Tutorial: Execute Commands and cmdutils

cmdutils *ExitIf* commands

Bob Zawalich June 4, 2021

Command Macros and Command Plugins often work well only if the selection is in a particular state before the macro is run. There are no built-in commands to check the status of the selection, so normally the only way to make such checks would be to create a **Command Plugin** and edit the generated code in the **Plugin Editor**.

The **ExitIf** commands in **cmdutils** were designed to allow a macro to state that there is a problem with the selection, and either exit after displaying a message, or provide a small set of other options.

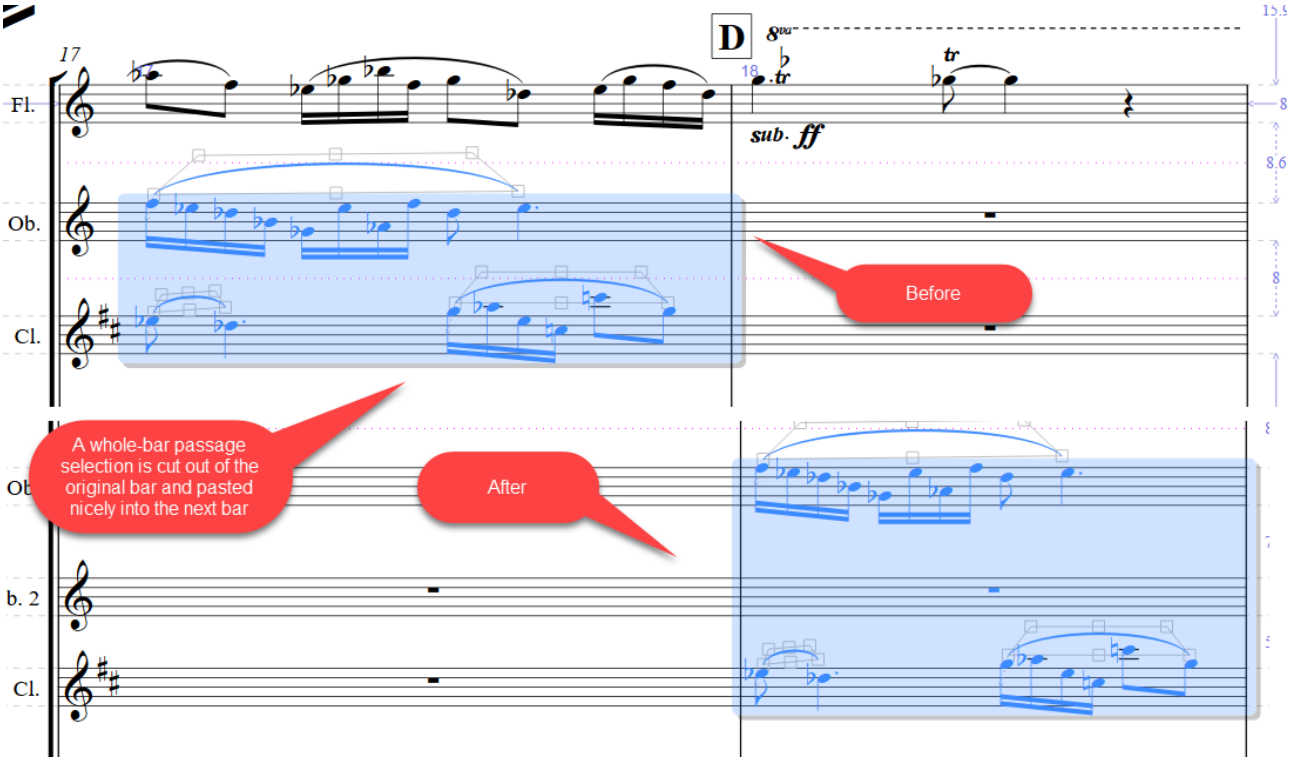
In my opinion you are not likely to need to add these checks for a macro you will only use yourself, but they can prevent problems when you distribute macros and plugins to other people. In "real" ManuScript plugins, I very commonly use the equivalent of these commands to prevent unwanted behavior.

