Flex Buttons Utility (V2.0)

Keith Regli K7KAR
keith@kregli.com

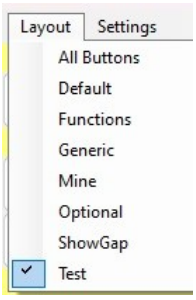
Simplicity of operation and screen real estate are things I care about when I'm operating. This little utility simplifies my screen layout and reduces the number of clicks I need to accomplish some common functions. I no longer have PGXL and TGXL utilities running all the time and don't need to run SmartSDR for mouse access to functions. The pictures below show different layouts that I set up to demonstrate possibilities. You can have as many or as few of these buttons as you like and you can arrange them as you like. **Note**: Start your radios, PGXL and TGXL *before* starting FlexButtons.

The two on the left have the same functions, but show how gaps can make things easier visually. The one on the right shows more of a vertical orientation and ads CW messages CWXPads. Others set up all bands/all modes or buttons for each of the nets they frequent. Still others set up buttons to replace profiles. It all depends on how you use your Flex radio.

ONE CLICK!

You can combine two or more functions on one button, such as: on slice A go to 14.329Mhz, USB with a 2700 Hz filter, the panadapter zoomed to 10 kHz, set POWER to 25 W, BYPASS the ATU, set the TGXL to TUNE and the PGXL to OPERATE. And now you're ready for the Flex Net.

Due to user interest FlexButtons has grown from just a few functions to buttons for nearly anything you'd want to adjust (and many things you have no desire to adjust). The picture below is my attempt to add just one variation of each button type.



As I've received requests from users, the number of actions has increased well beyond the original design. So, for version 2 I've made an effort to re-design Flex Buttons with the large number of actions in mind. The bad news is that I had to change the way layouts are stored so much that compatibility is lost – you'll have to redo any 1.x layouts from scratch – sorry. With version 2 the actions are organized into loosely defined groups to keep the selections more manageable. I haven't implemented every possible setting, but we may get there :) There were a couple of settings that I wanted to add, but couldn't get them to work. Below is the selection tree:

AF – audio level actions associated with a slice

- AFGain: Set the AF Gain level (0-100)
- APF: Audio Peaking Filter on/off/toggle or set a level.
- AudioPan: Set the audio L/R balance (0-100)
- Mute: Set the mute state on/off/toggle
- Mode: Set the operating mode
- Play: Play the record buffer.
- Record: Start recording audio for this slice
- RxDax: Set the DAX channel
- RxFilter: Set the filter to a width in Hz
- RxFilterBounds: Set the upper and lower bounds for filter in Hz. Use this with care.

Cosmetic – non-action layout features

- Blank: Insert a blank space for a button
- Gap: Insert a blank row for a title

CW – actions associated with CW operational

- Breakin: Set on/off/toggle for breakin.
- CWDelay: CW Delay 0-2000ms
- SendBuffer: Send a pre-programmed buffer CWX buffer in SmartSDR.
- SendText: Send a locally pre-programmed buffer. (Note: right click on any button cancels send)
- SidePan: Adjust the sidetone L/R balance (0-100).
- Sidetone: Set the sidetone on/off/toggle.
- SideVol: Set the volume of the sidetone (0-100).
- Speed: Set the CW speed for the internal keyer (5-100 wpm).
- SpeedDown: Decrease the CW speed (2-10 wpm).
- SpeedUp: Increase the CW speed (2-10 wpm).

External – actions associated with things outside the radio

- ANT: set the RX, TX or RX/TX antenna
- PGXL: Switch the active PGXL to operate or standby
- RUN: Run a program with arguments
- SCAN: Set timer and buttons to cycle through
- TGXL; Switch the active TGXL to operate or bypass

Frequency – actions that set the frequency of the radio

- BandMode: Set a frequency (MHz) and mode
- BandModeFilter: Set a frequency (MHz) mode and filter width (kHz).
- Frequency: Set the frequency in MHz

Panadapter – actions associated with the panadapter displayed

- Average: Set the level of averaging to smooth the panadapter display.
- Bandwidth: Set the panadapter zoom to Band, Segment or a specific width in kHz
- BlackAuto: Set on/off/toggle or threshold for removing weak signals from the display.
- DAXIQ: Set the DAXIQ channel for this panadapter (1-8).
- FallColor: Set the color scheme for the waterfall display (Basic, Add Purple, Dark, Grayscale, Deuteranopia, Trianopia).
- FallGain: Set intensity of the colors in the waterfall display (0-100).
- FPS: Set the frequency that the display is updated (0-30).
- Rate: Set the rate at which the waterfall falls (0-100).
- Weight: Set on/off/toggle for the waterfall to use the weighted average to emphasize certain signals.

