numeric Speech

Similar to command speech you can have X-KeyPad announce the value that was just stored. In this example we will have X-Key announce the frequency we set on the COM1 standby dataref:

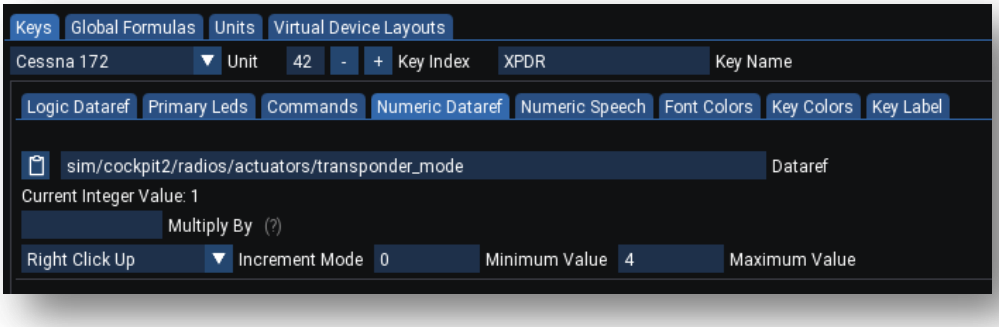


You could have this speech be conditional but in our case we will set the condition to always true as we want the speech anytime we set this value with the numeric entry. Note the [n] in the speech string. This tells X-KeyPad to insert the speech for the value of the number buffer into the full string. Using the upper case [N] will have X-KeyPad speak each number separately where the lowercase n will sound out the numeric value.

Data Manipulation

It is also possible to adjust the value of a numeric dataref using various increment modes. An example of this is the Transponder Mode Key:

In this example we will manipulate the transponder mode dataref between the values of 0 – 4 which represent the 5 transponder modes.

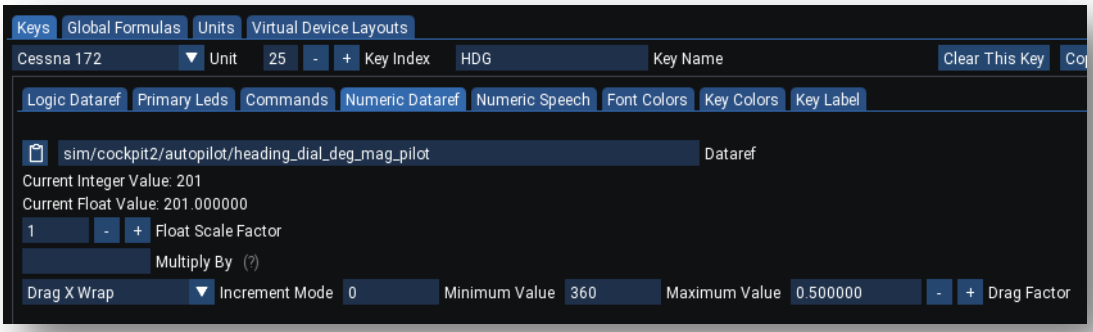


In this example we will increment the dataref by one when the key is clicked or touched in the right portion of the key and it will be conversely decremented by one when click in the left half. No increment or decrement will occur if the dataref is at the limits of 0 or 4.

Also note that we are using conditional speech on this key as we want the speech to change based on the new transponder mode being set.

Data Manipulation Using Mouse or Touch Drag

The Course and Heading keys are good examples of how you can either enter data using the number buffer or by using an increment mode with mouse dragging either horizontally or vertically on the screen.



In this example we will manipulate the pilot heading bug dataref by either entering a heading value into the number buffer and clicking the key or by dragging the mouse left or right once we have done a left click on the key.

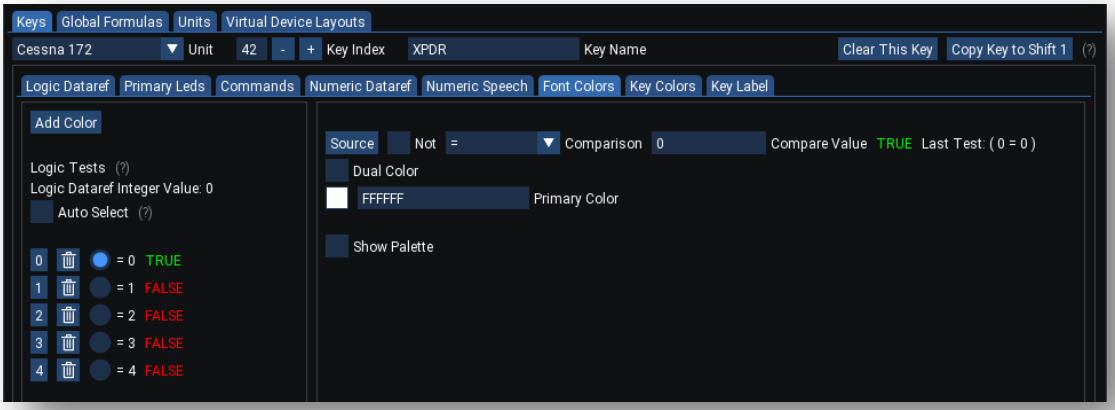
We use the Drag X Wrap to indicate that we want to increment or decrement the value depending on if we are dragging to the right or the left. The value will wrap when it gets to the limits. As an example if you try to increment past 360 it will wrap to 0 and if you try to decrement lower that 0 it will wrap to 360. The increment amount is controlled by the Drag Factor which is simply the number of pixels the mouse moved. In this example you need to move two pixels to get the heading to move by one.

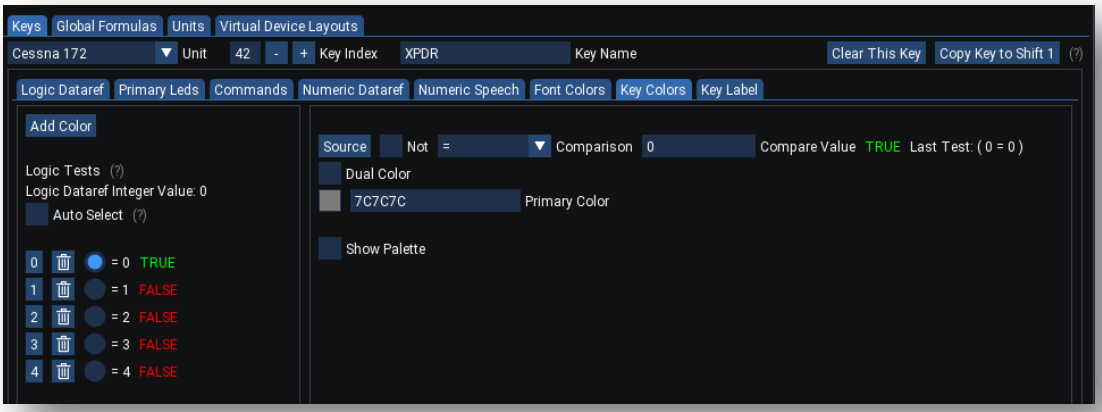
Complex Key Labels

Using Different Fonts

The XPDR Mode key is also a good example of a complex key label. If you observe the label behavior as you cycle through the various transponder modes you will notice that font and key background colors as well as the label text changes with each transponder mode.

This is all accomplished by specifying the transponder_mode dataref on the Logic Dataref tab and then creating conditions for the Font Colors, Key Colors, and Key Labels. The following two images show the Font and Key Color tabs:



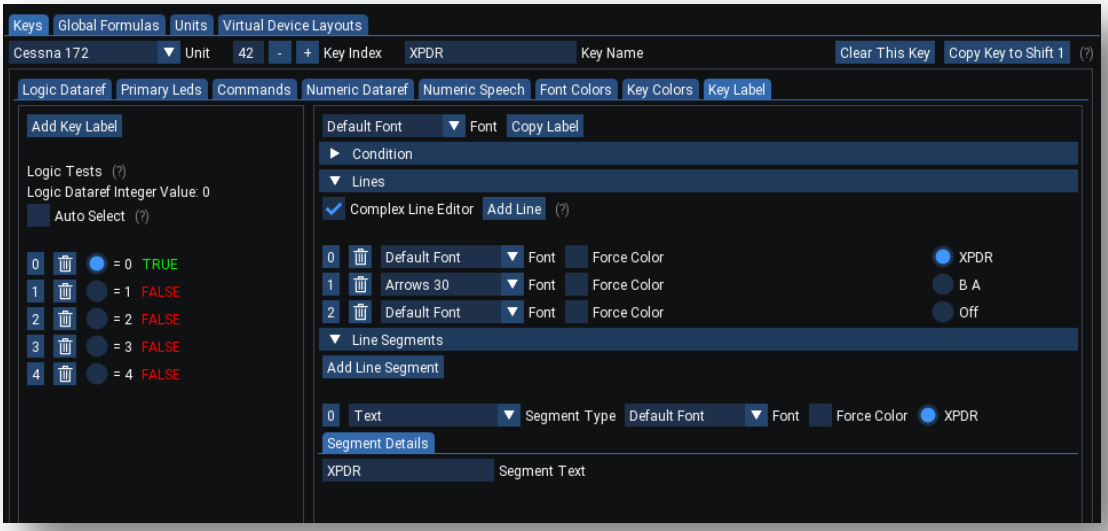


The Key Labels are using the Complex Line Editor option. With this option you can define label lines individually with more control over the lines appearance. As an example, you can force a line to use a specific font or even force it to specifically use one of the two font colors. That color override can be used to have some lines flash between two colors but other lines remain fixed on either the primary or secondary color.

Lines can also be portioned into line segments. Line segments can be either fixed text, a custom string created by a FlyWithLua script, numeric data from a dataref or formula, or the current contents of the number buffer. Furthermore, line segments can use a different font and have a forced color.

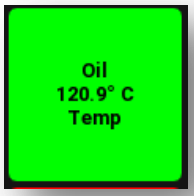
This combination of features gives you a great deal of flexibility over the key's visual representation.

The transponder mode key label definitions take advantage of multiple lines with some lines using the special arrow font to render the left and right arrows on the key. The appendix of this User Guide has the font glyph maps for the arrow and symbol fonts to you can see what characters need to be entered to render the various symbols.

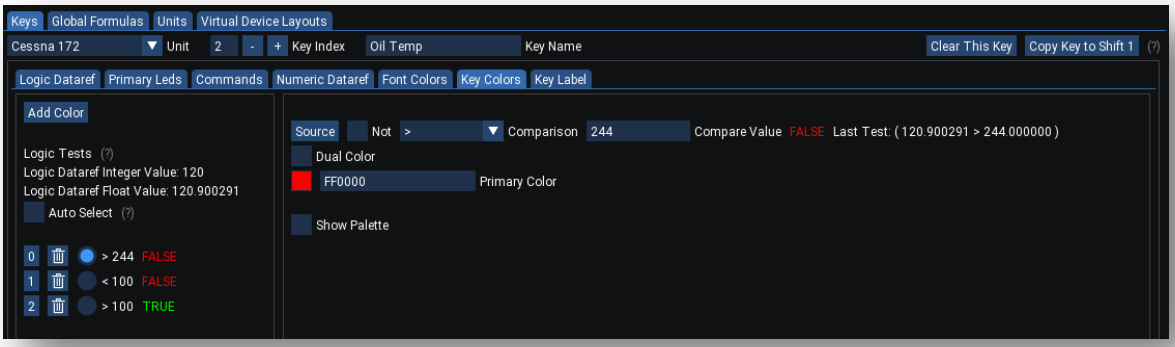


Including Numeric Data

The Oil Temp key is a good example of creating an enunciator key that has no operation behind it but the key colors change based on the oil temp values and the key contains the actual oil temperature.

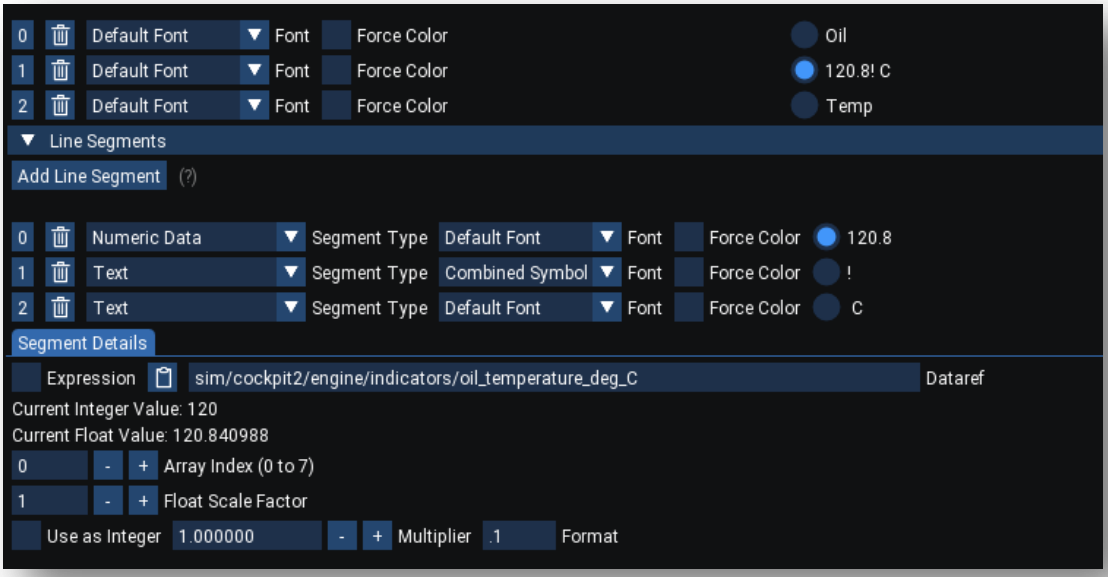


The key background colors are driven off the value of the value of the oil_temperature_deg_C logic dataref:



Depending on the value the key background will be either; Red, Yellow, or Green.

The key label uses three lines with the middle one displaying the actual oil temperature. That line is using multiple segments to show a formatted oil temperature data value, the degree symbol from the symbol font, and the fixed text for the C (centigrade).



The temperature segment is showing numeric data from the oil temp dataref. Note that the oil temp dataref is actually an array of floats representing the oil temperature of each engine. Since the C172 has only one engine we only need to look at index 0.

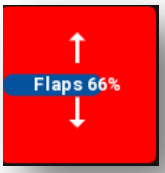
The multiplier field is useful when you want to show data in a different scale. As an example of a dataref had a range of 0.0 to 1.0 and you want to display it is a percent then you can use a multiplier of 100 to scale the value up. The Panel Light key is a good example of this. It is also used on the radio frequency keys to convert the KHz values to megahertz.

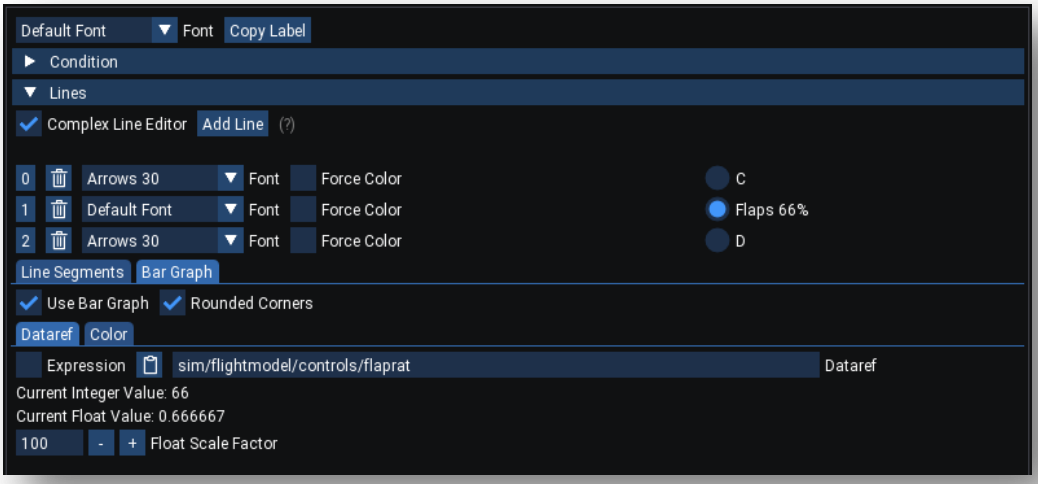
The format field gives you control over how the numeric data will be displayed. .1 indicates that we want only 1 decimal point of precision and no leading 0's. Here are some other examples of format:

- .0 No decimal precision, will just show as a whole number
- 7.1 The entire number will take up 7 character positions and will have one decimal precision
- 08.3 The number will use 8 character positions, have leading zeros and 3 decimal precision
- .2 The number will have 2 decimal precision and the field width is variable based on value

Displaying Numeric Data as a Bar Graph

It is possible to display a bar graph under a line based on the value of a dataref. The C172 flaps key is an example of this.





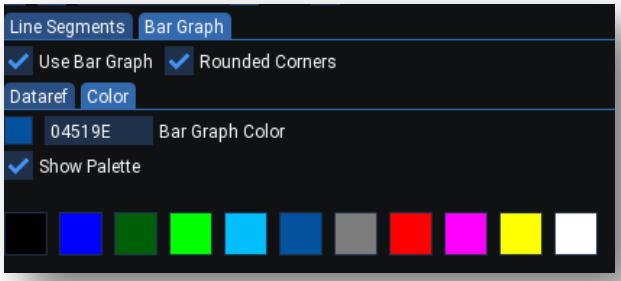
In this example we enable the Use Bar Graph option on the Flaps line. Once that is checked we will have the opportunity to enter a dataref that will be used to drive the amount of the bar graph that is displayed. The result from the dataref must be a value between 0 and 100 indicating the percent of the graph that should be displayed.

In this example we are using the dataref sim/flightmodel/controls/flaprat which has a value between 0 and 1.0. Since we need to get this into a range between 0 and 100 we can simply use 100 as a float scale factor.

You may need to use an expression when the underlying data does not lend itself to such a simple scaling operation.

The rounded corners option will draw the bar graph with rounded corners.

The color tab will allow you to pick a color for the bar graph.



Including Text from Byte Datarefs, Using Bit Testing, and Overriding a Data Source

The NAV key is an example of an even more complex key that gets data from multiple sources to render the key.

It is also an example of using bit testing to control the RED and BLUE primary led colors.

