This is the place to thank the very active communities around MuseScore and AutoHotkey. Both Open Source programs have a public history of about ten years. It is amazing to see how the original authors by their stamina and vision succeeded in enrapturing so many volunteers to enrich these software tools. And it would be great if the combined tools would foster the creative process of making music.

AutoHotkey Kit for MuseScore

F4 Positioning

F5 Alternating Time Signatures

F6 Advanced Dynamics

F7 Master Palette

for Windows: Language English US with US keyboard QWERTY

AutoHotkey Kit for MuseScore

Intro with descriptions of all DIY items

All the macros of this kit are actually more or less sophisticated workarounds.
Of course a native implementation will always be the best.

- there is an ongoing effort to make all Palette symbols accessible via shortcuts
- developers have expressed the desirability to add a macro facility *****
- plugins could be written adding new functionality
- MuseScore 4 is in the works promising optimized notation defaults
  A stable version will probably be released before the end of 2021.

* Until then: https://musescore.org/en/project/action-chain-shortcut

In the meantime the macros of AutoHotkey can provide a powerful alternative.
Meant for those users who can take it for what it is, a kit.
Prefabricated parts with bolts and nuts you have to tighten yourself.
Benefits of this kit:

- reduces repetitive tasks to pressing a shortcut ('hotkey combination'). *****
- minimizes manual mouse movements and reduces repetitive stress
- optimizes ergonomy for frequently used shortcuts
- adds a number of features which at the moment don't exist in MuseScore

***** *Often more than ten or even twenty actions are compressed into one.*

"The impact on the workflow is considerable" is an understatement.

As already announced future versions of MuseScore will add new features. New features pave the road to new macro possibilities. So presumably there is a place for AutoHotkey also after 2021 until a full-fledged macro facility will be integrated in the program.

DIY

Add macros step by step when you realise you have done the same action already much too often. Or when some new project makes it very attractive. Imagine having to create 300 measures with alternating time signatures.
BTW with just a little knowledge of AutoHotkey you can expand your own collection of timesaving shortcuts. Hopefully the examples serve you well.

DIY. *But what exactly do you have to do yourself to get the macros working?*

for Windows: Language English US with US keyboard QWERTY

AutoHotkey Kit for MuseScore

MuseScore: https://musescore.org/en/download Current version 3.6 (January 2021)
AutoHotkey: https://www.autohotkey.com/ Current version 1.1.33.02 (July 2020)
"The ultimate automation scripting language for Windows."
Download the Unicode version https://www.autohotkey.com/docs/Tutorial.htm#s11
Kit: a collection of files consisting of:

This document: AutoHotkey_for_MuseScore.pdf		
Independent_Hotkeys.ahk	Data	
Master.ahk	Data	
Coordinates.ahk		*Repository of coordinates*
PixelMousing.ahk		*Tool to determine coordinates*
Set_Surface_Coordinates.ahk		*Macro: help for Coordinates.ahk*
Hotspot_Coordinates .txt		*Help file for Coordinates.ahk*
Image_Coordinates.txt		*Help file for Coordinates.ahk*
HotKeys_in_prefix_order.txt		*Overview and more info*
Check MuseScore coordinates.mscz		*MuseScore help file*
Initialise_Inspector.mscz		*MuseScore help file*
Ornaments_Master_Palette.pdf		*Help for Ornaments*
F1_ColorSearchSelect.ahk	Data	*Macrogroup F1*
F2_Apply_Palette_Symbols.ahk	Data	*F2 cooperates with workspace:*
Advanced AutoHotKey.workspace		
F3_Navigation.ahk	Data	*Macrogroup F3*
F4_Positioning.ahk	Data	*Macrogroup F4*
F5_Alt_Time_Sigs_and_BeamProps.ahk	Data	*Macrogroup F5*
F6_Advanced_Dynamics.ahk	Data	*Macrogroup F6*
F7_Master_Palette.ahk	Data	*Macrogroup F7*
F8_Note_Input.ahk	Data	*Macrogroup F8*
F9_Specials.ahk	Data	*Macrogroup F9*
F10_Change_Score_Status.ahk	Data	*Macrogroup F10*
F11_Note Duration_MouseWheel.ahk	Data	*Macrogroup F11*

The Data files are 13 small .txt files. They list the DIY items for each hotkey and are meant as assistants in assembling them. See the Reference section.

In the attachments the .ahk files have been renamed as .txt files.
Give them again the extension .ahk When the extensions are not visible then (for Windows 10): Open Windows File Explorer, switch the ribbon to the View tab and tick the File name extensions box. Now you'll be able to change the actual extension.

Put the file 'Advanced AutoHotKey.workspace' in
C:\Users\Username\AppData\Local\MuseScore\MuseScore3\workspaces
As mentioned this is primarily the companion file to
F2_Apply_Palette_Symbols.ahk but several other macro groups use it as well.

Create a folder called 'AHK' in C:\.....\Documents\MuseScore3
This folder is called your '*working directory*'.

For first-time users

The package includes a few extra files. These are small. They are meant for some experiments. If you are new to AutoHotkey the best way to get the gist of basic concepts is a hands-on approach.

These files are:
Learn.ahk.
Master_stripped.ahk and its companion Master_stripped_DATA.txt Data
Again: the Data files are only assistants, helpers in keeping track.
Coordinates_stripped .ahk
F1_ColorSearchSelect_stripped .ahk and F1_Stripped_ DATA Data

Put these files in your *working directory*. Do the same with the tools PixelMousing.ahk and Set_Surface_Coordinates.ahk

Finally, if you want to have shortcuts for all symbols of the Advanced AutoHotKey.workspace:
F2_Apply_Palette_Symbols.ahk (put its companion Workspace file in the location described above)
This macro will *almost* work out of the box. See below for the DIY part.

Change the pathname of the .ahk files as descibed below.

For users experienced in AutoHotkey

Put all files of the collection - except the .workspace - in the working directory.

Change the pathname in most .ahk files but two

All hotkeys but those in 'Independent_Hotkeys.ahk' are context-sensitive.
They will only trigger their macros when MuseScore is active.
So the scripts have to know the location of MuseScore:
Open the script with Notepad. Almost at the top of it there is this line:

```
#IfWinActive ahk_exe S:\MuseScore 3.6 portable\MuseScorePortable\App\MuseScore\bin\MuseScore3.exe
```

with the comment `; Enables Hotkeys when MuseScore3 Window is Active`
Replace the path S:\etc. with your location of MuseScore3.exe

Do this in Master.ahk, PixelMousing.ahk and in all files from F1 thru F11.
It is *not* needed in 'Coordinates.ahk' and 'Set_Surface_Coordinates.ahk'.

The next paragraph is mainly meant for first-time users of AutoHotKey.
The section about the *prefix keys* is relevant for everyone.
For clarity the paragraphs about the Defined State are rather verbose.

LEARN HANDS-ON

PM: Don't forget to set the path of the .ahk files as described above.

Start MuseScore3
Launch Master.ahk and next Learn.ahk (select -> Enter or by double-click)
To edit this .ahk file: right-click -> Edit (or press E)

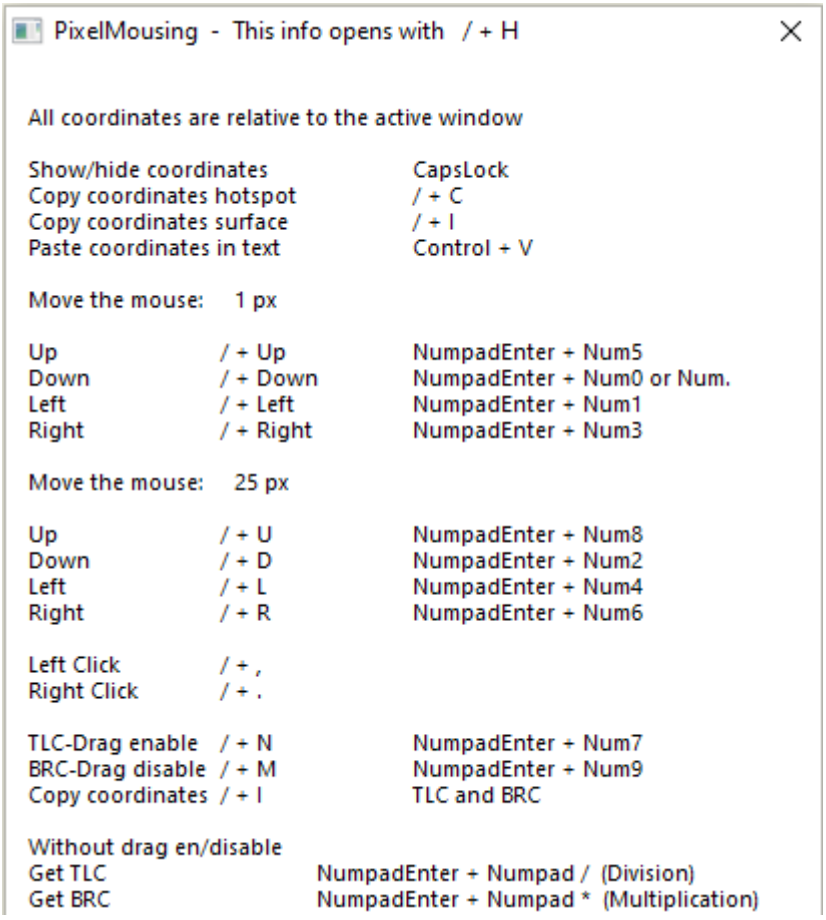
For the hotkeys to run and edit 'Learn' and other files see page 39.

Step #1 We are going to make a really superfluous macro:

Its function must be: "Click on the Menu Add to open it."

Of course MuseScore has the shortcut Alt + A to open this menu. The example just wants to illustrate basic principles and functionality.

Obviously we have to tell the program where to click.
We need the assistance of the tool PixelMousing to find the coordinates.
So we launch PixelMousing.ahk *Hotkeys on page 39.*
With *nothing selected* in MuseScore we press **/ + H** to get the help info.
Nothing selected: to prevent the creation of e.g. an acciaccatura.



The picture shows part of the Help info for PixelMousing.

There are two sets of hotkeys performing the same action.

The commands using **/** as first key use the arrowkeys for one pixel movements.
The keys **N M , .** and **/** are neighbours. **N** for **en**able.
U D L R for 25 pix movements.

In the Numpad variants the keys with arrows move 25 pix.

Before we continue let us focus on our pressing of **/ + H**.
Or more precisely our pressing of **/**.

The Prefix keys *See also the list on page 40*

In MuseScore we have the key **/** as the default shortcut for Add acciaccatura.
Seen from the perspective of AutoHotKey that's a waste of possibilities.
In AutoHotKey we can make a multitude of hotkeys with **/** as first key.
/ + 1 **/ + 2 ...** **/ + A** **/ + B...** **/ + Up** **/ + Home ...**
/ + LeftMouseButton ... etcetera.

The combination **/ + H** is called a '*hotkey combination*'.
The first key is called the *prefix* key.
The prefix key must be free in MuseScore meaning the key is *not in use as a shortcut* within MuseScore.
At the same time we don't want this key to be hijacked by AutoHotkey.
For instance we must still be able to use it when typing text.

The solution is simple: we can 'liberate' the **/** key by redefining the shortcut within MuseScore: e.g. Add acciaccatura → **Shift + /**
And AutoHotKey has its own solution to prevent this prefix key to be stolen:
The **/** must be preceded by the tilde sign **~**

So the tip would be: **redefine** for now 'Add acciaccatura' in **Shift + /**

Actually you could opt for not redefining this shortcut at all.
Typing **AC** in the InputBox of F2_Apply_Palette_Symbols.ahk produces an acciaccatura.

Another instance is the **Z-key,** one of the easiest accessible keys on the keyboard. In MuseScore **Z** is the shortcut for 'Show symbol palette'.
So we're going to liberate this key. Redefine this shortcut e.g. in **Shift + Z**.

Step #1 continued

We start PixelMousing with the hotkey **[+ /** and steer the mouse cursor first manually and then for precision sake via keystrokes to position it in the middle of the second 'd' of Add. At this spot the PixelMousing tooltip reads say:

It turns out that for the testscreen the numbers are x=133 and y=41.
In Learn.ahk input the numbers valid for your screen.

The numbers are input in the hotkey **T + A**. Here we have another *prefix* key.
The T-key is 'free' after a default installation of MuseScore.
But the only macros in this kit using the T prefix key are those in Learn.ahk
Page 40 shows a list of all prefix keys. See also HotKeys_in_prefix_order.txt

Save and reload Learn.ahk. Test the hotkey by pressing **T + A**.

Step #2 Replace hotkey #1 by hotkey #2.

In hotkey #2 **T + A** triggers the MuseScore shortcut Alt + A. See Learn.ahk.
Save and reload Learn.ahk. Test the hotkey by pressing **T + A**.

Step #3 Replace hotkey #2 by hotkey #3.

Now we want to use the more general approach with *Variables* and *Values*. The numbers of coordinates and other things get their own repository. This is Coordinates.ahk. In this file you assign the values valid for your system to the variables. The values will be the numbers of X- and Y-coordinates or e.g. the numbers of colors. For Learn.ahk we use Coordinates_stripped.ahk. So the macros themselves don't have these specific data. The names of the variables are references. Any ahk file containing macros will read the coordinates file at startup. That's where the #Include command almost at the top of Learn.ahk is for. See this ahk file and the info accompanying hotkey #2 and #3.

Step #4 We use hotkey #3 to make hotkey #4

In this one **T + C** triggers the - superfluous - macro 'add Composer text'.
We meet the *Sleep* command for the first time. We must give MuseScore some milliseconds to open the Add menu. When many programs run simultaneously CPU-load is heavier and *Sleep* times have to be longer. See Learn.ahk.
About *Sleep* time see also the **Technical Supplement** of this pdf.

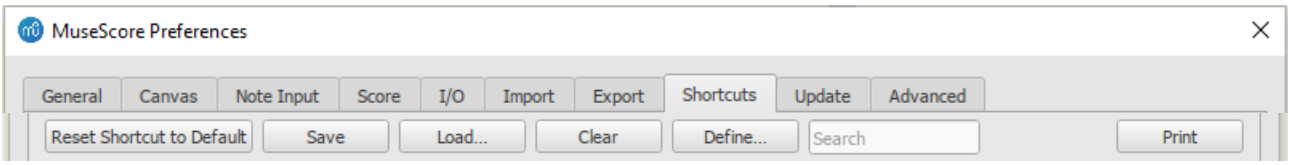
Step #5 Hotkey #5 liberates the P-key. NB: in **Master_stripped.ahk**

P + K toggles on screen Piano Keyboard

You could redefine 'Piano keyboard' **P** in **Shift + P** or you could opt for not redefining this shortcut at all.

By liberating the P-key another multitude of hotkey combinations becomes available.

Step #6 Hotkey #6 operates in a window. NB: in **Master_stripped.ahk**



CapsLock + S Open Preferences Tab Shortcuts and click in the search field.

In order for this hotkey to work the window Preferences must be in a *defined state. I*n this case (as in most cases) that means: *minimized*.
Now the mouse will always hit the Tab 'Shortcuts' and then the Search field.
So first minimize the window and only then determine the coordinates.
MuseScore remembers the last state of the window when it reopens.

In Coordinates_stripped.ahk:
Enter the coordinates of Tab_Sc_X, Tab_Sc_Y (in the middle of the word 'Shortcuts'.)
Do the same for Pref_Search_X and Pref_Search_Y (the centre of the Search field.)

Save and reload **Master_stripped.ahk**. Test the hotkey, press **CapsLock + S**.

For now close Learn.ahk and the other running .ahk files.
We continue with a recapitulation of the topic Coordinates as preparation for the topic Defined State. We'll return to Learn.ahk and the stripped files on page 14 to practice what we read in the next paragraphs.

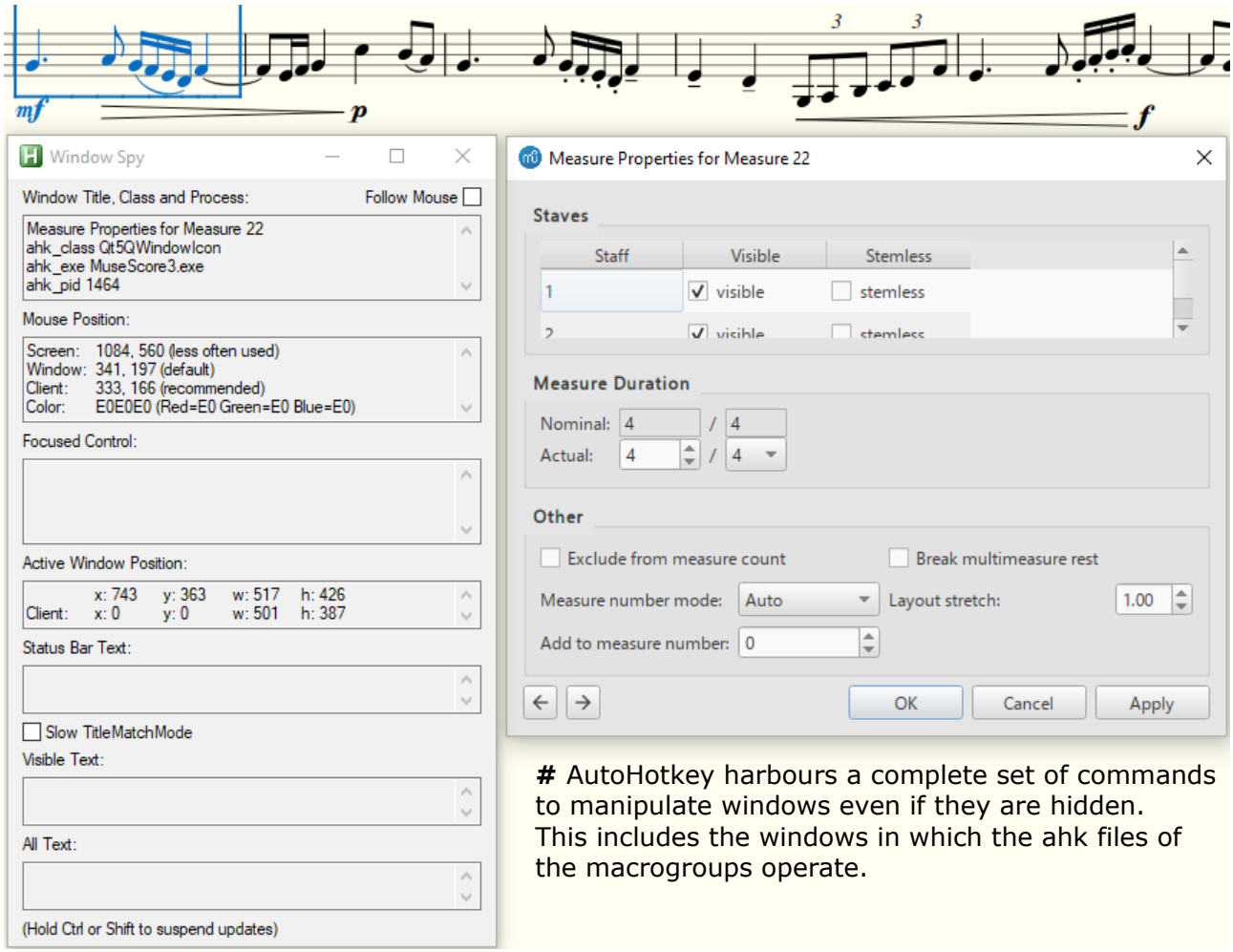
COORDINATES - screen or window

The test screen has a resolution of 1920 x 1080 at 96 DPI (dots per inch), 1920 pixels horizontally (X) and 1080 pixels vertically (Y).
The top left corner of the screen has the coordinates x = 0 and y = 0.
The bottom right corner of the screen has x = 1919 and y = 1079.

This tooltip shows the mouse position. The coordinates are relative to the active window. The tooltip is created by the tool *PixelMousing* ***** which is part of this AHK-kit. Left number shows X, right number Y

AutoHotkey comes with an extensive tool of its own: *Window Spy*.

It reports that the mouse currently is at spot x=1084, y=560 relative to the screen. More important for us: it reports that the mouse currently is at spot x=341, y=197 *relative to the active window*. That means the mouse can do all sorts of things in this window *if it always has the same size* **#**. And just as important: the position of the window on the screen is **not** relevant.
Window Measure Properties is active, its title is black.



AutoHotkey harbours a complete set of commands to manipulate windows even if they are hidden. This includes the windows in which the ahk files of the macrogroups operate.

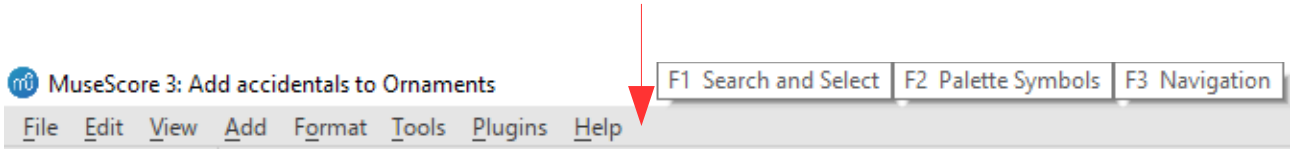
Window Spy has a rich feature set with info needed when you would want to write your own macros.

***** In *PixelMousing* you steer the mouse by keyboard presses. This counteracts small hand vibrations and gives us the pixel precision we'll sometimes need.

COORDINATES - screen layout

If your screen has the same resolution - 1920 x 1080 - as the test screen there is a slim chance that the number of your DIY parts is minimal.
In that case you could use the coordinates of Coordinates.ahk without further ado. Ultimately it all depends on which *screen layout* you prefer.

The coordinates given in Coordinates.ahk presuppose Full Screen, picture B.



A - Screen maximized - MuseScore title ribbon with tooltip reminders of active macro groups

B - Full Screen - No title ribbon. Menu ribbon with tooltip reminders of active macro groups. The arrow shows the 'toolbar selection spot'. *See page 29*

Height

Of course your choices have a direct influence on the coordinates.
Say you want a macro to click a menu title like View or Add.
Compare pictures A and B.
All horizontal (X) numbers probably are the same. But the height (Y) of all items in pic A is that of Pic B *plus* the height of the MuseScore 3: title bar.

The choice for Maximized or Full Screen also influences the position of all hotspots and their coordinates in the Inspector.
The same is true for the Palettes and the Selection Filter.

And the final factor: which (horizontal) toolbars are present, if any.

Width

The position of all items in the side panels is influenced by their width.

From the margins to the centre

All these choices together determine the surface of your Canvas Real Estate.

DEFINED STATE - intro

Many macros contain mouse clicks on hotspots within the inspector.
Colors play an important role in macros which add *new* functionality to MuseScore so let us take 'Set Color of one element' just as an example.

Two types of images for elements.
Only one type has a Style button.

We look for a spot in the black rectangle where the mouse will click the target irrespective which of **three** is visible: the third one is of a Note Anchored Line, with different X-coordinate. Luckily there is an *overlap* area valid for all three.
So we get - for the test system - these lines in the file Coordinates.ahk

```
IN_012_X := 1862          ; I(012) color picker rectangle (black)
IN_012_Y := 86            ; OVERLAP all ELEMENTS - dynamic, note, note anchored line
```

Obviously the chosen hotspot of Inspector item 012 is here at 1862, 86.

BTW everything behind the semicolon and the space is a comment. Comments are not read by AutoHotkey so they don't influence the speed of execution.

Now you have to determine these coordinates for your screen.

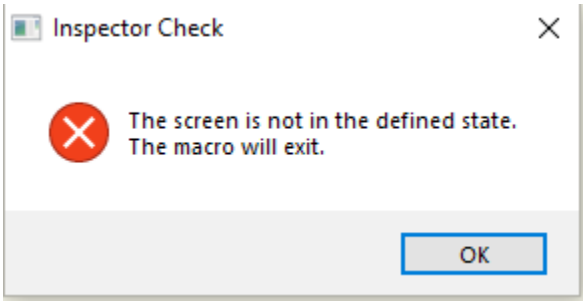
The kit has several tools for determining coordinates: **Window Spy** which comes with the installation of AutoHotkey and **PixelMousing** optimized for use with MuseScore.

In the Master file of the kit and also in the stripped Master we have some other tools:

[+ NumpadEnter	run the macro 'Set Surface Coordinates' SSC - see page 19
Shift + NumpadEnter	and create immediately images - *hotkey embedded in SSC*
[+ NumpadAdd	run the snipping tool - see page 21 if it refuses to run.
Z + F11	get coordinates of hotspots - see page 27
Z + F12	get coordinates of surfaces

Notice the two prefix keys: **[** and **Z**.

Of course the mouse will only click the right target if the two rectangles 'Set Color' will *always occupy the same surface* on the screen.
One of those conditions to attain that aim could be that the Inspector panel is docked at the right hand side of the screen.
All conditions together result in the **defined state** of the screen.
The layout of the screen, that state, is the result of your choices. In a way it reflects your workflow. That state must be easily reproducible with pixel precision.
Just as important: when the screen is *not* in the defined state all commands using the Inspector must be ineffective. We don't want unpredictable edits of our score. That's why we maybe can appreciate a message like this one:

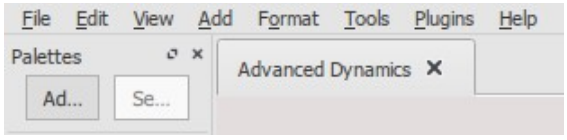


DEFINED STATE: Inspector

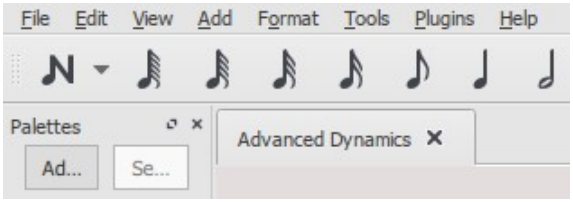
First step to reproducibility of the screen state: the Inspector is *docked*. In general this will make the canvas surface bigger. It also makes the AHK commands shorter, simpler and faster.

Docked: FWIW my personal preference is on the right, its default position.
When undocked the Inspector is a MuseScore window that must be activated before each use.
When docked the Inspector is just a part of the screen.

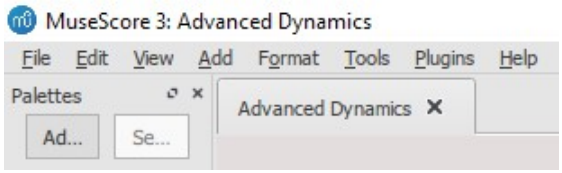
Height of the Inspector: obvious choices influence the height of the Inspector and thus all coordinates in this side panel:



Full screen, no toolbars = MAX CANVAS



Full screen, toolbar Note Input *****



Screen maximized, no toolbars

No Full Screen, Taskbar eats real estate...

* The height of the Note Input toolbar lowers the position of Canvas and Inspector. An additional toolbar like Playback of course even more.

Screen: Full or Maximized? Toolbar(s): visible or not? If yes, which toolbar(s)?

The situation where MuseScore operates in a floating window is not taken in consideration.
FWIW: my personal preference is *MAX CANVAS.*

Width of the Inspector, **the wide variant**: select a dynamic. In the Inspector you see a column of S-buttons. (Set as Style). Adjust the Inspector Separator with PixelMousing *using the method described on the next page*.
Widen the Inspector until the S-buttons are fully in view. The horizontal scroll-bar at the bottom of the Inspector must just have disappeared. Now you have found your *Defined Inspector Width*, the left number in the tooltip. Make a note. It is the variable **`Def_Insp_Width`** to which you assign your found value.

My prefence is **the narrow variant** because it makes the Canvas 5% wider . Select a hairpin. Adjust the Inspector Separator - *using the method described on the next page* - until the horizontal scrollbar at the bottom of the Inspector just disappears. Make now a note of the x-number (the X-coordinate). With this width the mouse can still hit *almost* all relevant spots while stealing only a minimum of Canvas real estate. By giving the mouse an extra horizontal scroll command we can shift the Inspector to the right or the left. Now all relevant spots - e.g. text properties - can be clicked. See page 78 about scrolling.

DEFINED STATE: Palettes/Selection Filter

This concerns mainly the Selection Filter. The reason is that *all palette symbols can be input via hotkeys* (macro shortcuts) so the width of the palettes is actually not relevant.
Palettes and Selection filter are *docked* for the same reasons as the Inspector.

Docked: FWIW my personal preference is on the left, the default position.
When undocked these panels are MuseScore windows. They must be activated before each use. When docked these side panels are just a part of the screen.

The width of the picture Selection Filter will determine the width of Palettes/Selection filter.

Adjust the Palettes/Selection Filter separator with PixelMousing. Choose the minimal width which makes the whole image visible.
Make a note of this width (X-coordinate).
The name of this variable *Defined Palette Width is* **`Def_Pal_Width`**
In the file Coordinates_stripped.ahk respectively Coordinates.ahk you assign your found value to this variable.
E.g. for the testscreen **`Def_Pal_Width := 111`**

Measuring the coordinates

Determine the width using PixelMousing. First step is to minimize the width. For a left side panel approach the separator with the mouse cursor moving from left to right. Enable drag at the pixel where the mouse cursor changes shape. Move the separator to the desired width.Disable drag. Make a note of the X-coordinate. For a right side panel approach the separator with the mouse cursor moving from right to left.