RF-Slice – RF related actions that are attached to a slice

- AGCMode: Automatic Gain Control speed may be set to Fast, Medium, Slow, None, Off
- AGCThreshold: Set the maximum gain available to the AGC function (0-100)
- ANF: Automatic Notch Filter on/off/toggle or set a level{0-100).
- DialStep: Set the dial step rate in Hz (1-1000).
- DIV: Diversity on/off/toggle
- RFGain: -8, 0, 8,16,24,32dB
- RIT: Set the offset in Hz for the RIT (0-10 Hz).
- NB: Noise Blanker on/off/toggle or set a level {0-100).
- NR: Noise Reduction on/off/toggle or set a level{0-100).
- WNB: Wideband Noise Blanker on/off/toggle or set a level{0-100).
- XIT: Set the offset in Hz for the XIT (0-10 Hz).

RF-No Slice – RF related actions that are independent of a slice

- ATU: Invoke Tune or Bypass
- ClearSpots: Clear all spots
- FDX: Full duplex on/off/toggle
- MemoryRecall: Select a memory profile to use
- MOX: Turn MOX on/off/toggle
- Power: Set the transmit power level (0-100).
- TNF: All Tracking Notch Filters on/off/toggle.
- Tune: Radio TUNE on/off/toggle
- TunePower: Set the tune power level for this band (0-100)
- TXFIL: Set the bounds for the TX Filter
- VOX: VOX on/off/toggle or set a threshold.

Slice Operations – actions that manipulate whole slices

- COPY slice X to slice Y
- SLICE: Set the active transmit slice
- SWAP slice X with slice Y

The functions that turn things on and off generally allow you to “toggle” as well. In addition, those buttons that have on/off states and have a color attached will have the background changed to the color when the device is ON and standard gray when the device is OFF. Other buttons with colors attached will always show the background in that color. Clicking on a “Gap” label will either hide or reveal the buttons between it and the next “Gap”.

Installation

If you have a version 1.x of FlexButtons installed the installation will *delete* any previously created layouts.

You can download this documentation and the software from my website (http://www.kregli.com). It's free – if you feel the need to pay for it, donate to some good charity.

Once you get the zip file (FlexButtons.zip) create a directory in any convenient location (e.g. C:\Programs\FlexButtons). Then extract the files from FlexButtons.zip to that directory (right click FlexButtons.zip and select Extract All…). Depending on the privileges you have, you could run into your security system. You'll have to work your way around that. Choosing a directory in your documents folder may make that easier.

You'll see FlexButtons.exe, FlexButtons.pdf (this document), FlexButtonsRepair.exe and six dlls from Flex. There are also some .dat files that aren't needed for installation.

Double click on FlexButtons.exe and you should see FlexButtons appear in the upper left of your screen with the default set of buttons (be patient, it takes a few seconds to search for radios). The information about the buttons is stored in the Windows Registry and the program will setup the necessary keys when it runs the first time (for those who know about the Registry it's [HKEY_CURRENT_USER\Software\K7KAR\FlexButtons]). If your account is short on privileges, you may have to use "Run as Administrator…" the first time you run the program. Windows will tell you. It may also object (the first time) with a warning that the program is unsigned – this is true. You're allowed to run it anyway (I don't feel the need to register it in the Microsoft Store). See the description of FlexButtonsRepair.exe towards the end of this document.

If your radio is on and connected to the same LAN as your computer the buttons should work. With this base you can change the configuration to meet your needs.

Configuration

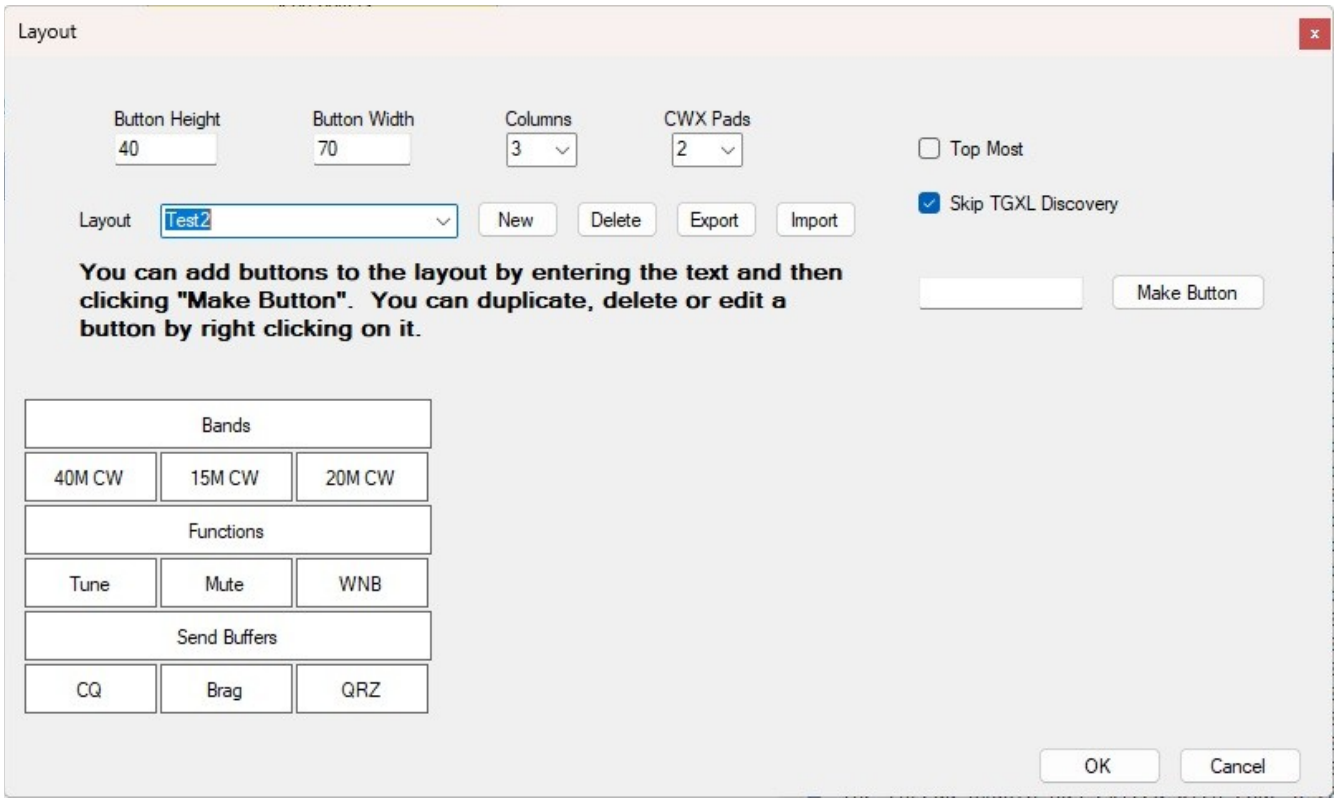
Once the basic program is working you'll probably want to start on your own layout. You do this by clicking on Settings at the top of the window. That will bring up the settings window with the current layout. The only way to get comfortable with configuring layouts is to try it. Expect to throw away your first few efforts.