The **ExitIf** routines will check for an empty or non-passage selection, or a passage selection that does not include specific staves or bars, or whether a plugin is installed. If found, they will give a warning and either Exit, or ask if you want to continue, possibly after selecting the entire score.

Here is one example of a macro that uses **ExitIf** commands:

```
// ShiftSelectionRightOneBar
// by Bob Zawalich
// Move All selected objects one bar to the right
ExitIfSelection_NotPassage_cu(A passage selection is required. This plugin will now exit.)
ExitIfSelection_Needs_FullSelect_cu(The selection must have all bars fully selected. This plugin will now exit.)
ExitIfSelection_Avoid_LastBar_cu(The selection may not include the last bar. This plugin will now exit.)
cut
ShiftSelectionNextBar_cu()
paste
```

This plugin starts with a selection, cuts the contents to the clipboard, moves the selection "shape" one bar to the right, and pastes into the resulting selection. Here is a snapshot of it in use:



The actual working code is these lines

```
cut
ShiftSelectionNextBar_cu()
Paste
```

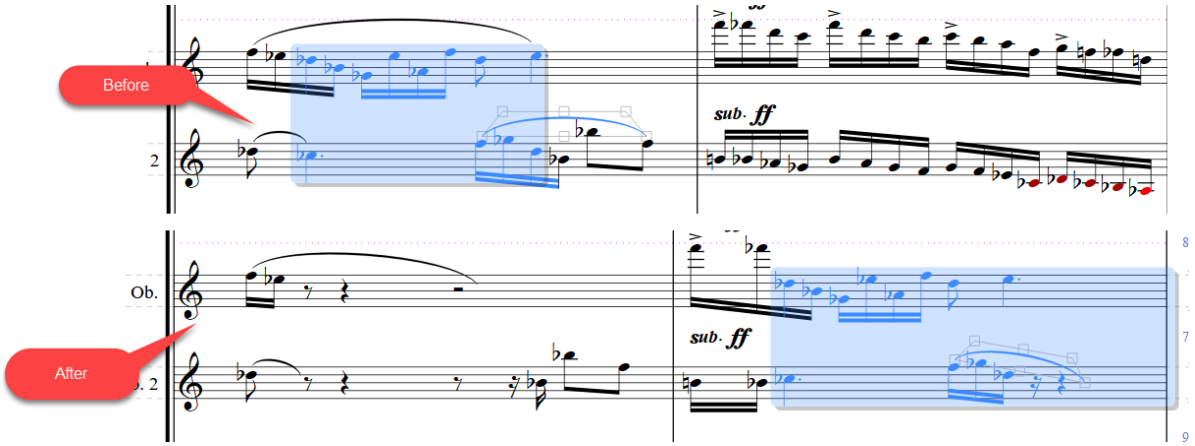
and if you are moving passage selections of fully selected bars it does its job quite nicely.

The **ShiftSelectionNextBar_cu()** command can handle a non-passage selection. It will create a passage selection that includes all the selected objects, and will move that selection. That might not be exactly what you had in mind, but this is how most of the cmdutils routines deal with non-passage selections.

Let's look at some situations that might come up using just these 3 lines of code.

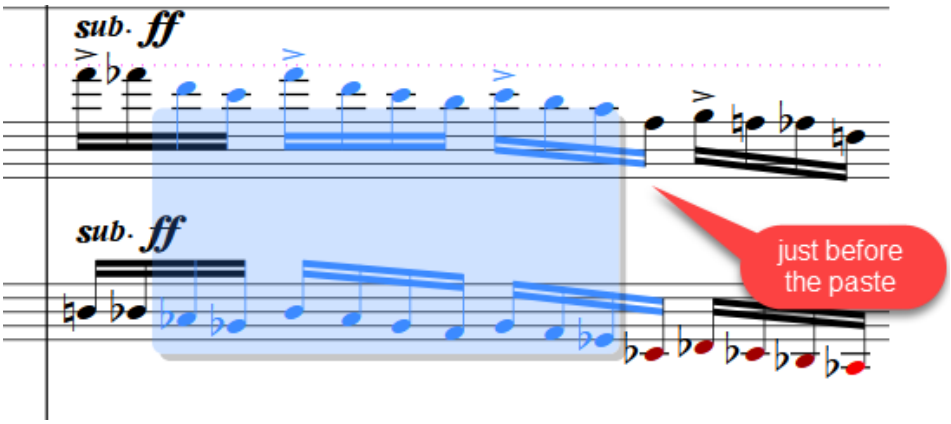
What if you started the plugin when you were in the last bar of the score? If would, in fact, do nothing, because it cannot advance to the next bar. In general these commands will not add additional bars to the start or end, or staves to the top and bottom of the score, but will simply stop, sometimes, but not always, providing an error message.