Pixelmousing

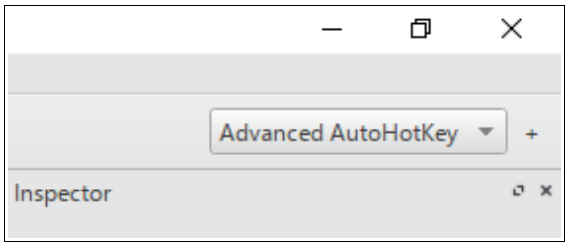
/ + N Drag enable
/ + M Drag disable
/ + C Copy coordinates to clipboard

```
Def_Insp_Width :=
Def_Pal_Width :=
```

In danger of stating the obvious: when determining coordinates **stick to the choices of your Defined State!**

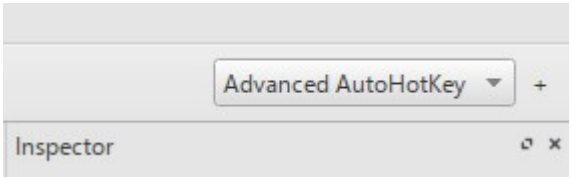
How to restore the Defined State?

First let us imagine our screen is in the defined state.
You have chosen the height and width of the Inspector as per your workflow.
In the top right corner we see (light theme assumed) something like A, B or C.



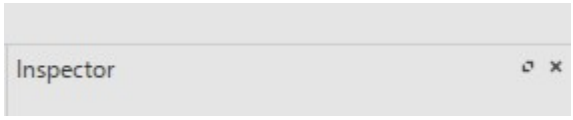
A: Maximized Screen with toolbars Note Input and Workspace

or



B: Full Screen with toolbar

or



C: Full Screen, no toolbar - MAX CANVAS

For now we are interested in the position of Inspector

Whichever defined state you may have chosen in each defined state this Inspector image *will always occupy the same surface*. This will serve us well.

AutoHotkey has the following command:

ImageSearch

Searches a region of the screen for an image.

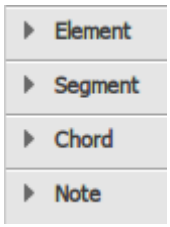
```
ImageSearch, OutputVarX, OutputVarY, X1, Y1, X2, Y2, ImageFile
```

All macros using the Inspector include this line:

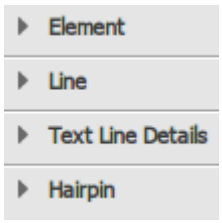
```
ImageSearch, , , IM_01_X1, IM_01_Y1, IM_01_X2, IM_01_Y2, *40 IM_01_Inspector_Inspector.png
```

"Search for the image IM_01_Inspector_Inspector (Inspector) within the rectangular surface determined by the top left corner with the coordinates IM_01_X1 and IM_01_Y1 and the bottom right corner with IM_01_X1 and IM_01_Y2 and allow 40 shades of color variation in the search. (*** 40**)."

If the image is not recognized the message 'The macro will exit' appears. If the image is recognized it's in the right spot (surface), meaning the Inspector is visible and its width and height are correct. Execution of the macro may proceed. But it's still possible that something in the inspector is not quite right. For we don't like it if a section of the Inspector is collapsed. We don't like this



or this



We need a macro to check that for the selected element all sections of the Inspector are in their expanded state. If sections are collapsed they have to be expanded to restore what we could call the defined Inspector state.
We'll return to this topic on page 31.

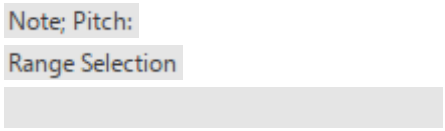
Summary: to restore the Defined State of the *screen* we have to restore

The global layout of the screen	- full or maximized
The width of Palettes/Selection Filter	- left panel
The presence of the Palettes	- as default left panel
The width of the Inspector	- right panel
The presence of the Inspector	- as default right panel
The expanded state of the Inspector	- for selected elements
The height of the side panels	- which toolbars (if any)

Related to the Inspector:
The Play Panel docked within the Inspector or as independent window?
On a side note:
The chosen theme - Light or Dark - belongs to your Defined State as well.
General related topic:
The Defined State of the *Muse Score Windows*.

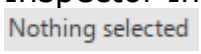
Intermezzo - the Function of Images

Statusbar Images give the program information about selected elements.



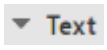
Statusbar_Note_Pitch.png
Statusbar_Range_Selection.png
Statusbar_Nothing_Selected.png

Inspector Images sometimes perform the same job better



IM_03_Inspector_Nothing_Selected.png

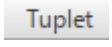
Or their presence is associated with different kinds of elements. E.g.



IM_17_Inspector_Text.png
occurs with Staff text, System text, Tempo text, Frame text, Dynamics

IM_14_Inspector_Segment.png
occurs only with notes, rests, barlines, timesignature or clefs

Because the text is black and not grey we know something about a note

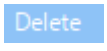


IM_06_Inspector_Tuplet.png

The meaning of this one depends on which surface it occupies

CTS_Checked_ticked_sign.png

This blue contextual menu image will prevent terrible edits



Delete_Blue.png

This one tells us many different things. Notice also 'All' is not blue

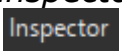
SLF_01_SelectionFilter_All_ticked.png

HANDS-ON - Restore Defined State - 1

Learn - make image 'Inspector' and determine search area

Our screen is in the Defined State and we have liberated the **/** and **Z** key.

We're going to make *IM_01_Inspector_Inspector.png*


or in the black theme


Launch PixelMousing. If the tooltip does'nt show press CapsLock.
Launch the Snipping Tool. The tooltip now shows the coordinates relative to the active window Snipping Tool.

Press Control + N to start a new snip. The crosshair cursor appears  or 
Steer this cursor to the **T**op**L**eft**C**orner of 

and finetune the position with keypresses. **/ + arrowkeys** moves one pixel, **/ + L,R,U,D** moves 25 pixels.
Press Enter. This makes the screen the active window.

The tooltip now shows the screen coordinates of the **TLC** e.g. 

Write the coordinates down.
Or press **/ + C** to copy them to the clipboard, run Notepad and paste them.
Steer this cursor to the **B**ottom**R**ight**C**orner of 

and finetune the position with keypresses.

The tooltip now shows the screen coordinates of the **BRC** e.g. 

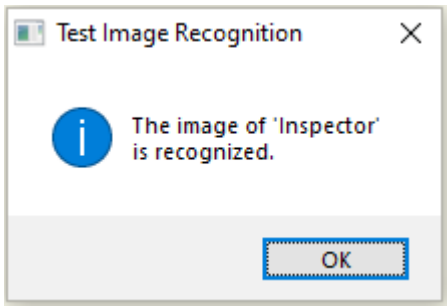
Again make a note of the coordinates or copy/paste them in Notepad.
Press Enter and save the image as .png in your working directory.
Give it exactly the same name: IM_01_Inspector_Inspector.png
Next step: determine its search area. You could use the found coordinates of **TLC** and **BRC**. (I prefer the search area to be one or two pixels bigger.
To me that works as an extra check to keep the attention focused!)

```
IM_01_X1 := 1639      ; one px more to the left       TLC X minus 1
IM_01_Y1 := 23        ; one px higher                 TLC Y minus 1
IM_01_X2 := 1697      ; one px more to the right      BRC X plus 1
IM_01_Y2 := 43        ; one px lower                  BRC Y plus 1
```

Enter your found values in Coordinates_stripped.ahk

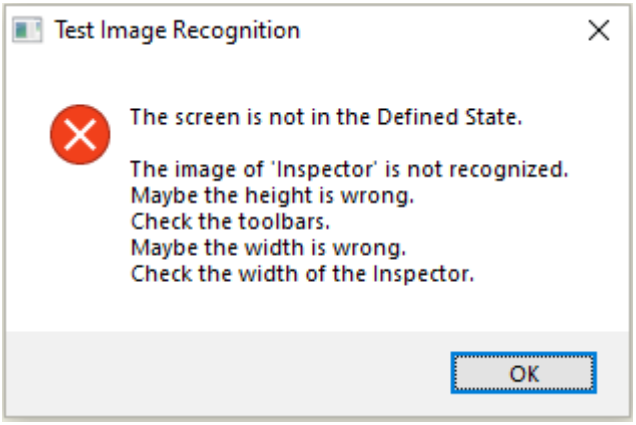
To **test the image** we have a hotkey in Master_stripped.ahk.
Z + 5 Test Image Recognition IM_01_Inspector_Inspector.png 

NB: again **Z** as prefix key. See the *List of prefix keys*.



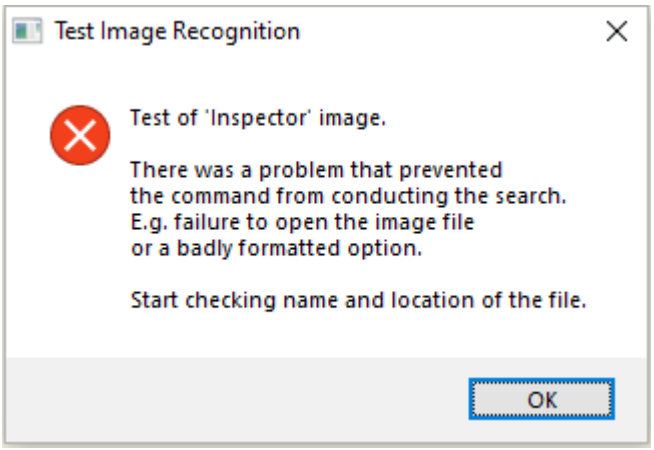
The recognition tells us that the image itself functions correctly and is recognized within 40 shades of variation - the ***40** command option.

Also our coordinates of the search area are correct.



BTW: This error message does'nt mention the possibility that we have forgotten to make the Inspector visible!

Anyway the image has not been found in the search area.



Something prevented the search. Several possible causes. Almost all ImageSearches have the ***40** option. No other options are used.

The images work correctly on the test system but they are all in the Light theme. In the dark theme only a few images were tested. They functioned all right as well.

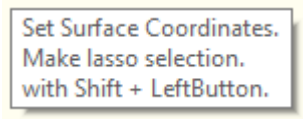
You are invited to test each image before you start a hotkey which uses it.
In the big Master.ahk there is a utility to test all images:
Test Image Recognition: **Z + U** → (InputBox:) **TIR**
This is of course especially important after an update of MuseScore.

Alternative method to make images and determine their search area.

When you realise the potential of AutoHotKey a moment can arrive where you want to make *a series of images in a row together with their surface areas*. As mentioned there is a dedicated hotkey for this job in Master_stripped.ahk. After your first try-outs it works remarkably fast. Practice the examples!

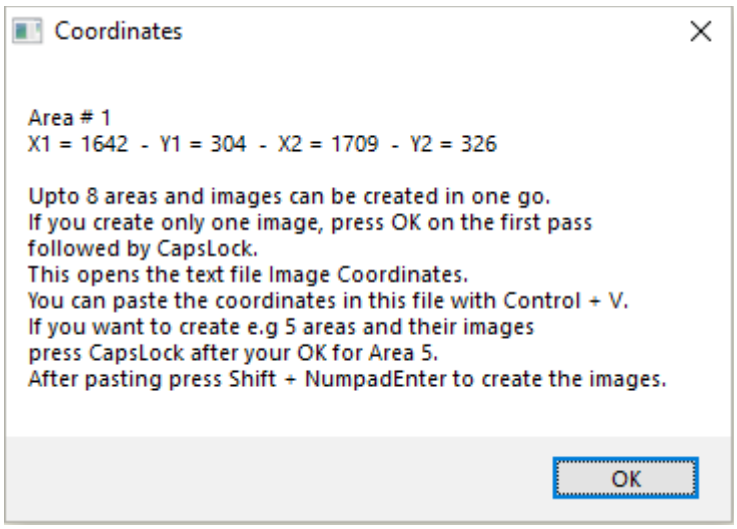
[+ NumpadEnter This command is in Master_stripped.ahk. It launches the separate file 'Set_Surface_Coordinates.ahk'.
Once you have determined the coordinates you can create the images with **Shift + NumpadEnter** which resides in 'Set_Surface_Coordinates.ahk'.

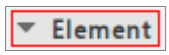
Tooltips and messages guide you through the process.



Example: we want to make these 4 surfaces and images.

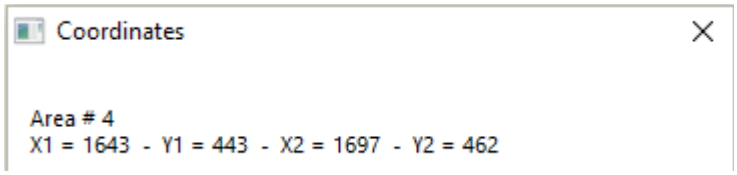
The thin red border marks the image but is not part of it.





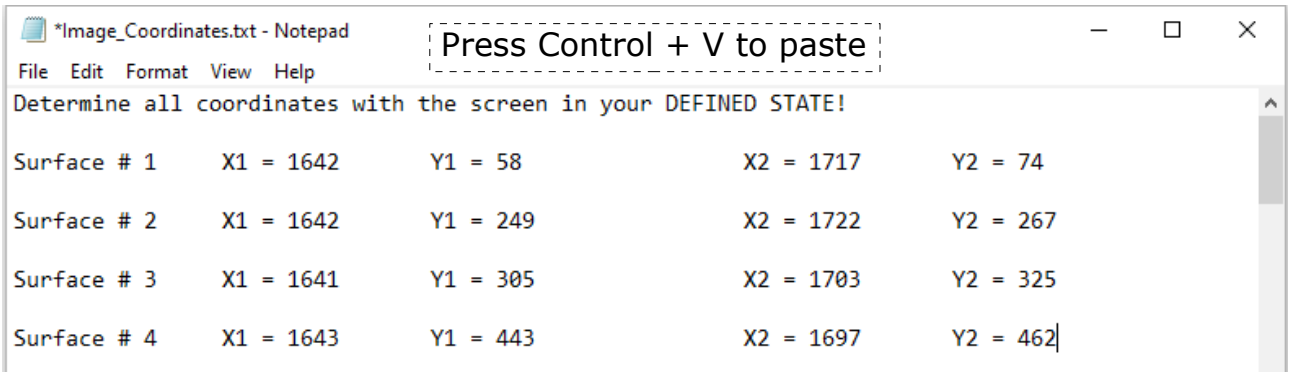
Aided by the thin red border making a close fitting rectangle is a no-brainer.

When you release the left mouse button the message appears about Area # 1.
We press OK and make the second rectangle:

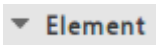


The picture shows only the top of the message.

After making this fourth rectangle press **CapsLock**.
A supporting text file appears in which we paste the coordinates.

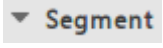


Image_Coordinates.txt is the image part of Coordinates.ahk. It includes the variables of all surfaces. After entering the numbers in this text file you can copy them later to Coordinates_stripped.ahk c.q. Coordinates.ahk.

Example 

surface #1

```
; ImageSearch, , , IM_13_X1, IM_13_Y1, IM_13_X2, IM_13_Y2, *40 IM_13_Inspector_Element.png
IM_13_X1 := xxxx    ; enter here        PM:    TLC X subtract 1 or 2
IM_13_Y1 := yyyy    ; the values               TLC Y subtract 1 or 2
IM_13_X2 := xxxx    ; found                    BRC X add 1 or 2
IM_13_Y2 := yyyy    ; by you                   BRC Y add 1 or 2
```

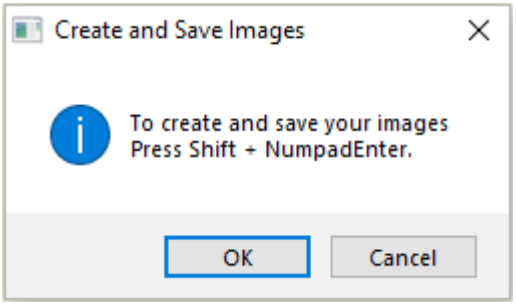
Example 

surface #2

```
; ImageSearch, , , IM_14_X1, IM_14_Y1, IM_14_X2, IM_14_Y2, *40 IM_14_Inspector_Segment.png
IM_14_X1 := xxxx    ; enter here        PM:    TLC X subtract 1 or 2
IM_14_Y1 := yyyy    ; the values               TLC Y subtract 1 or 2
IM_14_X2 := xxxx    ; found                    BRC X add 1 or 2
IM_14_Y2 := yyyy    ; by you                   BRC Y add 1 or 2
```

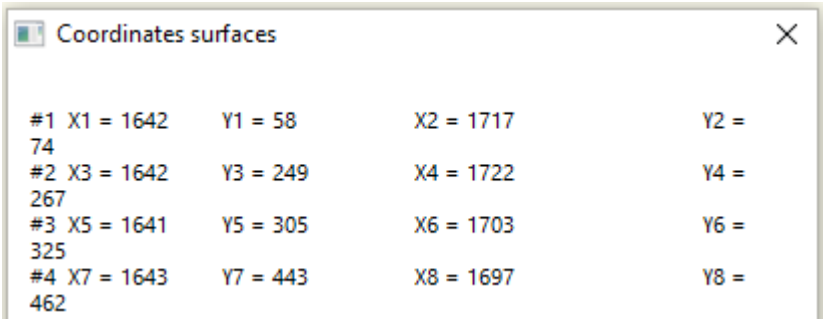
The images 'Element" and 'Segment' are actually needed. 'Chord' and 'Note' **not**.

After saving Image_Coordinates.txt and exiting it this message appears:



Pressing Cancel will exit the macro.
The coordinates will remain in the clipboard until the next copy action.

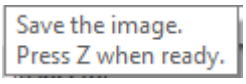
After pressing OK we press
Shift + NumpadEnter and again we get a message.



This message shows the coordinate pairs for each of the four surfaces.
If something went wrong at the prior pasting in Image_Coordinates.txt we can still make notes manually.

The picture shows only the top of the message.

After pressing OK the Snipping Tool* appears and creates immediately the image of Area #1. A tooltip appears:



Save the image in your working directory. Give it exactly the same name as in Coordinates.ahk

If you don't want to save just press **Z**. Do not exit the Snipping Tool!
After pressing **Z** the Snipping Tool makes immediately the image of Area #2.
The process repeats itself, in this case until the image of Area #4 has been created.

* When the Snipping Tool fails to launch you need to change the path.
The usual path is `Run, C:\windows\system32\SnippingTool.exe`
Change this in `Run, "C:\Windows\Sysnative\SnippingTool.exe"`
To change the path:
Select the file 'Set_Surface_Coordinates.ahk'. Rightclick to edit.
Press Control + F and copy the usual path in the search field. Press 'Find Next'.
Enter a semicolon followed by a space before this line to outcomment it.
Delete the semicolon before the next line to make it active.

Things you maybe want to change: like the position of tooltips, the size and position of a unique window - e.g. that of 'Image_Coordinates.txt '. - and some other unique items where using variables is superfluous.
In this and other macros you can search for this type of things. They are marked with **(*change*?)** and give additional info.

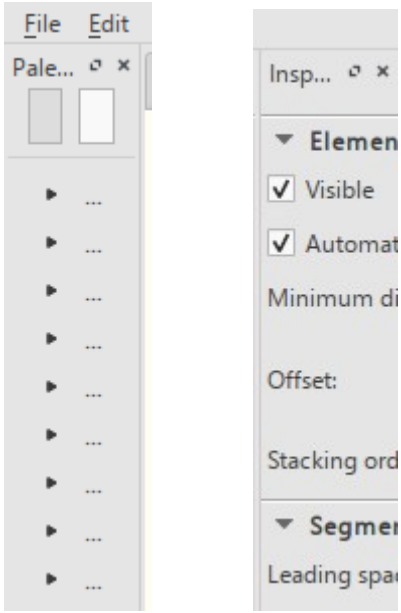
We'll return to **'Learn'** on page 26 after the next topic:
Restoring the Side panels.

DEFINED STATE - Width - Restoring the side panels

Both side panels: method 1

Z + -	Reset Width Palettes	These hotkeys reside in
Z + =	Reset Width Inspector	Master_stripped.ahk

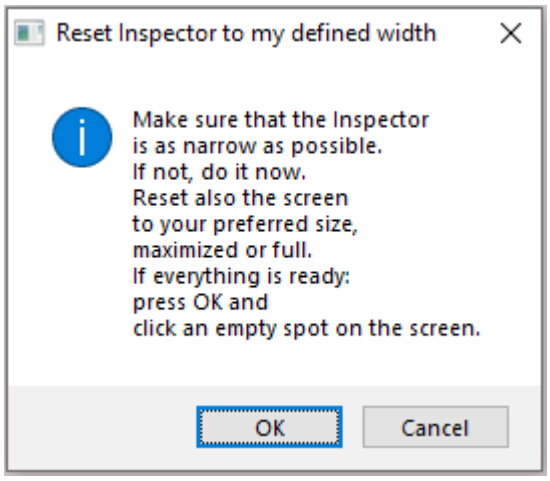
The pictures show Palettes and Inspector *at minimal width* in my personal favorite *MAX CANVAS* - Full Screen, no toolbars.



On any system the *narrowest* width of the side panels will always be the same. When we have to reset the width we could start with manually reducing the width to its minimum and press a hotkey combination. The macro steers the mouse to the X-coordinate of this minimum width, enables drag, moves the separator to the desired width of the defined state and disables drag.

To be able to do this we need to assign values to readable variables, like 'Minimal_Inspector_Width' and 'Defined_Inspector_Width'. So for the test system:

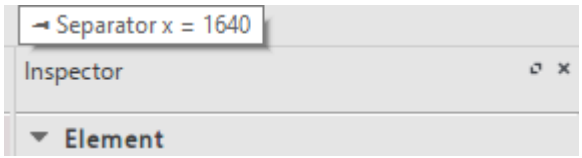
```
Min_Insp_Width := 1853          Min_Pal_Width := 67
Def_Insp_Width := 1640          Def_Pal_Width := 111
```



We press **Z + =**
This message appears.
We arrange the screen, press OK and click on an empty spot.
The width of the Inspector is adjusted.
The mouse returns to its point of departure.

We press **Z + -**
The reset of the Palettes width follows the same procedure.

There is a calming appeal in having an on-screen indication that the Inspector has its defined width:



This tooltip doesn't need new coordinates.

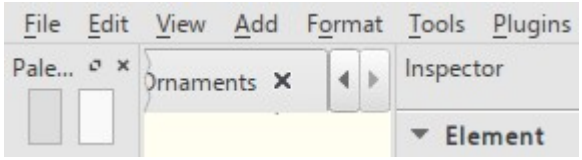
Its X is Def_Insp_Width.
The Y is 0, as high as possible.
In the tooltip text the number ('1640') appears automatically.

For the comparable Palettes width indication: see the next section.

Inspector panel: method 2

As we can surmise there is a second way to restore the side panels. With a bit more work we don't need a prior 'reset to minimum width'.

Irrespective of the width of the **Inspector** this image will always be visible: 
Notice it is a small part of the word without the dots which occur when the Inspector is very narrow.



Inspector at maximum width

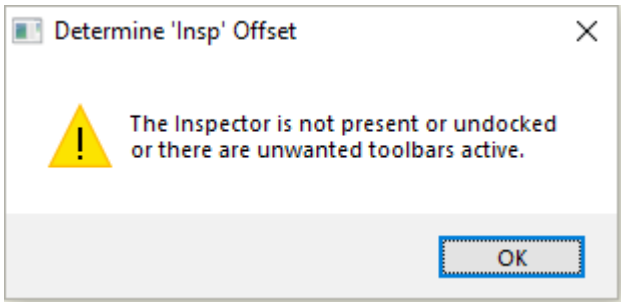
Look for this image in *the ribbon from minimum to maximum Inspector width*. AutoHotKey will find its **TLC**. Obviously this is a point a tiny bit above and to the left of the capital I of 'Insp'.

```
ImageSearch, X, Y, IM_01_2_X1, IM_01_2_Y1, IM_01_2_X2, IM_01_2_Y2, *40 IM_01_2_Inspector_Displaced_Insp.png
IM_01_2_X1 := 198   IM_01_2_X2 := 1919    The ribbon like surface with its 2 corners.
IM_01_2_Y1 := 23    IM_01_2_Y2 := 51      The TLC of 'Insp' is stored in 'X' and 'Y'.
```

With PixelMousing we determine the small horizontal distance between this **TLC** (on X and Y) and the point where the Separator becomes active. This distance is always the same irrespective of the width of the Inspector.

```
Insp_Loc_Dist_X := 2      This distance acts as an offset. The separator is to the left
                          of the TLC of 'Insp' so this distance will be subtracted.
```

To facilitate the whole procedure there is a hotkey for it: **Z + F2**



Oops! It looks like we forgot something. IM_01_2_Inspector_Displaced_Insp.png for some reason can't be found.

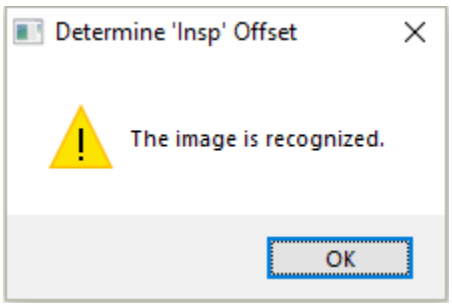
The search is comparable with the search for IM_01_Inspector_Inspector.png

So we better put things right first.

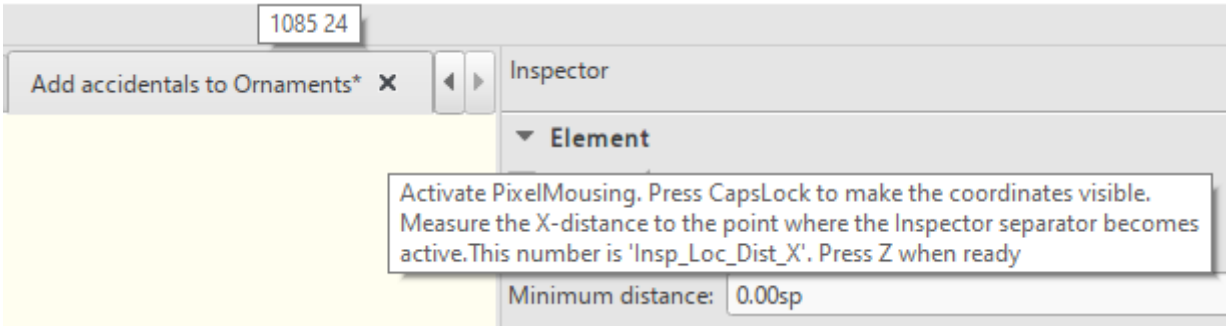
Next try: **Z + F2**

That looks better.

PM: a quick way to create and test images will be essential especially after an update of MuseScore. Rapid maintenace is key.
The kit has tools for this.



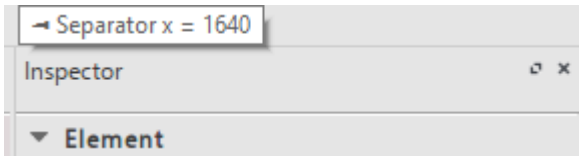
We press OK. Something like the image on the next page appears. The mouse lands on the **TLC** of 'Insp'. It is at 1085, 24



For the test screen we got: `Insp_Loc_Dist_X := 2` (Inspector Locator Distance)
The Y-coordinate we get for free. AutoHotKey has a built-in variable 'A_ScreenHeight'.
Its value it derives from the Windows operating system. The click will be at half screenheight.

Now AutoHotKey has enough data and is able to click and drag the Inspector separator to its defined width. PM: `Def_Insp_Width := 1640`

Hotkey: **Z + 2** **Reset Width Inspector**
It is attractive and calming to have an on-screen indication that the Inspector has its defined width:

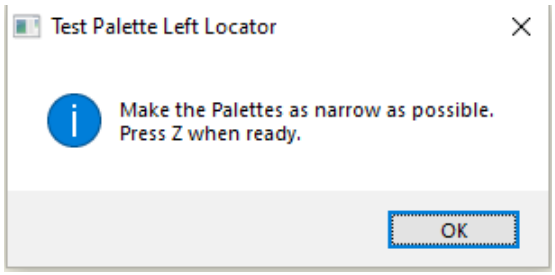


This tooltip doesn't need new coordinates.

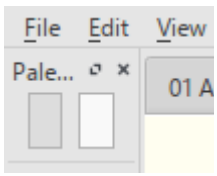
Its X is Def_Insp_Width. Its Y is 0, as high as possible.
In the tooltip text the number ('1640') appears automatically.

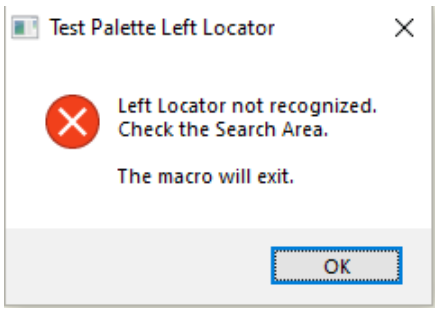
Palettes/Selection filter panel: method 2

To reset the Palettes/Selection filter we follow an almost identical way.
Hotkey: **Z + 1** **Reset Width Palettes**
To facilitate the whole procedure there is a hotkey for it: **Z + F1**

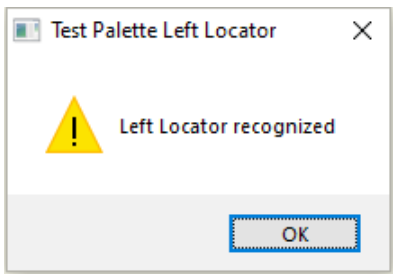


Palettes as narrow as possible.