The top row has the parameters that apply to the whole layout. The button height and width, the number of columns and the number of CWX Pads. The CWX Pads are free form text buffers that you can use to send interactive CW.

Below those you can select which layout you'd like to work on. You can Add, Delete or Export the selected layout and you can Import a layout in an appropriately formatted ".dat" file. There should be some layouts you can import in your installation directory.

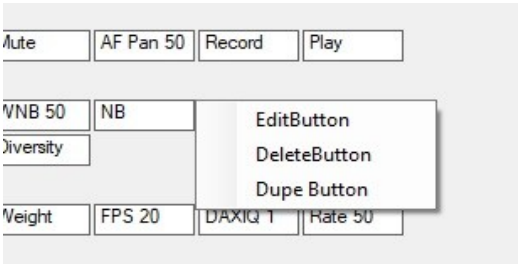
To the right of the window are check boxes labeled "Top Most" and "Skip TGXL Discovery". Top Most allows Flex Buttons to appear on top of all other windows on your desktop. Skip TGXL Discovery can save several seconds on startup if you don't have a TGXL.

The bottom portion of the window is devoted to a representation of the current layout.

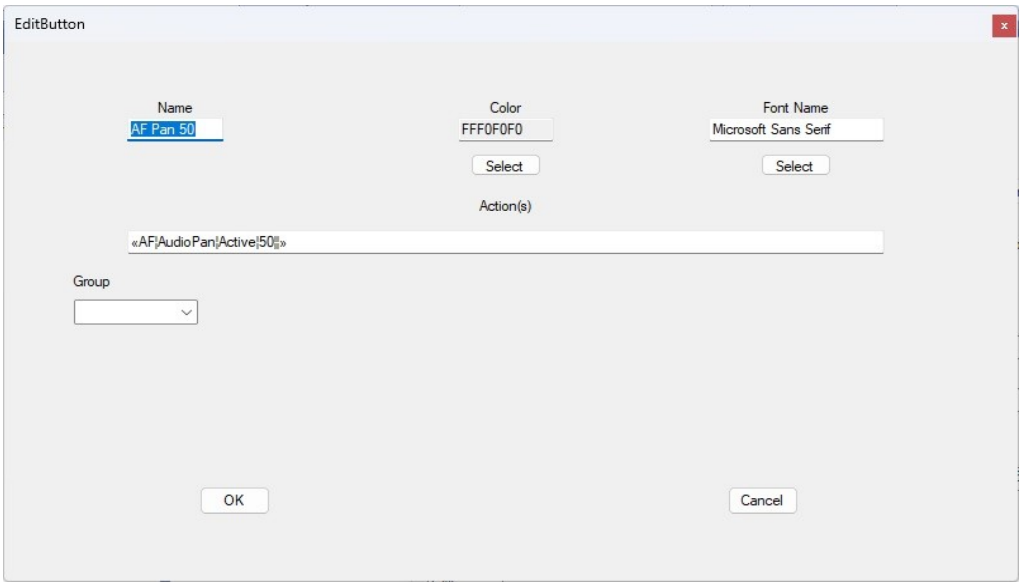


You'll notice in the picture above that some buttons are alone on a row. Those are "Gap" buttons and can be used to separate groups of buttons in the display. If you drag a button (just the cursor will move) you can place it where you'd like it to display. If you right click a button a menu will pop-up that will allow you to Edit, Delete or Duplicate the button. Dupe creates a button with the same characteristics as the existing button. Delete removes the button completely. Edit allows you to adjust the characteristics of the button.