Bit Testing

On occasion you may run into datarefs where the meaning of their value is determined by which bits are set in the integer dataref. The Logic Dataref for the NAV key is: ***sim/cockpit/autopilot/autopilot_state***. This integer dataref shows the state of the default autopilot. Each bit position indicates the condition of a certain autopilot operation depending on the value of the bit position. See the following link for the various bit positions:

http://www.xsquawkbox.net/xpsdk/mediawiki/Sim/cockpit/autopilot/autopilot_state

In our example we want the NAV key to show as blue when the HNAV mode is neither armed nor engaged. We want it to flash red when HNAV is armed and solid red when HNAV is engaged. In order to accomplish the logic test we need to use the AND test operation. See:

https://www.guru99.com/c-bitwise-operators.html

You can read more about bitwise operations here:

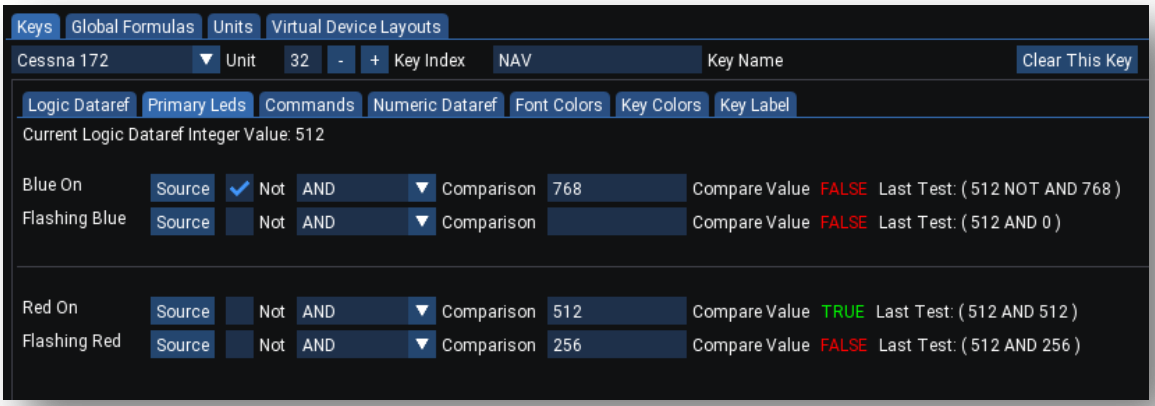
https://en.wikipedia.org/wiki/Bitwise_operation

You may also find this logical AND calculator useful:

https://www.rapidtables.com/calc/math/binary-calculator.html

The X-KeyPad logic test considers any non-zero test result to be a TRUE test. Any result that ends in a zero value is considered FALSE.

Consider the Red/Blue led logic tests for this key:



Regarding the Blue Led On test the value of 768 decimal is the sum of 512 and 256. Those are the two bits in the status that indicate if the HNAV function is armed or engaged. If we AND 768 with the dataref we will get a non-zero result if either bit is set and zero if neither of them are set. By using the AND operation we are effectively ignoring the condition of all the other bits since they have no bearing on the state of the HNAV function and we don't want their values to interfere with the setting of the led states. We set the NOT check box to invert the test logic thus if neither of the two bits are set we will get a blue led lit.

The Red On test checks to see of the HNAV is engaged by ANDING 512 with the status. If that evaluates to false we will then AND with the value of 256 to see if HNAV is armed and if that evaluates to true then the red led will flash.

Bit testing can be complicated and definitely takes some practice.

Alternate Data Source on a Logic Test

The NAV key's data is dynamic and will change based on the CDI mode selected on the GPS:

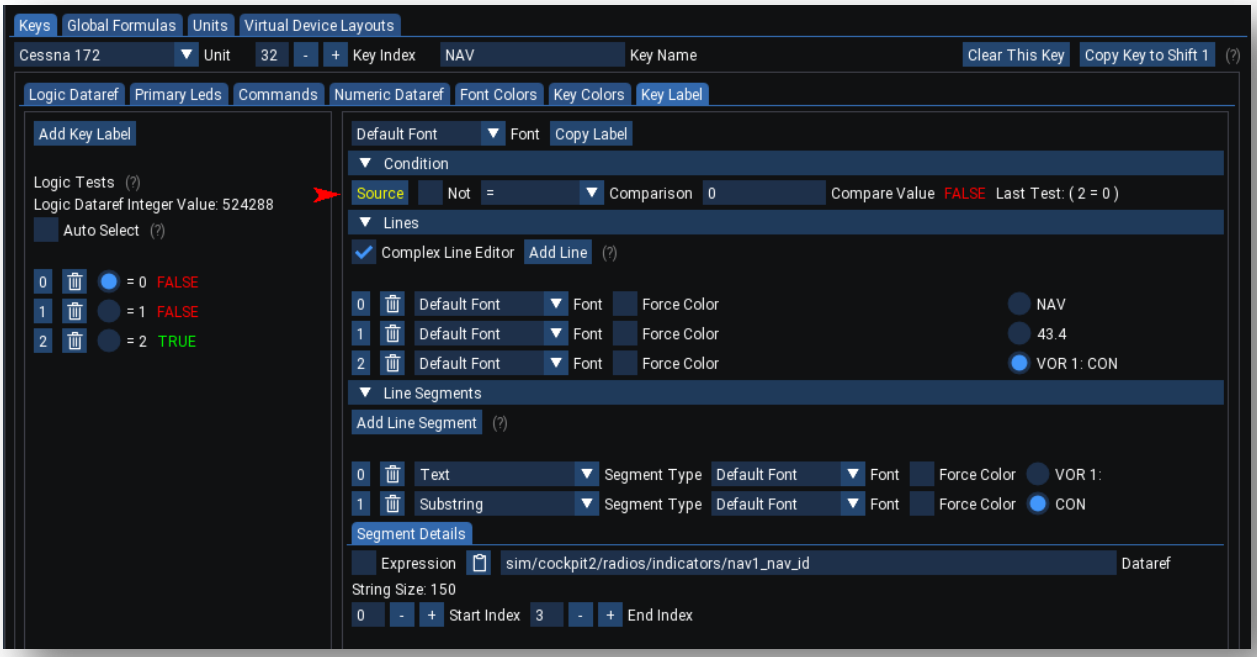
When in GPS mode it will show the distance to the next GPS waypoint as well as the ICAO code for the waypoint. When in NAV mode it will show the distance to the NAV aid (VOR etc.) as well is the ICAO code for the navaid, (KEEN for the Keene New Hampshire VOR).

In order to make the key choose the right data we need to select a label based on the value of the dataref: ***sim/cockpit/switches/HSI_selector***

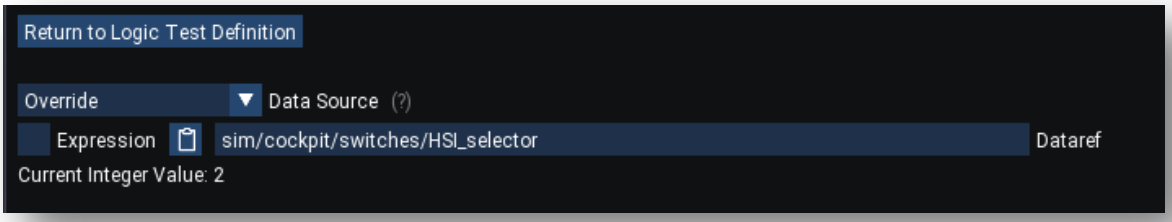
- 0 VOR1
- 1 VOR2
- 2 GPS

Normally this dataref would be selected in the key's Logic Dataref tab, however, we are already using that option to decide of the autopilot HNAV mode is off, armed, or enabled.

In these situations X-KeyPad allows you to select an alternate data source for a Logic Test. Refer to the Key Label tab in the following image:



On the left we have three logic tests for each of the three CDI modes, VOR1, VOR2, and GPS. On the right under the Condition expander you will see a Yellow Source button on the Logic Test, it is highlighted with a red arrow. You can use this button to configure the source of the data for the Logic Test. When clicked you will see a window which allows you to configure the data source:



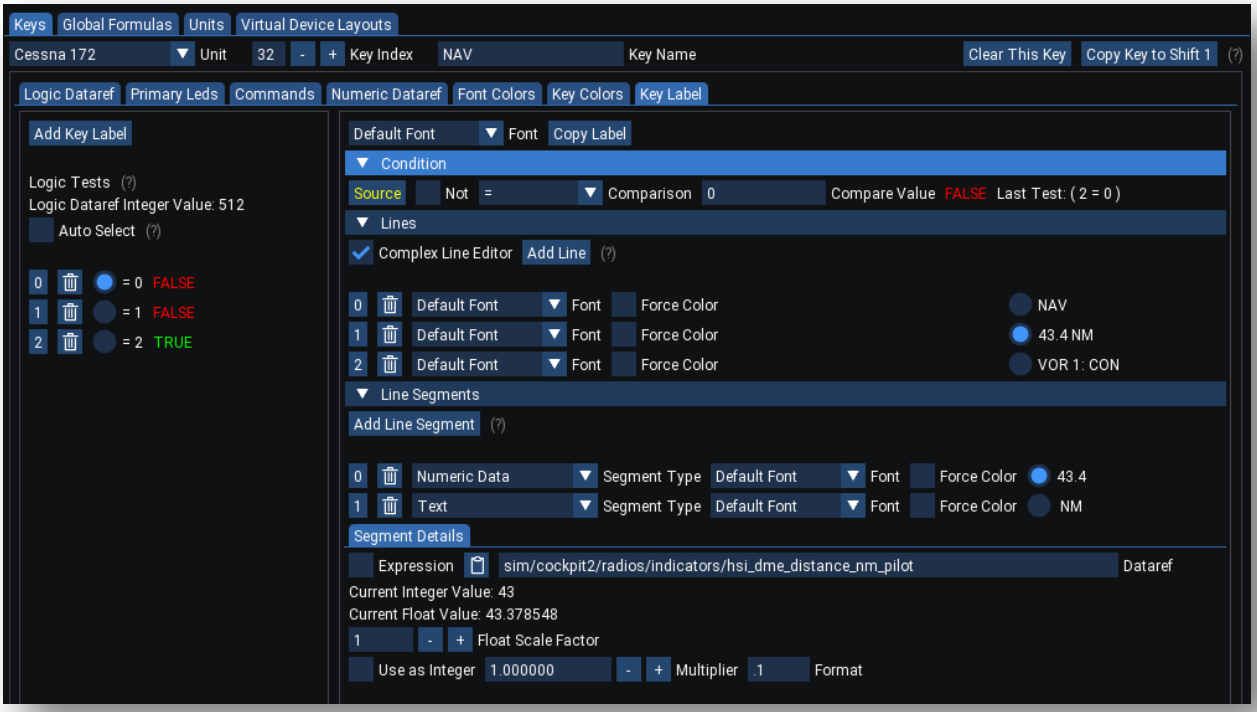
The two values in the dropdown are default and Override. In the default mode the Logic Test will use the dataref configured on the key's Logic Dataref Tab. In the Override mode you can specify the dataref you want to use for the test. In our case we want to use: ***sim/cockpit/switches/HSI_selector***

Each of the three Logic Tests will be configured to use this Override dataref. Depending on which value tests TRUE then the corresponding key label will be selected.

Each label is comprised of three lines. The first line is just the static text NAV. The second line is the Pilot's HSI DME distance which we get from the dataref:

sim/cockpit2/radios/indicators/hsi_dme_distance_nm_pilot

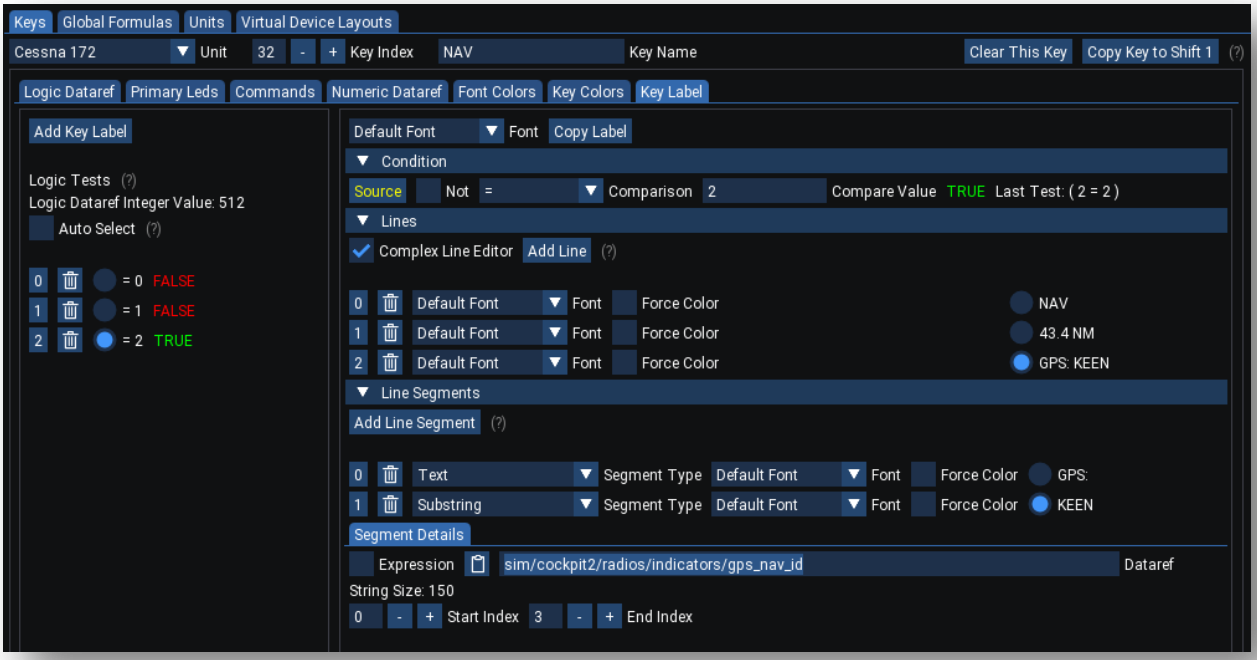
The segment for that value is configured to show one decimal point of precision:



The third line will show static text for the current CDI mode and then the text for GPS waypoint name or the ICAO name of the radio navaid. That text comes from three different byte array datarefs:

- sim/cockpit2/radios/indicators/nav1_nav_id
- sim/cockpit2/radios/indicators/nav2_nav_id
- sim/cockpit2/radios/indicators/gps_nav_id

In order to display that text we are going to use the Substring type which allows us to pick a range of character positions in the Byte Array to display.



In our case we will display the first four characters of the byte array so that will be index 0 through index 3.

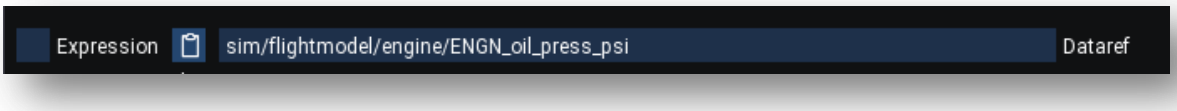
Note: You are allowed to set the End Index past the end of the byte array. When this occurs the substring will contain all the characters in the array starting at the start index and ending with the last character in the end index or the last character in the array if the array size is smaller than the specified end index.

Using Formulas and Expressions

Formulas and Expressions allow you to combine multiple datarefs and constants using mathematical expressions to create a value that can be used in a Logic Test, as data to be displayed on a key, or even data that will be written to a result dataref. Formulas and Expressions also allow you to conditionally perform math operations based on values of user defined variables.

Prior to X-KeyPad 1.5 this complex data manipulation needed to be done in a FlyWithLua script. You can still use FlyWithLua if you prefer but Formulas and Expressions may offer a simpler solution for the basics.

Formulas are expressions with more flexibility on what happens with the result. As an example, any dataref can optionally be defined as an expression by simply checking the expression checkbox:



The result of any defined expression will be used as the data in the dataref. It is only available in the context of that dataref.

Formulas are expressions where the result is stored in a global location. By default a formula's result will be stored in one of 100 formula datarefs:

- SRS/X-KeyPad/FormulaFloat[100]
- SRS/X-KeyPad/FormulaInt[100]

The index of the array is based on the Formula number 0 – 99.

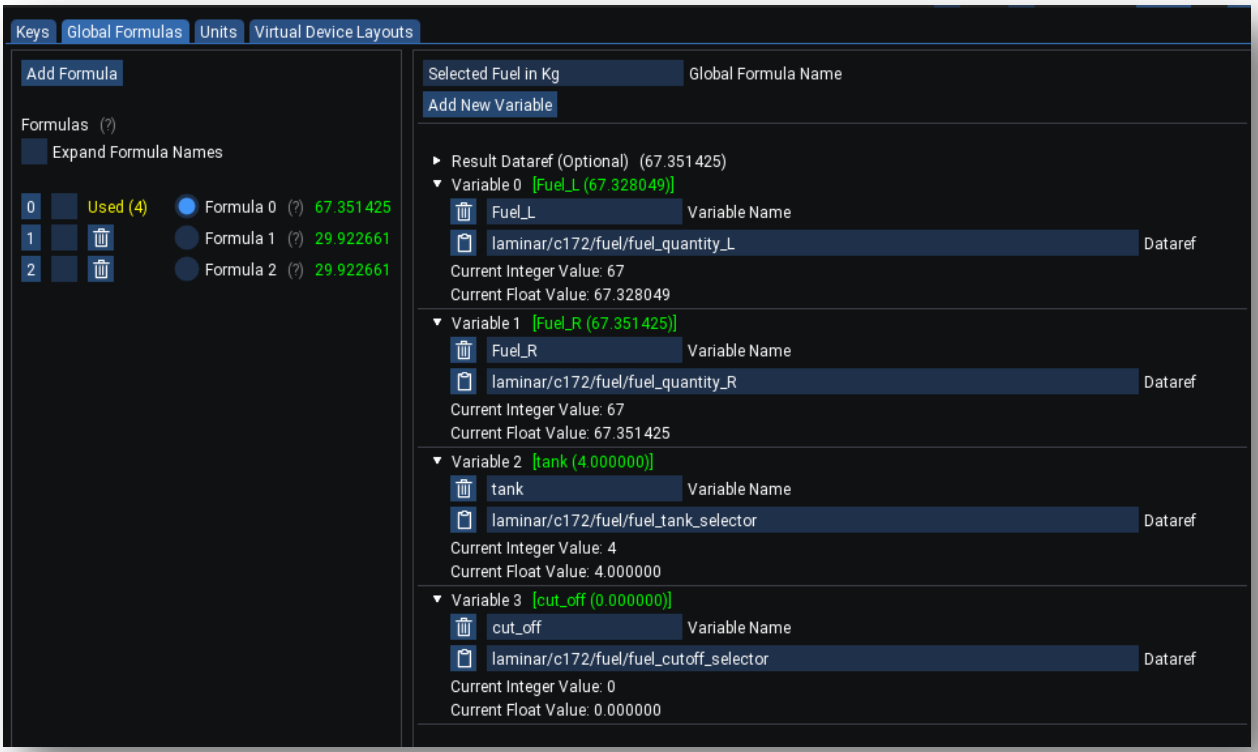
Optionally you can specify the dataref you want updated with the formula results with the formula is evaluated. Formulas are evaluated on every flight loop.

The fuel selector key is an example that uses a formula:

In this example the total fuel in gallons being shown on the fourth line is derived from a formula. The formula is used to mathematically calculate the fuel available to the engine based on the position of the fuel selector and the fuel shutoff:

When the fuel shutoff is off then zero fuel is available to the engine. When on the amount of fuel available is determined by the position of the tank selector.

On the Global Formulas tab Formula 0 is the formula that will make this calculation. It uses four defined variables to accomplish this:



- Fuel_L laminar/c172/fuel/fuel_quantity_L
- Fuel_R laminar/c172/fuel/fuel_quantity_R
- tank laminar/c172/fuel/fuel_tank_selector
- cut_off laminar/c172/fuel/fuel_cutoff_selector

The formula's expression is:

```
if[cut_off = 1,0                          // If Cut off then 0 fuel
    ,if[tank = 2,Fuel_L + Fuel_R          // If Both then Fuel Left + Fuel Right
        ,if[tank = 1,Fuel_L,Fuel_R]       // Else if left then Fuel L else Fuel Right
    ]
]
```

The syntax of an expression is very similar to how you use formulas in Microsoft Excel. If statements are conditions that are enclosed in [ ]. They can be nested.

In our example the first condition test is if the cut_off variable is 1 then we will return a formula result of 0. In this fairly simple test we can detect if the fuel cut-off knob is OUT (a value of 1) then the engine is starved of fuel so zero gallons.