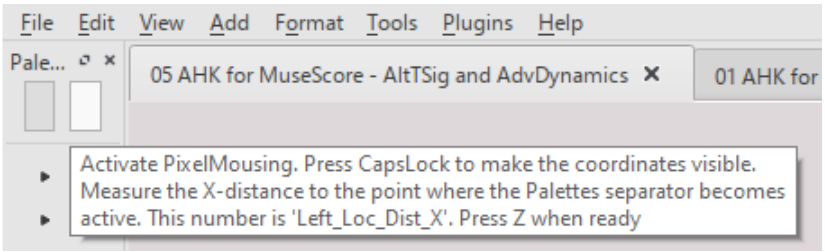


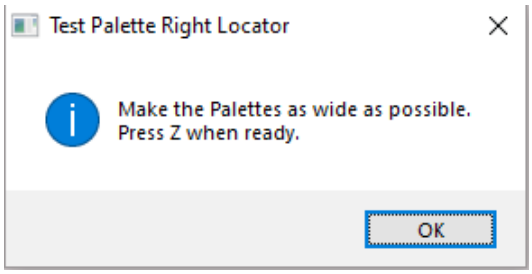


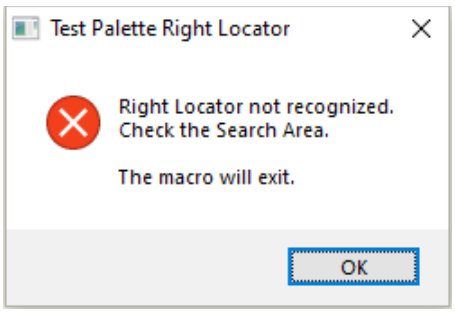
Something went wrong...



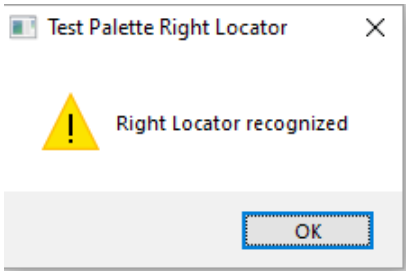
After next try: ***Z + F1***

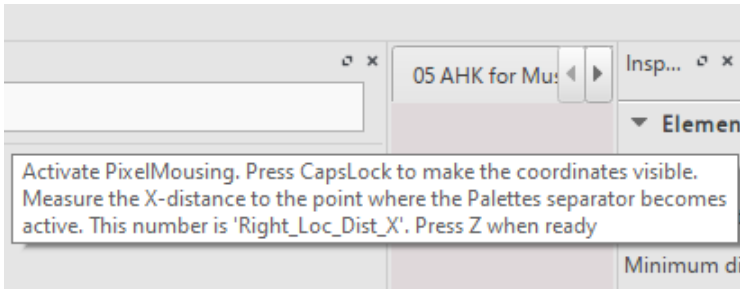






NB:
If the macro exits pressing **Z + F1** starts the Left locator part again. Also: 'Escape' acts as OK





In this case we need **two** images, the 'Right' variant has more 'leading space'.

Palette_Locator_Left.png Palettes at minimum width
Palette_Locator_Right.png Palettes at maximum width

Of course we have to determine the search area for these two commands.
The search area for both is the same ribbon described by its two corners.

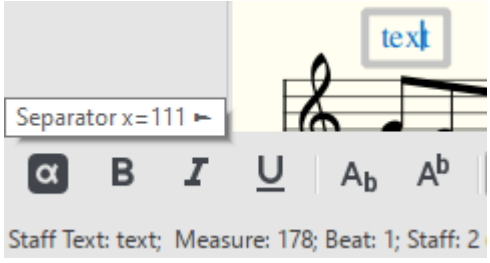
```
ImageSearch, X, Y, Pal_Loc_X1, Pal_Loc_Y1, Pal_Loc_X2, Pal_Loc_Y2, *80 Palette_Locator_Left.png
ImageSearch, X, Y, Pal_Loc_X1, Pal_Loc_Y1, Pal_Loc_X2, Pal_Loc_Y2, *80 Palette_Locator_Right.png
Search area for the test screen:
Pal_Loc_X1 := 36        Pal_Loc_X2 := 1728      The found values of the TopLeftCorner
Pal_Loc_Y1 := 18        Pal_Loc_Y2 := 43        are stored in X and Y.
```

After pressing **Z + F1** we have found two distances from two **TLC**orners

```
Left_Loc_Dist_X := 30     These distances act as offsets. The separator is to the
Right_Loc_Dist_X := 42    right of each TLC so this distance will be added.
```

Now we have enough data. After pressing **Z + 1** the macro tries first to find the Left Palette Locator. If this small image is not found it tries to find the Right Palette Locator. When one of the two images is recognized it drags the separator to the Defined Palette Width.

Here too there is a calming appeal in having an on-screen indication that the Palettes and especially the Selection Filter have their defined width.



This tooltip needs one new coordinate.

```
The X is 'Def_Pal_Width' which we have already got.
The Y is 'Pal_Sep_Height' which we have to choose.
```

The picture shows the optimal position

```
Pal_Sep_Height := 1000  (for the test screen)
```

HANDS-ON - Restore Defined State - 2

Summary DIY - the Side panels - All hotkeys are in Master_stripped.ahk

Semi-automatic
Z + - Reset Width Palettes
Z + = Reset Width Inspector

Determine your own coordinates and put them in Coordinates_stripped.ahk
```
Min_Pal_Width := 67
Def_Pal_Width := 111
Min_Insp_Width := 1853
Def_Insp_Width := 1640
```

Automatic
Z + 1 Reset Width Palettes
Z + 2 Reset Width Inspector

Z + F1 *Palettes tool*
Make these images and put them in your working directory

Palette_Locator_Left.png Palettes at minimum width

Palette_Locator_Right.png Palettes at maximum width

Determine height of 'Tooltip Separator ' : **Pal_Sep_Height := 1000**

Determine your own coordinates and put them in Coordinates_stripped.ahk
```
Pal_Loc_X1 := 36          ; TLC-X search area
Pal_Loc_Y1 := 18          ; TLC-Y
Pal_Loc_X2 := 1728        ; BRC-X
Pal_Loc_Y2 := 43          ; BRC-Y
```
Offsets
```
Left_Loc_Dist_X := 30
Right_Loc_Dist_X := 42
```

Z + F2 *Inspector tool*
Make this image and put it in your working directory


IM_01_2_Inspector_Displaced_Insp.png

Determine your own coordinates and put them in Coordinates_stripped.ahk
```
IM_01_2_X1 := 198         ; TLC-X search area
IM_01_2_Y1 := 23          ; TLC-Y
IM_01_2_X2 := 1919        ; BRC-X
IM_01_2_Y2 := 51          ; BRC-Y
```
Offset
```
Insp_Loc_Dist_X := 2
```

The values in Coordinates_stripped.ahk are for learning.
They have to be copied to Coordinates.ahk, the real depository.

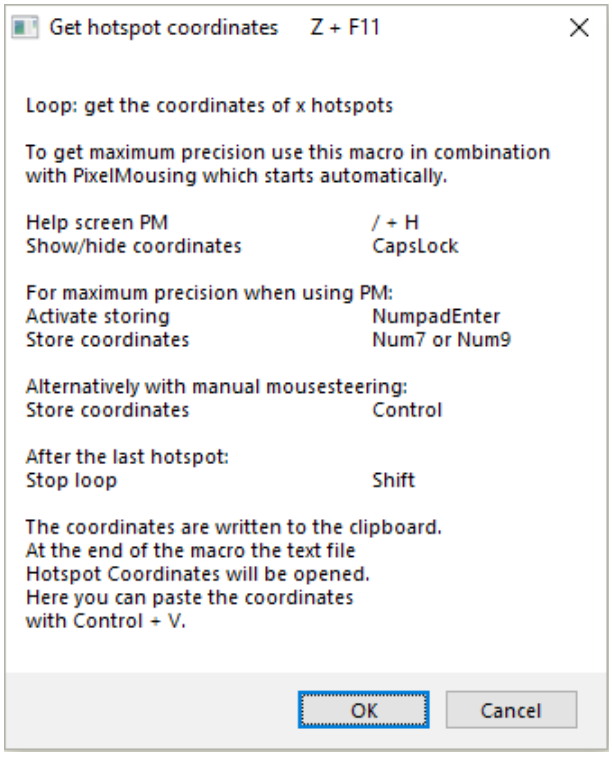
Supplement - determining coordinates

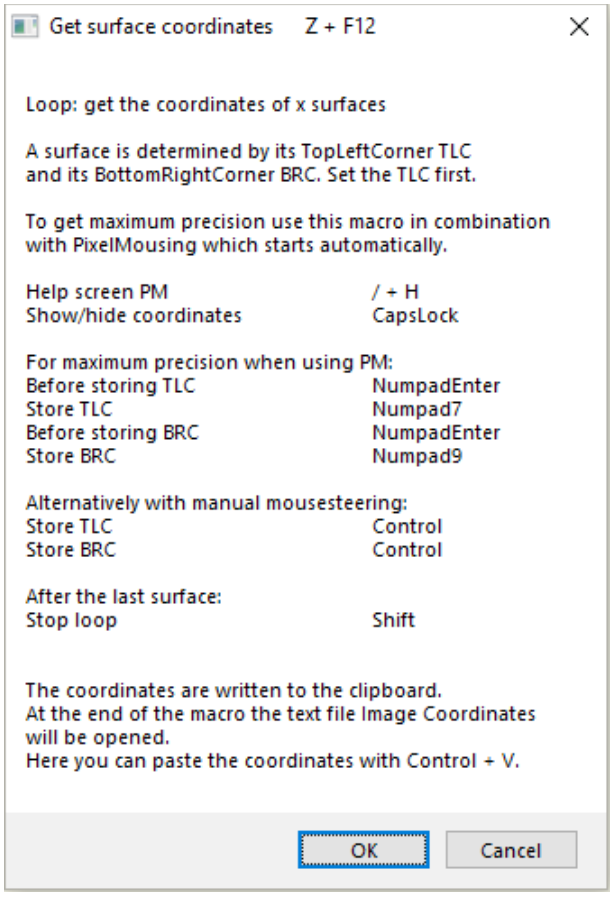
Maybe you prefer just to determine many coordinates of hotspots and surfaces in one go and to create the images later.

Z + F11 get coordinates of hotspots
Z + F12 get coordinates of surfaces

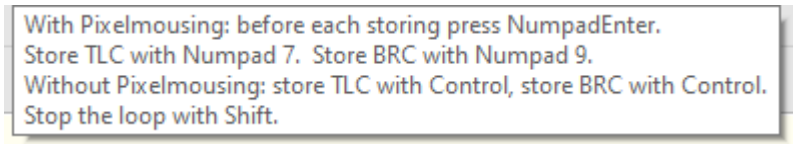
After having stopped the loop by pressing Shift you can immediately paste the coordinates in the supporting text files.
Keep track which number refers to which hotspot c.q. surface.

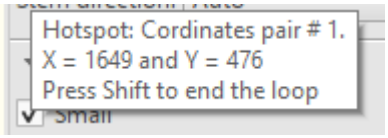
These macros operate independently.
They don't need additional data.





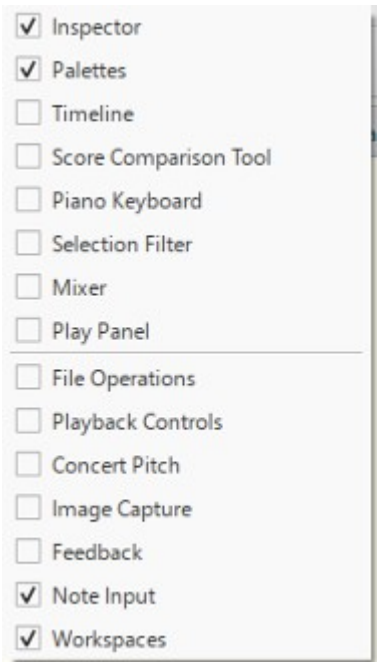
These commands are in Master_Stripped.ahk and in the utilities section of the full version.
Tooltips assist you in the process. The *Store* command has two variants.





In the next paragraph we focus once more on the Defined State, but now the vertical aspects: toolbars on the screen and collapsed sections in the Inspector.

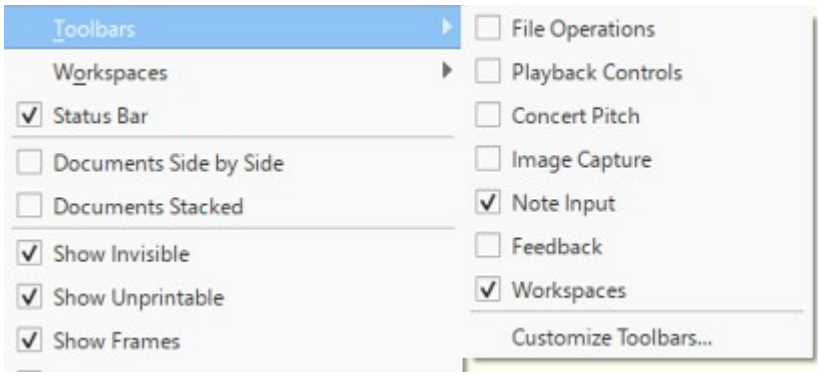
DEFINED STATE - Height - Toolbars



A: Right-click on 'Menu ribbon'

There are two ways to get info about the toolbars. The macros use picture A - the right-click method. Its execution speed is higher with less strain on the eyes when the windows flash by. And there is a bonus: Pic A includes extra info we can use for other macros.

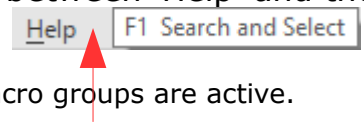
These images have been captured from a prior PrintScreen capture copied to a large size document.



B: Selection via Menu 'View'

NB: *At startup verify that the upper section of Picture A includes 'Play Panel'.*
If not, switch the Play Panel on and off. Now it will appear under 'Mixer'.
Only now the coordinates of the lower section will be right!

With the screen in your Defined State: (full? maximized?)

Choose the spot which must be right-clicked. Halfway between 'Help' and the tooltip and in in the middle of the height of the ribbon. 

In the highest ribbon tooltips can appear to remind you which macro groups are active.

The macro searches here for  . When it finds the image its **TLC** is stored in the Output Variables.

ImageSearch

Searches a region of the screen for an image.

```
ImageSearch, OutputVarX, OutputVarY, X1, Y1, X2, Y2, ImageFile
```

Now follows a description of two cases.
Case 1: In my Defined State there are no toolbars visible.
Case 2: In my Defined State the toolbars Note Input + Workspaces are visible.
If you prefer other toolbars to be visible these descriptions give sufficient info to adapt the macro to your preference.

In Coordinates_stripped.ahk you find extra information. Making this macro work may look challenging at first. But it is an excellent practice in mastering AutoHotKey for MuseScore. *More hands-on? Use Coordinates_stripped.ahk*.

About the making of the image ☑ Use the √ square of ☑ Play

Try PixelMousing for this precision job. The Snipping Tool sometimes refuses to create very small images. This explains the surface to the right of √ in ☑

Case 1: Defined State - no toolbars visible

In the big Master file and in Master_stripped.ahk there is the hotkey **CapsLock + 8** with the line:

```
ImageSearch, Tick_X, Tick_Y, CTS_X1, CTS_Y1, CTS_X2, CTS_Y2, *40 CTS_Checked_ticked_sign.png
```

The search area is a narrow column containing all small squares from File Operations to Workspaces. Preferably its width must be a tiny bitter bigger than that of the image. We right-click the 'toolbar selection spot' and determine the coordinates of the search area.

The coordinates of the 'toolbar selection point' are **ToolBarSel_X,ToolBarSel_Y**

☑ The **TLC** of the search area is **CTS_X1, CTS_Y1**.
The **BRC** of the search area is **CTS_X2, CTS_Y2**

And in the file Coordinates_stripped.ahk resp. Coordinates.ahk we'll input the actual values of the coordinates:

So *for the test screen* we got: (the position of **ToolBarSel** see p.28)

```
ToolBarSel_X := 330        CTS_X1 := 329        CTS_X2 := 357
ToolBarSel_Y := 9          CTS_Y1 := 177        CTS_Y2 := 369
```

When the macro finds the first ☑ it stores the coordinates of its found **TLC** in **Tick _X** and **Tick_Y.** As we have done earlier in similar cases we add a small horizontal and vertical *offset* to the **TLC**. We want it to click more in the centre of the square. This prevents accidentally hitting separators. A few pixels more to the right and down. We determine this offset with PixelMousing.

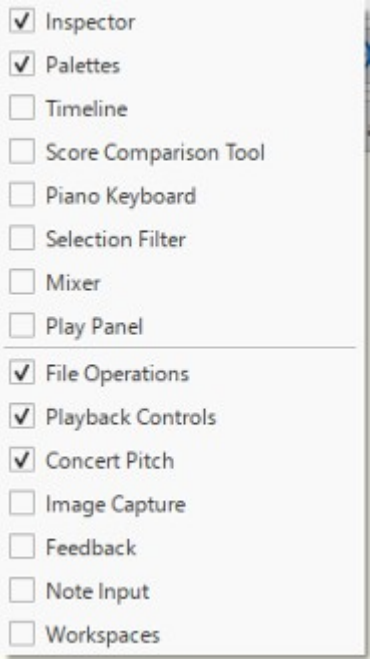
In the file Coordinates_stripped.ahk resp. Coordinates.ahk we get *for the test screen*:

```
ToolBTickOffset_X := 4
ToolBTickOffset_Y := 4
```

The macro operates in a loop. It will click on each ☑ it finds and exits when all ticks are unticked. Now the toolbars Note Input and Workspaces have disappeared. And no other toolbar is visible.
In Full Screen this macro restores MAX CANVAS if the width of the side panels is already in the defined state.

PS: In the big Master file there is a series of hotkeys to toggle individual toolbars or *combinations of toolbars*. **CapsLock + 1 resp. 2,3,4,5,6,7,8**

Case 2: Defined State - Note Input and Workspaces



When you input note durations, dots, rests, ties etc. via the computer keyboard you may prefer the toolbar *Note Input* to be visible so you can monitor your actions. And may be also the toolbar *Workspaces*. It is present in the same 'ribbon'. It will not diminish the height of the Canvas. But at the moment these toolbars are not visible for some reason.
Say you have just used an icon in 'File Operations', another in 'Playback Control' and checked for 'Concert Pitch'. The picture shows the state of your toolbars.

To return to your Defined (toolbar) State the two lowest squares have to be ticked and all others must get in the unticked state.

The job of the macro will by now be pretty clear.
Again we have to look for the Checked ticked sign √

With the screen in your desired Defined State: (full? maximized?) *after a right-click on the toolbar selection spot:*

"Search in the narrow column with the small squares from 'File Operations' to 'Feedback'. When a match of √ is found click in the small square to untick it. Next search the area in front of 'Note Input'. When √ is found do nothing, otherwise click the square. Finally repeat this procedure for the area in front of 'Workspaces'. "
In the big Master file and in Master_stripped.ahk there is an *alternative* hotkey **CapsLock + 8**.

Use as searchterm CapsLock & 8 (ck & 8 will work as well) to learn how you can simply activate this version and switch off (outcomment) the MAX CANVAS version.

There is a line describing the search area from 'File Operations' to 'Feedback':

```
ImageSearch, Tick_X, Tick_Y, FoFb_X1, FoFb_Y1, FoFb_X2, FoFb_Y2, *40 CTS_Checked_ticked_sign.png
```

And again in the Coordinates file(s) we input the actual values.
F*or the test screen:*
FoFb_X1 := 329 FoFb_Y1 := 203 FoFb_X2 := 357 FoFb_Y2 := 322

These values have been measured in Full Screen. A choice for Maximized Screen as your Defined State of course will produce different values.

Finally we have to determine the small search areas in front of 'Note Input' and 'Workspaces'. We use the same png with different variables.

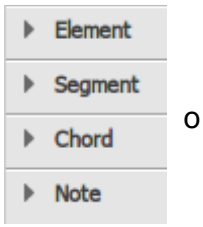
F*or the test screen:*
NoteInp_X1 := 329 NoteInp_X2 := 357 WorkSp_X1 := 329 WorkSp_X2 := 357
NoteInp_Y1 := 323 NoteInp_Y2 := 369 WorkSp_Y1 := 347 WorkSp_Y2 := 369

Many coordinates here have the same values since they describe the same corner. Or they share the same X, the left border of the contextual menu.

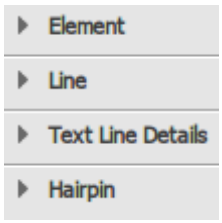
Defined State Inspector - Expanding Collapsed Sections

This finishes the description of the Defined State of the screen.

From page 16: We don't like it if the Inspector is in this state:

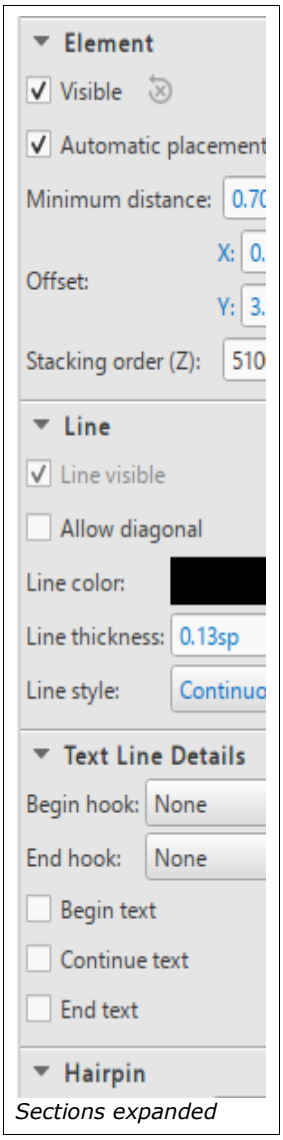


or this

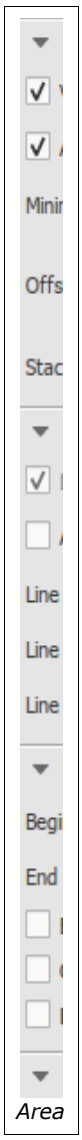


Of course we need all sections of the Inspector in their expanded state. We need a Command: *search* for the triangle *and click* on this image.
IM_18_Inspector_Triangle_Closed.png

Z + I Expand collapsed Inspector sections



Sections expanded



Area

ImageSearch

Searches a region of the screen for an image.

```
ImageSearch, OutputVarX, OutputVarY, X1, Y1, X2, Y2, ImageFile
```

The ImageSearch command searches for an Imagefile in this case **`IM_18_Inspector_Triangle_Closed.png`**.

It searches in an rectangular area determined by a Top Left Corner **TLC** (X1, Y1) and a Bottom Right Corner **BRC** (X2, Y2)

This will be the region of the picture 'Area' at the left. This area includes *all relevant triangles of all the Inspector macros* in the AHK kit.
The search starts at the top row of the region and moves downward. So the first matching graphical object found is the 'closed triangle' of 'Element'.

Once found the command stores its TLC coordinates in the containers named OutputVarX and OutputVarY.
Next: the macro steers the mouse to this found TLC and is able to click the *centre of the triangle* by adding a small horizontal and vertical *offset* to the TLC.
Result: the arrow changes shape and this Inspector section opens.

Once again: this kind of macro must operate in a loop. In its second pass it will find the closed triangle of 'Line', clicks it to open that section and so on until all sections are open and it's time to exit.

Z + I Expand collapsed Inspector sections

This hotkey is also in Master_stripped.ahk
The hotkey includes the important *PixelSearch* command. This will be described in the next topic: how to use colors in MuseScore.

Create the image  **Inspector_Triangle_Closed.png**
Put it in the working directory

In the macro the ImageSearch command looks like this

```
ImageSearch, TrX, TrY, IM_18_X1, IM_18_Y1, IM_18_X2, IM_18_Y2,
IM_18_Inspector_Triangle_Closed.png
```

In Coordinates_stripped. ahk and Coordinates.ahk we have *for the test screen*:

```
IM_18_X1 := 1639  IM_18_Y1 := 54  IM_18_X2 := 1663  IM_18_Y2 := 592
```

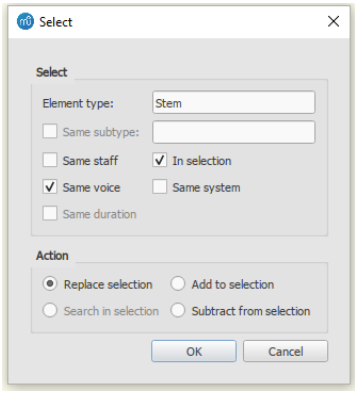
Determine the offset. *For the test screen*:

```
TriAngleOffset_X := 10          TriAngleOffset_Y := 10
```

The **TLC** of the *matching found image* is stored in TrX and TrY. These are *output variables*. Their content will change after each succesful search.

DEFINED STATE of MuseScore Windows - window size

PM: there are two types of coordinates, those *relative to the screen* and those *relative to the active window*.
For instance the window Select; we see it's active because its title is black.



The tool PixelMousing shows the coordinates of the mouse position relative to the active window.
As stated before: this means the actual position on the screen of this window is *not* relevant. But what is very relevant is the size of the window.
Most windows used in the macros have to be *minimized*. This is the easiest way to make the effect of mouse clicks reproducible. MuseScore will remember the size the next time the window will be opened. This guarantees that mouse clicks will be always on the intended spots.

There are a few exceptions where a minimized window does not suffice. In those cases its size must be told to AutoHotkey so that the program can take care of this job. Examples are the window *Staff/Part Properties*, the window *Special Characters*, the *Play Panel* and the window *Style*. Their size will be an item in the short descriptions of the related macros.

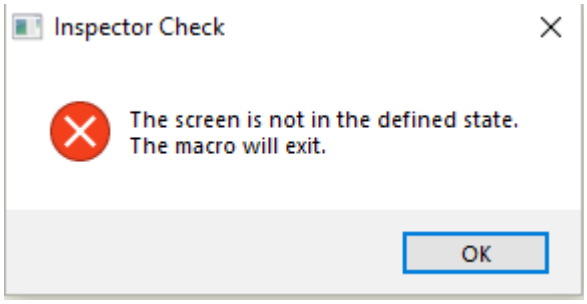
Defined State - Final note

Now the following question has a practical sense:
Is it necessary that each macro starts with a verification of the Defined State and restores it automatically?

In short: that would certainly be overkill.
1 Some macros are 'FREE CANVAS' meaning they don't depend on the state of the screen. They just operate on the Canvas, without reference to the Inspector or any toolbar.

2 Some macros operate only on MuseScore windows with local coordinates.

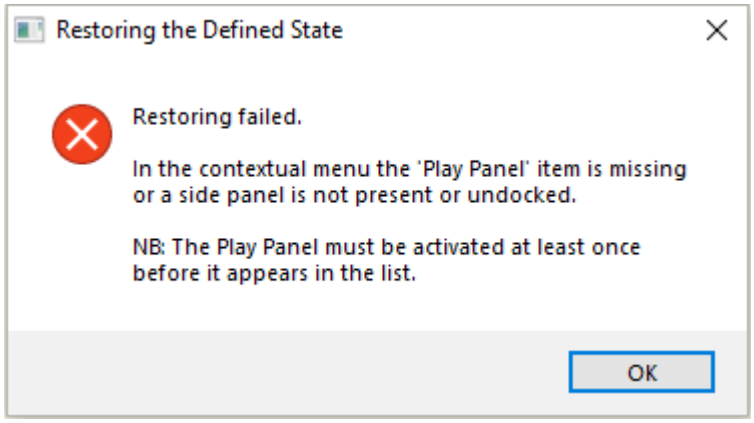
3 Some macros don't need coordinates at all.



The message at the left will only pop up if a verification is essential. Actually it reveals a choice: it's possible to restore all aspects of the Defined State but in practice a manual reset is usually more effective than an automated one.

Restore the Defined State manually (*via one or more hotkeys)*.
IMO this creates a new balance between a new - faster - workflow and automation. And it speeds up macro execution.
When you get used to working in your Defined State mistakes will become increasingly rare.
IMO it's enough to be protected against unpredictable and potentially desastrous *edits* and be warned. In some cases protection involves an extra check of the expanded state of Inspector sections. But in many other cases this is superfluous. One glance at the Inspector and you know: press **Z + I**.
On a related note: potentially desastrous *selections* must and will be aborted

NB: When MuseScore is not active, minimized **Control + Win + Z** restores the Defined State of the screen, width side panels, toolbar state and Inspector sections for the selected element. This hotkey is in *Independent_Hotkeys.ahk*.



Info Screen *Independent Hotkeys*

PM: At startup verify that the upper section of the contextual menu includes 'Play Panel'. Page 28.
If not, toggle the Play Panel on and off. Now it will appear as the last item under 'Mixer'. Only then the coordinates of the lower section will be right!

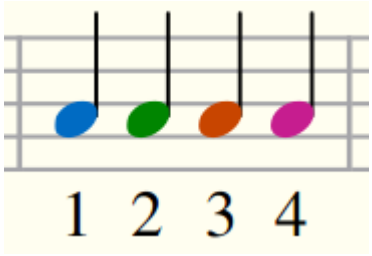
Color Search and Select

PixelSearch, the pivotal command, has much in common with ImageSearch.