You can also create a button with no characteristics by typing a name into the text box next to the Make Button button.

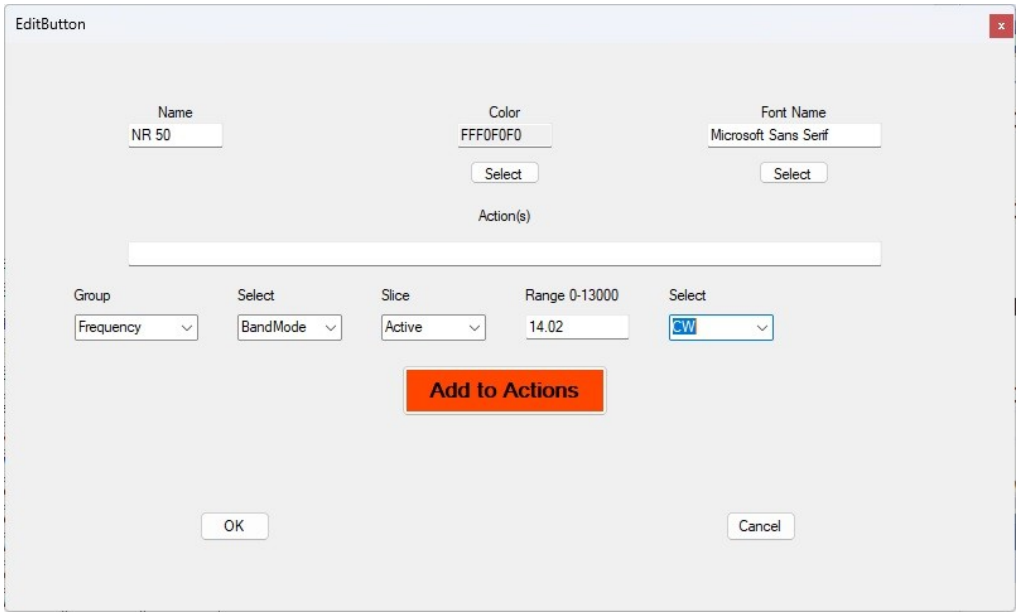


When you click on the EditButton menu the following window appears:



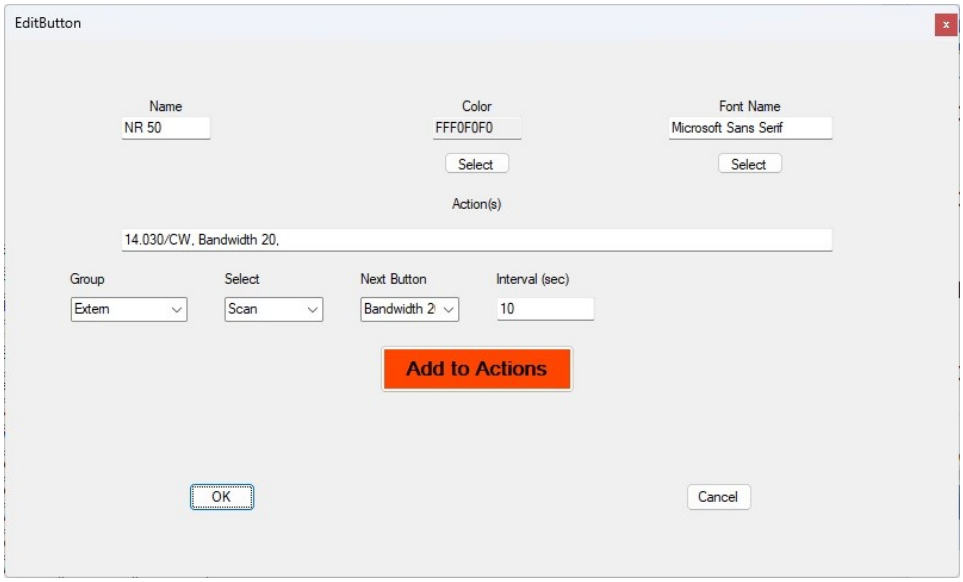
Across the top you can change the name, color and font for the button. The second line shows the current action{s} for the button. Once you get the hang of it you can edit this directly (say for a Dupe button). At first you'll want to use combo boxes below the Action text box. As you build a new action you can add it on the end of the existing actions. If you want a whole new action just delete the contents of the Actions text box.

When you pull down the Group combo box you'll see the groups of actions listed. When you pick one a second combo box will appear and you can select the action from that. Then, depending on the action, other combo or text boxes will appear allowing you to add the parameters for the action. **Note: use the TAB key to indicate you entry is complete in a text box.** Once you've added all the parameters the "Add to Action" button will appear. As each new parameter entry box appears some text explaining what to do will appear.