A test expression is followed by a command and a math expression that will be used if the test was true. Then there is another comma followed by the math expression (Or Another nested IF) that will be evaluated if the test was false. The left and right braces [] must properly enclose the three parts of the IF statement. If they are not properly positioned you will get an expression error. In this example we used indentation and comments to help see the nesting more easily.

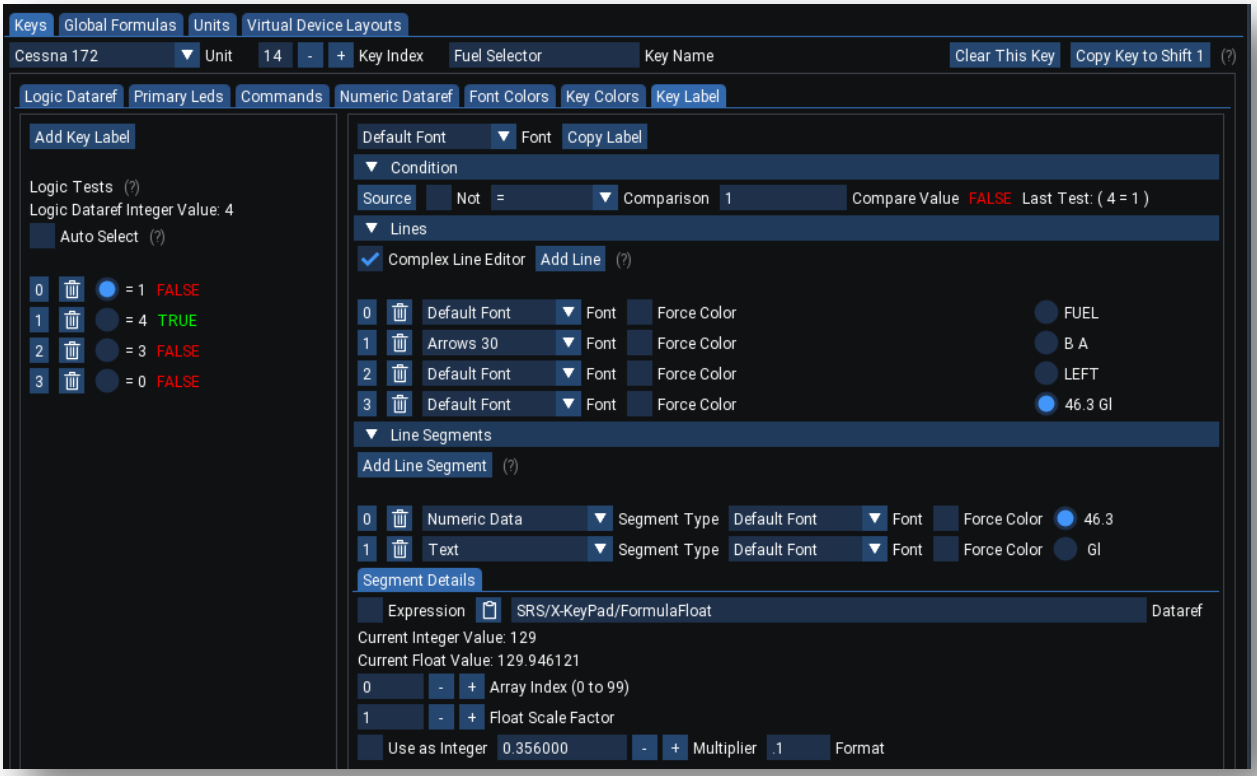
Following that logic we then test if the fuel selector is in the both position (value of 2). If it is 2 we will return the sum of the left and right tank quantities.

If that is false then the selector is either left or right so our third nested IF tests if the selector is in the left position (value of 1). If it is then we return the left fuel tank quantity. If that fails then the only other possibility is it is in the right position so we return the value of the right tank quantity.

Since we did not define an optional result dataref the formula's result will be stored in:

SRS/X-KeyPad/FormulaFloat[0]

We will use this value in the Fuel Selector key:



Note that the line segment containing the fuel quantity is using the Formula Float dataref index 0. Also note that we multiple this value by .356 before it is displayed on the key to convert it from kilograms to U.S. gallons using the appropriate conversion factor.

An even more complex use of formulas is used by the HG/HPA key and the Baro Key:

These two keys use global formula's 1 and 2 along with a shared float dataref to allow you to see and set the altimeter in with Inches of mercury or Hecto Pascals. Both formulas have comments that will describe how they work together.

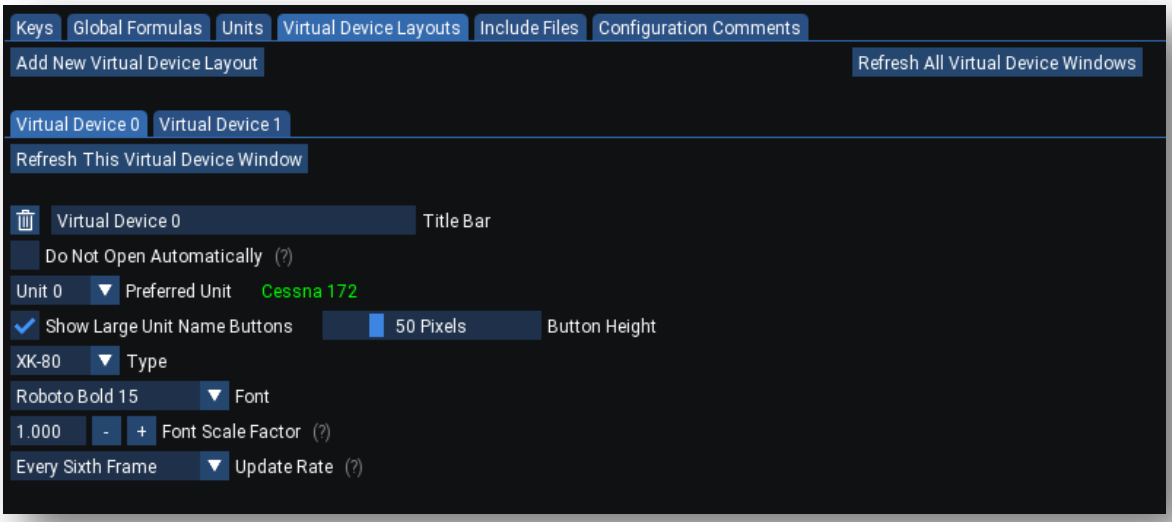
Please keep in mind. Much of what can be done with a formula can also be accomplished with FlyWithLua scripts and by using the Shared Float / Integer datarefs and / or the customer string datarefs. Which approach you use is entirely up to you.

Under the covers X-KeyPad is using the C++ library bcParser. You can read more about the library here:

https://www.mathparsers.com/docs/cppdoc/html/

Virtual Device Layouts

You can have up to 8 virtual devices windows. Each virtual device window can be defined with a different number of rows and columns. The Virtual Device Layout tab will allow you to define the Virtual Device windows:



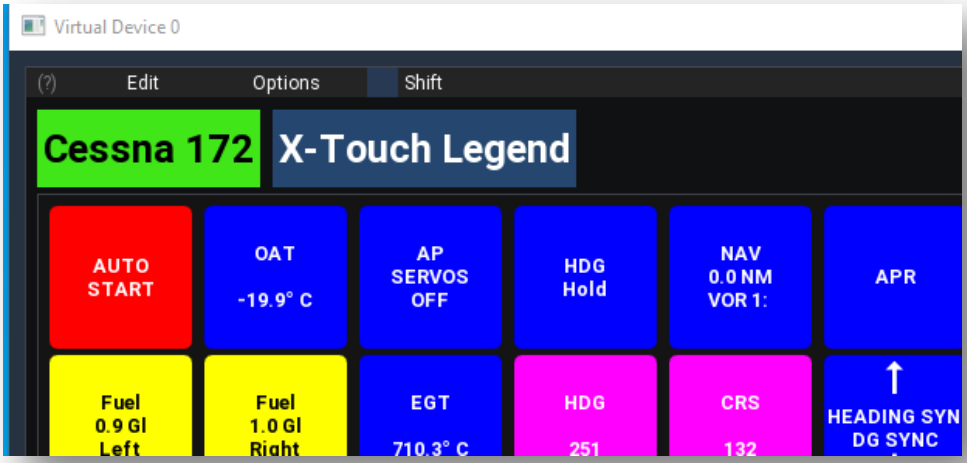
You can specify the text you want to see in the Virtual Device title bar.

You can also specify if you do not want the virtual device to be automatically opened when the aircraft loads. If enabled you will need to use the X-KeyPad plugin menu item “Open and Closed Virtual Devices” to create the windows.

You may recall from the concepts section that you can define up to 8 different units. You can specify which unit number should be selected by default with the VD window is created.
The type dropdown will allow you to define VD layouts that mimic the P.I. Engineering hardware keyboards. You can also use the custom type to define a layout with a custom number of rows and columns as long as the product of the rows and columns does not exceed 256 total keys.

The “Show Large Unit Buttons” option will display the unit names in larger buttons just below the Virtual Devices menu bar. This may be useful when using the VD on a touch monitor with multiple units. Without this option enabled the unit names are displayed as smaller button on the menu bar. They can

be harder to activate with your touch since they are not very tall. Here is an example of what they would like with the Larger Button option enabled:



The font scale factor allows you to scale the fonts on the keys. This may be useful when you are using a configuration that was designed to be used on a 1080P monitor but you wish to use it on a 4K monitor. Scaling the fonts will make them more readable on the higher resolution screen.

The update rate determines how often X-KeyPad will evaluate the underlying conditions of the key to determine how it should be displayed. The logic for this can be a bit time consuming and doing it every frame will impact your frame-rate. Lower update rates will use fewer resources and will not material impact the visual appearance of the key.

Note: Any time the mouse is down over the window or you touch a touch sensitive monitor X-KeyPad will revert to calculating the key's appearance every frame. This is necessary to make sure we don't miss mouse clicks and to support drag and drop operations.

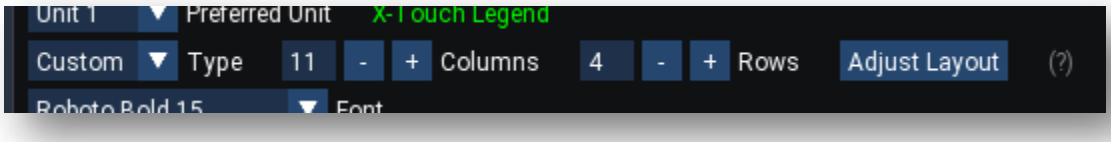
Changing the size of a Virtual Device grid

You may recall from the concepts that where on a virtual device grid a particular key index is displayed depends on the number of rows and columns on the virtual device. Key indexes start at 0 for the first row and first column and then increment through all the rows in the first column and then continue to all the rows in the second column.

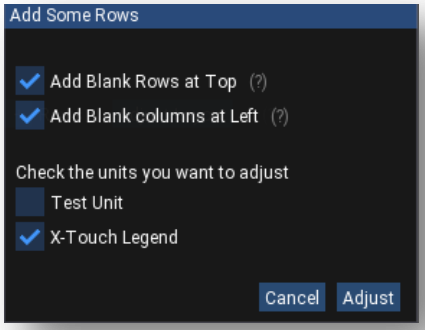
Generally if you define keys and then later on decide to change the number of rows on a virtual device, the keys will be in a new location on the grid because the number of rows per column will change. Adding columns is less of an issue as they will just be added to the right and since the rows per column does not change the keys will remain in same basic grid configuration.

X-KeyPad does have a feature for custom virtual device types that will rearrange the key indexes when you need to add rows. It can also do this for deleting rows although you will lose any key definitions in the deleted rows. It also allows you to optionally add blank columns to the left.

If you change the rows and columns on a layout when you have a virtual device window created you will see an additional Adjust Layout button.



Clicking this button will bring up a dialog that allows you to pick the units you want to adjust and where you want rows and columns added or deleted.



If all your units run in the same virtual device window or in multiple virtual device windows that have the exact same number of rows and columns then you will want to check all the units before clicking adjust.

On the other hand, if some of your units are designed to work in separate virtual device windows that have a different number of rows then you do not want to check those units as the adjustment will not look right.

The key factor in how key indexes are layout in a virtual device windows is dependent on the number of rows per column.

Configuring P.I. Engineer Keyboards

Many of the concepts discussed in configuring virtual devices apply to configuring a P.I. Engineering keyboard. Here are some of the main differences.

- The key layout on a keyboard is fixed based on the unit type. Unlike a virtual device you cannot change it.
- Keys only have a red and blue led under the keycap. You cannot choose colors beyond the primary led configuration.
- Key labels are obviously fixed. Most people will print out the legends using a laser printer and vellum paper and then place the cut out legends under the clear keycaps.

- Unit numbers and the key definitions for those units are directly assigned to the physical keyboard's unit number. You cannot dynamically change them on the fly

Unit Numbers

The P.I. Engineering keyboards can have either an assigned or unassigned unit number and X-KeyPad can work with a Hardware ID mode enabled or disabled. This can be selected on the settings UI on the X-Keys Settings Tab.

When not using Hardware ID mode unit numbers will be assigned to each P.I. Engineering keyboard as they are found on the USB bus starting with unit 0. This works fine when you only have one device on your system as it will always be associated with unit 0, however, when you have more than one device the unit numbers can get rearranged if you change the USB ports that the devices are plugged into.

If you have more than one P.I. Engineering keyboard we recommend assigning a unique unit number to the device. This can be done on the X-KeyPad settings page:

X-Touch Mini

Overview

X-KeyPad can support multiple X-Touch Mini MIDI controllers. The configuration of these devices is done entirely using a GUI. The resulting configurations are stored in json text files in the aircraft folder for the current aircraft being used.

The json files will be named X-Touch-0_AircraftType.json, X-Touch-1_AircraftType.json, etc. The –n suffix represents the X-Touch Mini Device ID number. When you have only one X-Touch Mini on your system then n will always be 0. When more than one X-Touch Mini is connected the n is the device ID assigned to the physical device and it can be a number between 1 and 16.

The AircraftType will be the same name as the aircraft ACF file name. The reason we key the configurations files to the AircraftType is so you can have different configurations for multiple aircraft variants that may reside in the same folder. A good example of this is the steam gauge Cessna 172 and the G1000 Cessna 172.

- Cessna_172SP.acf
- Cessna_172SP_G1000.acf

Assuming a single X-Touch Mini with a configuration for each variant would result in two json files:

- X-Touch-0_Cessna_172SP.json
- X-Touch-0_Cessna_172SP_G1000.json

X-KeyPad has the ability to support a common configuration for all aircraft assuming that you only use command and datarefs that work in all aircraft. If X-KeyPad does not find a configuration in the aircraft folder it will then look in the X-KeyPad folder for files named like this:

- X-Touch-0.json
- X-Touch-1.json

If it finds them it will load them.

Samples

X-KeyPad comes with an X-Touch Mini configuration sample for the C172. It also has an optional Virtual Device configuration that can be used as a dynamic legend for the X-Touch Mini. This Virtual Device can be placed on a small inexpensive monitor that you would place directly behind the X-Touch Mini.

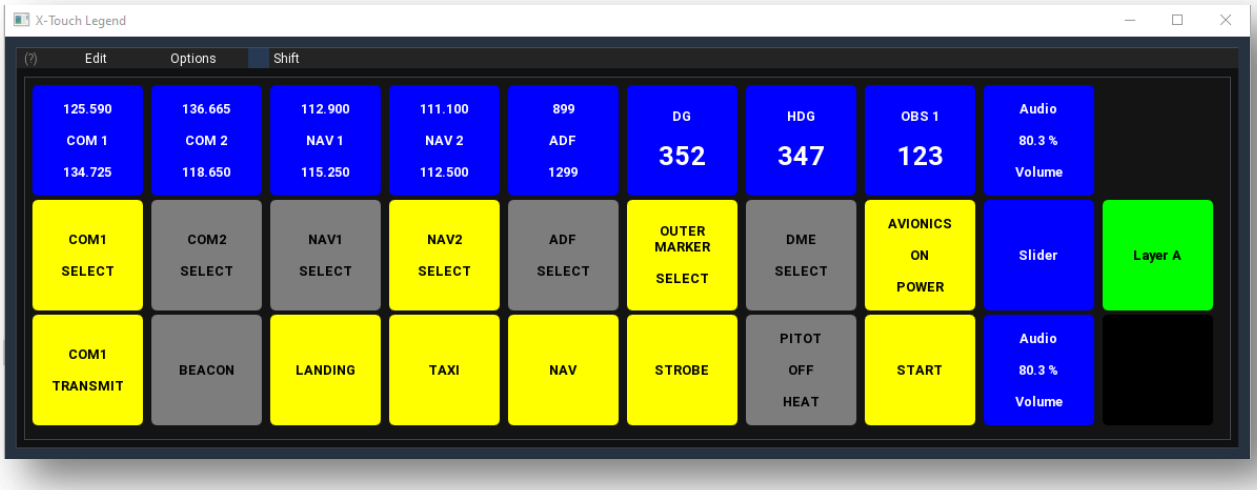
The sample files are located in:

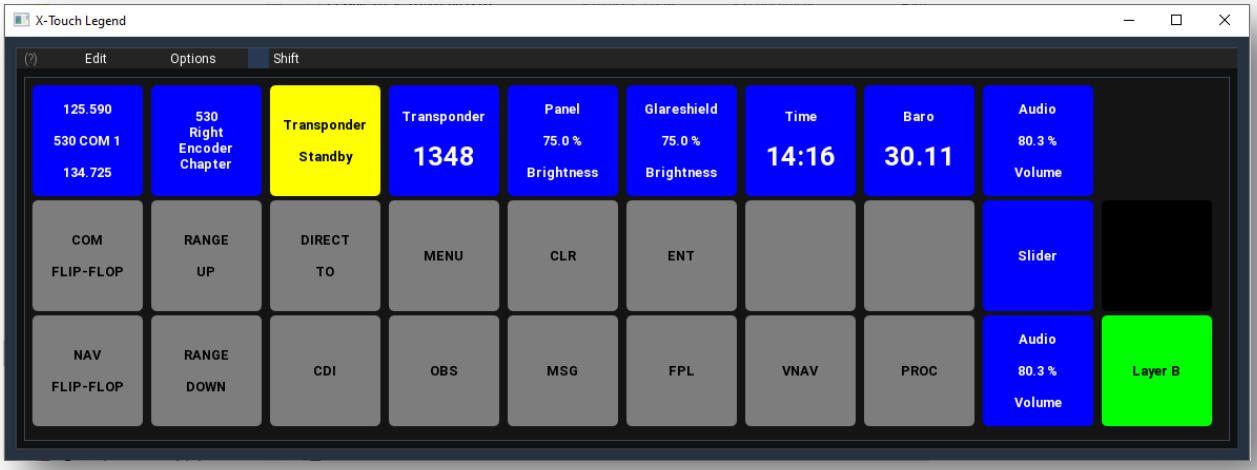
X-Plane 11/Resources/plugins/X-KeyPad/Samples/X-Touch Mini

Copy the file, X-Touch-0_Cessna_172SP.json, to:

X-Plane 11/Aircraft/Laminar Research/Cessna 172SP

The following images are examples of what the Virtual Device legend looks like:

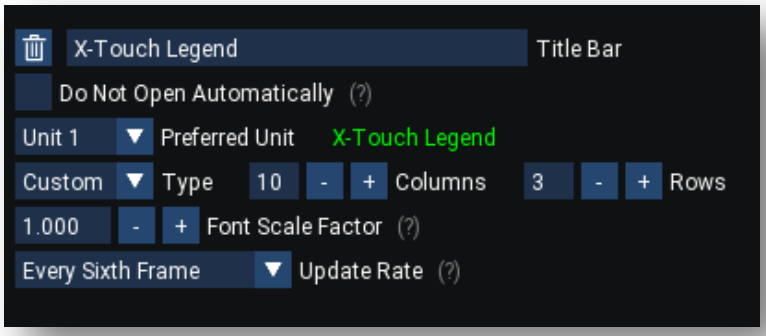




If you want to use that legend also copy the file, X-Keys_Cessna_172SP.json, to:

X-Plane 11/Aircraft/Laminar Research/Cessna 172SP

Note: If you are also planning on using the full C172 X-Keys sample along with this legend you may want to consider appending the legend configuration file to the main C172 X-Keys configuration. The legend will show up as Unit 1 and you can create a separate VD window for it using these parameters:



Configuration

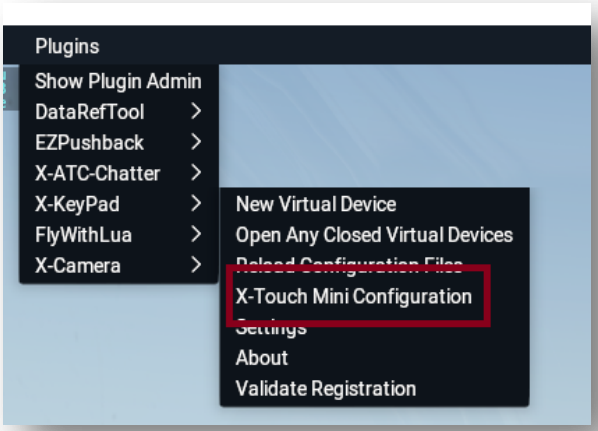
Commands and Datarefs

Configuring the X-Touch Mini involves assigning the various inputs to either a command or a dataref. You can learn more about commands vs datarefs here: Commands vs Datarefs.