PixelSearch

Searches a region of the screen for a pixel of the specified color.

```
PixelSearch, OutputVarX, OutputVarY, X1, Y1, X2, Y2, ColorID [, Variation, Mode]
```



The picture shows the *default* voice colors of the four selected notes. Note 1 is in voice 1, 2 in 2 etc. The colors have numbers which we find in Preferences -> Advanced, the last four entries of the list. ui/score/voice1/color
ui/score/voice2/color etc.

Click a color to see its number.

HTML: #0065bf
Voice 1 RGB
HTML: #007f00
Voice 2 RGB
HTML: #c53f00
Voice 3 RGB
HTML: #c31989
Voice 4 RGB

PixelSearch is key to a box of new features.
Out of a multitude of AutoHotkey commands this one scores high:
Move the mouse to the selected element and click it.
This eliminates an amazing amount of manual mouse movements and corresponding muscular stress.

When the selected note has the headtype of a quarter the note will be found even at a zoomfactor of 30%.

Alt + Z Search color voice 1 and click found element

```
PixelSearch, ElementX, ElementY, % CSA_X1, % CSA_Y1, % CSA_X2, % CSA_Y2, ColorV1, , Fast RGB
```

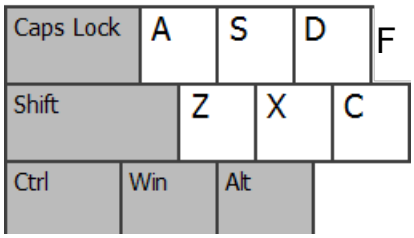
```
CSA_X1 := 122
CSA_Y1 := 55
CSA_X2 := 1639
CSA_Y2 := 1054

ColorV1 := 0x0065bf
ColorV2 := 0x007f00
ColorV3 := 0xc53f00
ColorV4 := 0xc31989
```

These are the coordinates of the *Canvas Search Area*, the surface of the Canvas. When the first pixel of the right color is found its coordinates are stored in ElementX and ElementY. The fast mode searches the screen row by row (top down) in about 50 ms. AutoHotkey writes **0x** instead of the **#** of MuseScore. If you use other voice colors change the default numbers in the Coordinates file. The next five hotkeys are also in F1_ColorSearchSelect_stripped.ahk

Alt + Z search color voice 1 **Alt + X** search color voice 2
Alt + C search color voice 3 **Alt + D** search color voice 4
Alt + S search successively color voice 1,2,3,4

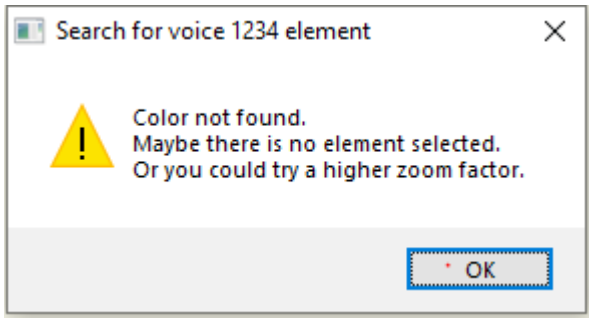
Helpscreen search voice color on page 40

A note about frequently used shortcuts

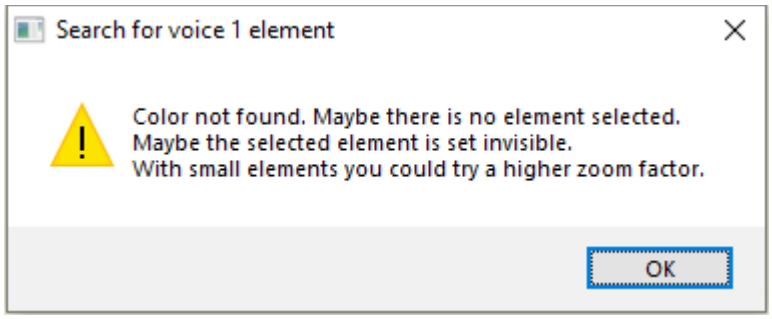


This is one of the most accessible areas of the keyboard. It houses a number of frequently used hotkeys. E.g.
Capslock + A Full Screen
Capslock + S Open MSc shortcuts
Z + A, C, D, F, S or X ...
Alt + Z, X, C, D or S voice colors

Control resp. **Win** resp. **Alt** resp. **Shift + CapsLock** etc.



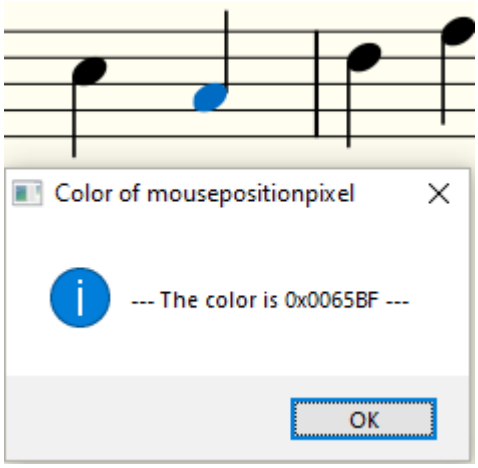
If the element is very small or thin this message can pop up. On the test screen staccato dots are easily found at 90% zoom. Barlines from 110%. Hairpins are found from 130%. Halfnotes from 80%. But the 'black notes' crown it with their 30%.



The same type of message for each voice color separately.

PM: the color of a selected invisible element differs from that of an unselected unvisible element. So other macros could search for these colors.

In Master_stripped.ahk: hotkey **Z + /** Get Color mouse position.



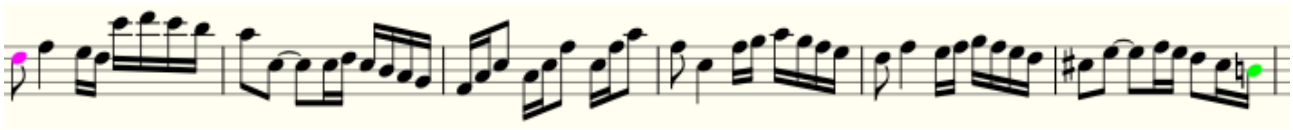
Some colors used by MuseScore are not in the list of colors in Preferences -> Advanced. **Z + /** gives us a useful alternative. Use PixelMousing to position the mouse on thin objects.

*We can expand the function of colors. By giving elements - preferably notes - a **custom color** we create new possibilities to search and select not only on the current screen but also in the whole score. Colored elements become memory spots.*

Example: attach *annotations* to notes with color 5. Make musical ideas searchable with color 6, color themes with 7. And so on: 22 'search colors'. The hotkeys to search the score are in the full version. See the reference section. The stripped down version is primarily meant to illustrate the new *selection tool* we get with AutoHotkey.

Select Colored Range

The colors of the first and the last note can function as a pair.



L + [	set left color of pair 1
L +]	set right color of pair 1
L + =	select colored range 1
L + -	reset colored range 1
L + LeftButton	mouse to left color 1
L + RightButton	mouse to right color 1
L + MidButton	select colored range 1

Experiment with these hotkeys in the stripped version to experience the impact on your workflow of instant reselection.
There is also a variant using the **Y**-key as prefix + Function and number keys.

In the full version there are eight colors to make four pairs. Four independent selections are possible which can be revisited at any moment. This is very useful when editing copied material in other staves.
For fast accessibility 'Colored range selection' has its dedicated hotkeys.
This selection method is also possible via the 'super macro' *Apply Colors*.

Available colors

4 default voice colors of MuseScore	V1	V2	V3	V4			
8 colors in 4 pairs	1 2	3 4	5 6	7 8			
7 'tone colors'	A	B	C	D	E	F	G
7 BWC 'tone colors' *	AB	BB	CB	DB	EB	FB	GB
1 LBC the LayoutBreakColor #	LBC						
1 MT pure white - 'Masking Text' ¤	MT						

* Boomwhackers convention
# Searching Spacers, layout breaks and the + or - sign of irregular measures
¤ Searching the 'invisble' color on a page or in the score

The 8 'paired colors' and the 14 'tone colors' make for 22 'search colors'
The 'tone colors' can also be used to color notes with the same note names following the schemes of the ColorNotes plugin resp. the BWC.

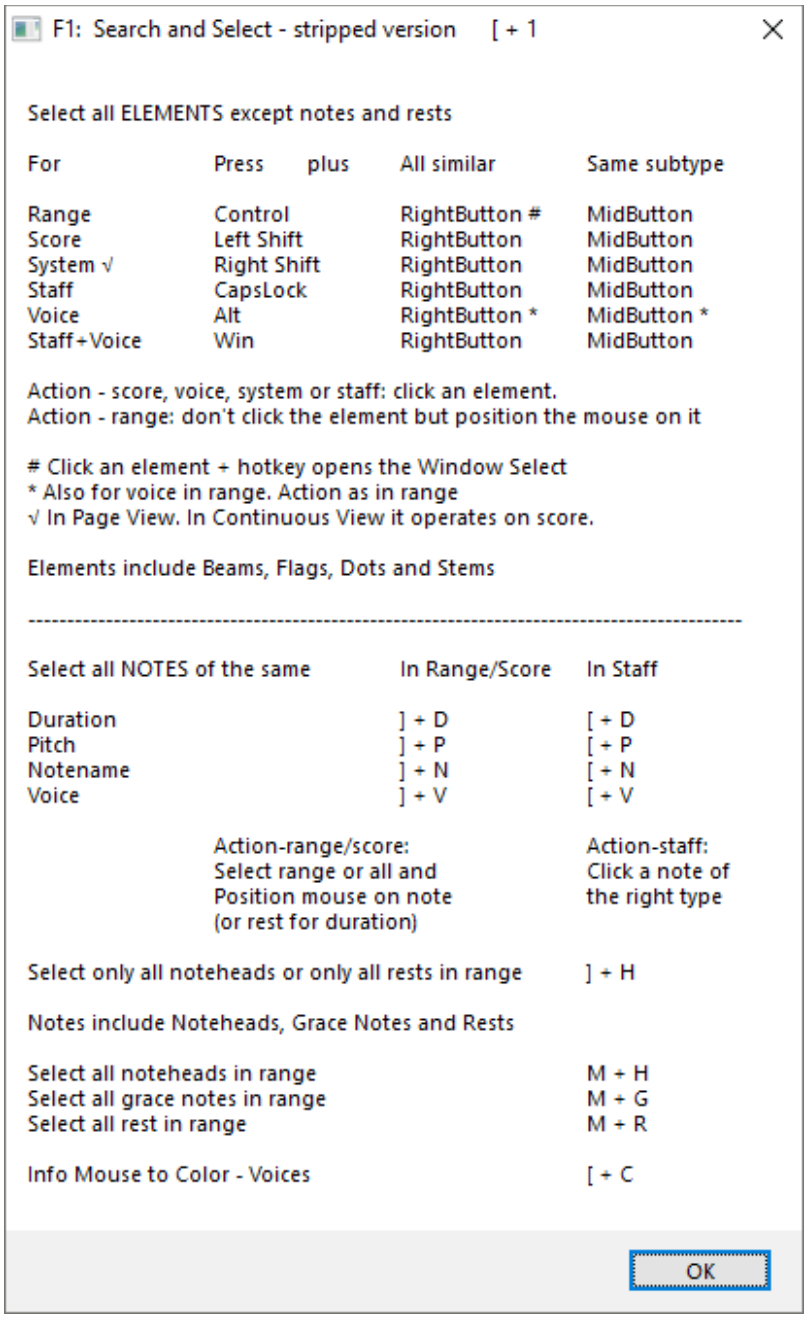
Z + C *'Apply colors'* launches the 'super macro' to set, reset, select, select range and search for colors on page and in score. *Not in stripped version.*

Selection Filter - not in the stripped version

Z + F The macro is completely keyboard driven which again speeds up the workflow considerably. It uses Colored range selection and has an inbuilt copy function to produce cue material which can be used by another macro for the actual production of cues. See reference section.

Selection via contextual menus

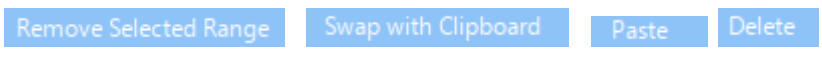
F1_ColorSearchSelect_stripped .ahk includes automated alternatives for the selection possibilities via the right-click contextual menus. IMO these are timesavers even if it concerns a domain in which manual mouse positioning still plays its role.

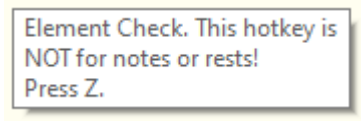


In the stripped version this help screen pops up after pressing **[+ 1**

As of MSc 3.5 there is a *native* alternative way for Range -> All similar selection.
For example:
Click on the first chord symbol in the desired range.
Shift + click on the last chord symbol in the range.
Result: all chord symbols in that range are selected, and nothing else.
This is in normal mode.

The macros have inbuilt protection from potentially desastrous selections.
These could happen when you apply a command meant for elements while you have actually a note or rest selected.
See the F1 Reference section for a further explanation and the determination of offsets on page 56.
In short: when one of the following blue images is recognized during the selection the macro will exit with a warning.





Remove_Selected_Range_Blue.png
Swap_with_Clipboard_Blue.png
Paste_Blue.png
Delete_Blue.png

The next page lists all DIY parts of F1_ColorSearchSelect_stripped .ahk

DIY parts of F1_ColorSearchSelect_stripped .ahk

HOTSPOTS on screen

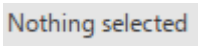
```
IN_002      ; I(002) rectangular Set Color = Reset Color - ELEMENT GROUP
IN_007      ; I(007) Select Notes - ELEMENT GROUP
IN_008      ; I(008) Select Grace Notes - ELEMENT GROUP
IN_009      ; I(009) Select Rests - ELEMENT GROUP
IN_012      ; I(012) color picker black rectangular - OVERLAP all elements
```

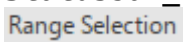
HOTSPOTS in MINIMIZED windows

```
RGB         ; Window Select Color HTML: #RGB rectangular

WS_01       ; (WS_01) Window Select Same subtype
WS_02       ; (WS_02) Window Select Same staff
WS_03       ; (WS_03) Window Select In selection
WS_04       ; (WS_04) Window Select Same voice
WS_05       ; (WS_05) Window Select Same system

WSN_02      ; (WSN_02) Window Select Notes Same pitch
WSN_05      ; (WSN_05) Window Select Notes Same duration
WSN_06      ; (WSN_06) Window Select Notes Same note name
WSN_07      ; (WSN_07) Window Select Notes Same staff
WSN_08      ; (WSN_08) Window Select Notes In selection
WSN_09      ; (WSN_09) Window Select Notes Same voice
```

```
Canvas Search Area:
CSA_X1
CSA_Y1
CSA_X2
CSA_Y2


Voice colors:
ColorV1
ColorV2
ColorV3
ColorV4
```

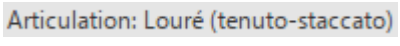
IMAGES 




```
IM_01_Inspector_Inspector.png                + TLC and BRC coordinates
IM_03_Inspector_Nothing_Selected.png         + TLC and BRC coordinates
Remove_Selected_Range_Blue.png               The coordinates of the blue
Swap_with_Clipboard_Blue.png                 images are derived from the
Paste_Blue.png                               mouse position. You have to
Delete_Blue.png                              determine three offsets *
Statusbar_Range_Selection.png                Our first Statusbar image #
```


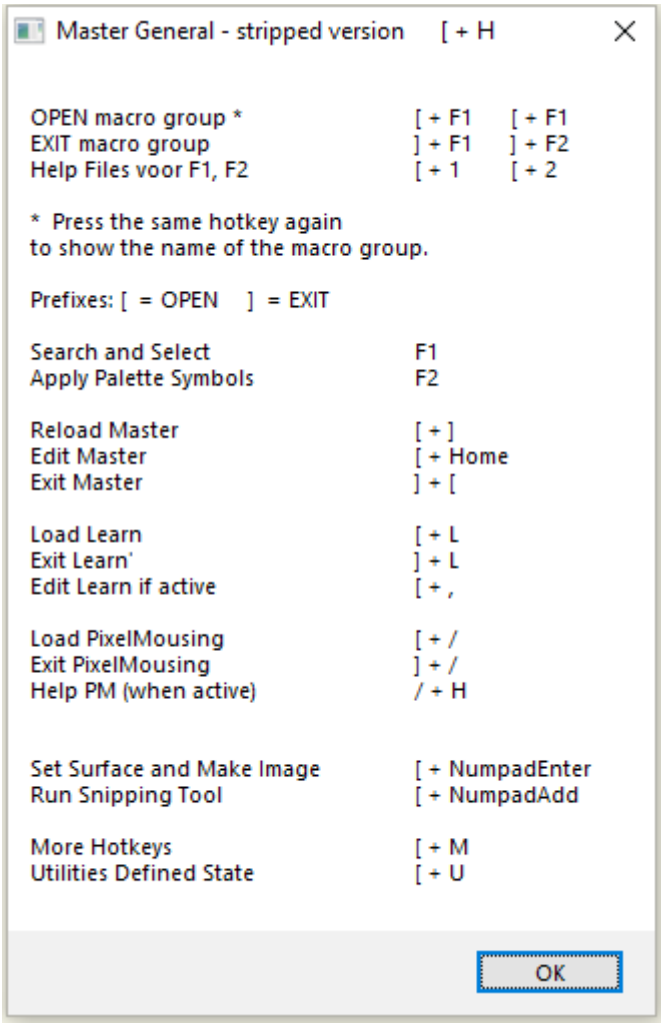

```
* Offsets of the 4 blue images in the Right-Click Contextual Menu:
RC_CM_Y1        See page 56 and the Coordinates file
RC_CM_X2        in the stripped version.
RC_CM_Y2        With info about how to create these images.
```

 # All Statusbar images have the same Search Area

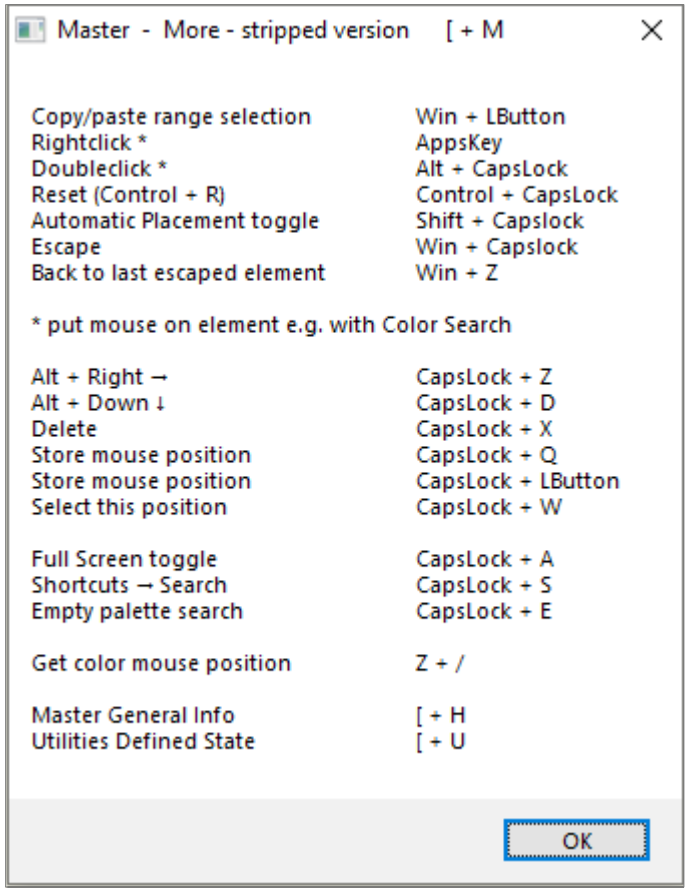
To determine the search area create and select an 'articulation tenuto staccato' (Louré). 

This is the widest surface we need for all Statusbar images. The coordinates are SB_X1, SB_Y1, SB_X2 and SB_Y2.
Make SB_X2 a few pixels bigger than the width of this png.
Set the **height** so that the surface can accomodate **two stacked statusbars**.
More info in the Coordinates file and in the stripped version.
Searchterm: SB_X1

The Master file



◄ Starting peripherals - The Master as Central Station and as General Store of wares and tools ▼

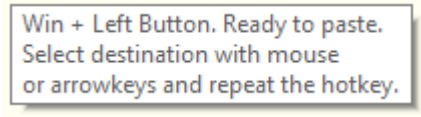


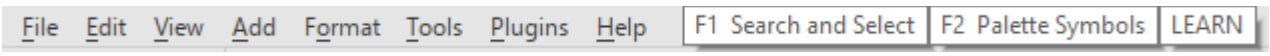
Stripped version

[+ H	Master - help
[+ M	More
[+ U	Utils Defined State
[+ Function key	Run macro group
[+ key	Open - Run
] + key	Exit

In 'More':
More ergonomical hotkeys, some of them with positional 'memory'. MuseScore equivalents remain.
'Empty palette search' useful for the F2 macro group. Tool 'Get color'.

The hotkey Win + Left Button was inspired by the workflow of some forum posters. In other workflows it could be considered superfluous.

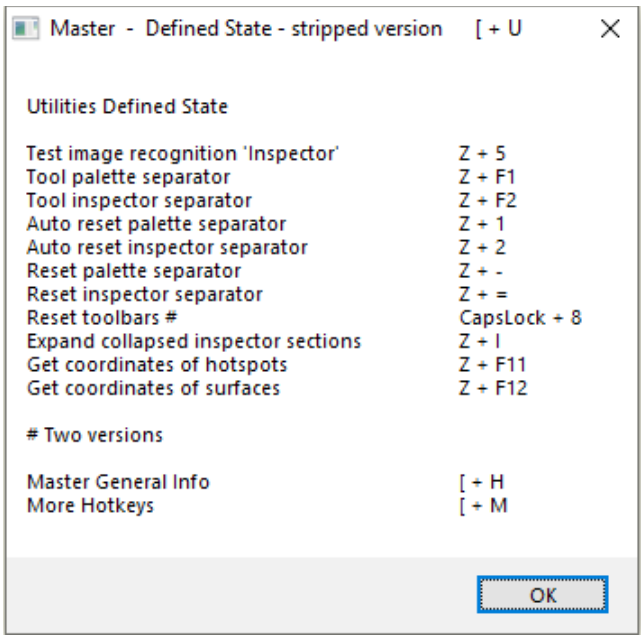




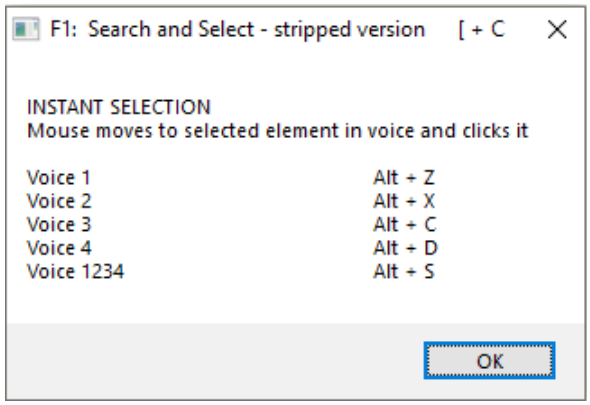
Determine the X-coordinate of the tooltips:

```
ToolT_F1_X := 340
ToolT_F2_X := 463
ToolT_Learn_X := 578
```

'Empty palette search' uses the MuseScore shortcut 'Palette Search': Control + F9



Help screen
Continuation from page 34

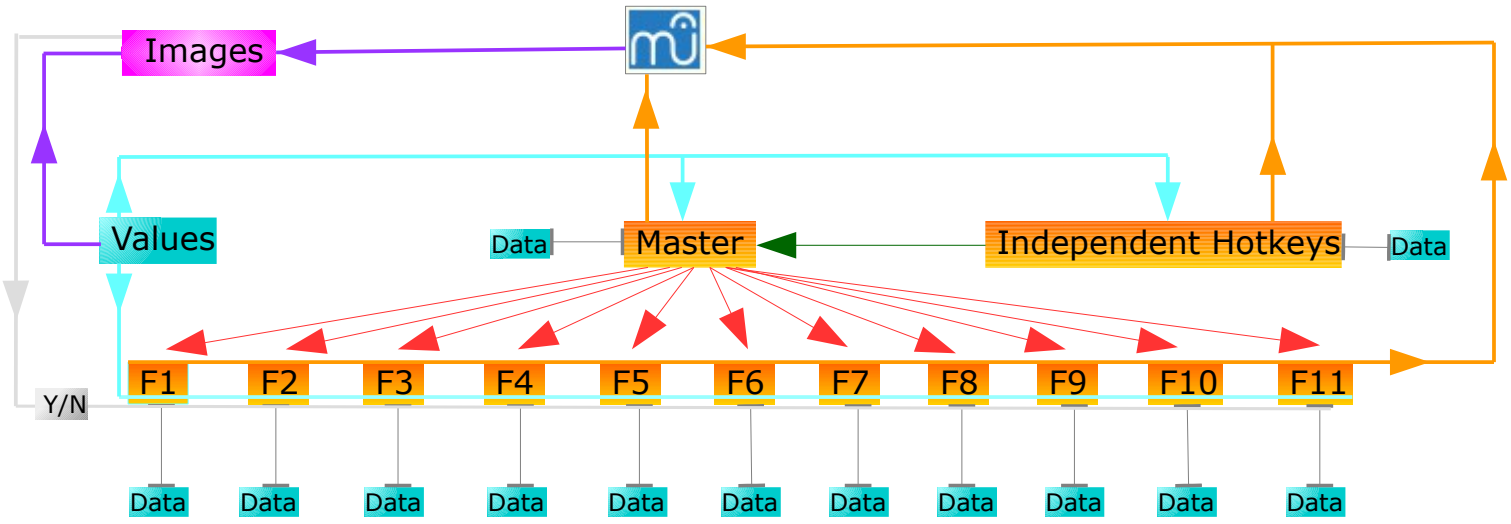


List of prefix keys

See also HotKeys_in_prefix_order.txt. with info about *how to liberate the key* and other keys as *prefix candidates*.

H J K L	are only used for fast colored range selection each prefix makes 7 combinations. See page 36.
M	redefinition MSc shortcut
P	redefinition and/or alternative hotkey **P + K**
Y	after default installation of MSc this key is free
Z	redefinition MSc shortcut
/	redefinition MSc shortcut
**[ ] , **	after default installation these keys are free
****	special backslash ScanCode 056, not on all keyboards only used in F11 macros: note duration by mouse wheel
CapsLock	free - not possible in MSc shortcuts
F2 F3 F5	**F3** and **F5** free after def. install. **F2** free when not in text
F11	playpanel - liberated by alternative hotkey **F11 + F10**
NumpadEnter	free after default installation of MSc
NumpadAdd	idem
NumpadSub	idem
LButton	mouse button is free - impossible in MSc
MButton	idem
RButton	idem
Control	3 combinations - default free - check the list
Alt	used combinations free by default - check the list
Shift	the 5 combinations are impossible in MSc
Win key	many combinations, all free - check the list
Control + Alt	check the 4 combinations, all related to text conversion
Control + Win	check the 2 combinations related to window positioning
Alt + Win	impossible in MSc

AutoHotkey Kit for MuseScore - Reference



Master, Independent Hotkeys and all macro groups F1 thru F11 have each their corresponding Data .txt file. The Data files list for each hotkey which DIY items it needs. Quite a few hotkeys don't need additional data.
The Data files are meant as little assistants in assembling the hotkeys and invite you to make notes of what you have already done. Help in keeping track.
Many hotkeys use the same data. So the actual work is a lot less than what it seems.

The DIY items can be images or values.
After the creation of images as .png they will reside in the Working Directory, the folder containing all .ahk files.
Values are numbers. They can refer to surfaces with 4 numbers, hotspots with 2 numbers or offsets, colors and dimensions having 1 number.
All values reside in Coordinates.ahk

Hotkeys for determining surfaces and creating images, see page 19.
Tool for surfaces and hotspots, see page 27.