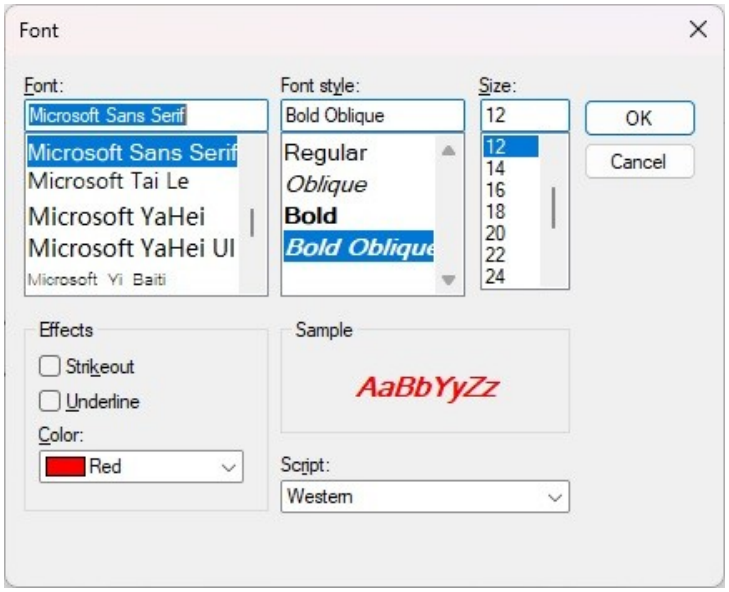


NOTE: If **you click the OK button before you add the action you'll have to do it all over.** You can then add another action or complete save the characteristics by clicking OK.

One special case. The Scan action. This action allows you to set a macro going that automatically clicks a sequence of buttons spread out over a time interval. You might use this switch to various net frequencies every few seconds. In this case the program allows you to repeatedly select the buttons to press until you complete entry of the interval. The button names are shown in the action text box as you add them. Once you add the time interval the "Add to Actions" button will appear. Clicking it will create the action. Note: you can't add multiple actions with a a Run action.



If you click the Select button below the Color or Font fields you will see the standard Windows dialogs for picking colors and fonts (the gray background is RGB(240,240,240).



Some of the actions could use a little explanation. For the most part you can find explanations of what the settings do in the Flex SmartSDR documentation (or 6x00M User Guide).

CW Speed – There appears to be an anomaly in the FlexLib that returns a speed of 30 wpm regardless of the actual keyer setting. So, I keep the speed internally. It starts at 25wpm, but if you use a knob on an M model or the slider in SmartSDR I won't see the change and the next button you click will go off my internal speed rather than the Flex speed.

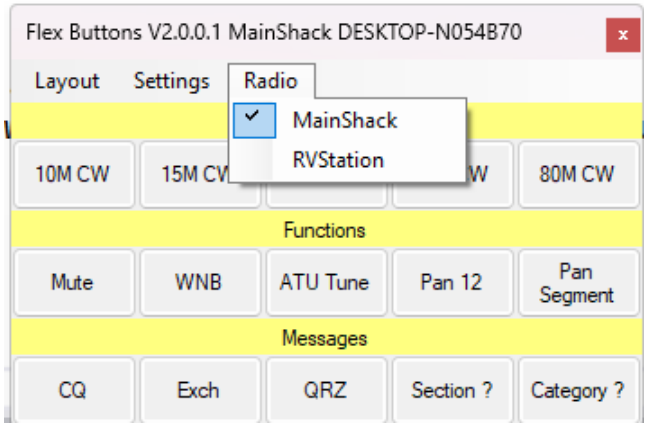
CW Send Text – This sends the text directly.
CW Send Buffer – This only works if SmartSDR is running and the CWX buffer is set up.
CWX Pads – These work like the CW Send Text, but you can enter the text any time. It will be deleted after it is sent.
Copy Slice – This copies all of the slice parameters from the source slice to the target slice so they end up looking identical.
Swap Slice – This trades all of the slice parameters between the two slices.

Memory Recall – When you use saved memories (for say repeaters using a transverter), this will let you recall one into the active slice. You will probably want to use the ANT action to send the in/output to the transverter in/output.

RxFilterBounds – This allows you to directly set the upper and lower bounds on the receive bandpass filter. You have to know how they should be set for the various modes or you won't get what you want. The RxFilter action allows you to set the width you want and worries about where to set the upper and lower bounds depending on your mode at the time you click the button.

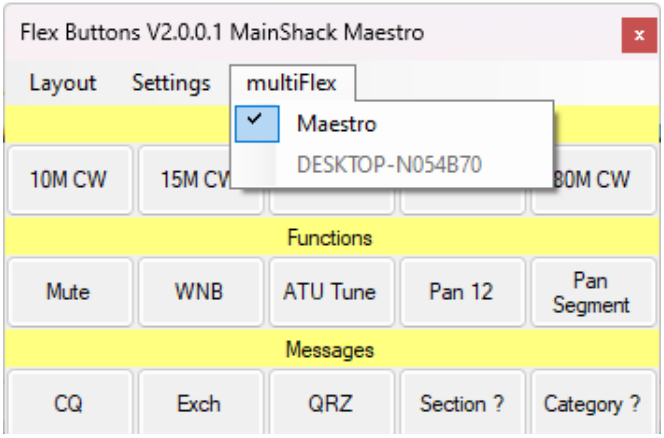
Multiple Radios/multiFlex

We deal with multiple radios. If, during the discovery process, we find more than one radio a third menu selection appears called “Radio”. Clicking on it will allow you to select which radio FlexButtons will connect to. The currently selected radio will be shown with a check.