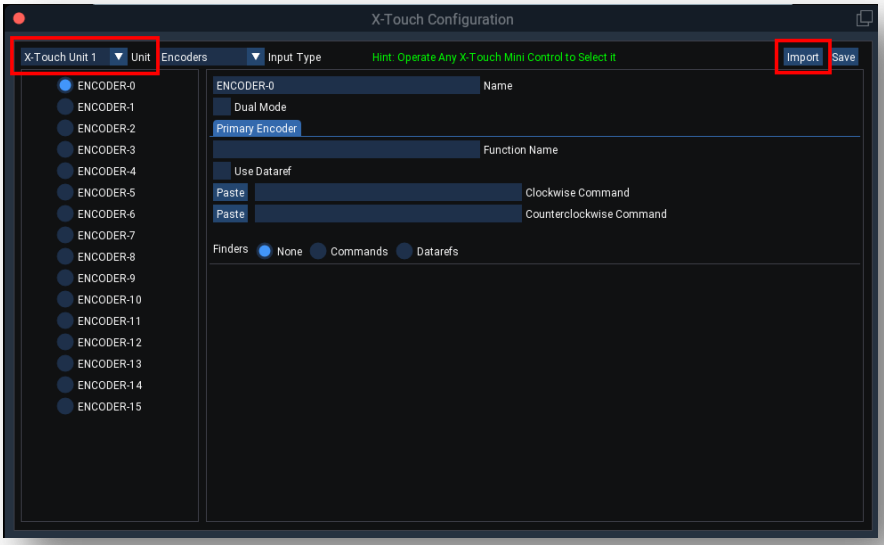
Sliders are always associated with a writable dataref. Encoders and Buttons can be configured to either operate commands or manipulate a writable dataref. The button led state is always governed by the state of dataref.

Sample X-Touch Mini Configuration

For the following configuration discussion will be using a Cessna 172 Steam Gauge example. You can import this sample using the X-Touch Configuration UI:

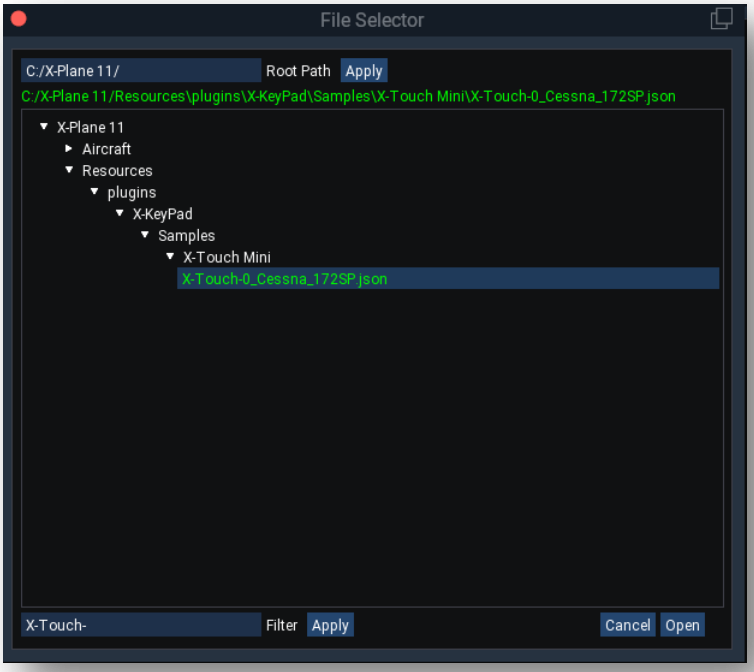


Use the Unit dropdown control to select the X-Touch unit you want to import the sample to and then click the Import Button:



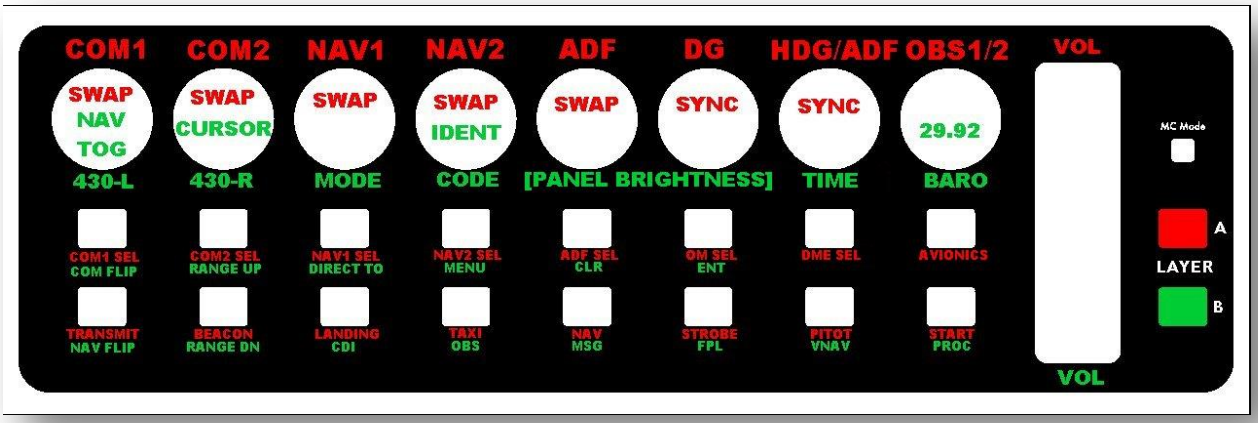
Use the file selector to navigate to the sample:

X-Plane 11/Resources/plugins/X-KeyPad/Samples/X-Touch Mini/X-Touch-0_Cessna_172SP.json



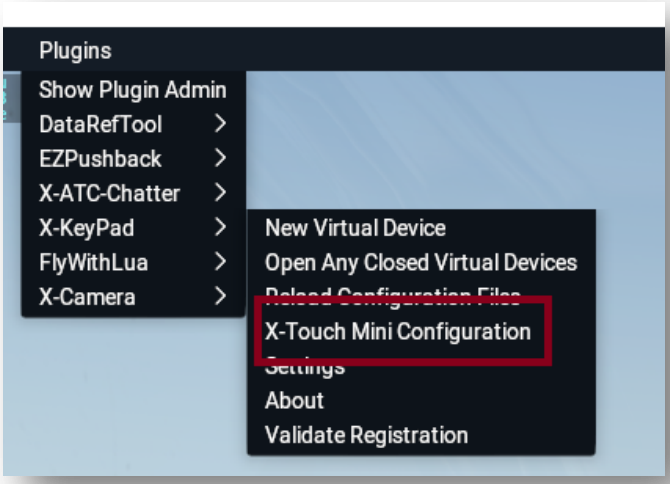
Click on the file name to select it and then click open to import the configuration.

This configuration supports modeling the radios, lights, Altimeter, OBS, Garmin 530, transponder, heading bug, DG, panel lights, audio volume, and time of day.

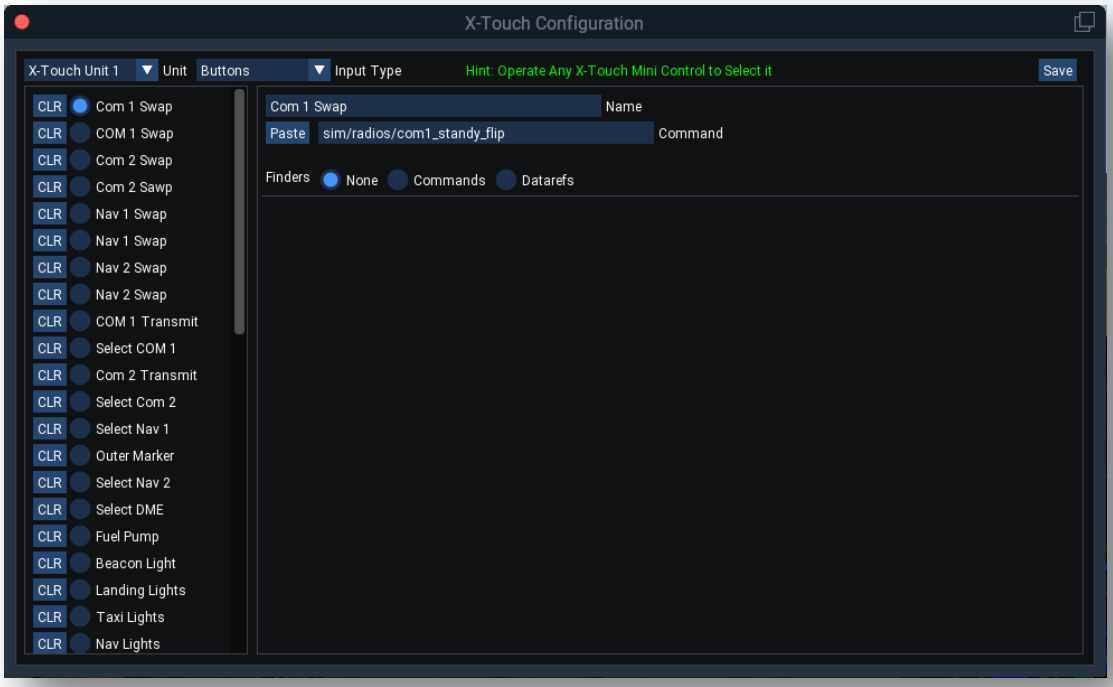


Using the Configuration UI

You can open the X-Touch Mini configuration UI using the X-KeyPad menu:



Once the UI is open you can navigate to the X-Touch Mini unit and input type using the dropdown controls and radio buttons on the left side of the UI. You can also simply activate one of the inputs on the X-Touch Mini and the UI will detect that activity and automatically select that input for configuration:



Using the Clipboard

You can hand enter the names of commands and datarefs but it is far easier to use the clipboard to copy the names from the datareftool or the built in finders. The Paste button next to the Command and Dataref fields will paste the contents of the clipboard into the field. The copy buttons on the datareftool and finders will copy the name to the clipboard.

Configuring Encoders

Dual / Single Encoders

Encoders can be operated as either single knob encoders or simulated dual knob concentric encoders. In dual mode you can enter both a primary and secondary operation specification.

When in dual mode the encoder's push button can be configured for three different modes:

Mode	Behavior
Toggle Encoder	When pressed and released the encoder will toggle between the primary and secondary encoder definitions
Momentary Toggle	When pressed and held down the encoder will switch to the secondary encoder definition. As soon as the button is released the primary encoder definition will again become active
Short Toggle / Long Command	A short press and release of the button will cause the encoder to toggle back and forth between the primary and secondary encoder definitions. A long press will active a user define command. If you setup the dual encoder to have more than two encoders then the short press will make encoder active. If you short press when the last encoder is active the Primary encoder will be made active next.

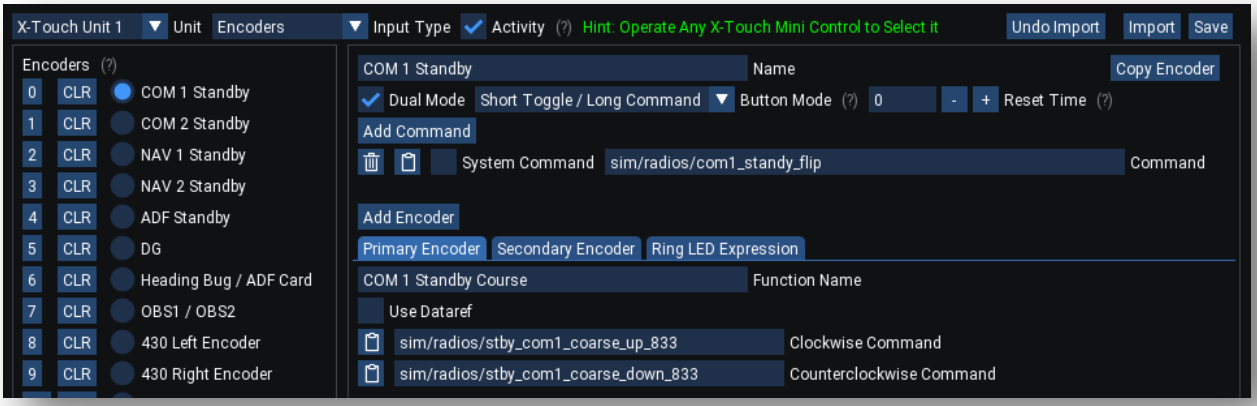
Encoder Configurations

You can configure encoders using two techniques. The first is to associate commands to the Clockwise and Counterclockwise rotations of the encoder knob. Each encoder detent will result in the execution of the command.

Using Commands

Commands can be found by using the Dataref Tool or by using the command finder.

A good example of using commands on a dual encoder is the COM and NAV standby frequency controls on layer A.



For the COM1 Standby course control we will be using the two commands on the primary encoder:

- sim/radios/stby_com1_coarse_up_833
- sim/radios/stby_com1_coarse_down_833

For the COM1 Standby fine control we will be using the two commands on the secondary encoder:

- sim/radios/stby_com1_fine_up_833
- sim/radios/stby_com1_fine_down_833

The encoder button is configured so that a short press will toggle the encoder between the course and fine configurations and a long press will execute the command, ***sim/radios/com1_standy_flip***, which will cause the COM1 standby and active frequencies to flip-flop.

The Reset Time can be used to automatically reset the active encoder to the primary encoder after the specified number of seconds has passed with no encoder rotations. If you set the Reset Time to a value of 0 then the either the Primary or Secondary encoder will remain active based on the last use.

Using a Dataref

On occasion you may find that the aircraft designer did not define commands for a particular operation but they may have defined a writable dataref that could be manipulated numerically for the operation.

The other situation where using a dataref might be desirable is to make the encoder more responsive to rapid turning. When using commands a rapid rotation of the encoder will result in multiple commands being queued for execution. Each command can only be executed once per flight-loop(frame) so if you have a bunch of commands queued it may take a number of frames to execute them. Depending on your frame rate the encoder operation may be a bit sluggish.

Let's take a look at the Heading Bug / ADF Card example on layer B.

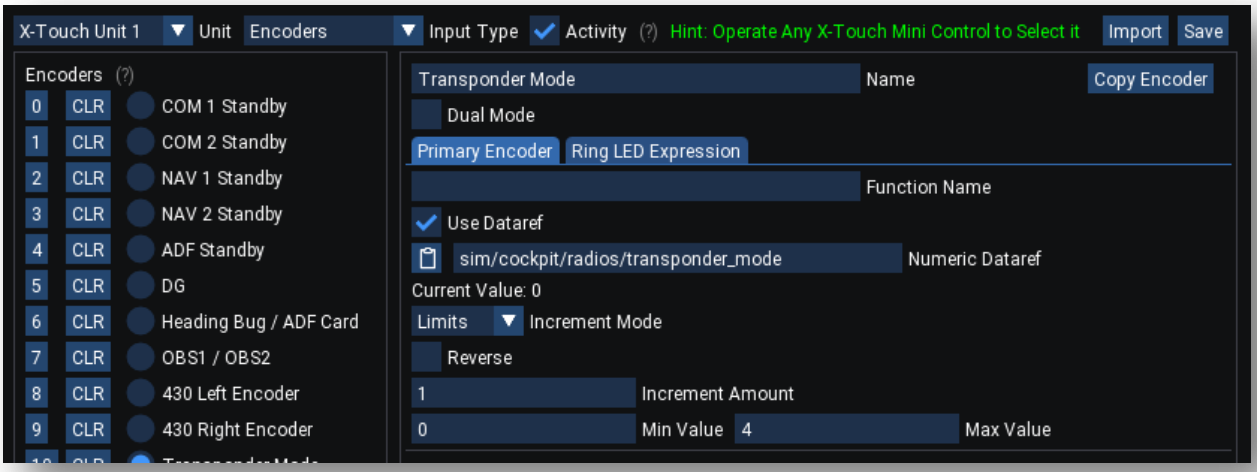
The heading bug has a writable dataref named:

- sim/cockpit/autopilot/heading

This is a floating point dataref that has values between 0 and 359 degrees. We can use the dataref mode of the encoder in Wrap mode to increment / decrement the dataref by a value of 1 between the min and max values of 0-395. Rotating the encoder right will add the Increment Amount to the dataref and rotating the encoder to the left will subtract the increment amount. If you increment or decrement past the limits the value will wrap. As an example, incrementing past 359 will cause the dataref to go to zero and decrementing below 0 will cause it to go to 359.

The Reverse checkbox can be used to swap addition and subtraction on the clockwise and counter clockwise encoder rotations.