You find a survey of all hotkeys in *HotKeys_in_prefix_order.txt*

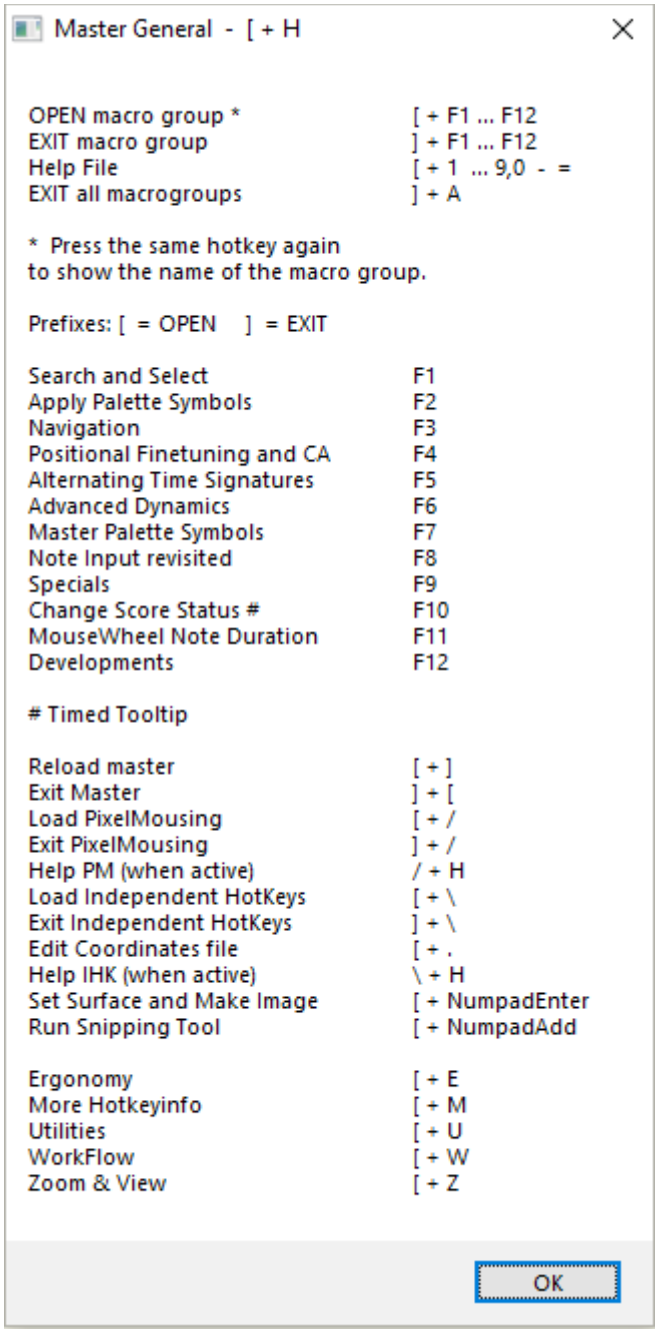
Search a command in a text file by typing it in the Searchfield.
Searching for the hotkey Z + M. Type z & m

Control	^
Alt	!
Shift	+
Win	#

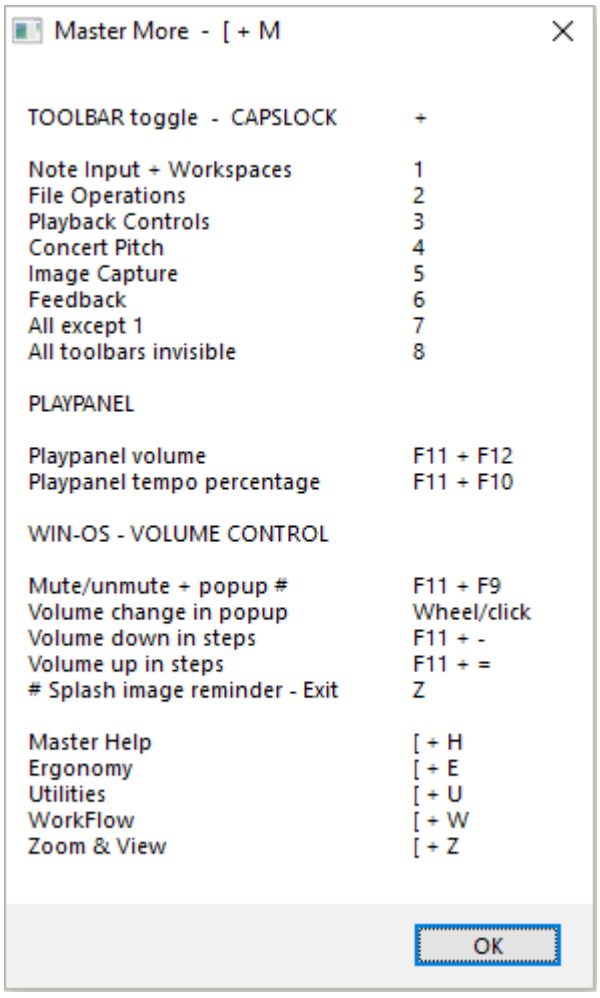
Reference section - Master

Supporting document: Master_DATA.txt. *Includes all DIY details*

[+ H



The Master can be launched in the usual way or with the hotkey **\ + M** which is located in *Independent Hotkeys*.

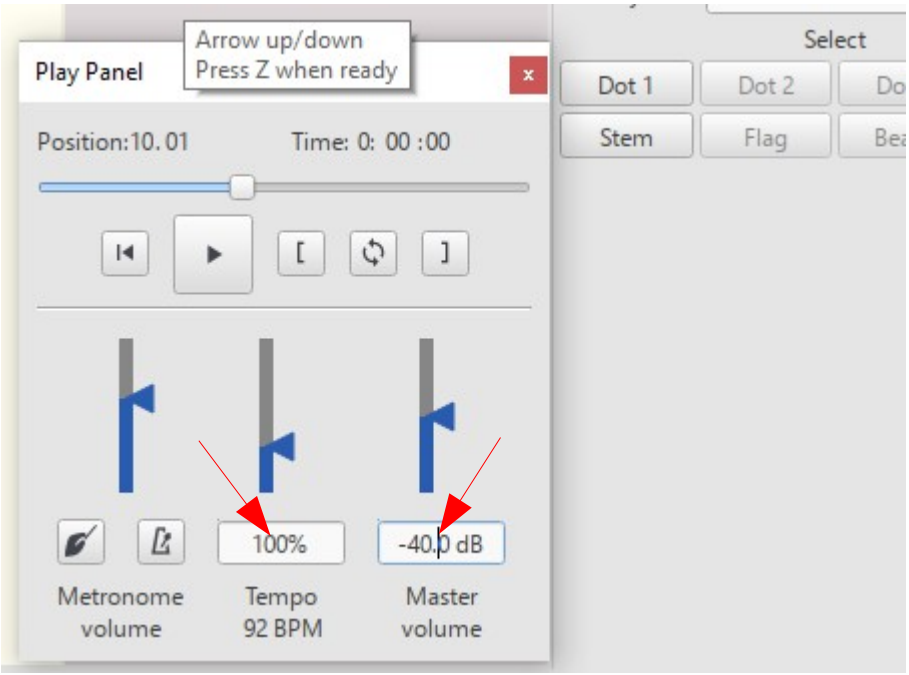


Open Master_DATA.txt and check a hotkey.
Now you can quickly verify that all commands from **[+ H** don't need external data. The hotkeys shown by **[+ M** however call for some DIY items.

Master - More

CapsLock + 1,2,3,4,5,6,7 and **8** need the toolbar selection spot. Page 29.
CapsLock + 8 For the two versions see page 29 and 30. A change to the alternative version is easy. More info also in Master.ahk. Search for 'ck & 8'.
PM: Edit the name of the command in the helpscreen of **[& m**

F11 + F12 volume changes by arrow keys
F11 + F10 idem speed as percentage



NB: It works best to treat the Play Panel as an independent window, thus not docked within the Inspector. In the Master utilities - to descibe later - there is a command IPP - Initialize PlayPanel. This moves the Play Panel outside the Inspector to a more fitting spot and sets its width and height. See **Z + U → IPP**

The hotspots within the rectangles - volume and speed % - must be determined relative to the Play Panel, being the active window.
See Master_DATA.txt for the names of the variables.

F11 + F9 Some commands use Splash images as eyecatcher. This is especially useful in score search commands. Here it could be a tooltip as well. The change to a tooltip is easy and included in the command. Yet it makes sense to experiment a bit with more attractive forms. Or even better: make your own flashes!

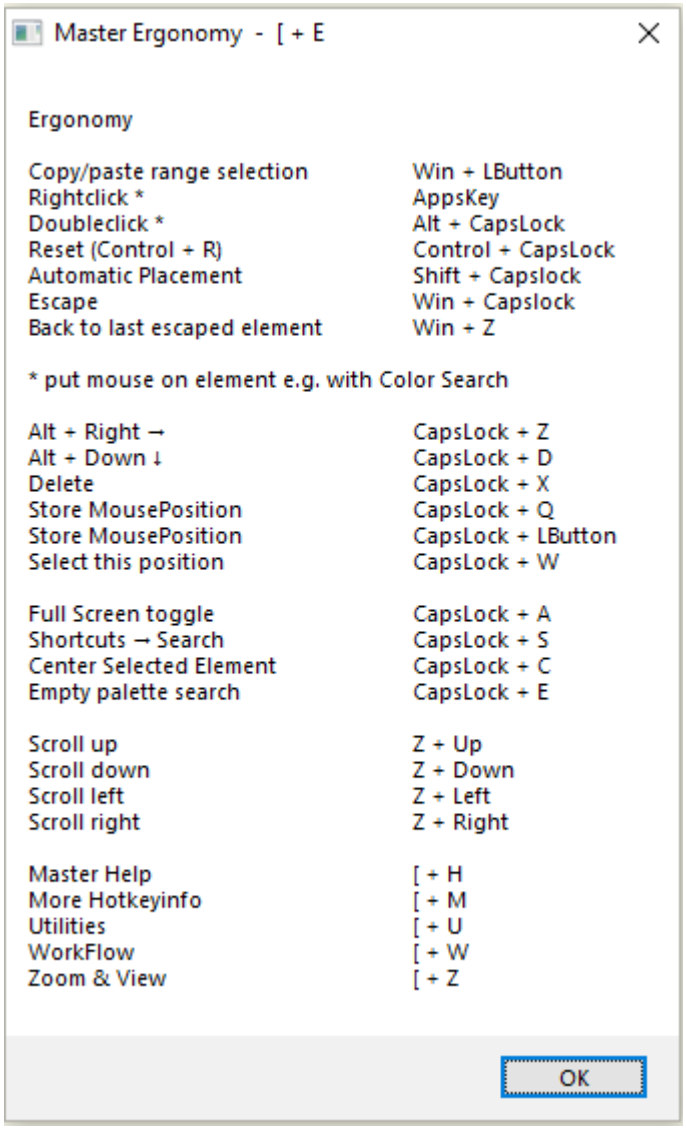
This is how it looks.

Basic form

Layout, font and color are in the AHK command. Search in the AHK Help for 'SplashImage'. Format must be .jpg

This example is **`MuseScore_logo_round.jpg`**

Master - Ergonomy



'Ergonomy' and 'Workflow', those terms refer to related practices.

The aim is to minimize manual movements compared with the MSc shortcuts. Compare **CapsLock + A** with Control + U in MuseScore. Or **CapsLock + Z** instead of Alt + →

A few hotkeys have a positional memory* assumed you don't shift the canvas in the meantime. **Win + CapsLock** makes **Win + Z** possible. **CapsLock + W** can be preceded by **CapsLock + Q** or **CapsLock + Left mouse button**.

After a color search in a densely packed score **CapsLock + C** helps to find lost elements.

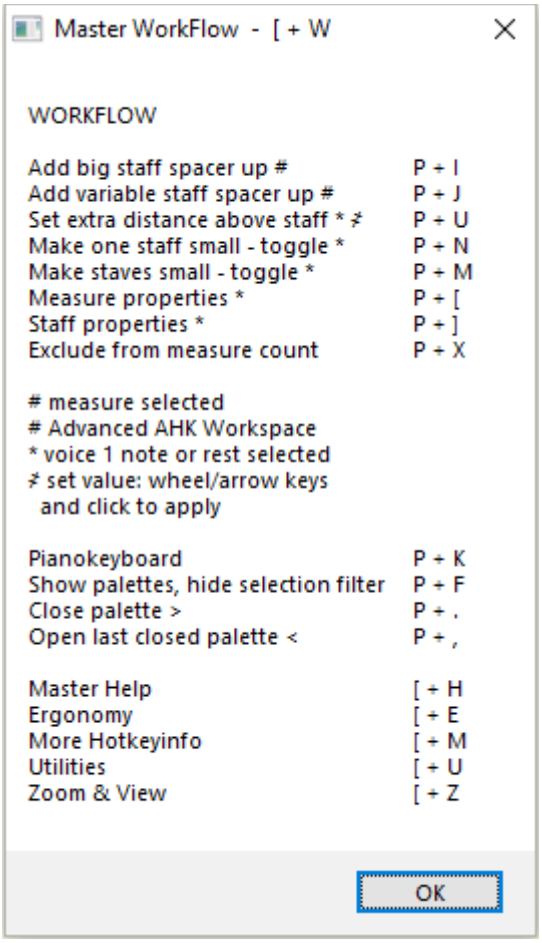
***** In the F9 group you can set and select 12 memory spots valid for the current session of MuseScore.

As mentioned Master DATA gives info about your actions to get the hotkeys working. A lot of them don't need any action at all. Others do. Here is an instructive one:

CapsLock + C *Center Selected Element*. How to read this hotkey?

Search in Master_**ahk** with 'ck & c'. After checking the Defined State Pixel-Search looks for the voice colors. If found the mouse travels to the selected element and clicks it. Its position is stored and used as the point of departure for a search. The macro starts a loop. Up to 15 passes it searches each time **10** pixels more to the left and **10** px higher for the **`ColorCanvas`** meaning it searches an 'empty' spot, not black, not an element and clicks this new spot. Next it inspects the statusbar for **`Statusbar_Nothing_Selected.png`**. If not found it starts a new pass. If found it clicks the empty spot and drags it to the centre. If each time during 15 passes some element was selected the macro changes direction. Maybe it was busy somewhere in the TopLeftCorner of the screen with many elements densely packed. Now it will search each time **10** pixels more to the right and **10** px lower for the **`ColorCanvas`** and will succeed in its enterprise. At least on the testscreen it never failed in super populated scores. In this description there is a custom element **10**. It will work on many screens but is marked **(*change*?)** to prevent the introduction of a unique variable. It's a simple experiment to find the optimal number for your screen resolution.

Master - Workflow



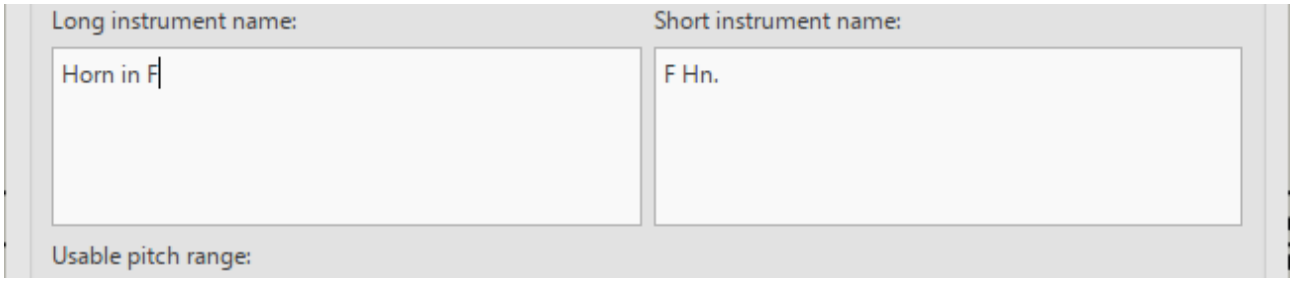
The hotkey **P + I** cooperates with the Advanced AutoHotKey workspace.
MSC Shortcut for Palette Search: *Control* + *F9*.
The macro sends 'bs5' to the Palette Search field. "Breaks and Spacers 5".
MSC Shortcut for 'Apply current palette element': *Control* + *Alt* + *P*.
PixelSearch finds the spacer and makes it higher. Very useful in Continuous View.
I has been chosen because of its form.
Notice the keyboard layout of **I**, **J**, **U**, **N** and **M**.

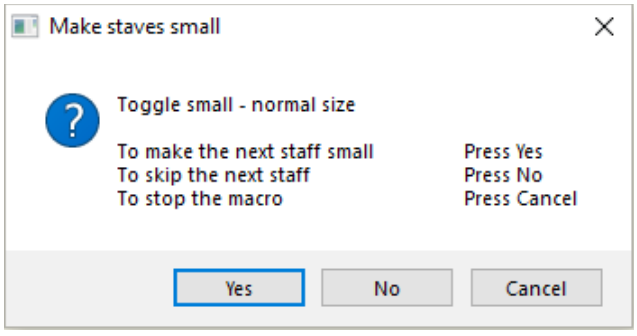
Wherever possible the letterform is used as mnemonic. As in opening bracket **[** and closing bracket **]**. Or in English letter combinations as in **P + K** and **P + F**. Or in oppositions as in **P + .** and **P + ,** together with their companions **>** and **<**

The first help screen of each macro group opens with **[+ #**group. A starting point to find your way.

P + U
P + N
P + M
P +]

These four hotkeys use the window *Staff/Part Properties*. The width of this window must be minimized but its height must be set to make room for more lines in the instrument. fields. In the picture there is enough space for five lines.





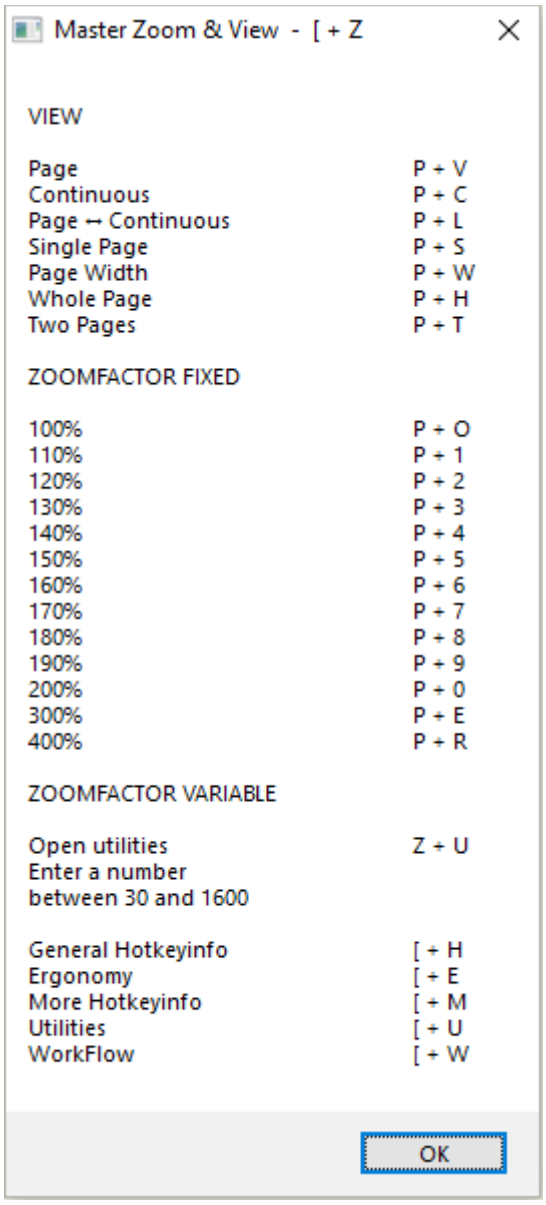
WSPPHeight := 675

P + M works downward. As in many commands selection by mouse is superfluous. The macro steers the mouse to the selected element and when its job is done a new PixelSearch finds the selected element in the changed layout.

P + F Show palettes, hide selection filter. It uses the image **CTS_Checked_ticked_sign.png** on two different surfaces: The tick areas in front of 'Palettes' and of 'Selection Filter' For more info see Coordinates.ahk

P + . Close palette uses **DA_Portal_Open_Down_Arrow.png** Again for more info see Coordinates.ahk

Master - Zoom & View



P + L simply uses the MSc shortcut Control + Shift + V to toggle between Page and Continuous View.

The other View hotkeys check first if the toolbar File Operations is visible or not. They use an image with much contrast.

PI_Print_Icon.png

They make the toolbar visible and will click within one of these two images.

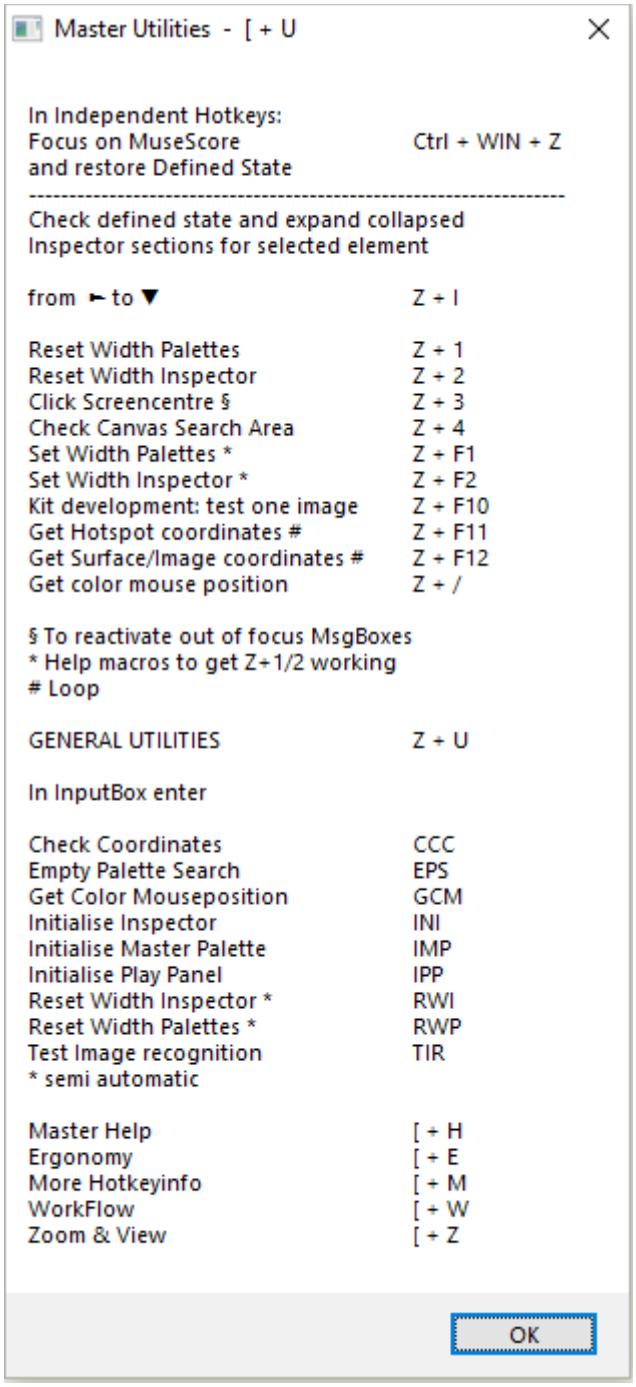
SPCView_X/Y
In the centre of the rectangle Single/Page/Continuous View.

ZoomArrow_X/Y
In the ▾ to the right of the % field.

Zoomfactor fixed:
P + O uses the shortcut Control + 0
The other hotkeys check first for the Print Icon and click next in the centre of the % field where they type the zoomfactor.
ZoomPerc_X/Y

Zoomfactor Variable is in *Utilities* and will be described below.

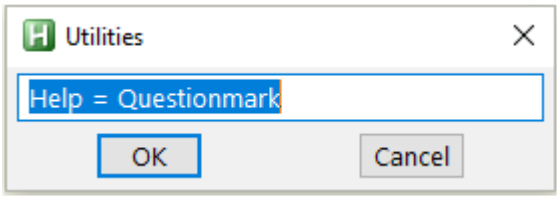
Master - Utilities



Control + Win + Z see page 33 for an explanation of what the hotkey does. *Independent Hotkeys* is in the last section of this document.

Z + I	page 31
Z + 1, 2, F1, F2	on 22 -26
Z + F10	example *How to test an image*
Z + F11, F12	page 27
Z + /	page 35

Z + U opens an InputBox



The InputBox is positioned in the Inspector. Search for z & u to **(*change*?)** its position relative to the mouse.

For almost all Inspector InputBoxes:

W_IB	**Width**
H_IB	**Height**
IB_X/Y	**Position TLC on screen**

See Coordinates.ahk for the exception.

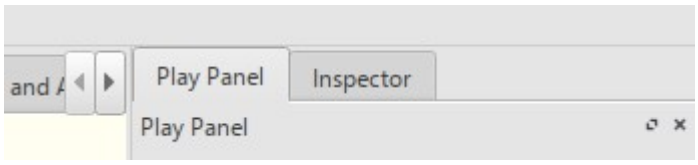
General utilities

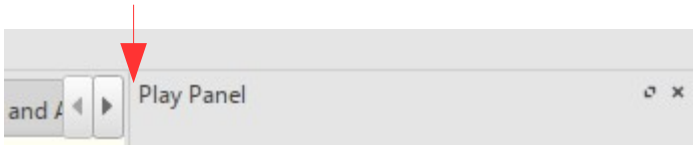
Some commands we have already encountered appear again in **Z + U**. Entering a **?** in the InputBox shows them. **EPS**, **GCM**, **RWI** and **RWP**.

Z + U → **IPP, IMP, INI** *Initialize Play Panel, Master Palette and Inspector*

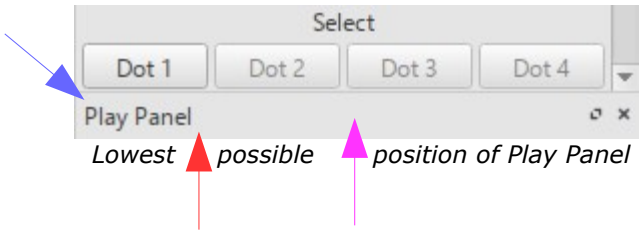
IPP Positions the Play Panel outside the Inspector.
IMP Selects a symbol, bringing the Master Palette in a Defined State.
IPP and IMP are preferably executed at the start of a new session.
INI Using a small .mscz file all Inspector sections are expanded from ► to ▼.

Z + U ► IPP Initialize Play Panel





Highest possible position of Play Panel



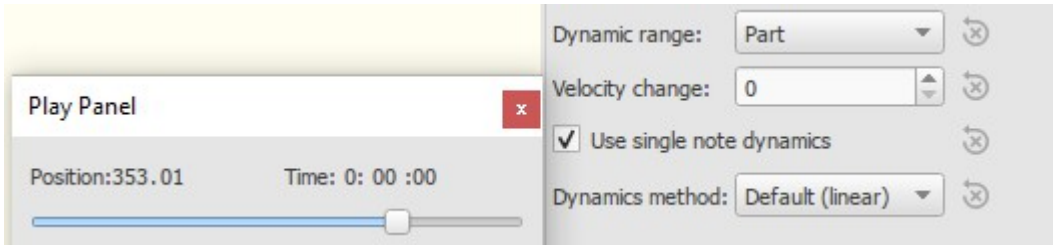
IM_19_Inspector_Play_Panel_Header.png

Search area of IM_19:
Put the Play Panel in the Inspector and switch the Inspector off. The word 'Play Panel' appears in the highest possible position. As in left pic here.
Switch Inspector on again and drag the Play Panel to the bottom of the Inspector until it docks. Go to the separator between Play Panel and Inspector and move the separator as far as possible down.

The search area is a rather high column of which the width is a bit bigger than the width of the word 'Play Panel'. The red arrows point to its TLC and BRC.
The **IPP** command tries to find the header 'Play Panel'. When the word is not found it switches on the Play Panel so that the header will be found. It stores the coordinates of the TLC of the found image, the point of the blue arrow.
We want to drag the Play Panel outside the Inspector. If it stays inside it could sometimes cover relevant data. We also want to have room for those InputBoxes which are best placed in the lower part of the Inspector.

To prevent dragging the separator  between Inspector and Canvas we add an X-offset to the found TLC, **PP_Off_X**.
To prevent dragging the separator between Play Panel and Inspector we add an Y-offset **PP_Off_Y** to the found TLC.
The macro steers the mouse to this spot, the point of the magenta arrow, double clicks and drags the Play Panel outside the Inspector to its new location.

On this new location the Play Panel has a TLC with **Play_Panel_X** and **Play_Panel_Y** and a size determined by **PlayP_Wide** and **PlayP_High**



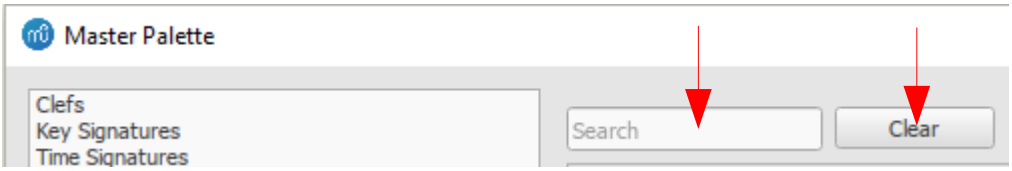
My favorite position of the Play Panel next to the Inspector

Z + U ► IMP Initialize Master Palette

The *Symbols* section of the Master Palette includes many glyphs specific for winds, strings, guitar, percussion etc. Each symbol you use frequently you'll probably want to be housed in a Palette.
At the other hand *fast acces* to and *positioning* of many less used Master Palette Symbols could be attractive. Compare it with the **Z +A** hotkey *Apply Symbols from Palette.* Macrogroup F7 handles the Master Palette and this utility prepares it for use.
Like most windows the size of the Master Palette has to be *minimized*. Minimize the left column from *Clefs* to *Symbols* as well.
The MSc shortcut to open the Master Palette Symbols is **Shift + Z**. See Page 9 about the liberation of the Z-key.