What if your original selection were only parts of a bar, like this:



First of all, it will leave some bits behind at the start of the original bar, which may or may not be what you intended. Secondly, the selection is expanded to the end of the second bar.

When **ShiftSelectionNextBar_cu()** ran, it set up the selection as I expected it to, with the same range as the original selection. It was **Paste** that extended to the end of the bar. **Paste** likes to fill things up.



So how do you deal with this?
If you are the only user, you can just remember that this is what happens and either try to remember to select full bars or check and repair the damage when it goes wrong.

If this is going to be distributed to the general public, though, expect to get a lot of questions about why it is doing the wrong thing.

Personally I would rather write more code than try to fix up things that do not behave as expected, and I can get some relief by employing some **ExitIf** commands at the start of the plugin.

These are the commands I added to the macro. What do they do?

ExitIfSelection_NotPassage_cu(A passage selection is required. This plugin will now exit.)
ExitIfSelection_Needs_FullSelect_cu(The selection must have all bars fully selected. This plugin will now exit.)
ExitIfSelection_Avoid_LastBar_cu(The selection may not include the last bar. This plugin will now exit.)

In all cases they check some state of the selection, and if they don't like it, they put up a message and quit. If the user wants to run the plugin they need to make an initial selection that satisfies the restraints.

Remember that I said that the code would turn a non-passage selection into one that includes all the selected objects? It would likely pick up other objects that were not originally selected as well. My thought was that it

would be better if it would only work with a passage selection so you knew explicitly what would be affected, so I added the first command:

```
ExitIfSelection_NotPassage_cu(A passage selection is required. This plugin will now exit.)
```

If you start with a non-passage selection this will stop you with a reasonably concise and polite message, and you would have to make a different selection and try again.

Remember the part where you selected a part of a bar, and Paste changed more than you wanted it to, and did not even tell you about it? There is another command that will not let you continue unless the selected bars are fully selected, which would prevent things like multicopy and selection expansion when the paste is done.

```
ExitIfSelection_Needs_FullSelect_cu(The selection must have all bars fully selected. This plugin will now exit.)
```

Finally what if you have selected the last bar in the score and run the plugin. You should not do that, but if someone you give this to does not really understand what it does and tries it out, nothing would happen. They would probably think it does not work. Thus the final ExitIf command will stop you if the last bar in the score is selected.

```
ExitIfSelection_Avoid_LastBar_cu(The selection may not include the last bar. This plugin will now exit.)
```

Now, it could be that these are excessive restrictions. In that case, you can comment out lines of commands (by putting // at the start of the line), like the descriptive lines at the start of the macro dat file.

```
// ShiftSelectionRightOneBar
// by Bob Zawalich
// Move All selected objects one bar to the right
```

Or just use the 3 commands and deal with it when things go wrong. But with these 3 **ExitIf** commands, I feel more comfortable giving this out for other people to use.

```
// ShiftSelectionRightOneBar
// by Bob Zawalich
// Move All selected objects one bar to the right
ExitIfSelection_NotPassage_cu(A passage selection is required. This plugin will now exit.)
ExitIfSelection_Needs_FullSelect_cu(The selection must have all bars fully selected. This plugin will now exit.)
ExitIfSelection_Avoid_LastBar_cu(The selection may not include the last bar. This plugin will now exit.)
cut
ShiftSelectionNextBar_cu()
paste
```

Here is the current set of **ExitIf** commands available in **cmtitils.plg** as of this writing.
Some allow you to continue if you change the selection to select everything. Some stop if you know the macro needs to run a plugin that must ne installed first. These are all scenarios I ran into while trying to write useful Command Macros, and they will not cover all scenarios but they handle a lot.