multiFlex allows multiple programs to access a Flex 6x00 radio and provide services similar to SmartSDR. Currently, a maximum of two programs (stations) is allowed. On receive they can operate independently (band, mode, etc.). On transmit, only one has control. FlexButtons is NOT a program that counts as a multiFlex client, but it interacts with the radio, so it could be associated with either of the two stations. If you are operating in multiFlex mode FlexButtons will add a multiFlex menu. If you click on the multiFlex menu it will show the (two) multiFlex stations. The one that is in control will be checked. The station that FlexButtons will try to relate to is black text, the other will be in gray text. When you select one of the clients, the FlexButtons actions will attempt to relate to that client. In particular, a command that says “set the mode to USB for slice A” will look for a slice A associated with the last station you clicked on and cause that station to switch to USB in slice A. Some actions relate to the station that has control and will go there regardless of which station you’ve selected in FlexButtons. For instance, setting the power level works this way.

The picture below is a multiFlex environment showing the multiFlex menu. It cannot change the controlling station. That must be done at the station desiring control.



Final Comments

Remember to start your radios first so that FlexButtons can pick them up when it starts. It can pick up some things while active, but not all.

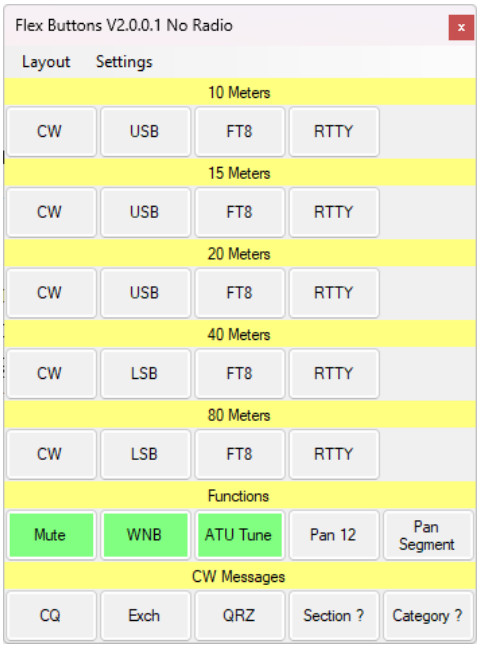
There is no reason you cannot run multiple copies of FlexButtons with different layouts at the same time, but there is only one *current* layout and that will be the last layout selected, so configuring with multiple copies running is a bad idea.

Here are a couple of layouts for really different operators.

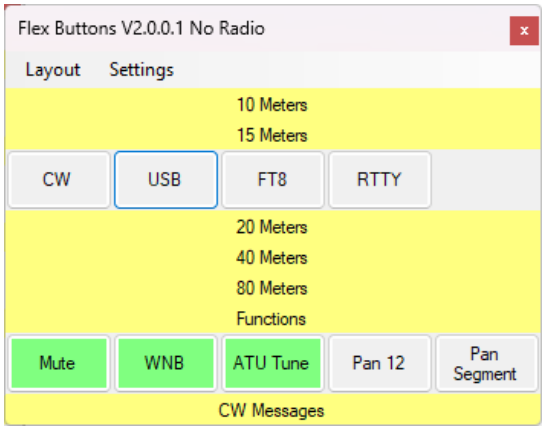
The one on the left is set for someone who checks into several nets and is interested in ARRL activities. Each of these buttons will set the radio to a specific frequency and mode (and possibly invoke the ATU).

The one on the right might be used by a contester. When they're running they'd mostly click the "RIT to 0" button. When they're doing S&P they'd either tune from the bottom of the band to the top and then move back to the bottom with a click (or the reverse). They might want to pan the entire band segment to see where the activity is or where the spots are and then switch back to 10 kHz pan to tune in stations.

Another layout scheme involves collapsing gaps. If you choose to have all bands/modes on buttons, you may like this scheme. The layout below:



Can be collapsed to something like:



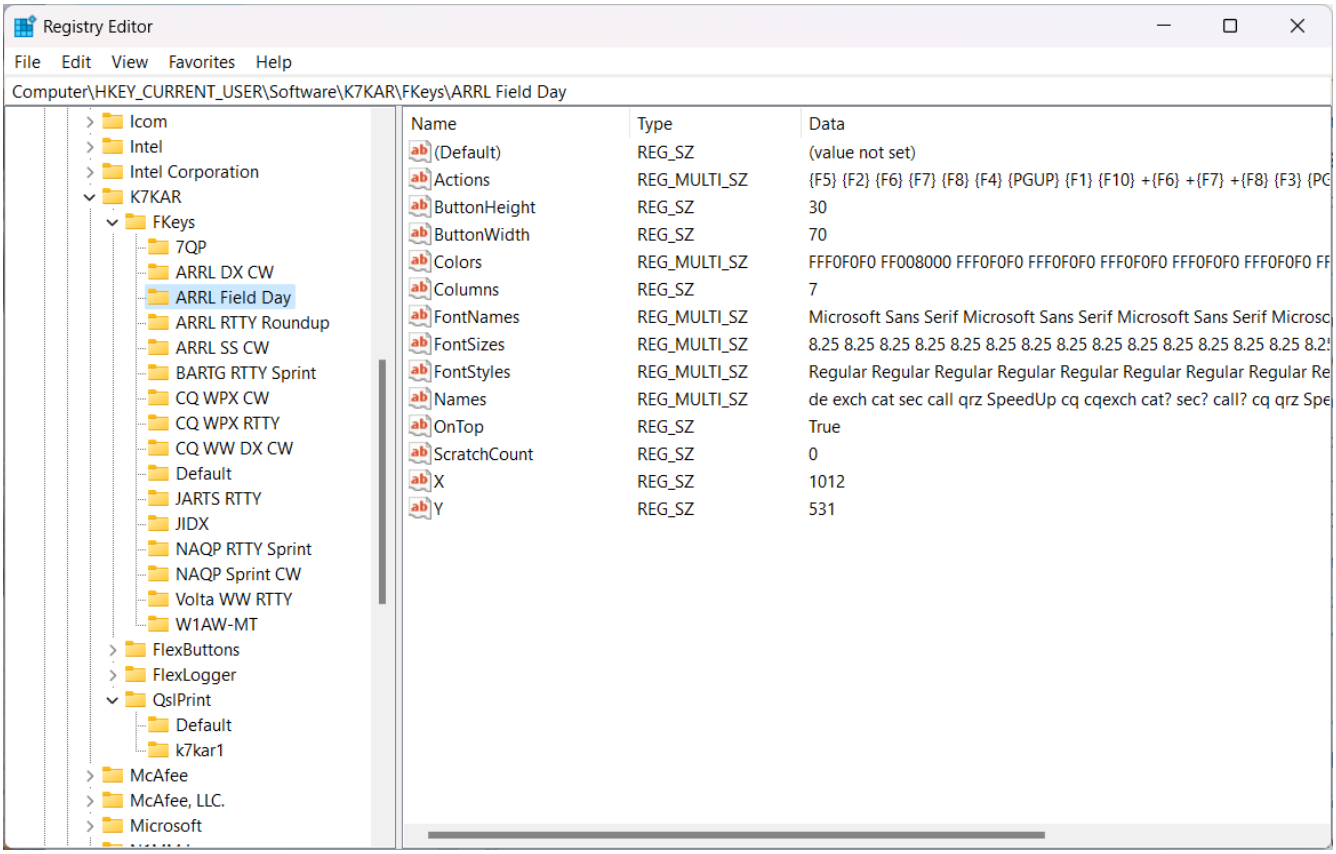
This allows you to save some screen real estate and focus on the band in question.

My original vision for this program was that I'd have five or six buttons that would do things that were a pain to do on the Maestro. After making it available to others, I found many would start with 20 or so buttons and want to increase. I suspect people will come up with all sorts of configurations that I've never imagined! Tell me about them – I might be able to make it easier to setup or use.

Have fun.! If you have questions or suggestions send an email to me at keith@kregli.com.