When you open the Master Palette Symbols after startup there is no symbol selected. The job of **IMP** is to select this gracious flying bird ornament:

PIN_Master_Palette_Pincé.png thanks to Couperin!
The search area is the surface occupied by the symbols. On the test screen 6 rows with 6 symbols each are wholly visible when the search field is empty.
The TLC is **MP_X1** and **MP_Y1** and the BRC is **MP_X2** and **MP_Y2**

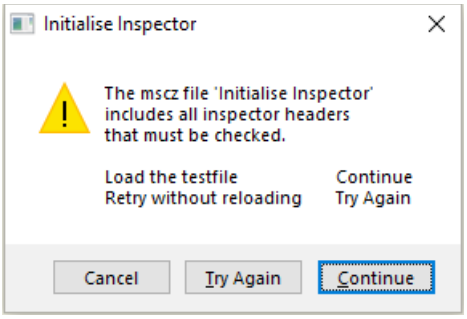


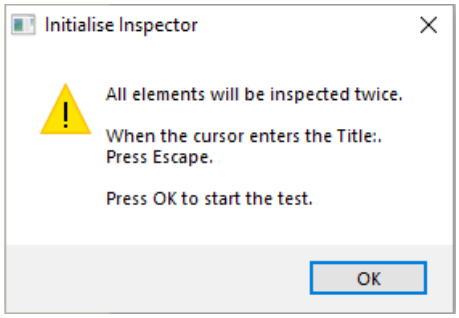
IMP sends a text to the search field. A new text demands a preceding *Clear*.

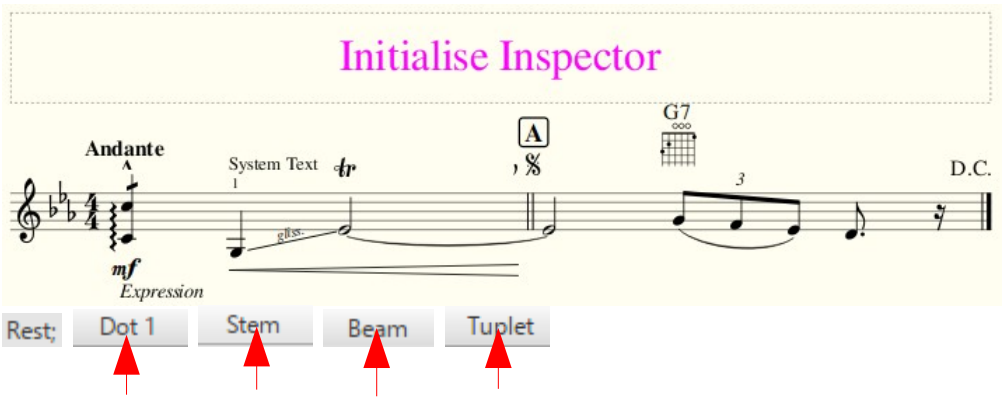
WMPS_01X/Y is the hotspot within the *Search* rectangle. **WMPS_02X/Y** of *Clear*.

In addition to these variables macrogroup F7 uses three more. See the F7 section.

Z + U ► INI Initialise Inspector

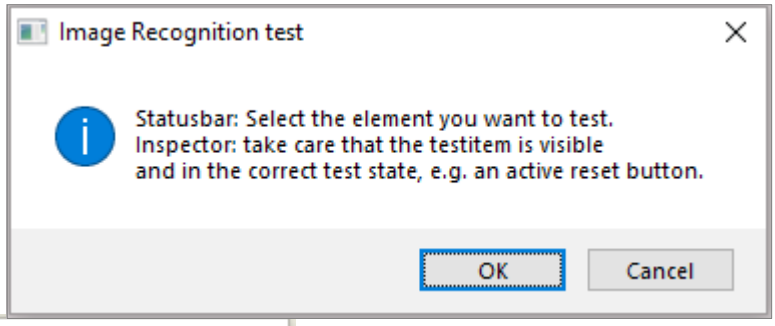






Steered by the mscz file the macro checks all Inspector sections and expands them if they are collapsed. It works the same as **Z + I** - page 31 - but applied to all relevant elements in a row. Initialising this way is more like a demo project. Or you could use it as another hands-on exercise. Check Master_DATA about the 4 hotspots.

Z + U ► TIR Test Image Recognition



Test Image Recognition

Type the short name of the image using this list.

STATUSBAR			
Acciaccatura	ACC	Note	N
Appoggiatura	APP	Nothing Selected	NSS
Articulation	AR	Range Selection	RS
Barline	BL	Rest	R
Beam	B	Slur	S
Bend	BE	S-tacca-to	ST
Dynamic	D	Stem	SM
Full measure rest	MR	Symbol	SYM
Grace Note	GN	Text	TXTS
Hairpin	H	Tie	T
List Selection	LS	Tremolo	TR
Master Palette Title	MP	→→ MP must be an INactive Window!	
Palette Locator Left	PLL	Palette Locator Right	PLR
Selectionfilter (NOT blue!)	ALL	-> Palettes must be invisible!	
Check ticked √ F- ops	CTFO	Check ticked √ Sel Filter	CTSF
Check ticked √ Palettes	CTP	Check ticked √ Workspaces	CTWS
Check ticked √ F-ops..Fb	CTFOFB	Check ticked √ Note Input	CTNI
Check ticked √ F-ops..Ws	CTFOWS	Check ticked √ Horiz. Beam	CTHB
NOTES CONTEXTUAL MENU - Item must be blue!			
Delete	DEL	Remove Selected Range	RSR
Paste	PASTE	Swap with Clipboard	SWC
INSPECTOR ▼ = section open			
Beam ▼	BM	Play Panel	PP
Beam Custom Pos √	BC	Play ticked √	PT
Clef ▼	CL	Segment ▼	SEG
Element ▼	EL	Select Augmentation Dot1	SAD
Empty Surface	ES	Select Beam	SB
Fingering ▼	FI	Select Flag	SF
Fretboard Diagram ▼	FD	Select Stem	SSM
Inspector	I	Select Tuplet	TU
Insp displaced	ID	Stem ▼	SMI
KeySig ▼	KS	Text ▼	TXTI
Line ▼	LI	TimeSig ▼	TIS
Nothing Selected	NSI	Triangle Closed ►	TRI
PALETTES		MASTER PALETTE	
Palette Open ▼	PO	Pincé - *not* blue!!	PIN

Input the abbreviation

OK Cancel

tacca

The making of *tacca* asks for the precision of PixelMousing.

Master Palette is pale.

Master Palette

By far the most elements have to be tested in the Defined State.

However it could be that some items are then invisible e.g. toolbars.

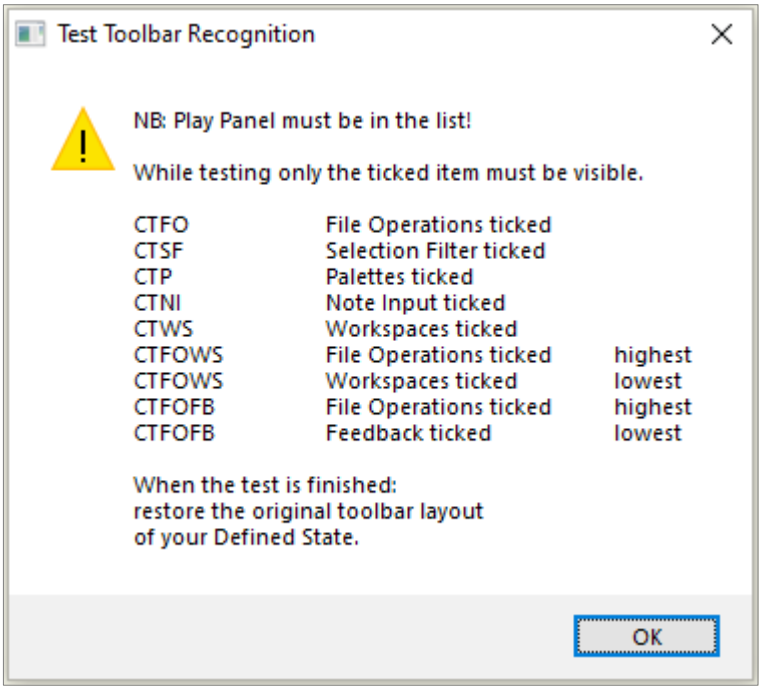
In such a case - while in the Defined State - press **Z + U ► TIR** , make the item visible and return to the big TIR window with **Alt + Tab**.

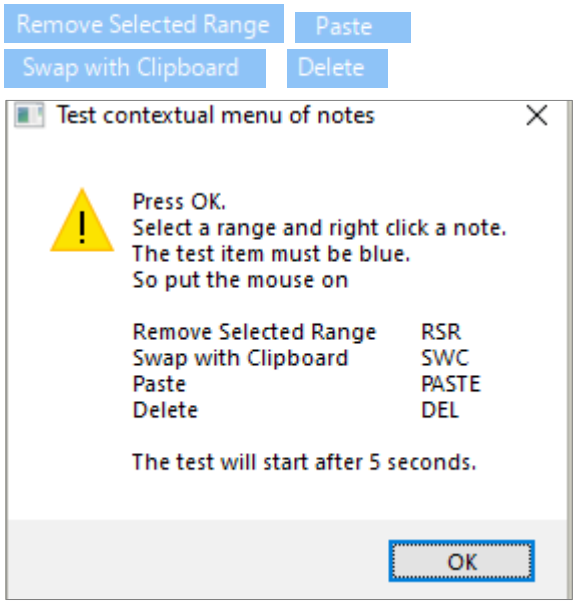
Added:
Print Icon
Test name
PR

Acciaccatura; Appoggiatura; Articulation: Barline: Beam; Bend; Dynamic: Rest; Duration: Measure;
Grace note Hairpin: List Selection Note; Pitch:
Range Selection Rest; Slur; tacca Stem; Symbol; Text: Tie; Start Tremolo:

Determine Surface of Statusbar by surface of Articulation: Louré (tenuto-staccato)
and make *Nothing selected* just fit.

Statusbar_Acciaccatura.png
Statusbar_Appoggiatura.png
Statusbar_Articulation.png
Statusbar_Barline.png
StatusBar_Beam.png
StatusBar_Bend.png
Statusbar_Dynamic.png
Statusbar_Rest_Duration_Measure.png
Statusbar_Grace_Note.png
Statusbar_Hairpin.png
Statusbar_List_Selection.png
Statusbar_Note_Pitch.png
Statusbar_Nothing_Selected.png
Statusbar_Range_Selection.png
StatusBar_Rest.png
StatusBar_Slur.png
Statusbar_tacca.png
StatusBar_Stem.png
StatusBar_Symbol.png
Statusbar_Text.png
Statusbar_TieStart.png
StatusBar_Tremolo.png
Remove_Selected_Range_Blue.png
Swap_with_Clipboard_Blue.png
Paste_Blue.png
Delete_Blue.png

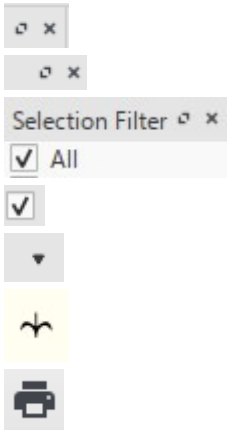




CTFOWS Surface from File Operations thru Workspace ☑
CTFOFB Surface from File Operations thru Feedback
CTNI Read more about all these tests
CTWS on page 29-30 relative to the hotkey **CapsLock + 8**.

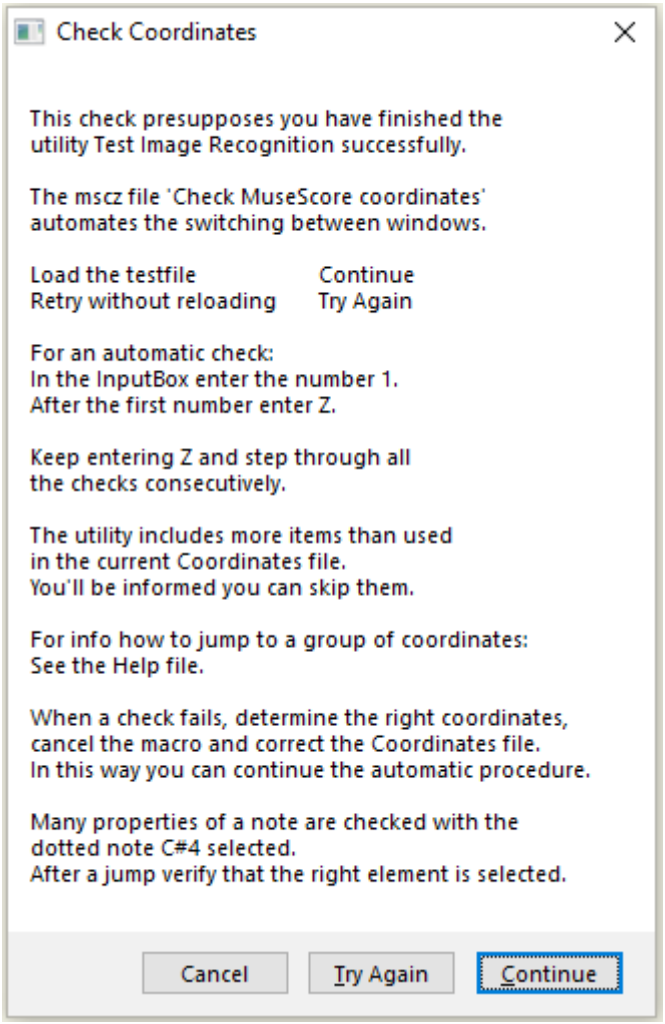
The commands **CTFOWS** and **CTFOFB** have to be performed twice.

IM_26_Inspector_Beam_Section_active.png
IM_09_Inspector_Beam_Custom_Position_ticked.png
IM_27_Clef_triangle_section_open.png
IM_13_Inspector_Element.png
IM_02_Inspector_Empty_Surface.png
IM_20_Inspector_Fingering.png
IM_22_Inspector_Fretboard_Diagram.png
IM_01_Inspector_Inspector.png
IM_01_2_Inspector_Displaced_Insp.png
IM_29_KeySig_triangle_section_open.png
IM_23_Line_triangle_section_open.png
IM_03_Inspector_Nothing_Selected.png
IM_19_Inspector_Play_Panel_Header.png
IM_25_Play_ticked.png
IM_14_Inspector_Segment.png
IM_07_Inspector_AugmentationDot1.png
IM_08_Inspector_Select_Beam.png
IM_05_Inspector_Flag.png
IM_04_Inspector_Stem.png
IM_06_Inspector_Tuplet.png
IM_21_Inspector_Stem_section_open.png
IM_17_Inspector_Text.png
IM_28_TimeSig_triangle_section_open.png
IM_18_Inspector_Triangle_Closed.png

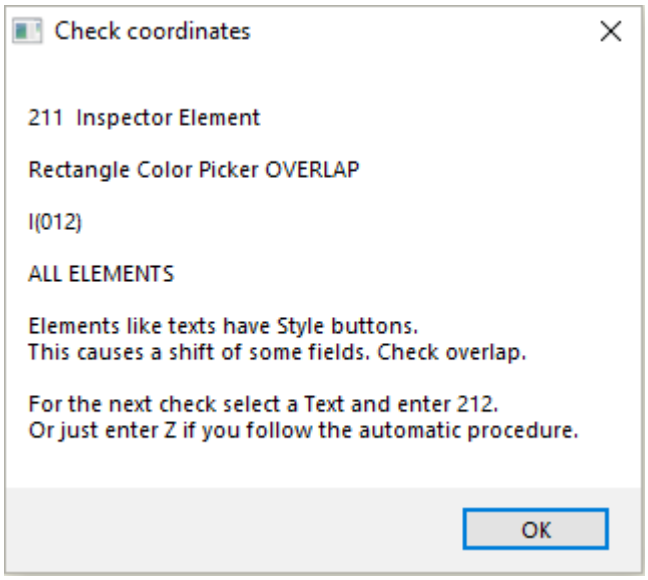


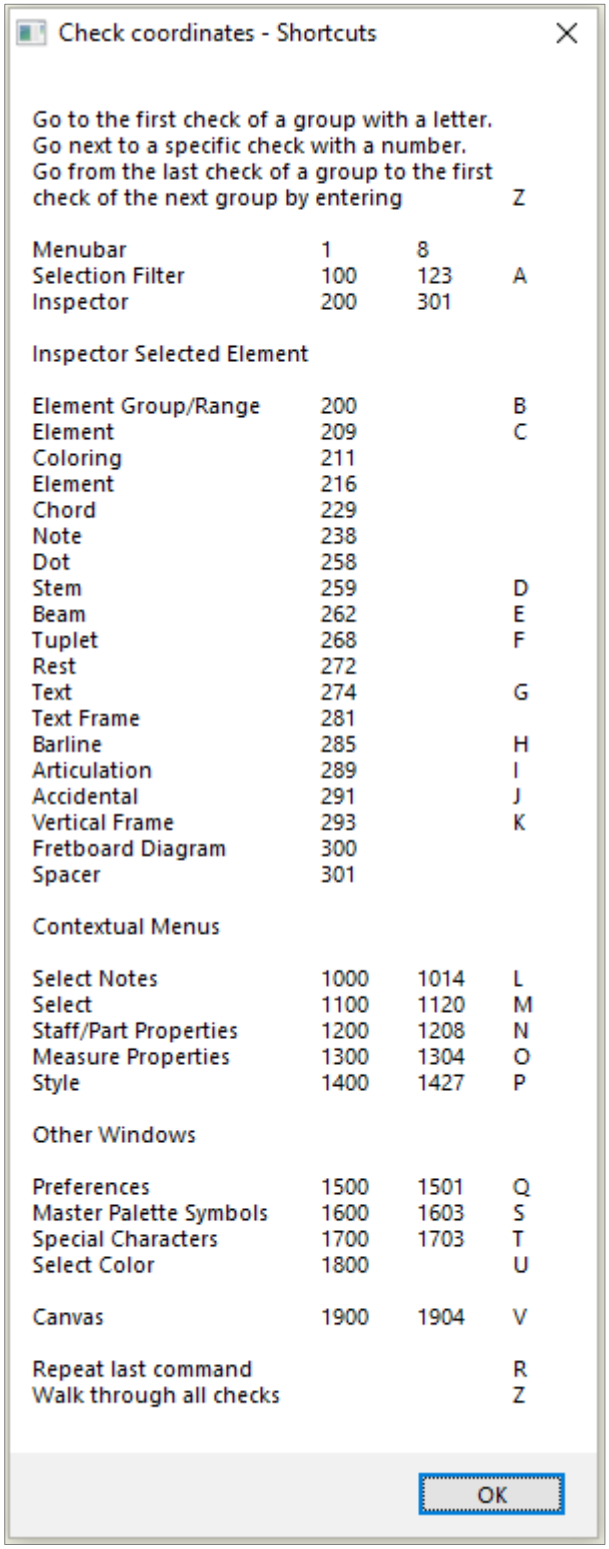
Palette_Locator_Left.png
Palette_Locator_Right.png
SLF_01_SelectionFilter_All_ticked.png
CTS_Checked_ticked_sign.png
DA_Portal_Open_Down_Arrow.png
PIN_Master_Palette_Pincé.png
PI_Print_Icon.png

Z + U ► CCC Check coordinates

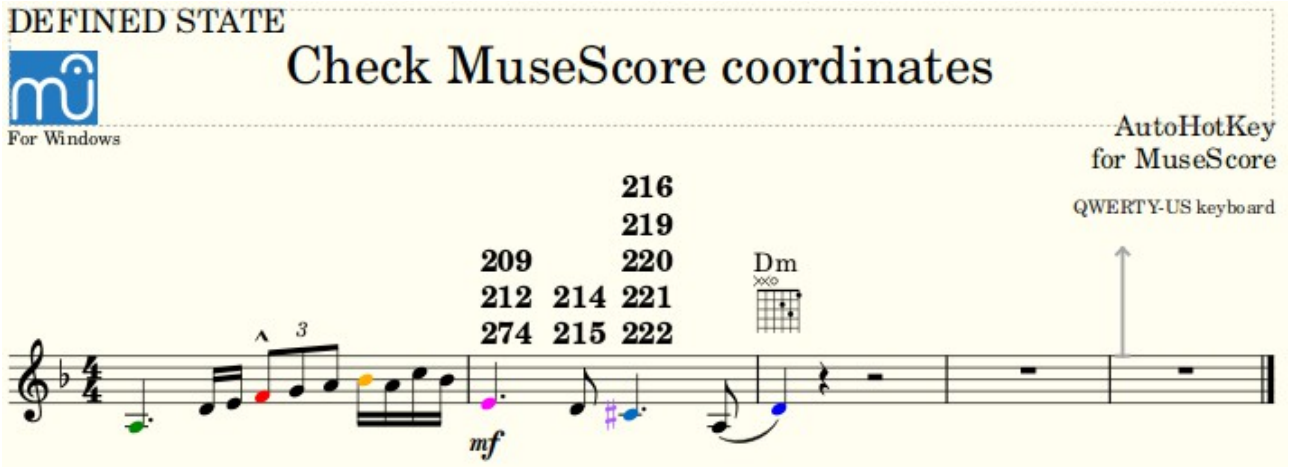


Help screens guide you through the test sequence

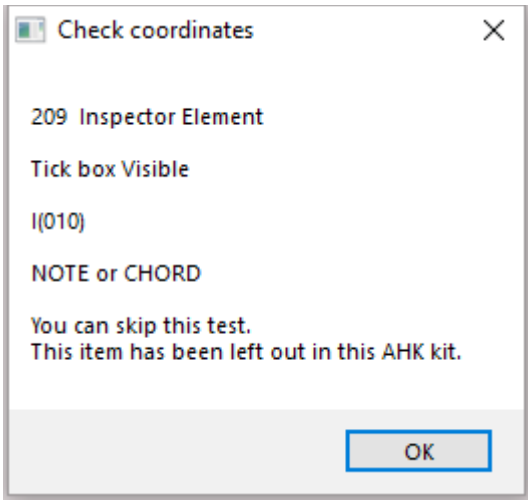




The numbers in the test score help to keep track. Especially with overlapping fields in everything color-related.



The colors in the small test score guarantee fast access to properties of e.g. beams, articulations, dots, texts, tuplets, barlines etc.
A previous Test Image Recognition of relevant images must have been successfull.



Unused hotspots and images Reference sections

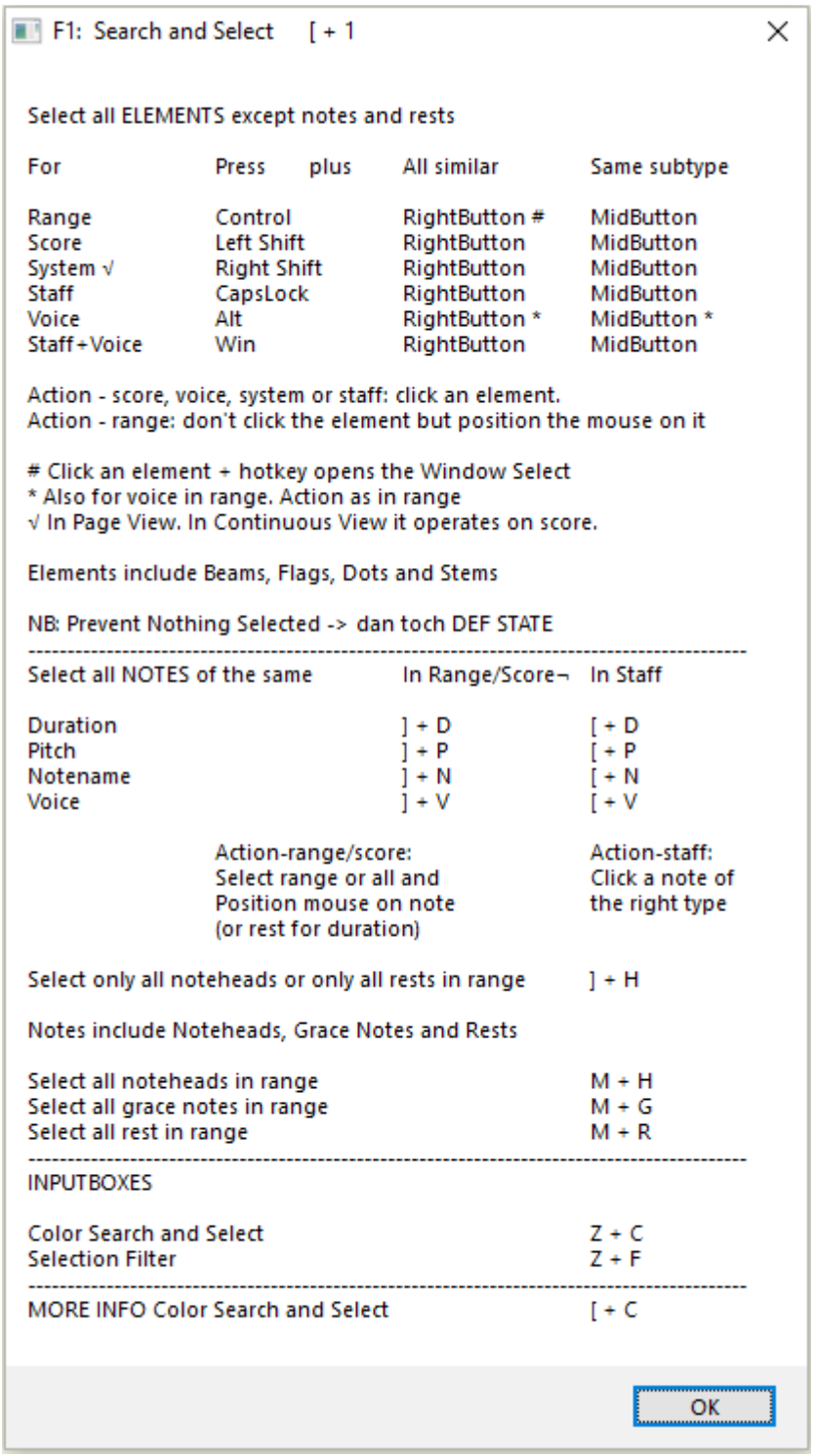
In *Check Coordinates* you can skip many test items. They are not included in the current version of this AHK kit. Skipping is fast. Just press OK and **Z**.

For possible future use their names are mentioned in Coordinates.ahk as the last section at the bottom of this file together with some unused images.
The sections are outcommented using **/*** and ***/** c.q. semicolons

Reference section - F1 ColorSearchSelect

Supporting document: F1_DATA.txt. *Includes all DIY details*

PM: Topics described on pages 34-38: The command PixelSearch - Voice colors and their numbers - The Canvas Search area - Hotkeys search and select voice1234, only V1, V2, V3, V4 - Select colored range - 22 + 4 colors - Selection via contextual menus - DIY parts of F1-stripped version



[+ F1 Run macrogroup F1 Master command

In F1_DATA.txt

List of hotkeys with their variables, if any.
Hotspots
Surfaces
Images
Colors
Offsets

In short all data which a particular hotkey needs.

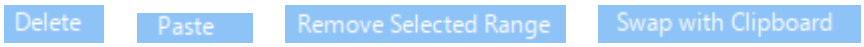
[+1 shows all selection hotkeys using the right-click contextual menus.. (page 39)

The 4 *blue* images are essential. (page 37) They use the offsets
RC_CM_Y1
RC_CM_X2
RC_CM_Y2
Read on the next page how to determine the offsets.

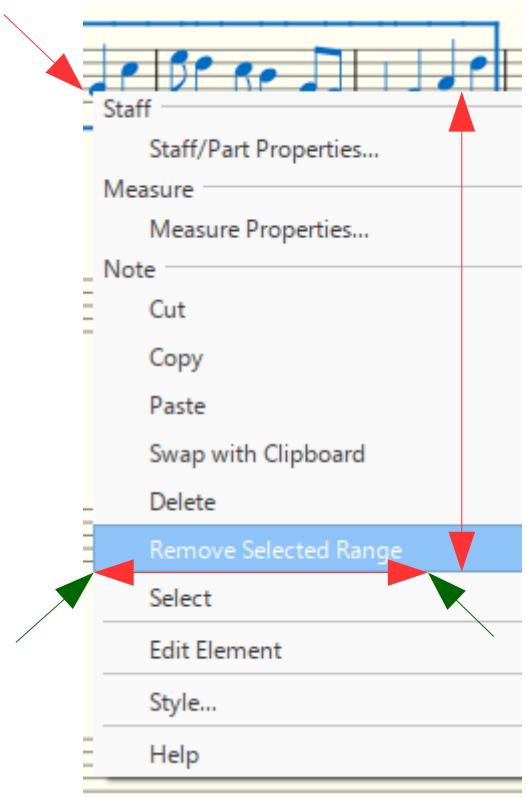
NB: Contextual Windows: Window Select and Window Select Notes must be minimized.

Right-click contextual menu: Offsets

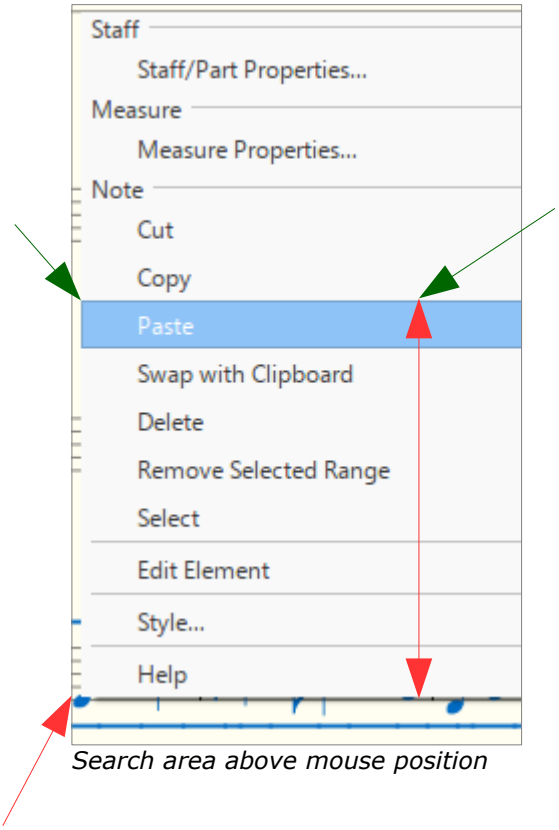
The macros have inbuilt protection from potentially desastrous selections. These could happen when you apply a command meant for elements while you have actually a note or rest selected. Dependent on the wrongly applied command the mouse could land on one of these four images:



Most of the time the Right-Click Contextual Menu *RCCM* will appear below the selection. But if the staff is low on the screen the menu will be higher than the selection. So the search area must take these two possibilities into account. In other words the search area is determined by offsets relative to the mouse position. The width of *Remove Selected Range* is the X-offset added to the X-mouse position. It is the width of the horizontal red line in the left picture. This is **`RC_CM_X2`**. The vertical arrow in the left picture is the Y-offset **`RC_CM_Y2`** which has to be added to the Y-mouse position. The vertical arrow in the right picture is the Y-offset **`RC_CM_Y1`** which has to be subtracted from the Y-mouse position. The whole search area is the rectangle determined by the four green arrows.