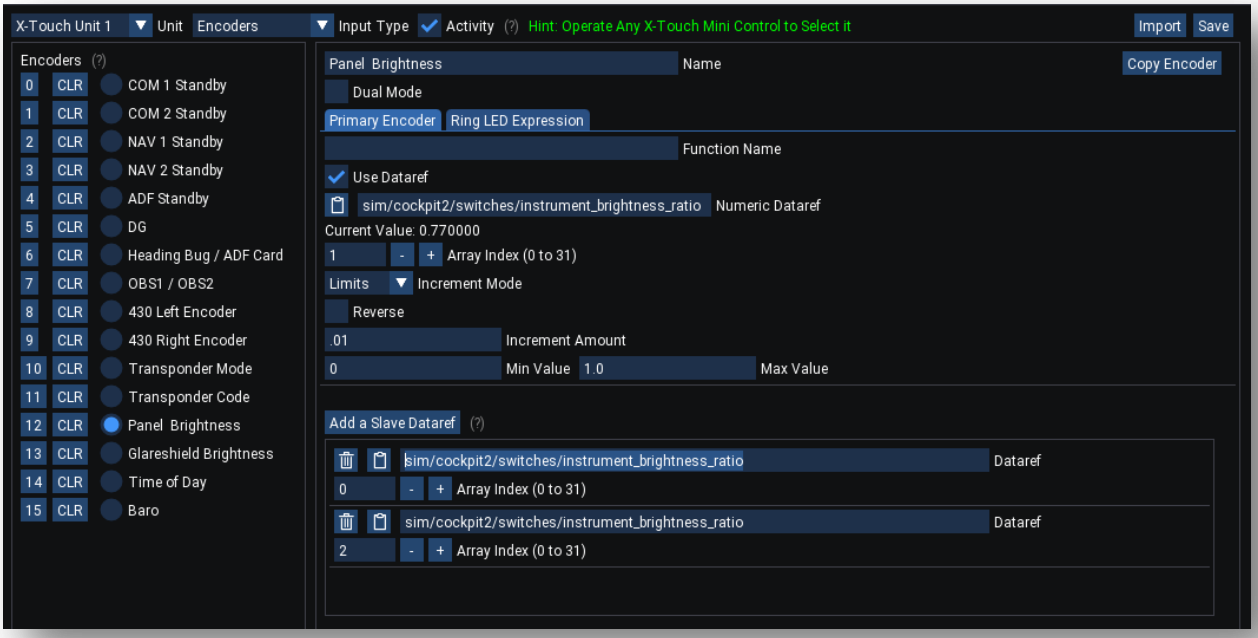
There is also a Limits increment mode where the dataref will only be incremented / decremented to the max and min limits. Further rotations of the encoder with not change the underlying dataref past its min and max values. The Transponder Mode encoder is an example of using limits. In this configuration the writable transponder mode dataref will be manipulated between the transponder 5 modes (dataref values 0 - 4).



Panel Brightness Example

Datarefs can also be represented as arrays of numbers. The Panel Brightness example illustrates this situation. The C172 has a number of controls to operate the panel lights. In the simulator these are all controlled by a float array named: ***sim/cockpit2/switches/instrument_brightness_ratio***

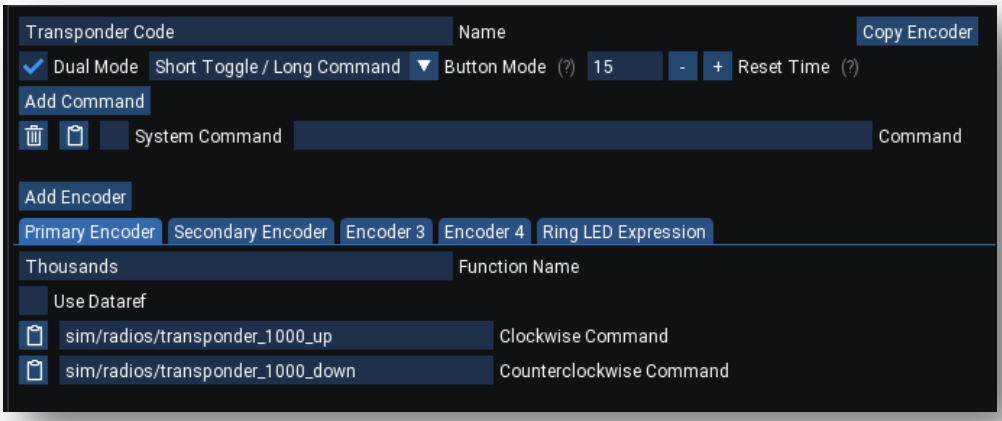
The Panel Brightness encoder has been configured to operate the various panel lights on index 0 – 2. This example is also showing you how to use slave datarefs. In this example we will manipulate the value at index 1 between the values of 0.0 and 1.0 (full brightness). Index values of 0 and 2 will be set to the same value as the primary dataref anytime the encoder is operated.



Multi-Encoder Example

A Dual Encoder can actually have more than two encoders although two is the most common use case. More than two encoders can be used for functions like adjusting each of the four transponder digits individually.

Take a look at the C172 sample transponder encoder definition on Layer B.



In this example we used the Add Encoder button to add Encoder 3 and Encoder 4 to the transponder dual encoder configuration. The primary encoder handles the 1000's digits, Secondary handles the 100's, the 3rd handles the 10's, and the 4th handles the ones's.

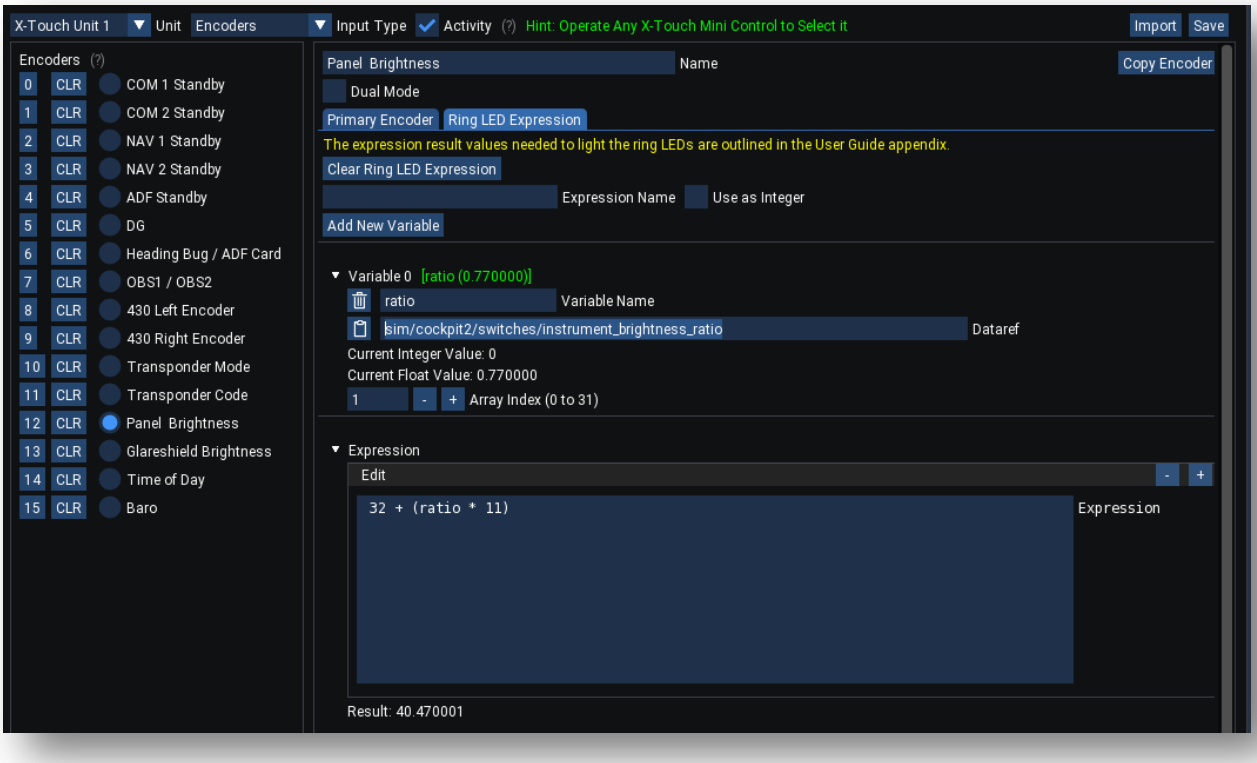
Each short press of the encoder's button will cycle between these four encoders allowing you to adjust each transponder digit individually. Also note that we set the Rest Time to 15 seconds. If no activity on the encoder is detected after 15 seconds the primary encoder will be made active.

Controlling the Encoder Ring Leds

The ring leds on an encoder can be manipulated using an expression. The ring leds have various numeric values associated with different ring patterns. The values and the ring patterns are outlined in the appendix of this User Guide.

An example of how to configure this is the ring leds on the panel brightness encoder. In this case we want the ring leds to progressively light as the brightness is turned up. For this pattern the values we need to use range between 32 and 43 for the fan pattern (see the appendix).

We can create an expression that will use the current value of the dataref, ***sim/cockpit2/switches/instrument_brightness_ratio[1],*** to generate a value between 32 and 43.



We first create the variable "ratio" that will be a reference to the panel brightness ratio at index position 1. This value will range between 0.0 and 1.0 depending on the current panel brightness.

We then create the math expression:

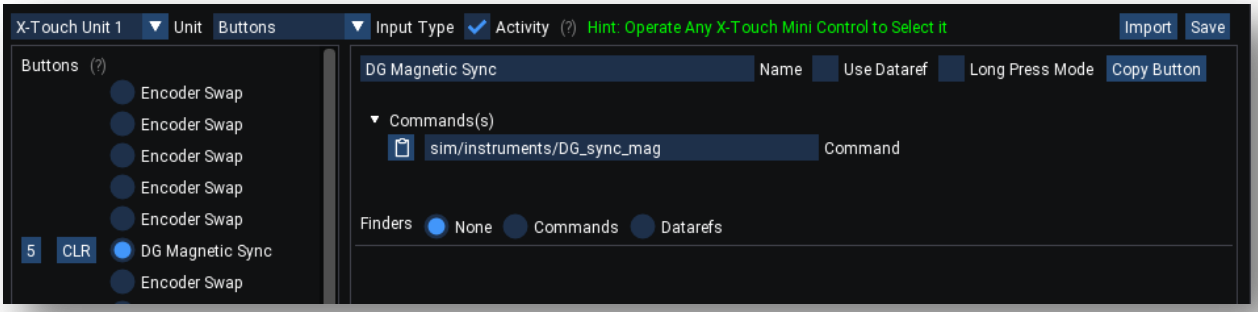
32 + (ratio * 11)

This will result in a value of 32 when the panel brightness is turned all the way down and a value of 43 when the panel is at full brightness.

Configuring Buttons

Each encoder has a center push button. There are also 16 led enabled push buttons. With both layers you will have 48 push buttons, 16 on the encoders and 32 led enabled buttons.

When an encoder is configured as a single encoder the encoder button will show up as one of the 48 configurable buttons. The DG encoder is a single mode encoder where its push button shows up as button #5 and it is configured to activate the command, ***sim/instruments/DG_sync_mag***, which will sync the DG to the magnetic compass heading.



Note: If the encoder is operating a dual mode you will not be able to enter a command definition for the encoder's button since the button is used exclusively to swap between the inner and outer knob definitions for the dual encoder or to actuate a command in long press mode which gets defined on the encoder definition.

Beacon Light Example

The buttons that have led support can also be associated with a dataref whose value can be used to determine the state of the led. Let's take a look at the Beacon light example:

In this example we are going to use the command, ***sim/lights/beacon_lights_toggle***, to operate the Beacon Light. Luckily there is an integer dataref named, ***sim/cockpit/electrical/beacon_lights_on***, that represents the state of the Beacon light. When it is a 1 the light is on and when it is a 0 the light is off.

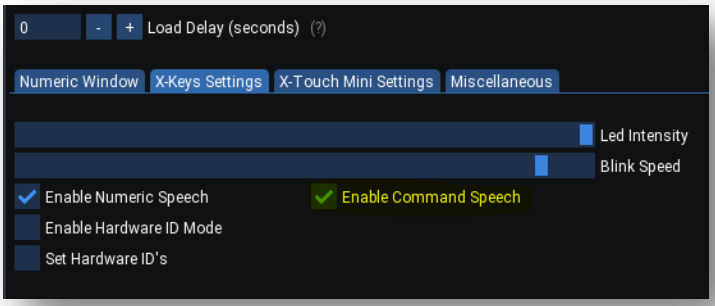
You can use the Not checkbox to invert the comparison. You can also use the Comparison dropdown to select the type of comparison you wish.

Also note the Add Command button. Additional commands can be configured so that when that one key is pressed multiple commands will be issued. This can be useful in aircraft where you want one button to turn on multiple items like dual fuel pumps or multiple lights.

Speech

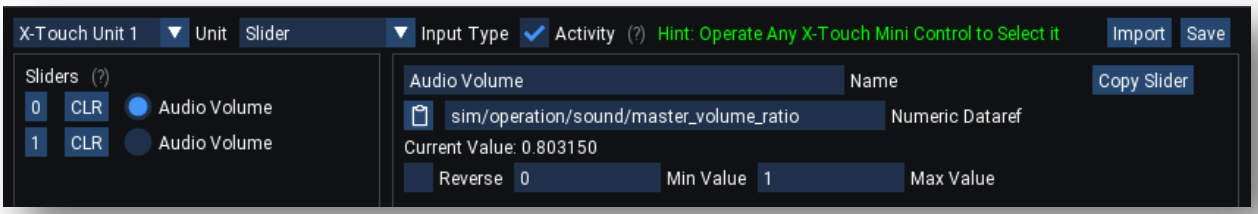
You can optionally add speech for the led on, off, and blinking conditions. When a button is actuated and defined speech will be processed by X-Plane text to speech engine.

Note: you can globally enable and disable command speech on the X-KeyPad settings dialog on the X-Keys Settings tab.



Configuring Sliders

Sliders are configured to manipulate the value of a writable dataref based on the position of the slider on the X-Touch Mini. Let's take a look at the Audio Volume slider configuration:



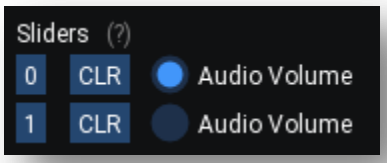
The X-Plane master audio volume can be controlled with the writable floating point dataref: ***sim/operation/sound/master_volume_ratio***. This dataref has a value between 0 and 1.0.

The slider is configured with these Min and Max values. As the slider is operated the dataref will be updated with a value between the min and max range based on the sliders position. The Reverse checkbox can be used to change whether the top of the slider is the Min or Max value.

You will note that we have the same configuration for both layer A and B. This is one of the limitations of the slider. Although you can define completely different layer configurations you end up with an unwanted side effect that causes the underlying dataref value to change to the current slider position as you swap between layer A and B.

Clear Button

The configuration for any input can be cleared using the CLR button:

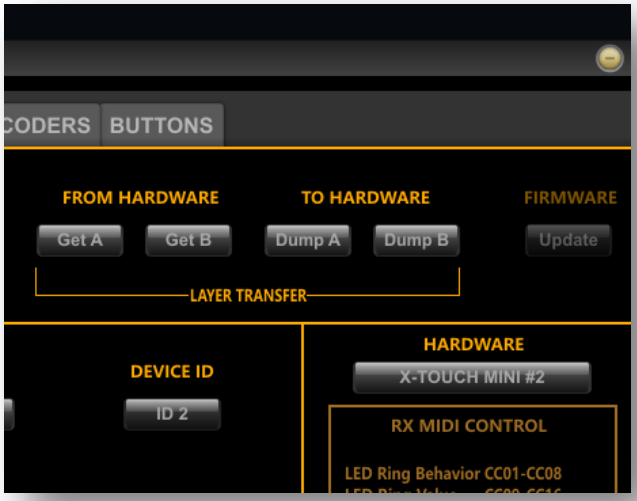


Once an input is cleared it will stop interacting with the simulator.

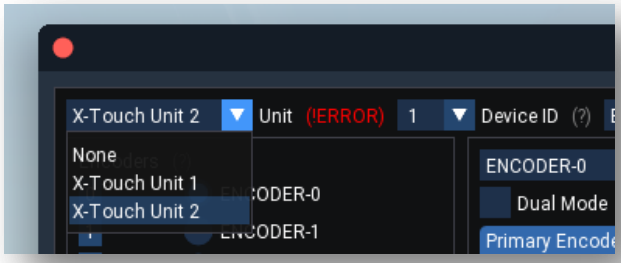
Using Multiple X-Touch Mini's

Unit numbers for multiple X-Touch Mini devices are assigned arbitrarily based on the order they are found on the USB buss. This can get very messy when you swap USB cables around or just through random chance the units are not detected in the same order consistently.

Each X-Touch device has an assignable device ID that can be used to identify and associated a physical unit to a corresponding json file. Behringer has a Windows only tool called the X-Touch editor that lets you change that assignment:

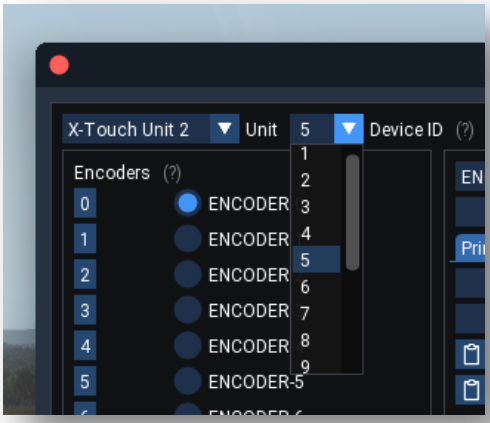


X-KeyPad also lets you assign an ID right from within the X-Touch Configuration editor. This is particularly convenient for OSX and Linux users who may not have access to a Windows system to run the Behringer editor. It is critical that Device ID's be unique when you have more than one X-Touch Unit connected. If you have duplicate ID's you will see an error in the X-KeyPad configuration editor:



Units with the same device ID will be loaded from a common json file.

You can change the Device ID assignment by simply using the Device ID dropdown next to the unit. The hardware will be updated as soon as you make and change and will persist in the devices firmware between X-Plane runs.



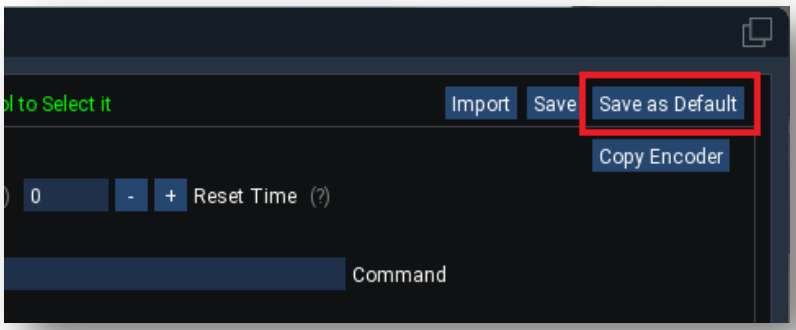
Note that the Device ID assignment does not need to match the unit number, as illustrated above; they just need to be unique. Also note that the order you see the units in the left dropdown is completely dependent on the order is which the operating system sees them on the USB buss so they can get flipped around in the list although the correct Json file name will follow the Device ID assigned.

Tip: If you started out with just one X-Touch device the json file name will use the -0 suffix. As soon as you add one or more subsequent devices X-KeyPad will start to use the Device ID as the suffix. This typically means that the first time you start X-Plane with more than one device the configurations will be blank as it will be looking to load json files that have a suffix of 1 – 16, it will not even attempt to load the original with the -0 suffix.

Simply select the unit you want to have your original configuration on, make sure it has a unique ID, and then use the Import button to locate and import the original -0 suffix json file and then do a save. A new json file will be created with a name using the Device ID suffix. Once everything is working you can manually delete the -0 json file as it will no longer be needed. The same process is necessary if you start changing device ID assignments after you saved configurations.

Saving a Default Configuration

If you chose to create a configuration that only uses commands and datarefs common to all aircraft you can use the Save as Default button to save the configuration to the X-KeyPad root folder.



If X-KeyPad does not find an X-Touch json file in the aircraft folder it will load the default from the X-KeyPad root folder if ones exists.

X-Touch Mini with FlyWithLua

One of the things you might consider doing to mounting your X-Touch Mini under a monitor where you could render data over each encoder showing what the encoder functions are. Since the encoders support dual mode it may be necessary to know how the encoder is configured and if it is dual mode is it currently on the primary configuration or the secondary configuration. The following datarefs can be used to drive your display logic.