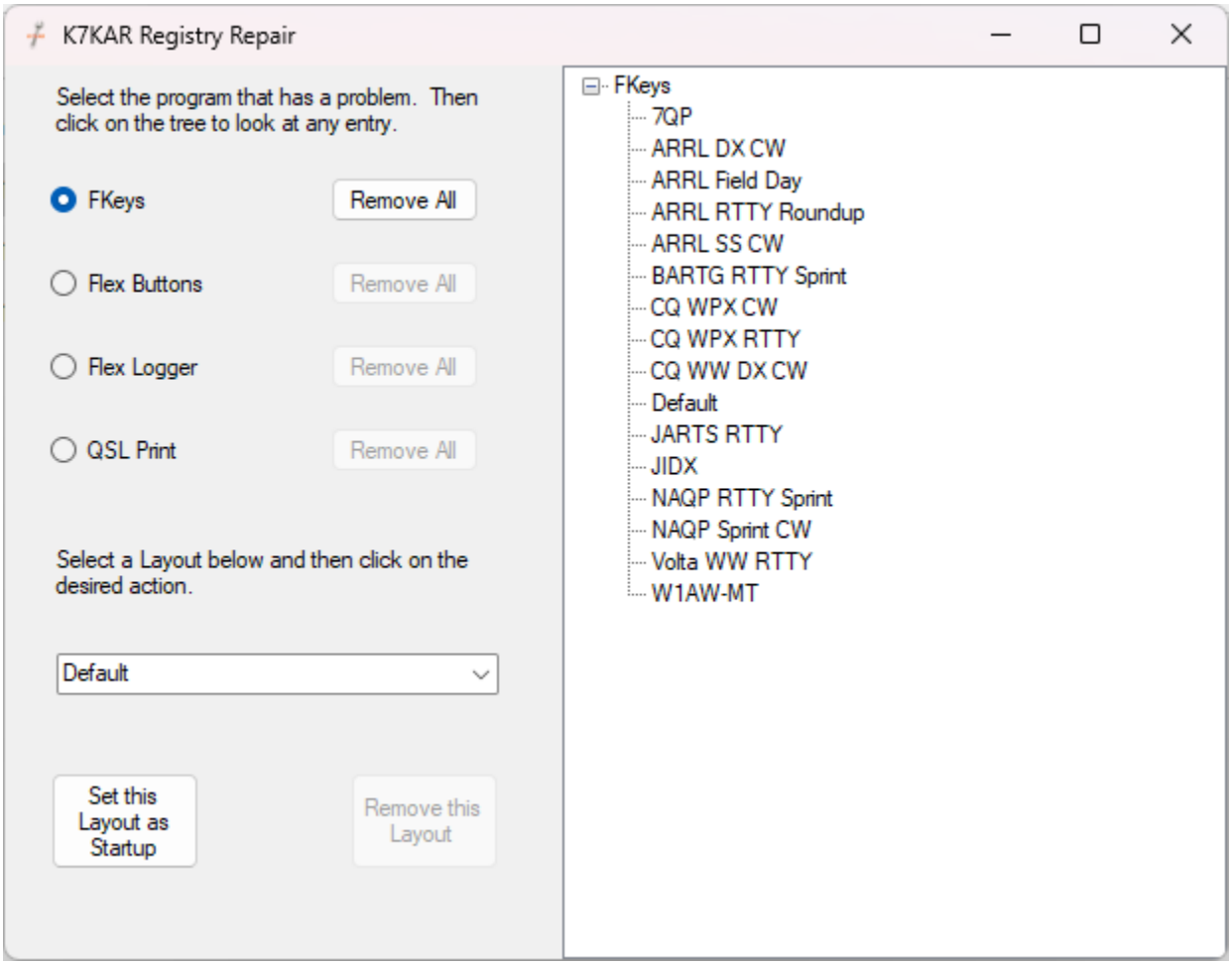
Repair K7KAR Utility (1.0)

This little utility can be used to fix up problems in the Registry associated with my programs. Generally these programs have Registry entries under HKEY_CURRENT_USER\Software\K7KAR. This is pretty typical for such programs. Under this key will be a key for each program (FKeys, FlexButtons, etc.). Under the program key there is some stuff that the program needs to operate. This stuff used to be stored in "*.ini" files, but now lives in the Registry. This makes it less accessible to users who might corrupt it, but makes it harder to fix when something goes wrong. Most of the problems with my programs are caused by me making changes to how things are stored from one version to the next (sorry). My programs have reached the point where each has some generic stuff stored at the top level (usually version # and last "layout" used). Beneath the top level is a key for each "layout". The layouts may be a collection of buttons associated with a contest or QSL configurations. Under each layout are fonts, sizes, text, actions, etc. Below is a picture of a sample Registry view showing what this looks like:



Users are discouraged from using the Registry Editor because you can mess up your Windows installation if you are not careful. I made this utility so you can see what's happening and fix most problems. This is version 1, so it doesn't do as much as I'd like; but it is a good start.

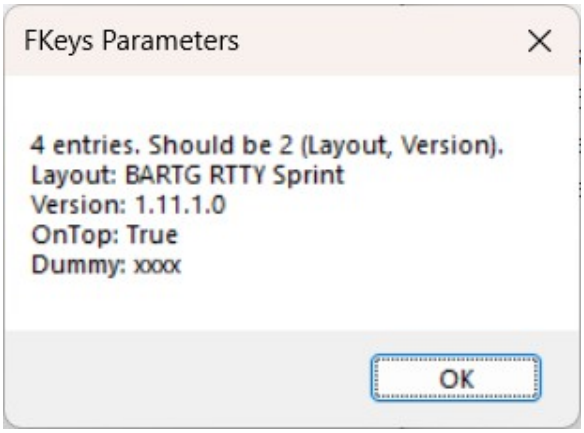
Let's take a look at the program.



This shows the screen with the FKeys program selected. The "Remove All" buttons will remove ALL reference to FKeys from the Registry. After doing that you can run the FKeys program and it should start with a blank slate using the simple "Default" configuration. This is a drastic solution as it will wipe out any layouts (contests) you have previously set up.

If you click on the program name (FKeys here) a message will pop up showing the top level stuff.

In this case I have a couple of incorrect entries (for testing) and it tells me that. It also shows the startup layout (BARTG RTTY Sprint). Usually the problem is that the startup layout has somehow gotten corrupted. Try switching the startup layout to the Default layout and see if FKeys (or other program) will startup correctly. If that works, you can go back into this utility and delete the offending layout. Both of these actions can be done on the lower left of the main screen.



If you'd like to see what is in one of the layouts you can click on it on the right side of the screen. A message box will appear showing the information from the layout and maybe give you a clue what was wrong with it.