Search area below mouse position



Search area above mouse position

[+ C Color Search and Select - *Info*

F1: Search and Select [+ C

INSTANT SELECTION
Mouse moves to selected element in voice and clicks it

Voice 1	Alt + Z
Voice 2	Alt + X
Voice 3	Alt + C
Voice 4	Alt + D
Voice 1234	Alt + S

COLORED RANGE - 4 pairs

With the 'liberated keys' H, J, K, and L

Range 1-2 managed by prefix	L
Range 3-4 managed by prefix	K
Range 5-6 managed by prefix	J
Range 7-8 managed by prefix	H

Set 1 (left)	L + [
Set 2 (right)	L +]
Reset 1-2	L + -
Select 1-2	L + =
Go to 1	L + L MouseButton
Go to 2	L + R MouseButton
Select 1-2	L + Mid Button

With the 'liberated key' Y

Set 1...8	Y + F1...F8
Go to 1...8	Y + 1...8
Select range 1-2	Y + 9
Select range 3-4	Y + 0
Select range 5-6	Y + -
Select range 7-8	Y + =

GENERAL

Colorpicker	Y + C
Reset Range Colors	Y + R
Reset Note Color	Y + N
General Reset Elements with/without Style Button	Y + S

INPUTBOXES

Color Search and Select	Z + C
Selection Filter	Z + F

OK

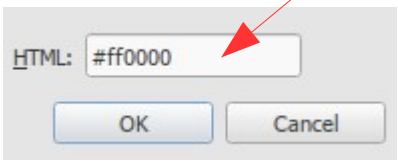
Select color voice 1-4: see page 34

Colored Range:
The hotkeys managed by prefixes **L, K, J, H** have an equivalent in the hotkeys of **Y + ...**

If you don't want the **L, K, J, H** hotkeys just outcomment them: e.g.
~h & [:: becomes **; ~h & [::**
and v.v. with the **Y**-hotkeys.
See *Hotkeys_in_prefix_order.txt*

To reset an **Y**-colored range select the range and reset with **Y + R**

`IN_012X/Y hotspot color picker rectangle (black) - OVERLAP all elements` see also page 13 and at the end of F1_DATA
(in the 'subroutine')
Here also the hotspot **`RGB_X/Y`**



In Window Select Color HTML:
#RGB rectangle

Bottom margin is the lowest rectangle of a Vertical Frame

The red surface is occupied by **`IM_02`**

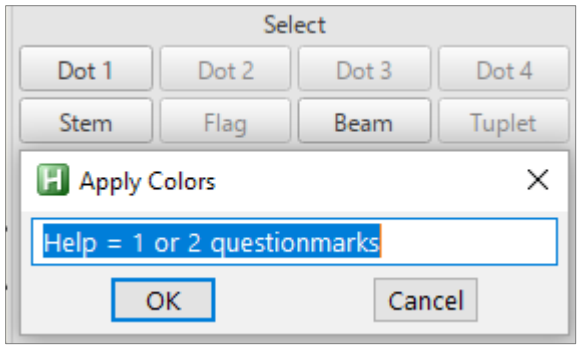
`IM_02_Inspector_Empty_Surface.png`

This dull image prevents the coloring of elements which can't be colored. In **Y + C** *Colorpicker*

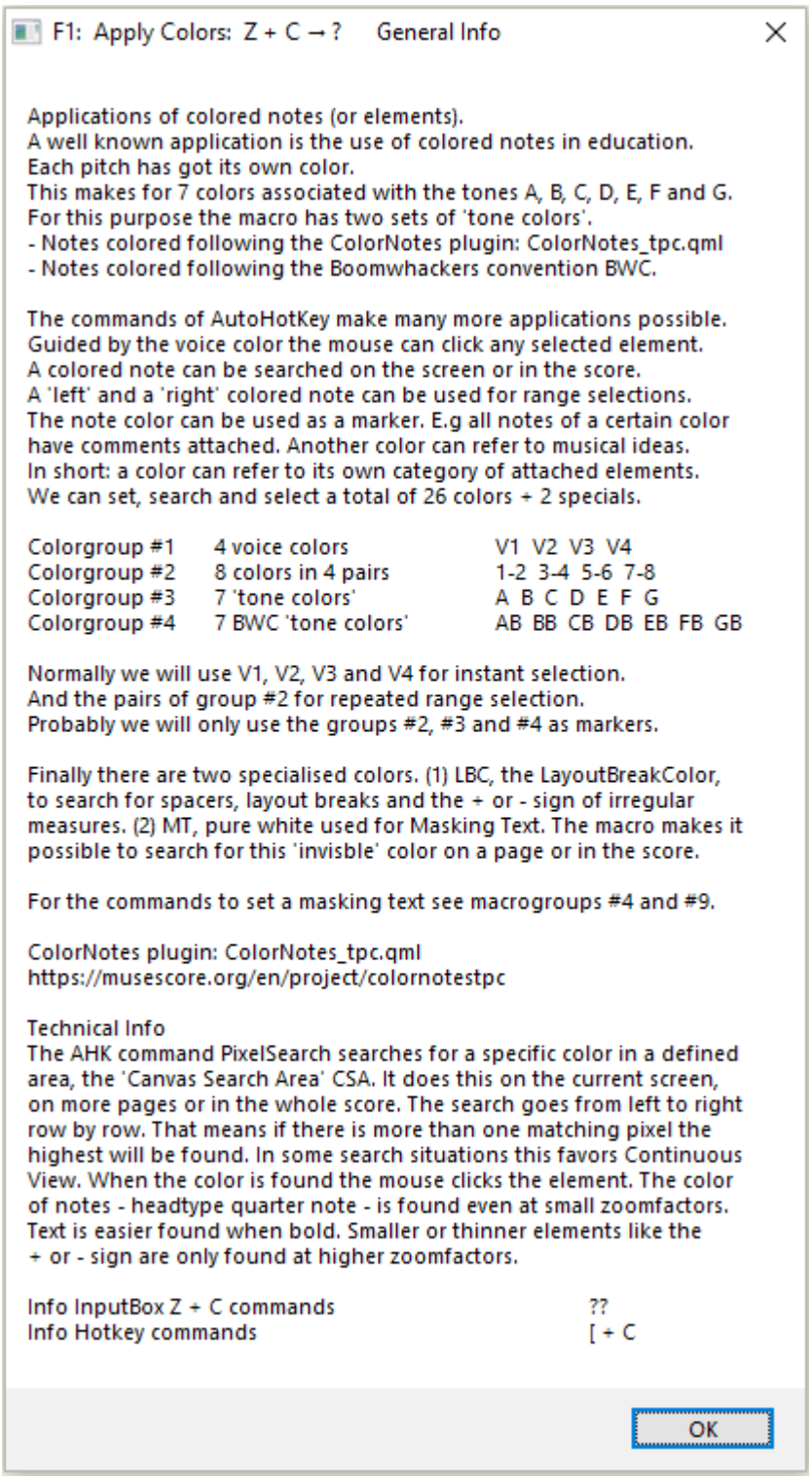
Z + C Apply Colors

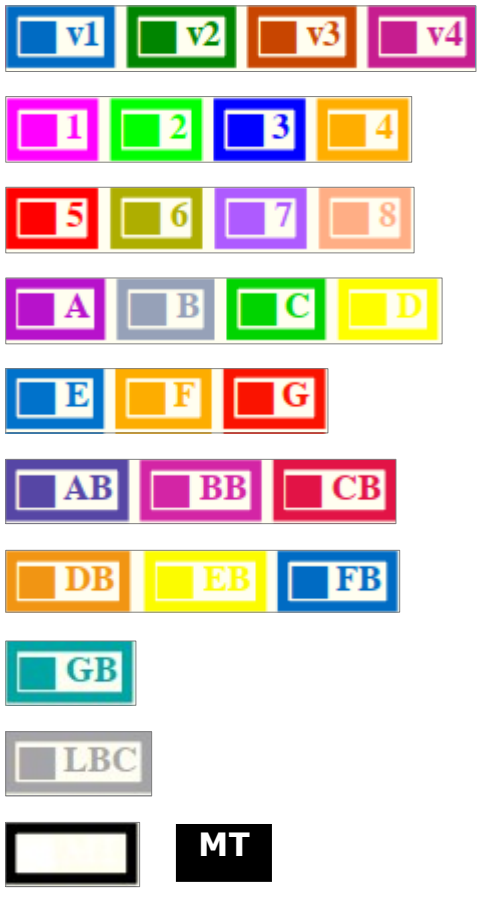
In F_1...DATA
Search for **z & c** to find the images to create and the coordinates to determine. Notice **(*change*?)**

Z + C ► ?
Info about color groups.



To reposition the InputBox near the mouse: (*change*?)





Z + C ► ??

F1: Apply Colors: Z + C → ?? Shortcuts

Colorgroup #1	4 voice colors	V1, V2, V3, V4
Colorgroup #2	8 colors in 4 pairs	1-2 3-4 5-6 7-8
Colorgroup #3	7 'tone colors'	A, B, C, D, E, F, G
Colorgroup #4	7 BWC 'tone colors'	AB, BB, CB, DB, EB, FB, GB

Specials: LBC, LayoutBreakColor and MT, pure white Masking Text

In InputBox enter:

SET COLOR SINGLE ELEMENT

Enter colorname	V1, V2, V3, V4, 1, 2, 3, 4, 5, 6, 7, 8, LBC, MT A, B, C, D, E, F, G, AB, BB, CB, DB, EB, FB, GB

SET COLOR RANGE

Color Noteheads	Same as above	colorname only
Color all similar *	Enter !	before the colorname
Color same subtype	Enter =	before the colorname

* E.g. beams, stems, articulations

SEARCH COLOR ON SCREEN

Search color	Enter []	before the colorname

Search the whole color family - reset possible

A, B, C, D, E, F, G	[]ABC
BWC colors	[]BWC
1.....8	[]123

SEARCH COLOR IN SCORE - reset possible

Search score	Enter /\	before the colorname
Stop Score Search	Z + Pause	

COLORED RANGE with color pairs 12 - 34 - 56 - 78

	Selection	Reset
In InputBox	Enter	Enter
Range 1	12	R12
Range 2	34	R34
Range 3	56	R56
Range 4	78	R78

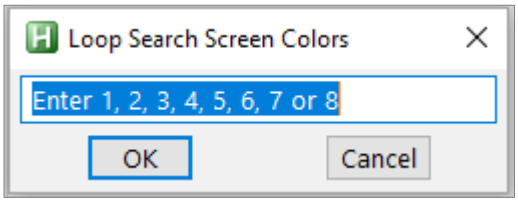
Info General	?
Info Hotkey commands	[+ C

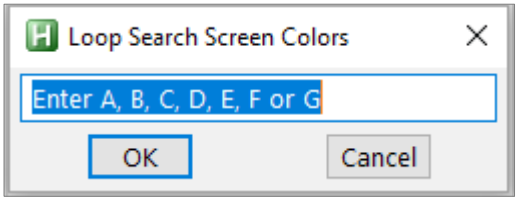
OK

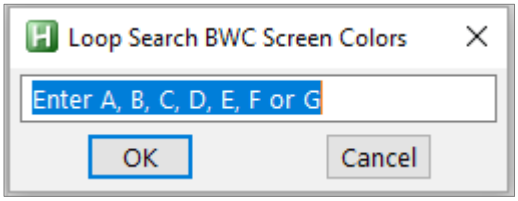
Mnemonics

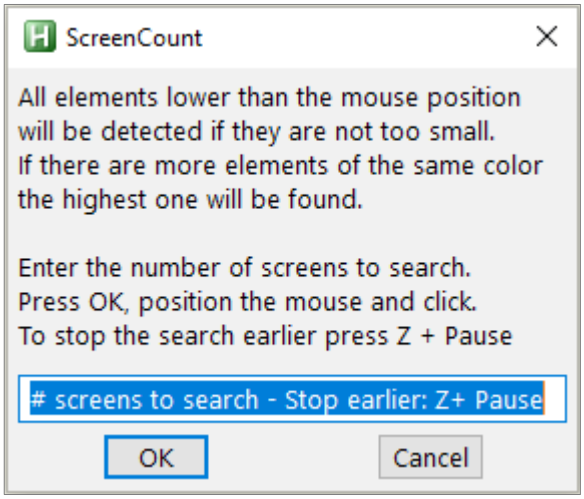
Set all similar	**!**
Set subtype	**=**
Search screen	**[]**
Search score	**/**

Range: Color *all similar* and Color *same subtype*
MessageBox
"Position the mousepointer on the right element."





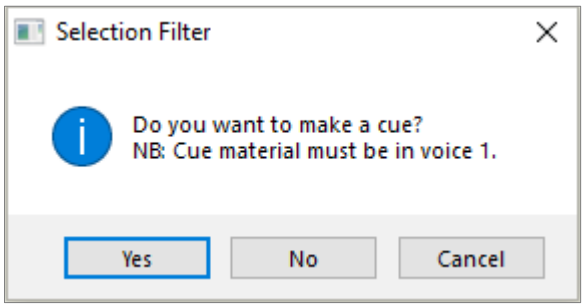




Though the search speed is high limiting the search ribbon will make it even higher. Consider using a dedicated staff with colored notes. E.g. the color AB for annotations, BB for ideas, CB for themes etc. Position this staff at the bottom of the screen in Continuous View and click just above the staff for optimal search performance.

Z + F Selection Filter

Select a note or rest first.
The messagebox asks: Cue?



Selection Filter - shortcuts

In the InputBox enter

Keep voice 1	K1 or Enter	
Keep voice 2	K2	
Keep voice 3	K3	
Keep voice 4	K4	
Toggle switches:		
Voice 1	1	
Voice 2	2	
Voice 3	3	
Voice 4	4	
All	*	
Dynamics	D	
Hairpins	H	
Fingerings	F	
Lyrics	L	
Chord Symbols	CS	
Other Text	OT	
Articulations	A	
Ornaments	O	
Slurs	S	
Figured Bass	FB	
Ottavas	VA	8
Pedal Lines	PL	
Other Lines	OL	
Arpeggios	ARP	
Glissandi	G	
Fretboard Diagrams	FD	
Breath Marks	BM	
Tremolos	T	
Grace Notes	GN	
Activate MuseScore to do something. E.g. set range colors	Z	InputBox
After your actions: Return to Filter	Z	Keypress
Copy filtered selection	C	InputBox
Select destination and Paste with	Z	Keypress
If you change your mind Put the mouse on an empty spot and press	Z	Keypress

Use ColorSelect for fast reselection

OK

Answering Yes colors the first and last element of the range. After filtering the remaining cue material is now ready for use.

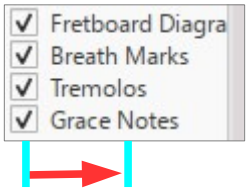
Z + Q *Make Cue*

This hotkey is in group F9 Specials
The copy in the destination is still a colored range. This makes automatic reselection possible for necessary edits like
Make notes small
Switch 'Play' off
Exchange voices
Move small rests up

Related:
Z + J *Copy Instrument name*
Also in F9 Specials

Sometimes it is desirable to temporarily suspend macro execution so you can do something else in MuseScore or run another macro.

SF_OffSet_X is the length of the red arrow



Before the macro finishes the mouse must click in the middle of *Grace Notes*.

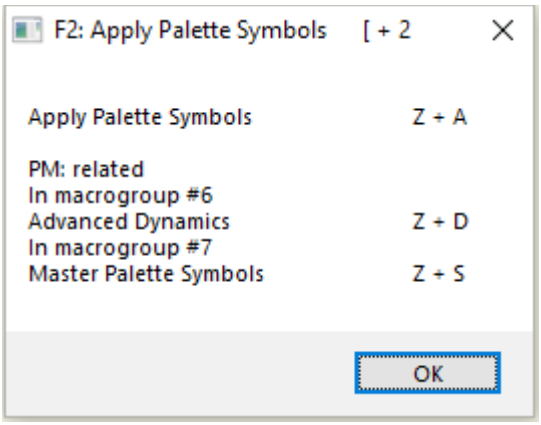
A click turns *Grace Notes* blue.
This ensures that *All* is not blue.

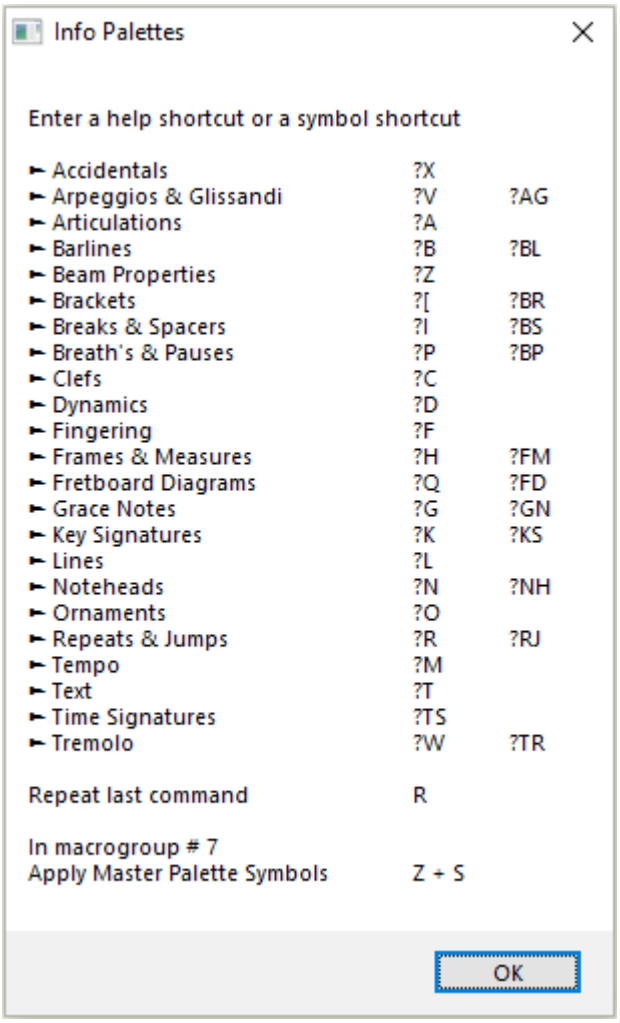
SLF_01_SelectionFilter_All_ticked.png

Reference section - F2 Apply Palette Symbols

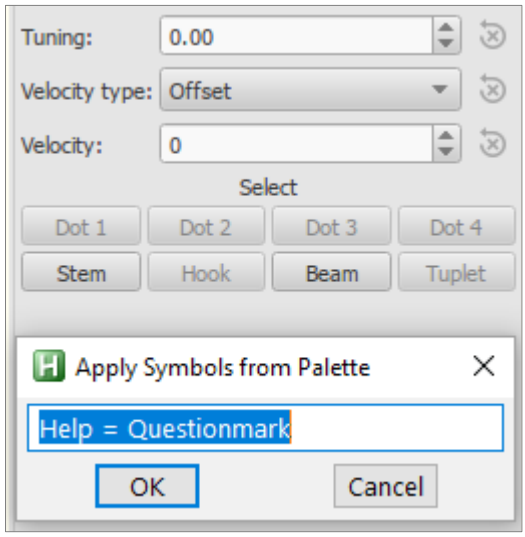
Supporting document: F2_DATA.txt. *Includes all DIY details*

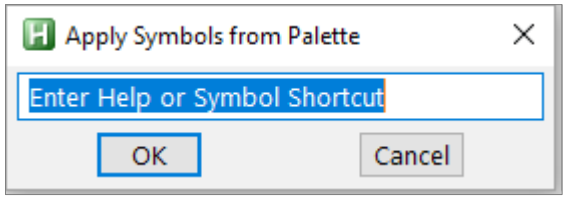
[+ 2 General Info

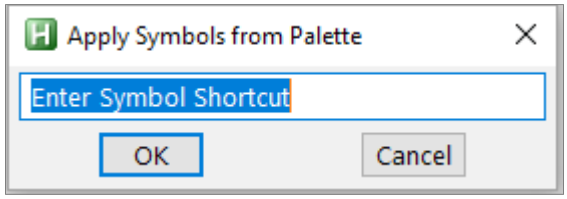




Z + A Apply Symbols from Palette







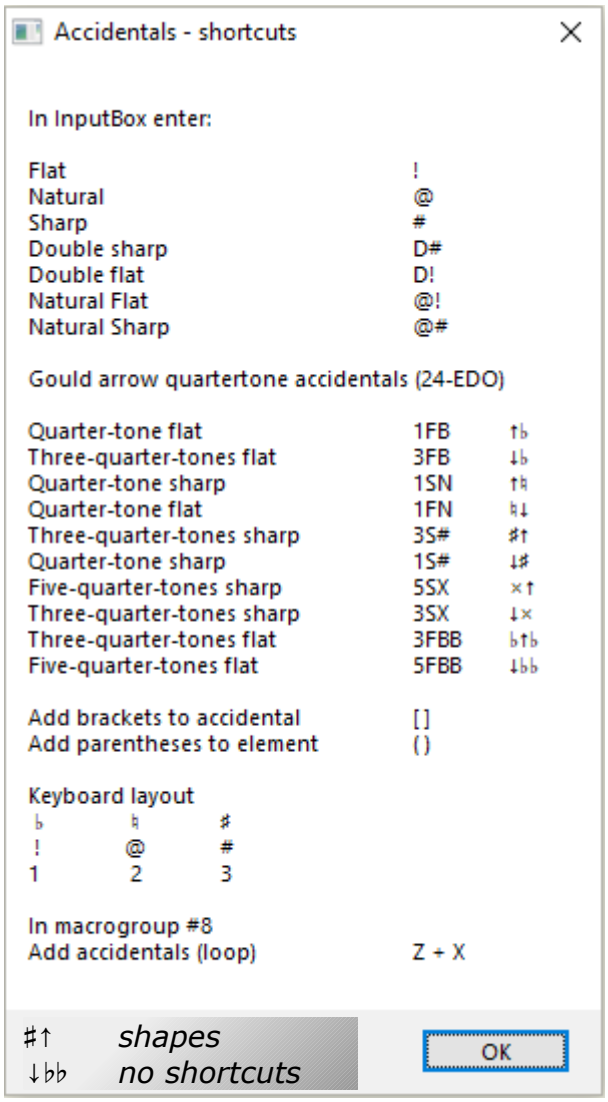
How the macro works:
With the Advanced AHK Workspace active open e.g the Palette Ornaments and hover the mouse over 

The symbol has been renamed in O-3-Trill. The macro will send O-3 to the Palette Search box.
The script uses Ctrl+F9 as the shortcut for Palette Search and Ctrl+Alt+P for 'Apply current palette element'.

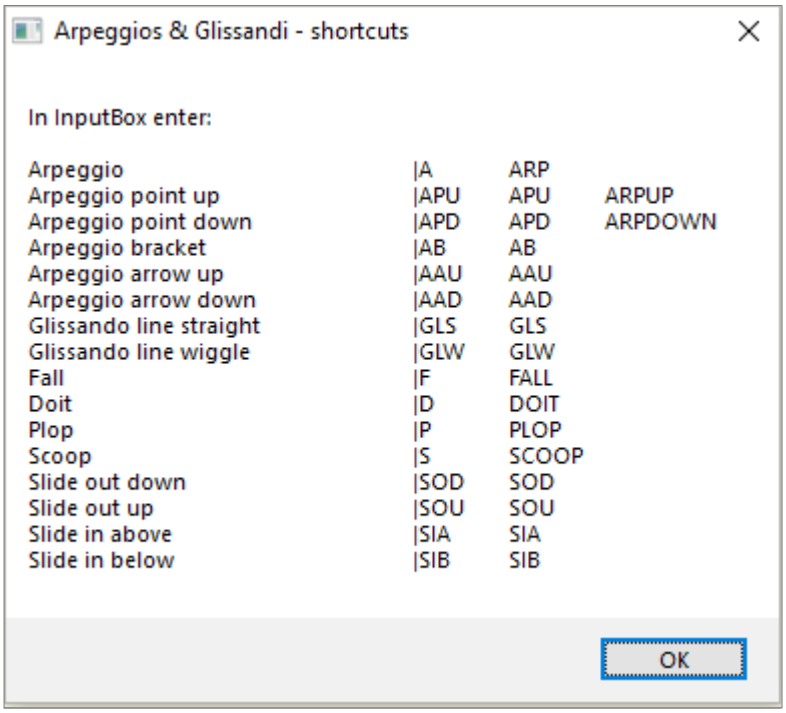
The macro is *Free Canvas*, independent of the screen layout. Two commands however will work better in the Defined State. See *Barlines*. NB: Added to the Workspace: *extra Time Signatures, Gould Arrow Accidentals and barlines.*

Some symbols have more shortcuts

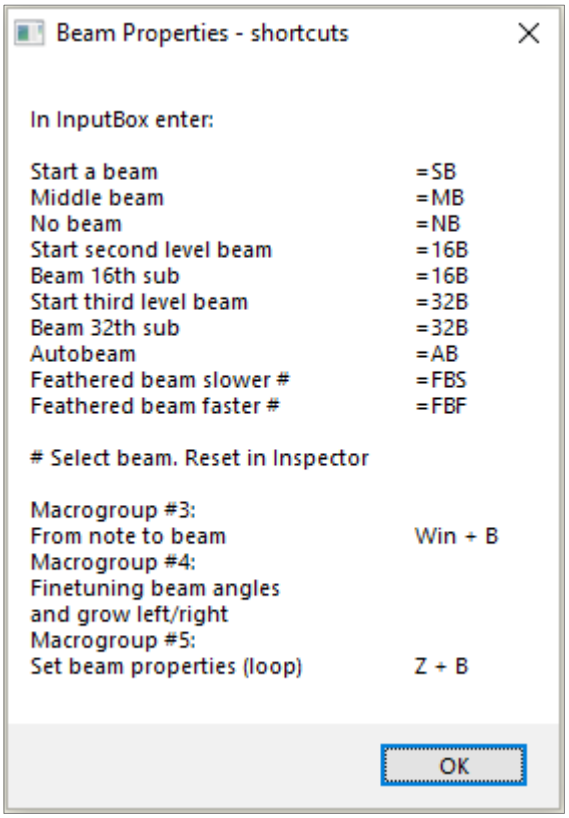
Accidentals



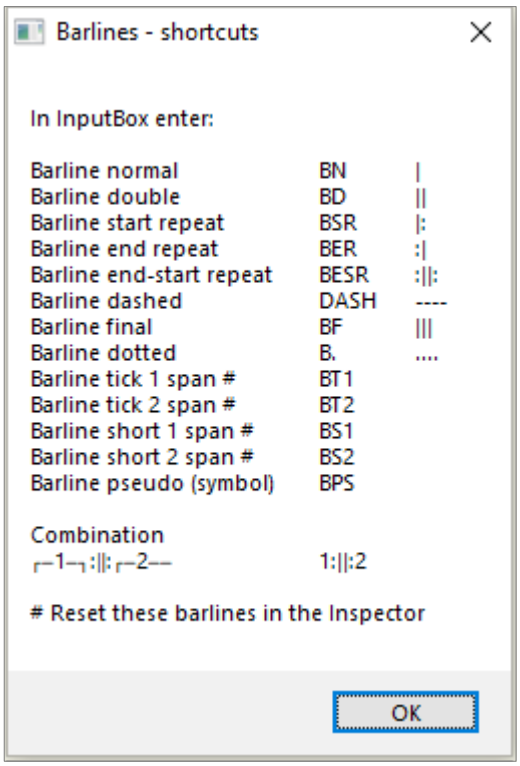
Arpeggios and Glissandi



Beam Properties



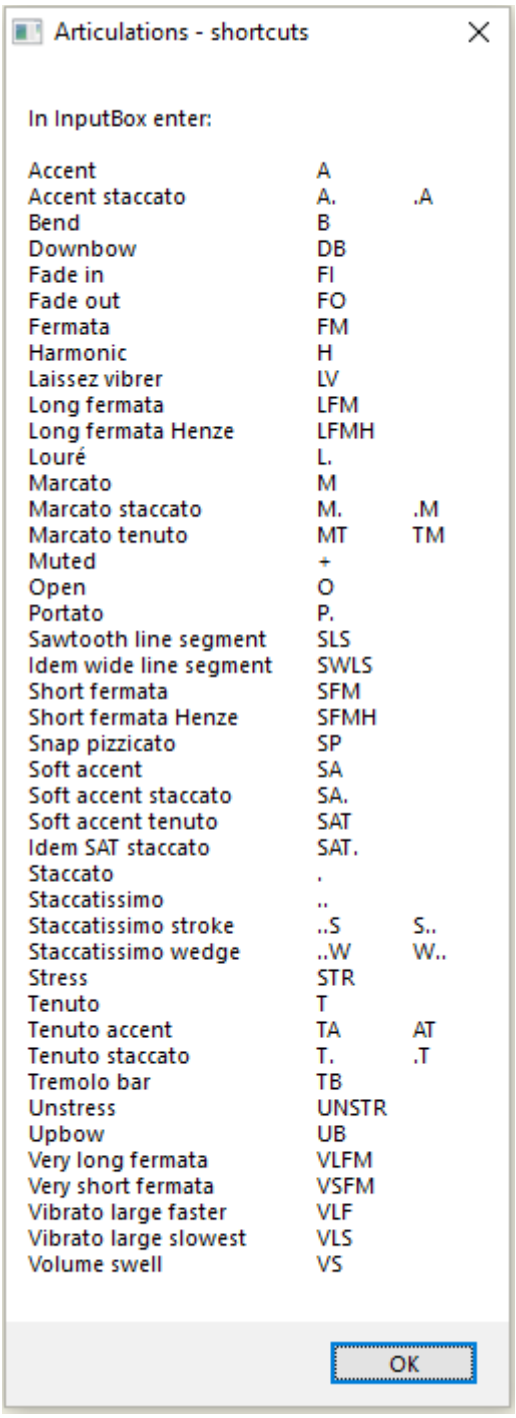
Barlines



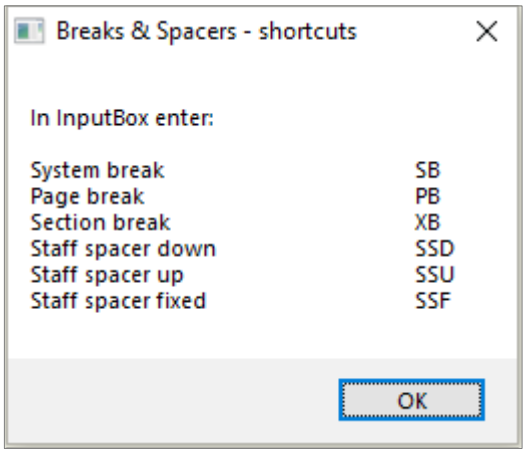
1:||:2 This command needs an image. **`Statusbar_Rest_Duration_Measure.png`** to prevent malfunctioning when the measure is empty. Defined State is not needed but in this case advised.