SRS/X-KeyPad/XTouch_Last_Active_Layer[8]

This in an integer dataref that represents the active layer on the X-Touch Mini unit indicated by the array index. A value of 0 indicates that the X-Touch is on Layer A and 1 indicates that it is on Layer B.

SRS/X-KeyPad/XTouch_Encoder_Mode[128]

This dataref is a 128 integer array indicating the encoder mode for the eight encoders for bot layer A and B. Layer A is index 0 – 7 while Layer B encoders are indexes 8 – 15 on the first X-Touch unit. For multiple X-Touch units the array index is offset by the X-Touch unit number * 16 encoders. A possibility of 8 units each with 16 encoders gives you an array size of 128.

Here are the value definitions:

- 0 – Encoder is in Single
- 1 – Encoder is in Dual Mode and the primary definition is active
- 2 – Encoder is in Dual Mode and the secondary definition is active
- >2 More than two encoders have been defined and the number represents the current active encoder

You can see examples of using these datarefs with the X-Keys X-Touch Mini legend Virtual Device. They are used to drive the key layout and appearance based on the state of the X-Touch Mini

Keyboard Only Support

One of the legacy features of X-KeyPad versions prior to 1.5 was support for setting and manipulating datarefs using just a keyboard. It was a rarely used feature but support for it is still in X-KeyPad 1.5 but the configuration is still done with a CSV file.

X-KeyPad.csv File Format

X-KeyPad.csv is used to define the set of numeric datarefs that you wish to manipulate in response to directly executing one of the:

SRS/X-KeyPad/Set_Sim_Variable_1 through **SRS/X-KeyPad/Set_Sim_Variable_50**

or when a specified X-Plane command is detected. It can be used in conjunction with any game controller, keyboard, or keyboard emulator.

The X-KeyPad.csv file is a simple comma delimited file that can be edited with any text file editor. If possible we would recommend using an editing tool that specifically handles comma delimited files. Microsoft Excel, Google Sheets, and Open Office Sheets are such tools. There are also a number of free tools available that are specifically oriented to editing CSV files.

The first line of the file is the column headers. The remainder of the file can contain up to 50 rows defining 50 sim datarefs that can be set. X-KeyPad ships with a number of sample files. We are using the Samples\Keyboard Only\ X-KeyPad.csv for this discussion. We suggest you open that file now and follow along as we outline the file format.

The following table outlines the columns in each row.

Column	Data Type	Description
Name	Text	A descriptive name for the dataref. Note that you can place – in the front of the name to indicate that this line is commented out and won't be read.
Comment	Text	Some comments about the dataref. Note: don't use commas in here
Dataref Number	Integer	A number between 1 and 50 that maps to the X-KeyPad commands SRS/X-KeyPad/Set_Sim_Variable_1 through SRS/X-KeyPad/Set_Sim_Variable_50
Dataref	Text	The X-Plane writable dataref that will be set to the value collected in the numeric buffer when the X-KeyPad command Set_Sim_Variable_n is executed Specific elements of an array dataref are represented with a [n] at the end of the dataref name where n is the zero based index of element you want. See the Zibo 737 sample for examples.
Piggyback Command	Text	An optional X-Plane command that will be monitored for activity by X-KeyPad. See the section on piggyback commands for further

		detail
Multiply By	Number	A numeric factor that will be applied to the numeric buffer before setting the dataref. This is useful for datarefs like COM frequencies where it is more user friendly to enter frequencies in mega HZ but the dataref needs the data in 10K HZ units.
Min	Number	The minimum allowed value for the dataref
Max	Number	The maximum allowed value for the dataref
Increment	Text	If set to the word "WRAP" means the current value of the dataref will be incremented by 1 until greater than the max value and then it will be set to the min value. This is useful for enumerations such as the HSI selector. If set to the word "UP-DOWN" means the current value of the dataref will be incremented up to MAX and then it will be decremented down to MIN and then back up to MAX in a cycle. This tends to model a rotary switch or multi-position toggle switch with a fixed set of positions. If set to the word "DOWN-UP" incrementing will behave the same as UP-DOWN except that the first increment will be down. If set to the word "UP" means the current value will be incremented up to its max value. Further executions will not increment the value past its max setting. If set to the word "DOWN" means the current value will be decremented down to is min value. Further executions will not decrement the value below its min setting. If set to the word Set_n where n is an integer value then the dataref will be set the value of n. An example of this would be Set_1200 which could be used in combination with the transponder code dataref: **"sim/cockpit/radios/transponder_code"** To set the transponder to the VFR squawk code.
Numeric Speech	Text	One or more strings separated by ; that can be selected based on comparison logic. The selected string will be spoken by the simulator if the Numeric Speech setting is enabled.

Table 2

Loading the Numeric Buffer

The number buffer is filled when X-KeyPad detects the commands associated with the number buffer. As soon as one character is entered a floating window will appear showing you the numbers being

entered. The window will remain visible until you close it, clear the buffer with the clear command, or a command is executed to set a dataref.

The numeric window has a number of characteristics which can be adjusted using the Settings menu item. See the Settings section of this manual for more details.

Command Execution

When X-Keypad detects the execution the commands, Set_Sim_Variable_1 through Set_Sim_Variable_50, it will use the command number to find the corresponding entry in X-KeyPad_Datarefs.csv.

Using the included sample file you will notice that at line 10 we have defined dataref # 9 as referencing the barometer dataref. Assuming you mapped a button or key to SRS/X-KeyPad/Set_Sim_Variable_9 X-KeyPad will detect that command and will look at the data referenced on line 10.

If no data is present in the numeric buffer the command will be ignored.

When data exists in the numeric buffer it will be converted to a floating point value and will be set in the X-Plane dataref “sim/cockpit2/gauges/actuators/barometer_setting_in_hg_pilot”. As soon as this is done the numeric buffer will be cleared in preparation for the next entry.

Piggyback Commands

Piggyback commands allow you to use a keyboard key or joystick button for two purposes. Let’s take the XPlane command “sim/transponder/transponder_ident” as an example. This command would be mapped to a key or button that when activated would toggle transponder ident. Refer to line 8 of the sample file.

When X-KeyPad has a piggyback command specified it will constantly monitor and intercept any activation of that command. When it sees activation it will look to see if any numbers have been collected in the numeric buffer. If the buffer is empty X-KeyPad will allow the processing of the command to continue and the X-Plane transponder will start its IDENT cycle.

If numbers exist in the numeric buffer when the piggyback command is intercepted then the specified dataref will be set to that value and the piggyback command will stop execution and not be passed on to X-Plane.

Note:

- Whenever you use a piggyback command your key or joystick button should be mapped to the piggyback command **NOT** the SRS/X-KeyPad/Set_Sim_Variable_x command.

- If you are not using a piggyback command then your key or joystick button should be mapped to the SRS/X-KeyPad/Set_Sim_Variable_x where x is the Dataref Number in column three of the dataref CSV file.

Advanced Dataref Manipulation

There are more advanced capabilities that can be used to control numeric datarefs:

- Multiply by a factor before storing
- Incrementing
- Min and Max testing
- Using a set of shared datarefs to interact with Lua scripts
- Selecting strings that can be spoken using the simulator's built-in text to speech capabilities

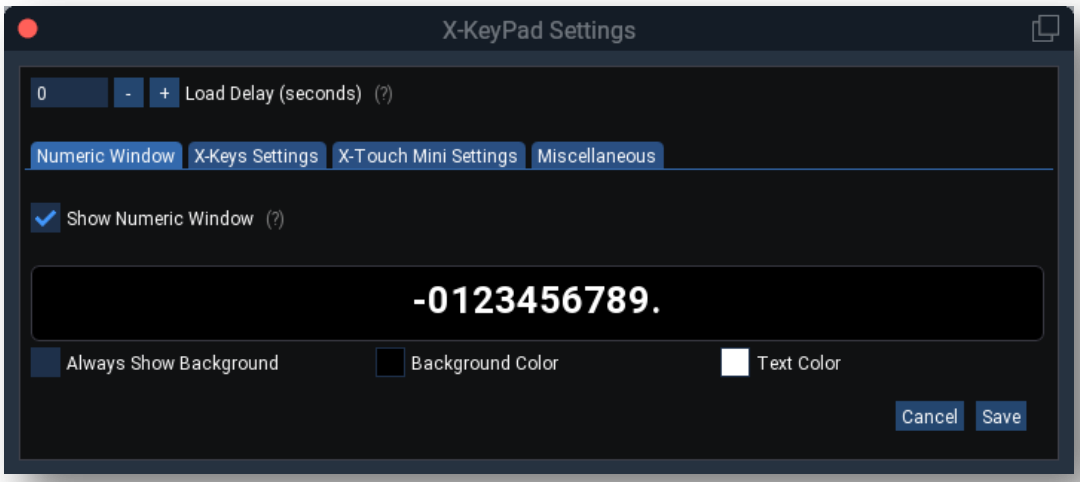
Making Changes to the File

X-KeyPad will read either the aircraft specific version or default version of X-KeyPad.csv whenever a plane is loaded or if you request a refresh from the menu item plugins/X-KeyPad/ Reload Configuration Files.

Settings

There are a number of settings which can be adjusted using the Settings menu item.

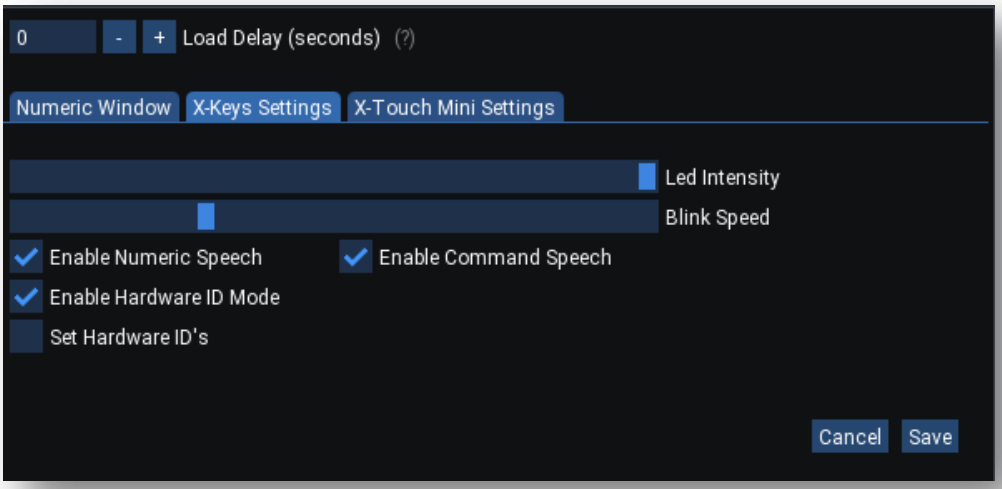
- Configuration Load Delay
- Data Entry Window background and text color
- Ability to have a translucent or transparent background on the Data Entry Window
- Device Led flash frequency
- Device Led brightness
- Numeric Speech
- Command Speech
- Hardware ID Mode (For use with multiple physical X-Key devices)
- X-Touch Dual Encoder long press delay
- Enable / Disable X-Touch support
- Leave the X-Touch and Mackie mode on exit
- Enable / Disable Auto Save



The figure above shows the settings dialog for the numeric window.

- Load Delay – Some aircraft need a few flight-loop cycles to fully create and define their custom commands and datarefs. You can adjust this value to delay the loading of the configurations so that X-KeyPad does not report errors about missing commands and datarefs before they were created.
- Show Numeric Window – By default X-KeyPad will display any numeric data you enter into the numeric buffer in a floating window on your X-Plane display. If you are using Virtual Devices it is possible to display that data on a key, like we did with the CLR key on the C172 Virtual Device sample. In that situation you may prefer to uncheck this box so that numeric buffer data will not be displayed on your X-Plane display.
- Sample Area – This shows what the color impact will be on the data entry window
- Background and Foreground color pickers will allow you to choose the colors for the numeric buffer window.
- Always Show Background – When set the numeric data entry window will always have a translucent or solid background. When not set the background is transparent and only the numeric data will show when there is data in the number buffer.

The following image shows the possible settings on the X-Keys tab.



- Blink Speed – Adjusts the blink speed of the X-Keys Leds and between Virtual Device key font colors if more than one color was specified
- Brightness – Adjusts the intensity of the X-Keys backlighting
- Enable Numeric Speech will use the simulator's text to speech function to announce the numeric value being set in the dataref as specified by the Numeric Speech column in the X-Keys.csv or X-KeyPad.csv files
- Enable Command Speech will use the simulator's text to speech function to announce any command speech that was specified in the currently loaded X-Keys.csv file.

The Led Blink Speed and Led Brightness can also be adjusted using two integer data refs:

SRS/X-KeyPad/LedBrightness
SRS/X-KeyPad/LedBlinkSpeed

The valid values for these datarefs are 0 – 255.

Working With Multiple X-Key Devices

When you have more than one physical X-Key device connected to your system, X-KeyPad will assign the unit ID's sequentially from 0 – 7 in the order that it finds the devices on your USB controllers. If you only have one device connected it will always be assigned unit ID 0.

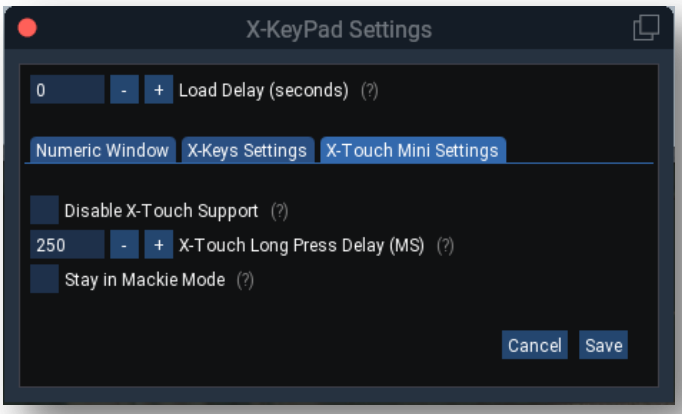
One potential issue with this approach is that the unit ID's assigned can change if you change the USB ports your X-Key devices are plugged into, thereby forcing you to change the unit IDs in your configuration files.

The "Enable Hardware ID Mode" allows you to assign a hardware ID between 0 and 7 to each of your X-Key devices and then use that ID as the unit number in your configuration files. This allows you to plug

your devices into any USB port or hub without the need to change the unit numbers in your configuration files.

Check the Set Hardware ID's checkbox to set the assigned ID's on your devices.

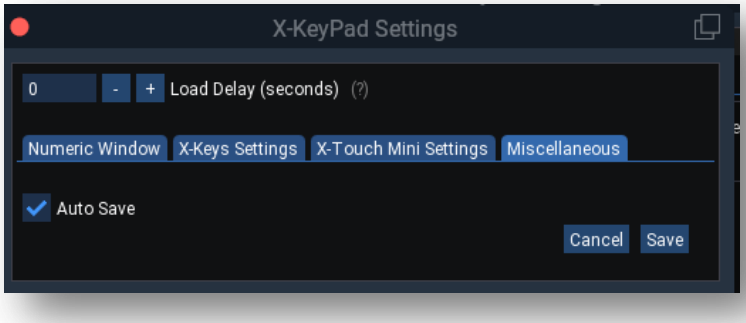
The following image shows the X-Touch settings.



- Disable X-Touch Support – If you have X-Touch Mini devices on your computer but you are using them with something other than X-KeyPad you can enable this option to prevent X-KeyPad from attaching to the devices.
- Long Press Delay – When using the Dual Encoder button long press mode this value determines how long you need to hold down the encoder button before X-KeyPad will detect the action as a long press.
- Stay in Mackie Mode – X-KeyPad puts the X-Touch Mini in Mackie mode so that it has greater control over the devices operation. By default, X-KeyPad will put the device into standard MIDI mode when you exit X-Plane. Check this box of you prefer that your X-Touch mini remaining in Mackie mode when X-Plane exits.

Auto Save

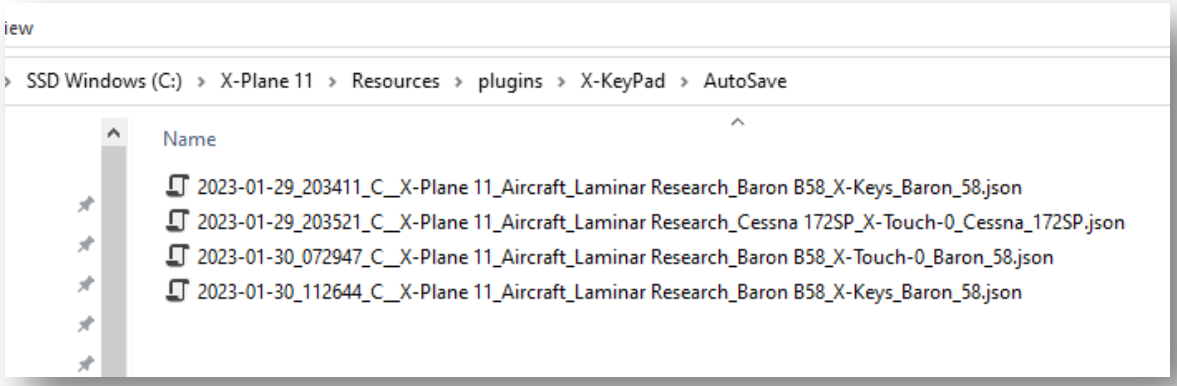
Auto Save is enabled by default.



When Auto Save is enabled X-KeyPad will save a time-stamped copy of your aircraft X-Keys, Stream Deck, and / or X-Touch json configuration any time you change aircraft or exit X-Plane assuming that one of the editor windows was opened while that aircraft was loaded.

If you never open either the X-Keys configuration editor or X-Touch configuration editor, X-KeyPad will assume you did not make any changes to the configuration and will not create a time-stamped copy.

The time-stamped copy is saved to the X-KeyPad/AutoSave folder. The name of the json file will have the date and time as the first part of the file name with the remaining portion of the file name representing the full path of the aircraft configuration.



Should you ever forget to save a configuration or wish to retrieve an earlier version you can use the clear and append file operations to select a backup from the AutoSave folder.

There are many free text file tools that can be used to compare two json files to see what the difference is between them.

https://www.diffchecker.com/

These can be used to determine what changes you may have made to various versions.

Keep in mind that if you are doing a lot of editing the AutoSave folder can contain a fair number of versions over time.

AutoSave will only auto save the aircraft the definition. The X-Keys configuration editor allows you to have multiple configuration files open at one time. Only the primary aircraft configuration will be auto saved.

Stream Deck Devices

X-KeyPad also supports an interface to the Elgato Stream Deck Devices.



The devices feature tactile buttons that can display images and/or text. In the case if the Stream Deck Plus it also has an additional four rotary encoders with a display panel above each encoder that can also display images and/or text.

Integration Overview

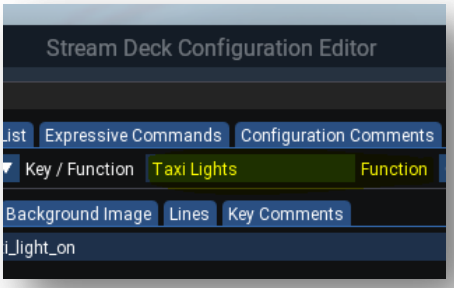
X-KeyPad interfaces with the Stream Deck devices using the Stream Deck SDK. This allows you to add buttons and dial operations to your Stream Deck device that can control X-Plane while still being able to have other buttons operate functions outside if X-Plane. This approach also allows you to use all the built-in Stream Deck organizing features like profile, pages, and folders to layout your interface.