If you find you need something that is not available, you can use **New Plugin** to turn a macro into a plugin, and then edit the generated plugin, adding your own ManuScript code. You might find a test you wanted to use a lot, so you could write a plugin that just does the check you want. On failure it might do something like change the selection, and you could follow it with an **ExitIf** command that would stop in that case.

There are lots of possibilities. Good luck!

Command Exit Methods

These routines will check for an empty or non-passage selection, or a passage selection that does not include specific staves or bars, or whether a plugin is installed. If found, they will give a warning and either Exit, or ask if you want to continue, possibly after selecting the entire score.

ContinueIfSelection_Empty_cu (strMsgYesNoContinue)
ContinueIfSelection_Empty_Full (score, selection, strMsgYesNoContinue, fIncludeSystemStaff, fNoteRestRequired)

- Exits the plugin if there is no selection and the user responds No in the message box. strMsgYesNoContinue is the message the user will see.

ExitIfSelection_Empty_cu (strMessageIfEmpty)
ExitIfSelection_Empty_Full (score, selection, fIncludeSystemStaff, fNoteRestRequired, strMessageIfEmpty)

- Exits the plugin if there is no selection

ExitIfSelection_NotPassage_cu (strMessageIfNotPassage)
ExitIfSelection_NotPassage_Full (score, selection, strMessageIfNonPassage)

- Exits the plugin if there is no passage selection

ExitOrAll_Selection_Empty_cu(strMessageIfEmpty)
ExitOrAll_Selection_Empty_Full(score, selection, fIncludeSystemStaff, fNoteRestRequired, strMessageIfEmpty)

- Exits the plugin if there is no selection, or selects the entire score (non-system selection) and continues

ExitOrAll_Selection_NotPassage_cu(strMessageIfNotPassage)
ExitOrAll_Selection_NotPassage_Full(score, selection, strMessageIfNonPassage, fIncludeSystemStaff)

- Exits the plugin if there is no passage selection, or selects the entire score (non-system selection) and continues

ExitPlugin_cu

- Exits the plugin immediately. Can be useful when debugging as a way to run a part of a macro and then stop.

ExitIfPlugin_Unavailable_cu (strPluginMenuName)
ExitIfPlugin_Unavailable_Full (score, strPluginMenuName, strMessagePluginUnavailable)

- Exits the plugin if a required plugin is not installed.

ExitIfSelection_Avoid_BottomStaff_cu(strMessage)
ExitIfSelection_Avoid_FirstBar_cu(strMessage)
ExitIfSelection_Avoid_LastBar_cu(strMessage)
ExitIfSelection_Avoid_TopStaff_cu(strMessage)

ExitIfSelection_Avoid_GrandStaff_Bottom_cu(strMessage)
ExitIfSelection_Avoid_GrandStaff_Top_cu(strMessage)

- These will exit the plugin or macro if there is a selection that includes a "forbidden" staff or bar. If the selection is not a passage selection, it will be temporarily converted into a passage selection that includes all the selected objects, and then restored after the tests are complete.

ExitIfSelection_Needs_GrandStaff_All_cu(The selection must include all the staves of a multi-staff instrument, including ossias. This plugin will now exit.)
ExitIfSelection_Needs_GrandStaff_Any_cu(The selection must include only staves of a single multi-staff instrument, including ossias. This plugin will now exit.)

- This will exit the plugin or macro if there is a selection that does not contain specific staves in a multi-staff instrument, such as a grand staff. If the selection is not a passage selection, it will be temporarily converted into a passage selection that includes all the selected objects, and then restored after the tests are complete.

ExitIfSelection_Needs_OneStaff_cu(strMessage)

- This will exit the plugin or macro if there is a selection that contains anything other than a single staff. If the selection is not a passage selection, it will be temporarily converted into a passage selection that includes all the selected objects, and then restored after the tests are complete.