BPS The 'barline' will stay above the staff when the screen is not in Defined State.

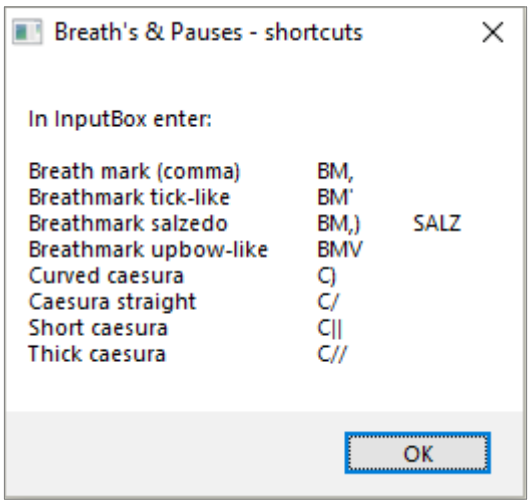
Articulations



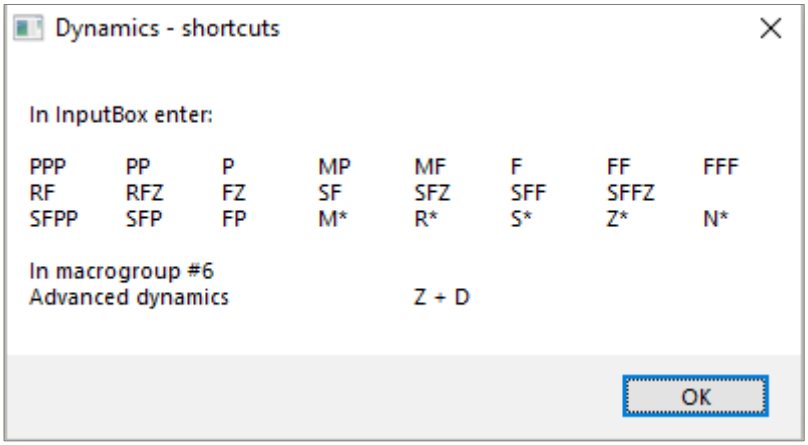
Breaks and Spacers



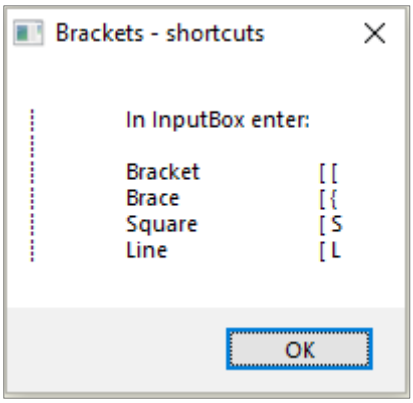
Breath's and Pauses



Dynamics



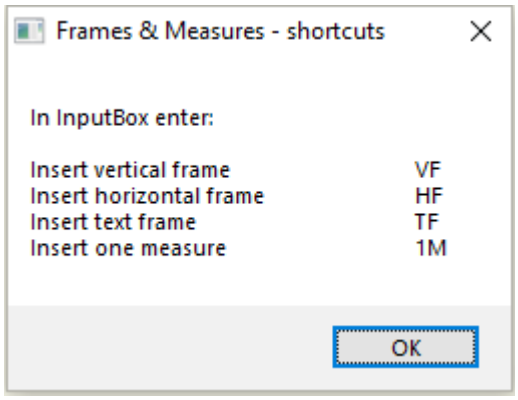
Brackets



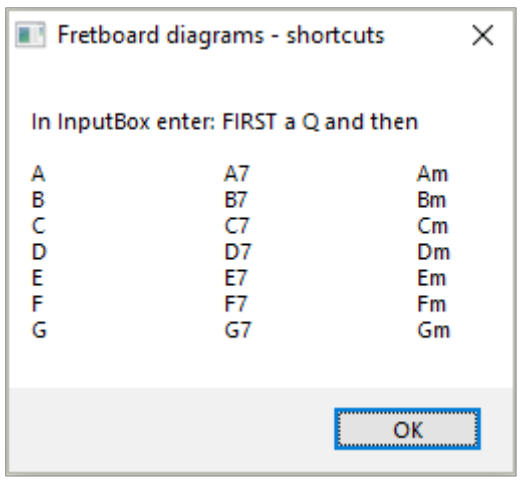
See **supplement**: customizing the Advanced AHK Workspace.

Clefs

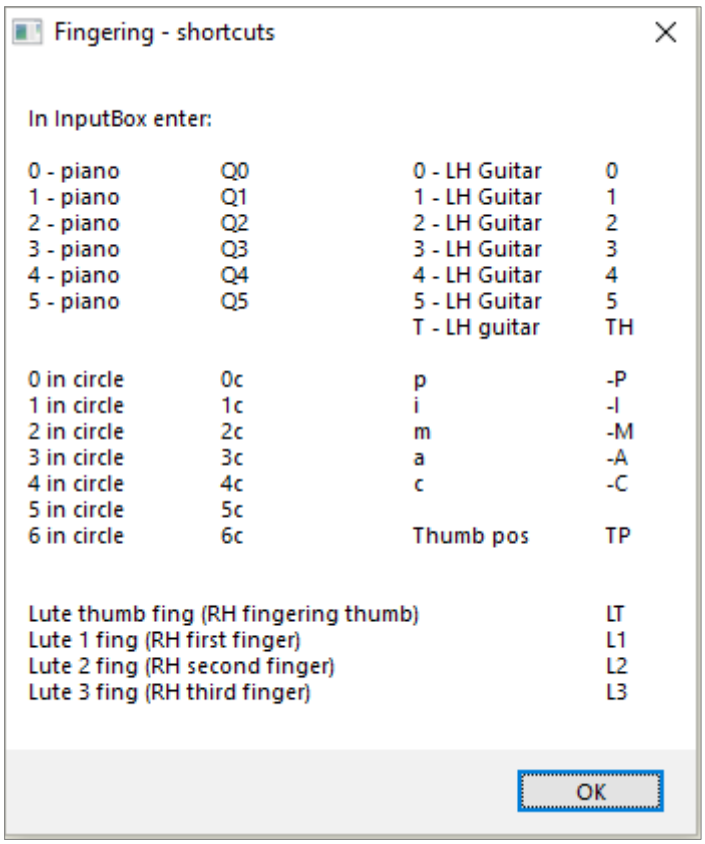
Frames and Measures



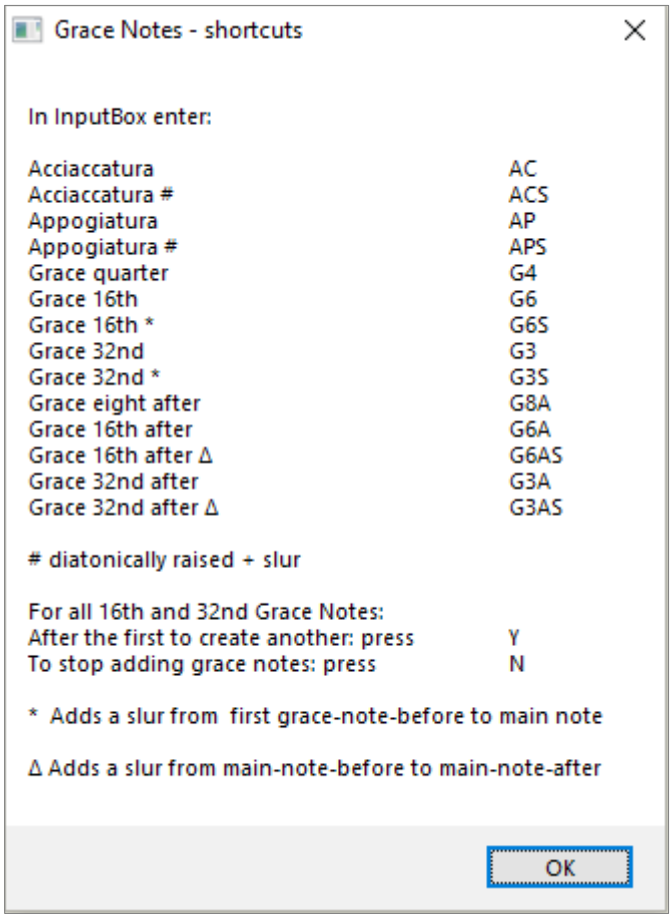
Fretboard Diagrams



Fingering

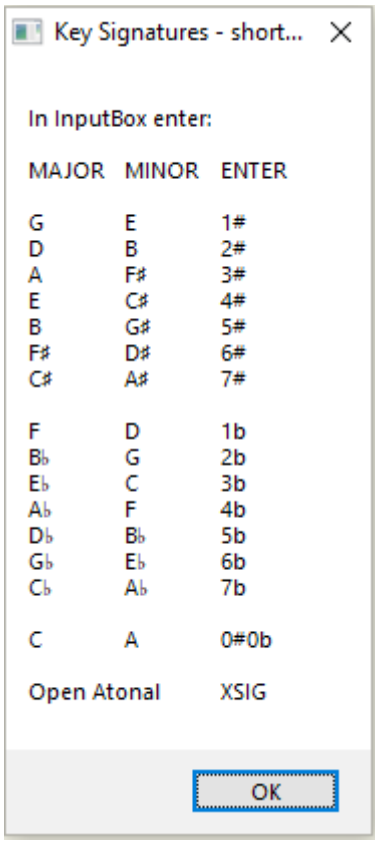


Grace Notes



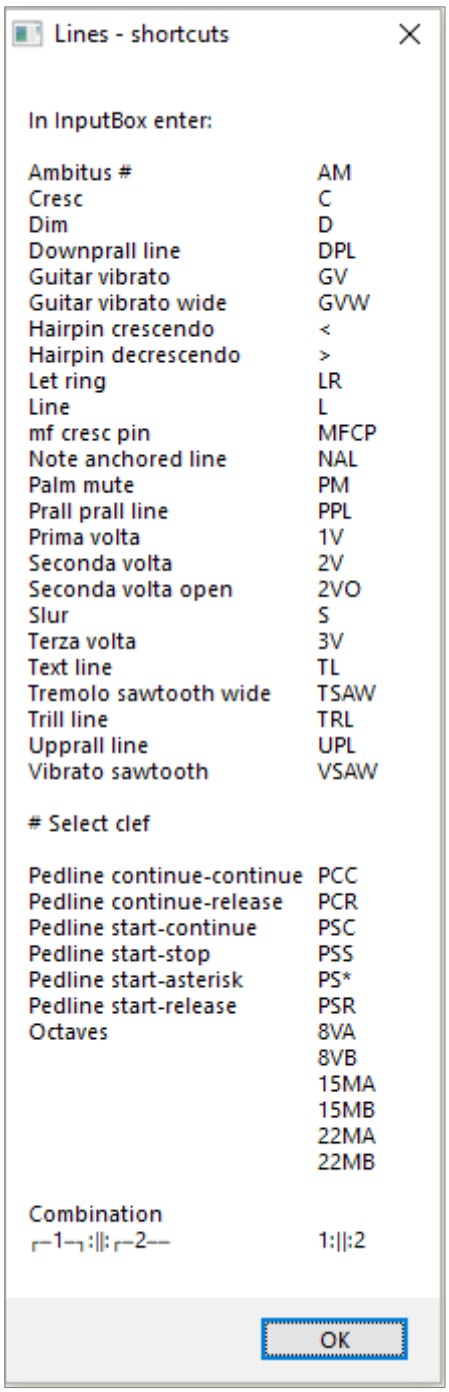
Some Grace Notes shortcuts have variants.

Key Signatures



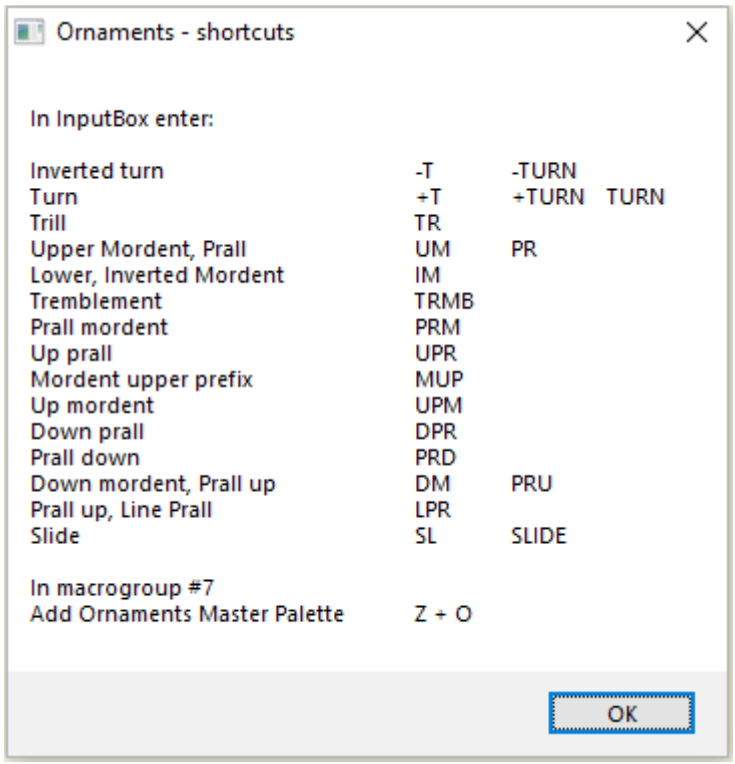
Noteheads

Lines

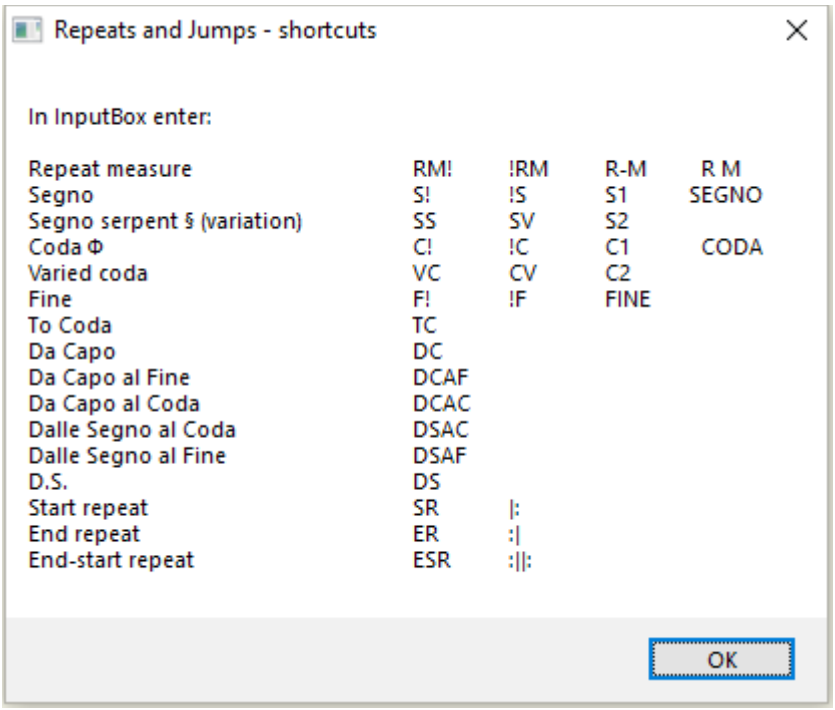


1:||:2 See note *Barlines*

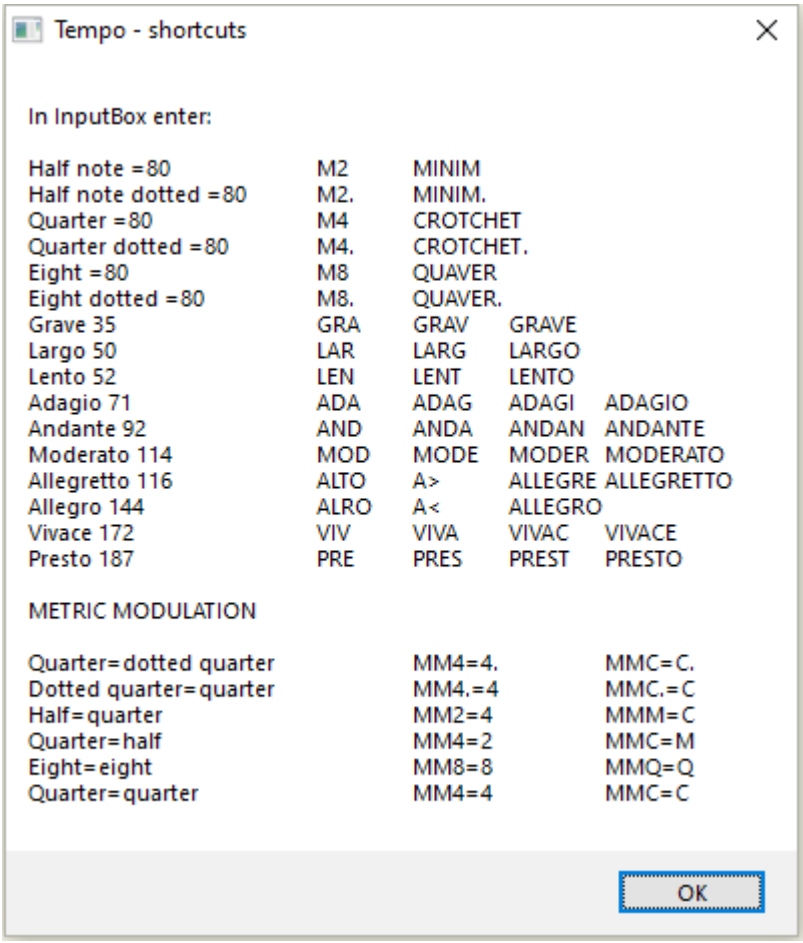
Ornaments



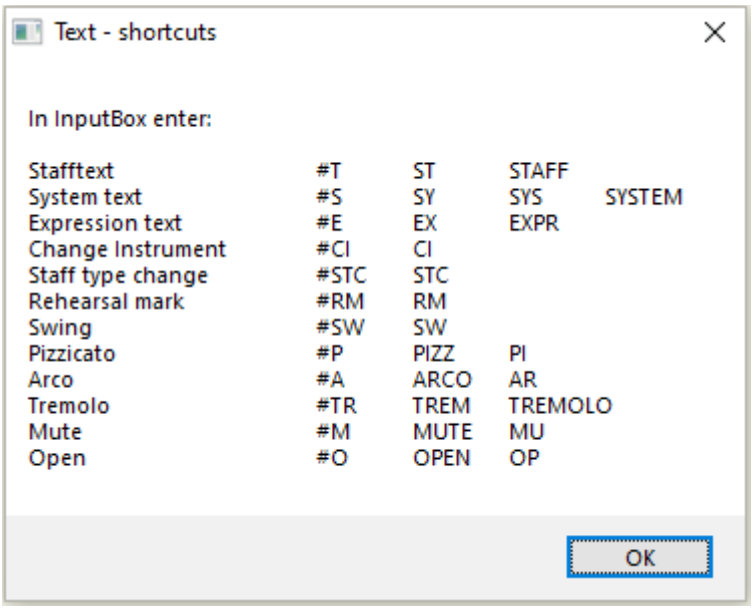
Repeats and Jumps



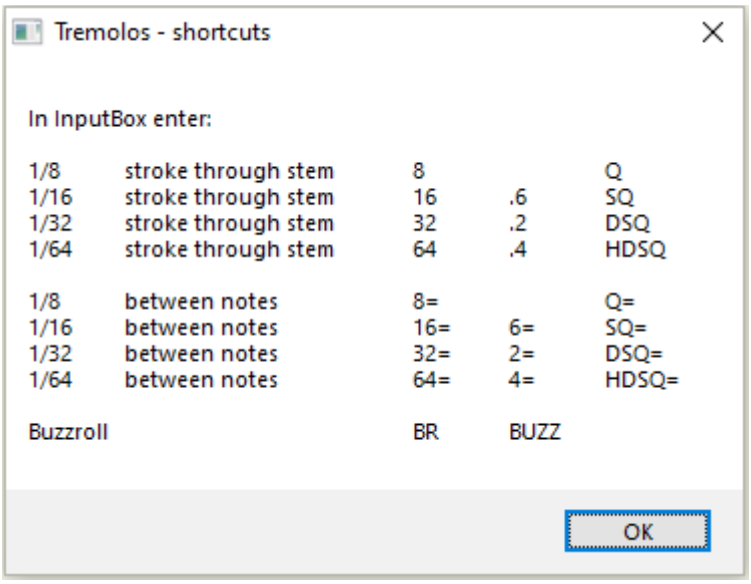
Tempo



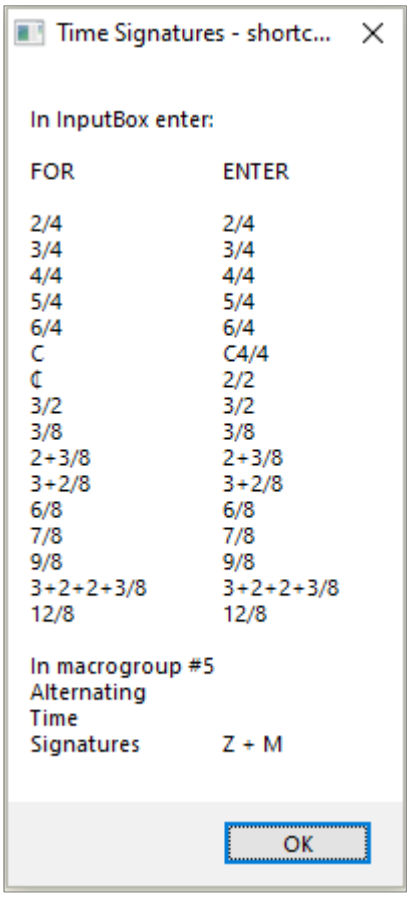
Text



Tremolo



Time Signatures

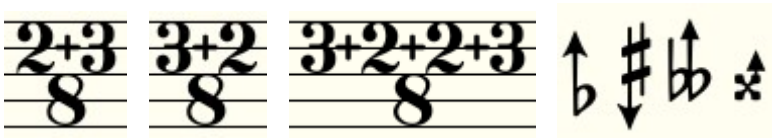


Supplement - customizing the Advanced AHK Workspace

How to add shortcuts for new symbols?

The first publication of the macro was https://musescore.org/en/node/303798
Later some new symbols were added.

1 3 new *Time Signatures*
2 The 10 Gould Arrow *Accidentals*
3 The pseudo barline from the Symbols Master Palette added to *Barlines*.



How to customize a palette:
https://musescore.org/en/handbook/palettes#custom-palettes

How to fit the symbols into the macro:

Time Signatures

https://musescore.org/en/handbook/time-signatures#palette-create

In the *Advanced AHK Workspace:* select the Time Signatures Palette.
Enable editing: select e.g. 2+3/8.
In the Palette Cell Properties we give it the name 2+3/8.
Disable editing.

Names are identifiers, so they have to be unique. The macro must send un unambiguous name to the search field.

In F2_Apply_Palette_Symbols.ahk almost at the top of the file there is an alphabetical shortcutlist to check the uniqueness. As we suspected **2+3/8** is a unique name and so are **3+2/8** and **3+2+2+3/8**. *See also the note* **(*)**.
That means that in this case the names of the symbols can be used as a shortcut. This is of course the most ideal mnemonic. Comparable with **MF** for mezzo forte.

Accidentals - see also page 112 for an alternative way

With the Accidentals we do it slightly different.
The new accidentals are members of a family whose Palette names all begin with an X. Clearly a sensible choice of prefixes makes unique name giving easier. The quarter-tone flat -↑♭ **-** got the name *X-10-Quarter-tone flat*.
After a check of the shotcut list and some reflexion about systematic namegiving the shortcut was coined **1FB** to distinguish it from **1FN**.

In the F2.ahk file this resulted in these lines:

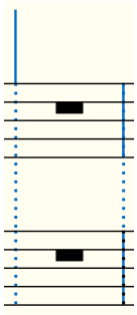
```
If (PaletteSymbol = "1fb")              ; ↑♭ - Gould Quarter-tone flat
Send x-10
```

The shortcut **1FB** - as any new shortcut - *must be entered in three places*.
To find these places search in the F2.ahk file for **1FB**.
- **1FB,** in the outcommented big shortcutlist almost at the top of the file.
- **1FB,** in the big list after the hotkey **~z & a::**
- **1FB,** in the third big list a bit under the last Input Box having the text *Enter Symbol Shortcut.*

Barlines

The added 'pseudo barline' symbol got the shortcut **BPS**.
The barlines are members of a family whose Palette names all begin with a B.
So in the Barlines Palette its name became *B-13-Symbol*.
In the F2.ahk file this resulted in these lines:

```
If (PaletteSymbol = "bps")              ; pseudo barline (symbol)
Send b-13
```



The pseudo barline makes it possible that a Grand staff has a dotted barline between its staves. After selecting a barline and pressing **Z + A ▸ BPS** the pseudo barline is initially positioned above the staff.
If the screen is in the Defined State the macro succeeeds in adding an Y-offset of 4 spaces to get the barline on the right height within the staff.
Adding the pseudo barline was inspired by a forum discussion about Dotted Connected Barlines.

(*) *Note* about sending the new Time Signatures:
In the F2. ahk file the line for sending a TimeSig like 3/8 is

```
If (PaletteSymbol = "3/8")
Send 3/8
```

But for the Time Sig 2+3/8 it is

```
If (PaletteSymbol = "2+3/8")
    Send {Raw}2+3/8
```

The difference is explained by the presence of the **+** sign.
Normally the **+** sign means *Shift* in AutoHotkey.
{Raw} must be added to send the plus sign literally.
More about Time Signatures in F5_Alt_Time_Sigs_and_BeamProps.ahk

Use info screens

It makes sense to add new shortcuts to the Info screens as well.
Search for the info screens using its *Help shortcut*. E.g. **?X** for Accidentals, **?BL** for Barlines and **?TS** for Time Signatures.

In NotePad an Info screen - a MessageBox - may look odd.
Maybe you have to add or remove some Tabs in the text of the MessageBoxes. to make everything look right on the screen. Font Consolas 11 pt.
This is a 'known limitation'.

Switching between Workspaces

During the last years MuseScore's handling of Palettes has become much more versatile in many aspects while adding the shortcuts for *Palette Search* and *Apply current palette element* made the macro **Z + A** possible.

You could think of more applications. Assume for instance you have build an extensive library of fretboard diagrams and several customized palettes. Depending on your favorite workflow you could build a workspace containing all your customisations with appropriate names, copy the essential part of the script, enter the **Send** commands and make all your palette items accessible via say the unused combination **Z + Y**.
The speed of switching between workspaces in 3.6 has increased much compared with earlier versions of MuseScore. And a hotkey for the switch is easily made.

In Master.ahk there is an outcommented hotkey:
Win + 1 Switch workspace - search *#1*

In this way you could keep the 'standard function' of the Advanced AHK workspace and its shortcuts separated from your customisations.