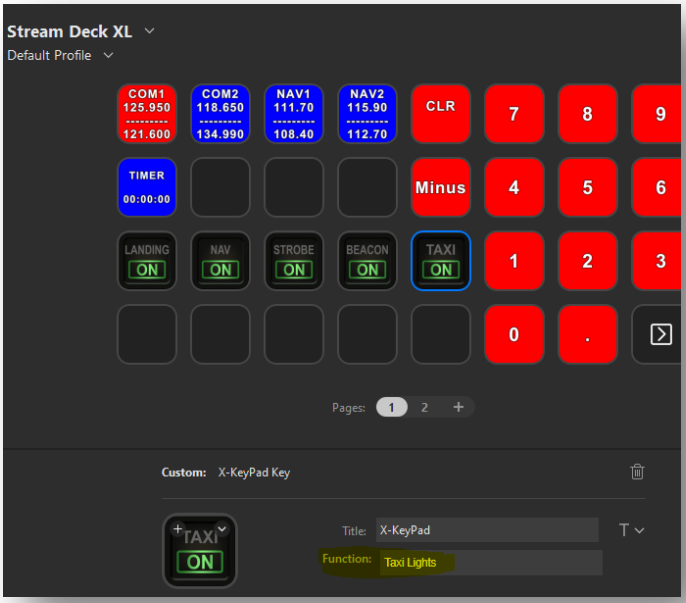
There are some limitations using this interface.

- There is no support on Linux as the Stream Deck software is not available on the Linux OS.
- Button text color, font, size, and alignment are controlled on the Stream Deck. This means that X-KeyPad can change the text contents dynamically but not the color, font, size, or alignment.

X-KeyPad is not really aware of the layout of the keys on the device. The X-KeyPad configurations basically defines what will happen in the simulator when an associated key or dial is operated and what text and/or image should be displayed on the key or dial LCD panel based on the state of datarefs in X-Plane. The key / dial layout is completely controlled by how you setup your Stream Deck profiles, pages, and folders.

The association of the key/dial instance on a Stream Deck device is made through the use of a unique function name for the key/encoder definition in X-KeyPad and setting that function name property on the Stream Deck key's instance.





This tutorial video has an overview of the Stream Deck devices and covers the concepts of the interface.

https://youtu.be/hiofS9Xg-xI

Limitations

- Support for Stream Deck devices is not available on Linux since Elgato does not offer a version of its software for Linux
- When using background images for the Stream Deck only PNG format files are supported

Comparison to X-Touch Mini and Virtual Devices

Creating a configuration for a Stream Deck device is very similar to creating configurations for a Virtual Device or X-Touch Mini. For this reason this section of the user guide will not cover the intricate details of commands, logic datarefs, numeric datarfs, logic test, and expressions as they are the same with the Stream Deck as they are for those other devices.

The primary differences revolve around the functional capabilities of the devices

- The Virtual Devices have the ability to change the font, size, and color on a line by line basis for each key. Stream Deck keys can only have one font, size, and color per key instance. You can compensate for this to some degree on Stream Deck keys by using multiple images with the font details drawn on the image.
- The X-Touch Mini Encoders have no way to display information near the encoder with the exception of the ring LED's. On the other hand the Stream Deck Plus dials have a small touch enabled LCD panel above each dial that allows you to display data. That touch enabled panel

also allows you to define extra actions on either a dial short/long press or the touch panel short/long touch.

Installing the Stream Deck X-KeyPad Plugin

In order for X-KeyPad to interact with the Stream Deck devices you must install the X-KeyPad Stream Deck plugin into the Stream Deck application. Make sure you have the Stream Deck application installed first:

https://www.elgato.com/en/downloads

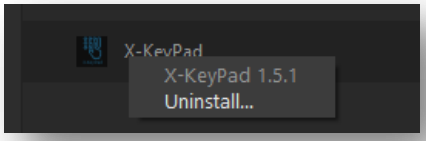
The plugin file can be found here:

X-Plane 11-12/Resources/plugins/X-KeyPad/SD-Plugin/com.stickandrudderstudios.x-keypad.streamDeckPlugin

After the Stream Deck application is running you can use Windows Explorer or OSX Finder to navigate to the X-KeyPad/SD-Plugin folder and then double-click the streamDeckPlugin file. That should install the plugin. This part of a tutorial video outlines this process.

https://youtu.be/hiofS9Xg-xI?t=792

Note: The Stream Deck plugin and the X-KeyPad X-Plane plugin communicate with each other using a protocol API. Both plugins need to be using the same version of the protocol. The X-KeyPad plugin will verify that the versions match when it connects to the Stream Deck software. If they do not match you will get an error message. Should this occur simply uninstall the Stream Deck Plugin by right clicking the action. Then reinstall the matching plugin using the procedure outlined above.



Stream Deck Sample

We will be using the generic Stream Deck sample for the following discussions.

X-Plane 11-12/Resources/plugins/X-KeyPad/Samples/Stream Deck/X-KeyPad-SD_Generic.json

You can import the generic configuration by following these steps.

1. Load the default C172 aircraft which is what we will use for the examples
2. Use the X-KeyPad menu to open the Stream Deck Configuration Editor
3. Use the File menu on the editor to do an Append operation
4. Use the File Selector to choose the Generic Sample in the path above
5. Click Open

The following tutorial video at the specific time stamp shows you how to import the sample.

https://youtu.be/hiofS9Xg-xI?t=901

Append vs Selective Append

When you append a configuration, X-KeyPad will replace any existing key/encoder in the configuration that has the exact same function name with that definition found in the file being appended. If that function name does not exist in the existing configuration then it will be added to a free slot.

Selective Append gives you a bit more control over what to do when a function name already exists in the configuration. On a function by function basis you can choose to have it replace the existing function or append it with a new name so that both the original definition and new definition will both be available to experiment with.

Binding Keys to Functions

The process of adding keys to your Stream Deck devices involves dragging the X-KeyPad action and dropping it on the Stream Deck key/encoder location where you want an X-Plane function. You then use the property inspector to enter the matching function name on your X-KeyPad Stream Deck configuration. The function name is case and space sensitive so make sure they match. The following video at the specific time stamp illustrates that process:

https://youtu.be/hiofS9Xg-xI?t=1256

Creating a New Key

New keys are created by finding an unused key slot and giving the key function a new unique name. You can then define the commands, background image and text lines that you want to have displayed on the key. The following tutorial video at the specific time stamp shows you how to create a new key:

https://youtu.be/hiofS9Xg-xI?t=1777

Encoders

A stream Deck encoder can be configured as either a single encoder or a Multi-Encoder. A Multi-Encoder can have more than one definition. You can switch between those definitions by doing a short press in the dial. A common use for a Multi-Encoder is to emulate a Dual Concentric Rotary encoder that is very common in real-world aircraft. Examples of a Dual Concentric encoder are:

- Radio Tuning Controls
- Garmin 430/530/G1000 Page and Chapter control
- Some airliner MCP controls

Defining an encoder involves configuring the following parameters:

- Which commands should be executed when the dial is turned clockwise or counter clockwise

- Optionally configure how a writable dataref will be incremented / decremented when the dial is turned
- Defining the command that should be executed when the dial is pressed or when using a Multi-Encoder what command should be executed on a long press
- What background images should be shown on the encoder LCD panel
- What text should be shown on the LCD panel
- Defining the command that should be executed when the LCD panel is touched
- Optionally configure an advanced layout for the LCD panel

The following tutorial video at this time stamp will show you how to define encoders.

https://youtu.be/hiofS9Xg-xI?t=2372

Encoder Layouts

Encoder Layouts give you a way to control what is displayed on the small touch panel above the Stream Deck Plus dials. By default you can specify a background image and up to four lines of text using a white, center aligned font.

You can define more complex layouts that allow you to:

- Control the font face, color, and size
- Position the image
- Display progress bars

The following encoder definitions in the sample are good examples of using Layouts:

- XPDR
- BARO
- AUDIO VOLUME

The following video will show you how to use Layouts.

https://youtu.be/hiofS9Xg-xI?t=2990

The Stream Deck SDK documentation contains greater detail on how layouts work. That said, it is very important to remember that the way X-KeyPad sends data to a custom layout is through the use of the line and value key names as out line in the tutorial video above.

https://developer.elgato.com/documentation/stream-deck/sdk/layouts/

Expressive Commands

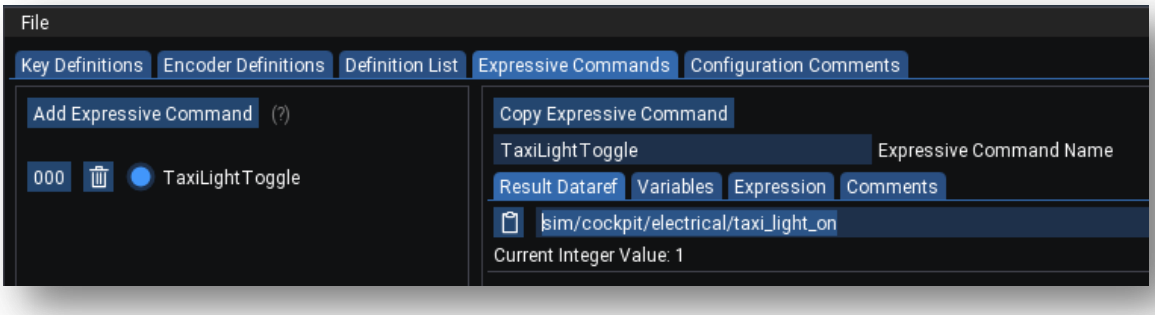
Expressive commands have very similar capabilities as global formulas with the primary difference being that they are evaluated only when a key associated with the expressive command is pressed. On the other hand, formulas are evaluated constantly on every flight loop.
Expressive Commands are useful when you have to control an aircraft function when a key is pressed but there is no X-Plane / aircraft command available to operate the function. I most cases there will be a writable dataref that can be used to operate the function.

We have created an Expressive Command named TaxiLight Toggle in the sample that demonstrates the concept. The use case is somewhat made up because X-Plane has a Taxi Light toggle command but the simplicity of the sample is helpful to get the concept across.

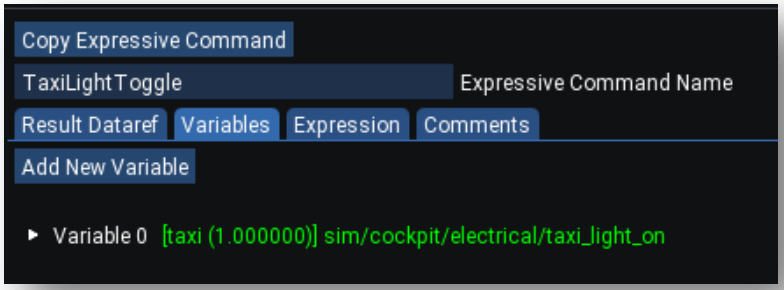
The sample will evaluate an expression whenever a key associated with that expressive command is pressed. The output of the expression will be written to the dataref:

sim/cockpit/electrical/taxi_light_on

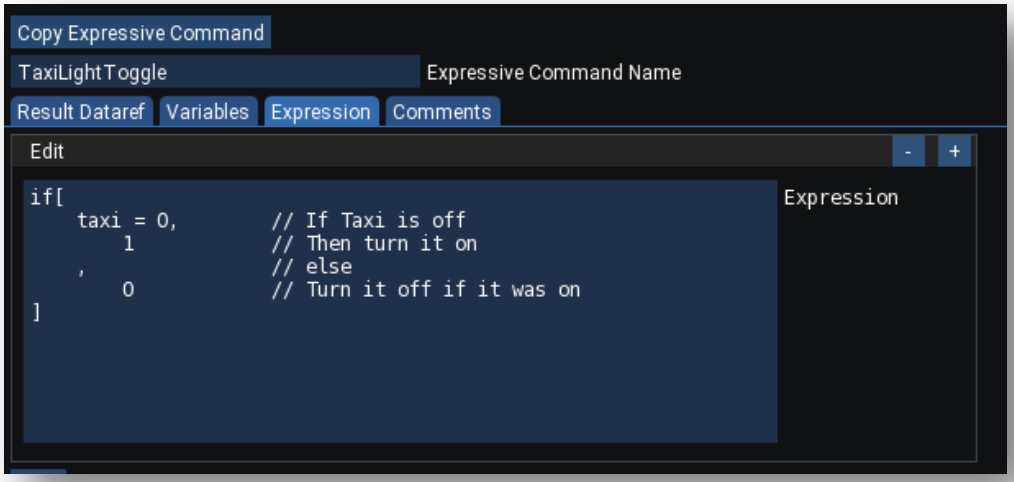
This dataref controls the state of the Taxi Lights, 0 means off, 1 means on.



The expression will also use that same dataref as one of its variables.



When the key associated with this expressive command is pressed a simple evaluation will be performed and the result will be written to the taxi light dataref. If the taxi light is off then return 1 which will turn it on and if it was not off (meaning it was on) then return 0 which will shut it off.



The following tutorial video will demonstrate the use of this sample.

https://youtu.be/hiofS9Xg-xI?t=3792

Sharing Configurations

It can take considerable effort to create X-KeyPad configurations. If you are new to X-KeyPad we recommend that you browse the shared configurations on X-Plane.org to see if someone has created a configuration for the aircraft you are interested in.

We also encourage you to share your completed configuration with the community. Here are some best practices for doing that:

- Use the comment fields on keys, encoders, expressions, etc. to document what you are doing in the configuration.
- If possible avoid including copyrighted material such as purchased images that may limit your ability to share the configuration.
- If you are using Lua scripts or if your configuration is relying on other plugins make sure you document those requirements and any additional installation steps.
- If you plan to share icons / images in an icon package try to prefix the PNG file names with something unique, possibly your initials, MXE-Taxi-On.png. This allows users to install your ICON pack in the images folder without file name conflicts.

Here is the link to the download forum in X-Plane.org where you can share your configurations.

https://forums.x-plane.org/index.php?/files/category/187-x-keypad-configurations/

Tips and Other References

Commands vs Datarefs

It is easy to get confused by the difference between X-Plane's commands vs datarefs. Here is a good description from the X-Plane developer's blog.

https://developer.x-plane.com/2009/04/datarefs-vs-commands-i-whats-the-difference/

Dataref Tool

You may be familiar with a popular tool called the Dataref Tool. This amazing plugin really helps to find which datarefs and commands impact the simulator. It is also invaluable if you are trying to get X-KeyPad to work with any custom planes that may very well use a set of custom datarefs and commands.

https://datareftool.com/

Bitwise AND Operations

If you have to work with bit masks you may find this online binary converter valuable.

https://www.mathsisfun.com/binary-decimal-hexadecimal-converter.html

FlyWithLua

FlyWithLua is a powerful scripting language that can be used to dramatically extend X-KeyPad's ability. Programing knowledge is required but there are a lot of examples.

https://forums.x-plane.org/index.php?/files/file/38445-flywithlua-ng-next-generation-edition-for-x-plane-11-win-lin-mac/

XSaitekPanels

X-KeyPad and XSaitekPanels have a lot in common. If you are trying to get a custom complex plane configured for X-KeyPad you may often be able to look at the Lua scripts and XSaitekPanel configurations for ideas.

https://forums.x-plane.org/index.php?/files/category/182-xsaitekpanel-configurations/

Hardware vs. Software Mode

Running your X-Key device in hardware mode without using MacroWorks is preferred. We also recommend setting your X-Keys device to the factory default with all key definitions blank and then specifying your command execution in the X-Keys.csv file.

Once your X-Key device is configured you can bring it to any PC and it will work without the need to install any other software. We recommend not keeping Macroworks on your system and just use the X-Keys Basic setup utility to configure your device. You can download it from here:

http://xkeys.com/PISupport/SoftwareMacroWorksHW.php

Note: MacroWorks may be necessary to reset your device to factory defaults but it won't be needed after that unless you want to define complex macros.

Background Images

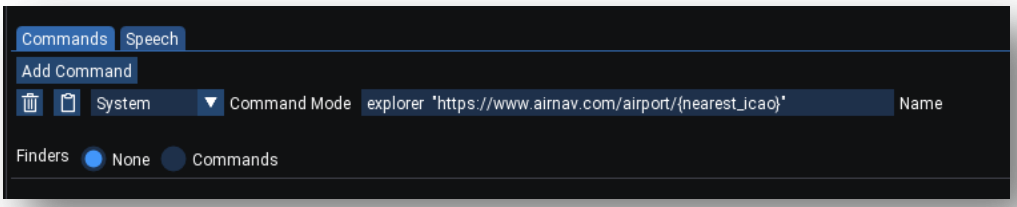
You can use background images for both the Virtual Devices as well as the Stream Deck devices. You can create your own PNG files using tools like Gimp or Photo Shop. You can also purchase images from companies like sideshowfx.

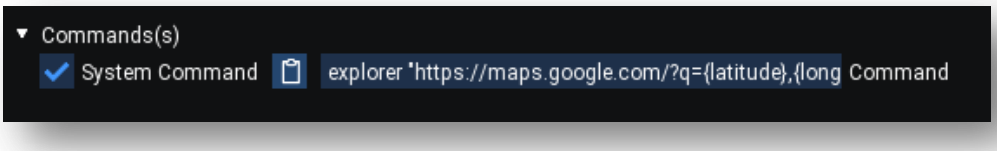
https://www.sideshowfx.net/flight-sim-icons

If you do choose to purchase an image collection make sure you follow and copyright restrictions if you plan to share your configuration with the community.

Running System Commands

Commands on either an X-Keys configuration, Stream Deck configuration, or X-Touch Mini configuration can optionally be system commands rather than X-Plane commands. You can select the command type by either a drop down or a check box:





The command format is similar to what you would see if running the command from a DOS box or a unix terminal. The first part is the program to execute and the latter parts are the command arguments. The program must be able to be found in your environment PATH. If it is not then you will need to specify the full path to the executable.

The normal convention is that the program name and path and the arguments are delineated by a space. In many cases the path might have an embedded space in it as can also be the case for command line arguments. When you have that situation you should enclose the path or argument with quotes like:

“C:\Program Files (x86)\Google\Chrome\Application\chrome.exe” "https://maps.google.com/?q={latitude},{longitude}"

On Linux and OSX X-KeyPad uses the C++ system() function to execute the command. The function will block further execution of X-Plane until the command completes. Some commands, like opening a browser session may very well open the browser and then return immediately. Other commands may not exit until they are closed. In the latter case you can use the & character at the end of the command to tell the OS shell to run the command in the background so the system() call will return right away allowing X-Plane to continue its execution.

There are three token strings that will automatically be replaced with current values.

Token	Value
{latitude}	Current numeric latitude of your aircraft
{longitude}	Current numeric longitude of your aircraft
{nearest_icao}	The ICAO code of the nearest airport

These can be used when opening browser windows to show a map of the airport or get airport information. Here are two examples of using them:

explorer "https://maps.google.com/?q={latitude},{longitude}"
explorer "https://www.airnav.com/airport/{nearest_icao}"

In the two examples above the tokens will be replaced by actual values before the command is executed.

Support

You can obtain support for X-KeyPad by visiting our support page:

https://stickandrudderstudios.com/support/

or the support forum on X-Plane.org:

https://forums.x-plane.org/index.php?/forums/forum/435-x-keypad/

We also have a Discord server that allows for voice communication and screen sharing:

https://discord.gg/KBDYEMUAsz

You can also contact us using the contact form:

https://www.stickandrudderstudios.com/contact-us/

Appendix

Defined Commands

X-KeyPad defines a number of commands that are used to control the plugin. The commands and their purpose have been discussed in previous sections of this user guide. The following table is an aggregated list of all those commands.

Command	Purpose
SRS/X-KeyPad/Keypad_0	Adds a 0 character to the numeric buffer
SRS/X-KeyPad/Keypad_1	Adds a 1 character to the numeric buffer
SRS/X-KeyPad/Keypad_2	Adds a 2 character to the numeric buffer
SRS/X-KeyPad/Keypad_3	Adds a 3 character to the numeric buffer
SRS/X-KeyPad/Keypad_4	Adds a 4 character to the numeric buffer
SRS/X-KeyPad/Keypad_5	Adds a 5 character to the numeric buffer
SRS/X-KeyPad/Keypad_6	Adds a 6 character to the numeric buffer
SRS/X-KeyPad/Keypad_7	Adds a 7 character to the numeric buffer
SRS/X-KeyPad/Keypad_8	Adds a 8 character to the numeric buffer
SRS/X-KeyPad/Keypad_9	Adds a 9 character to the numeric buffer
SRS/X-KeyPad/Keypad_Decimal	Adds a . character to the numeric buffer
SRS/X-KeyPad/Keypad_Minus	Adds a - character to the numeric buffer
SRS/X-KeyPad/Keypad_Clear	Clears the numeric buffer
SRS/X-KeyPad/Keypad_BackSpace	Clears the last character entered in the numeric buffer
SRS/X-KeyPad/Toggle_Shift	Toggles X-Keys global shift mode on and off. All units will be placed in the same global shift state
SRS/X-KeyPad/Momentary_Shift	Momentarily toggles X-Keys global shift mode. It will toggle again as soon as the key or joystick button is released. This is a global unit command, all units will be shifted together
SRS/X-KeyPad/Reload_Configuration	Reloads the configuration file. Same as using the menu option
SRS/X-KeyPad/Virtual_Device_Next_Unit	Advances all virtual devices to the next unit
SRS/X-KeyPad/Virtual_Device_Previous_Unit	Retreats all virtual devices to the previous unit
SRS/X-KeyPad/Virtual_Device_Next_n	Advances the virtual device n (0 -7) to the next unit
SRS/X-KeyPad/Virtual_Device_Previous_0	Retreats the virtual device n (0 -7) to the previous unit
SRS/X-KeyPad/Toggle_XKeys_Editor	Toggles the visibility of the X-Keys configuration Editor
SRS/X-KeyPad/Toggle_XTouch_Editor	Toggles the visibility of the X-Touch Mini configuration Editor
SRS/X-KeyPad/Toggle_Shift_Unit_0 Through SRS/X-KeyPad/Toggle_Shift_Unit_7	Toggles the shift state of individual units

Defined Datarefs

X-KeyPad defines a number of datarefs that are used to control the plugin and to interface to FlyWithLua scrips. The datarefs and their purpose have been discussed in previous sections of this user guide. The following table is an aggregated list of all those datarefs.

Name	Type	Comments
SRS/X-KeyPad/LedBrightness	Integer	A value between 0 and 255 which indicates the Led brightness level on all X-Keys units
SRS/X-KeyPad/LedBlinkSpeed	Integer	A value between 0 and 255 which indicates the Led flash speed on all X-Keys units
SRS/X-KeyPad/LedBlinkState	Integer	A value that alternates between 0 and 1 at the led blink rate
SRS/X-KeyPad/CommandSpeechEnabled	Integer	A 1 indicates that command speech is enabled on the settings dialog, 0 indicates that it is disabled
SRS/X-KeyPad/SharedInt	Integer Array[0-49]	Array of 50 integers that are shared between X-KeyPad and FlyWithLua
SRS/X-KeyPad/SharedFloat	Float Array[0-49]	Array of 50 floating point numbers that are shared between X-KeyPad and FlyWithLua
SRS/X-KeyPad/Virtual_Device_Selected_Unit	Integer Array[0-7]	-1 = The virtual device at that index is missing or is not open 0-7 = The current selected unit number of the virtual device at that index not shifted. 8-15 = The current selected unit number (Plus 8) of the virtual device at that index shifted. Note: Bit 3 of the integer indicates if the unit is shifted or not. Bits 0-2 represent the unit number.
SRS/X-KeyPad/UnitSelector	Integer	A value between 0 and 7 indicating which X-Keys unit you are addressing for all the following datarefs in this table
SRS/X-KeyPad/current_key_states	Read Only Integer Array [0-127]	A one indicates that the key is currently pressed
SRS/X-KeyPad/key_pressed_flags	Integer Array[0-127]	A one indicates that the key pas pressed. You must clear it in your script after process the key press
SRS/X-KeyPad/blue_leds	Integer Array[0-127]	0 = Blue led off 1 = Blue led on 2 = Blue led flashing
SRS/X-KeyPad/red_leds	Integer Array[0-127]	0 = Red led off 1 = Red led on 2 = Red led flashing
SRS/X-KeyPad/Unit_Shift_State	Integer Array[0-7]	A writable array of 7 integers that

	represent the shift states of units 0 – 7. 0 = not shifted 1 = shifted

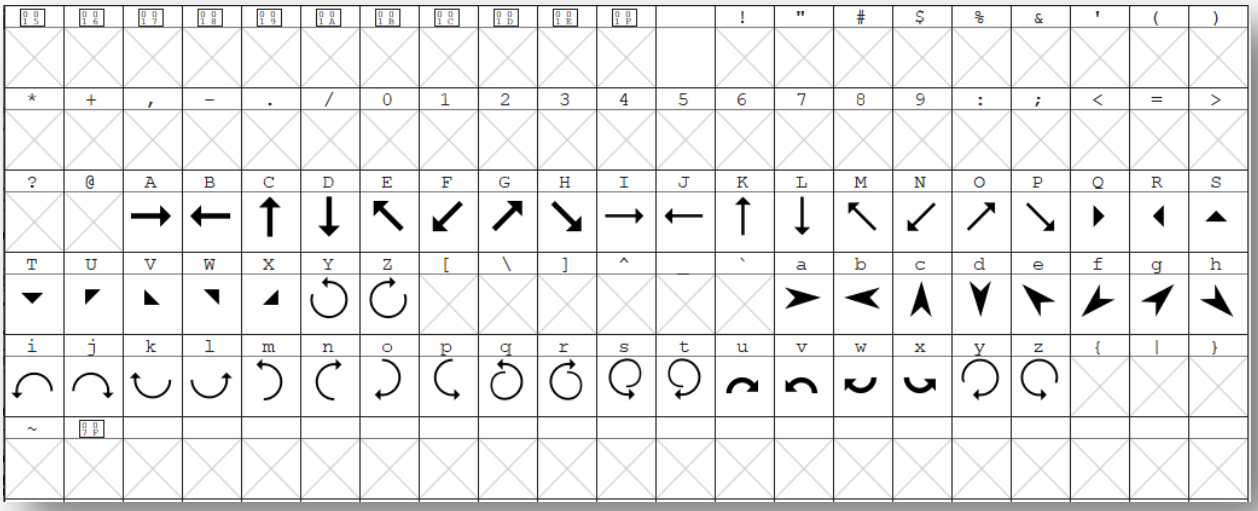
SRS/X-KeyPad/Virtual_Device_Selected_Unit is a writable and can be used with the numeric dataref set mode or with an expressive command to select the current unit and desired shift state on a specific virtual device.

Font Maps

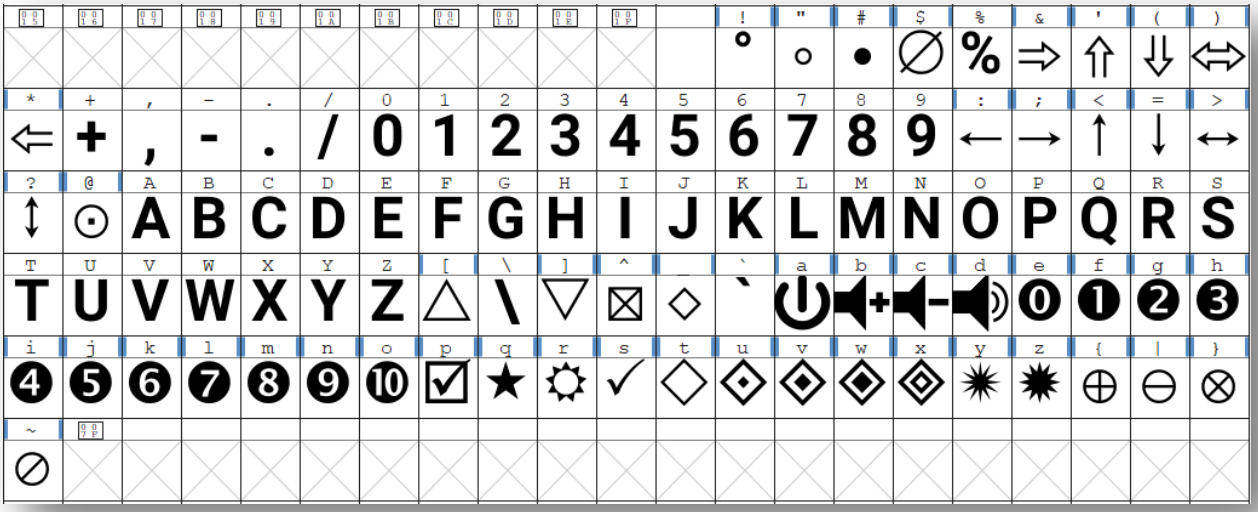
The virtual devices have four fonts where certain character glyphs have been replaced by symbols. In order to know what glyph will be rendered you need to know what character maps to the desired glyph. The virtual device options menu has a “Show Font Sample” entry that will open a pop-up window where you can select a font and then type sample text to see what glyphs will be rendered.

The following images also show the glyph maps for the four fonts. The character over the glyph is the character you must enter in the line editor to get that glyph.

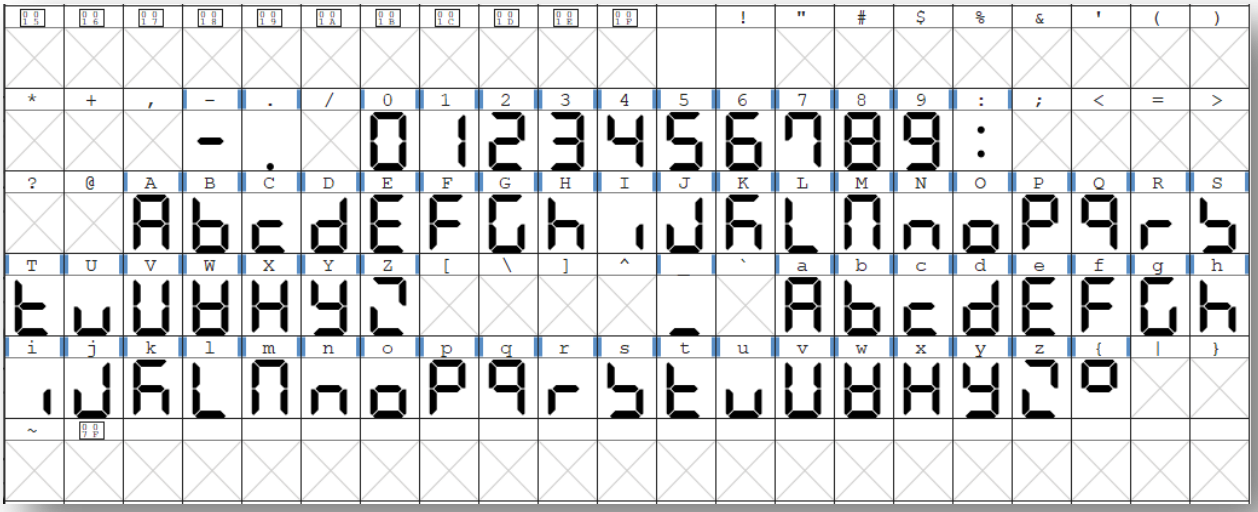
Arrow Font



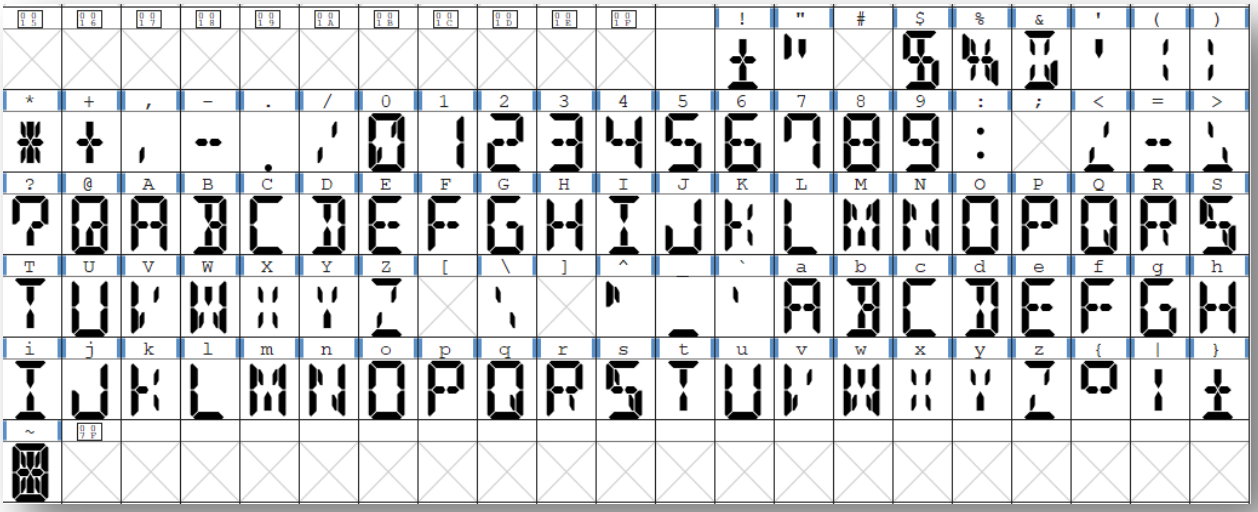
Combined Symbol Font



7-Segment Font

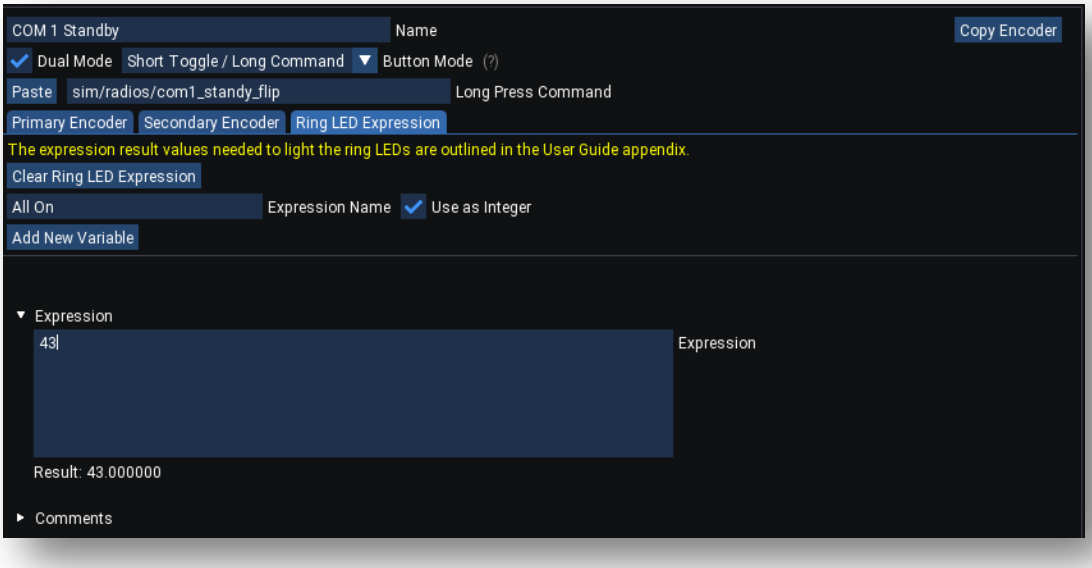


14-Segment Font



X-Touch Mini Encoder Led Ring Values

The X-Touch Mini encoder ring leds can be controlled by either writing a value to the associated dataref for that ring or by defining the ring led expression on the encoder definition. The following simplistic ring led expression will always return a value of 43 which will light all the ring leds on that encoder using the Fan mode value:



There are 11 leds on the rings and the values can represent four modes:

- Single Led Lit Values 0 -11
- Trim Values 16 - 27
- Fan Values 32 - 43
- Spread Values 48 - 59

Single and Fan are the mostly like modes you will use.

The dataref you write to is:

SRS/X-KeyPad/XTouch_Encoder_Leds[128]

This dataref is an array of 128 integers. This supports 8 X-Touch Mini units each with 16 encoders, 8 for each layer. Index 0 – 15 is for the 16 encoders on X-Touch unit 0 and index 16 – 31 is for the 16 encoders in X-Touch unit 1 and so on.

You can use the dataref tool to experiment with writing values to this dataref to see how the leds will behave. You could also experiment with it using the X-Touch Mini. The following image shows how to test the Fan mode on encoder 15, the right most encoder on layer B on X-Touch Unit 0.

The following tables illustrate the four modes and the values needed to get the indicated led pattern.

Single Mode

Value	Led lit indicated by *
0	
1	*
2	*
3	*
4	*
5	*
6	*
7	*
8	*
9	*
10	*
11	*

Fan Mode

Value	Led(s) lit indicated by *
32	
33	*
34	**
35	***
36	****
37	*****
38	******
39	*******
40	********
41	*********
42	**********
43	**********

Trim Mode

Value	Led(s) lit indicated by *
16	
17	******
18	*****
19	****
20	***
21	**
22	*
23	**
24	***
25	****
26	*****
27	******

Spread Mode

Value	Led(s) lit indicated by *
48	
49	*
50	***
51	*****
52	*******
53	*********
54	**********
55	**********
56	**********
57	**********
58	**********
59	**********

Index

Active Configuration, 34
Auto Save, 95
Back Ground Images, 43
Bar Graph, 56
bit masks, 104
Bit Tes, 57
Commands, 19
configuration editors, 18
Configuration Load Delay, 91
Continuous Commands, 19
Controlling the Encoder Ring Leds, 81
Copy and Paste, 39
copyrighted software, 12
Dataref Tool, 22, 104
Datarefs, 20
Device Led brightness, 91
Device Led flash frequency, 91
Drag/Drop, 39
Dual / Single Encoders, 76
Encoder Layouts, 101
encoder ring leds, 112
expression, 21
Expressive Commands, 20, 102
File Operations, 32
File-Append, 33
File-Open, 33
Finders, 22
FlyWithLua, 30
font scale factor, 68
formula, 21
hardware device, 29
Includes / Templates, 34
key background colors and key font colors, 27
Key Labels, 27
Key Numbering and Layout, 31
license key, 13
Linux, 15
logic dataref, 22
logic test, 21
Multi-Encoder, 100
multiplier, 55
number buffer, 26
numeric dataref, 22
Overlay Mode, 29
Piggyback commands, 90
Registration, 14
Repeat Commands, 20
Selective Append, 100
Shared Datarefs, 30
Shift State, 29
Single Commands, 19
Stream Deck sample, 99
Stream Deck SDK, 97
Substring type, 61
System Commands, 20
text to speech, 46
unit number, 29
Units, 28
update rate, 68
virtual device grid, 68
Virtual Devices, 17
X-KeyPad Stream Deck plugin, 99
X-Touch Mini, 11
X-Touch Mini encoder ring leds, 112