Here is my analysis of the problem of automatically renaming chord symbols when changing the fret of a fretboard diagram.

I assume that it is possible to recognize if to that note where our diagram is attached to there is also a chord symbol attached, too. Further I assume that the character of the key of the piece (of the score) can be recognized (whether it uses sharps or flats – or if it is completely absent which results in C-Major). At last I assume, that it is possible to recognize, how many frets a diagram has been moved by a keyboard or mouse action and the direction of that movement (a mouse action would always result in a movement by 1 fret, I guess).

At first glance it seems to come down to simply change an “E” into an “F” or an “Ab” into a “G” or something like that. It would indeed be as simple as that if there was not the possibility to input the intended fret directly into the fret-number box via keyboard. This makes it a bit more complicated but not so much that there wouldn't be a satisfying solution.

Because of the possibility of direct input via keyboard you have to calculate the number of frets to be moved and the direction (down or up). I guess this isn't really complicated if you are able to get the value of the fret box before and after direct input as well as before and after mouse action on one of the arrow-keys of the fret number box. Are you able to do so? “Value before” minus “value after” shows the number of frets moved and a positive value means moving upwards, whereas a negative value would mean moving downwards.

You could create a method for moving the chord symbols name just one “letter” up or down. That would be a “single-letter-move”, i. e. a move by 1 fret.

Once you know in which direction to move and how many frets, you could iterate x-times the same process as for a single-fret-move. However, this leads to certain naming problems for the target symbol name.

Therefore iteration isn't a good idea, in my opinion. The following questions must be answered in advance:

I. must the target symbol have an accidental or not, or has an accidental that existed in the original symbol’s name to be omitted for the target name or not? (This depends from the number of frets to move. Simplified an even number of frets preserves the state of the original root name – if without an accidental, then the target name is also without – but when with an accidental, then the target name has the same accidental. An odd number of frets moved leads to adding an accidental when the original symbol’s name had been without – or removing the existing accidental from the original symbol’s name.)
II. Do you cross the gap between "E/F" or "B/C" while finding the new root letter for the symbol? (Because then an accidental may have to be omitted that would otherwise be added according to the simple evaluation method.)
III. If an accidental has to be added, which one, a sharp or a flat?
IV. Do you exceed the upper or lower boundary of note names of a scale while moving the fret? (This is dependent from the method for counting root letters for the chord symbol. Whether you use a limited list of root names or not, and if so, with which letter it starts and ends. For example, moving upwards by 5 frets from "A" lands you on "D". This proves to be problematic if your list is only from "C" to "B" but not if it is from "A" to

"G#". But moving downwards by 7 frets from "D" lands you on "G", whereas moving down by 3 frets from "A" would bring you to "F#"). Again it is crucial whether your list starts with “C” or “A”.

All those sophisticated pondering can be circumvented, could even conveniently be solved by using an array of either 12 note names for one octave (and wrap the counting at the upper / lower end) - or an array of, let's say, 36 note names for all possible note names throughout the fretboard (without having to wrap the counting). So you just have to repeat the calculated number of steps on the fretboard within the array! Provided, of course, that you have initially retrieved the original name of the symbol, have identified it in the array and have set the step-pointer to it as a starting point. When the elements of the array contain already the complete note name, i. e. letter plus accidental, you just have to replace the original root letter in the symbol with that from the array where the pointer points to after having done all steps.

Now for the naming problems: Please get at first my basic idea. The key in which a piece is set is usually reflected in the accidentals of the used chords. Harmonizing scale tones of that key results in chord names (we speak of the names only) that all bear the same kind of accidental - if at all. In the C-Major scale doesn't occur any chord with an accidental. From that can be derived, that pieces in C-Major usually contain no chords that bear an accidental in their names. In F-Major there is only one chord with accidental and that is a flat accidental. Other keys down the "cycle of fifth" add only more flats, i. e. when harmonizing scale tones of those key signatures, you'll find only chords with flat accidentals in their names but never with sharps. This holds true even when speaking of double dominants and the like, which are not necessarily scale-chords any longer. For G-Major it is vice versa. For complex pieces as classical ones there may sometimes occur chord names with flats in it along with chord names using sharps – if chord names explicitly written at all, but that’s normally not the case and anyway you can create chord names to your liking via CTRL+K. Concerning diagrams – which are our topic – it is mainly about pop or jazz and for this purpose my claims made above would proof sufficiently valid.
A second basic and rather an obvious idea is, that all chord names consist of 2 elements. The first (mandatory) is the root note upon which the thirds are layered. This is not necessarily the lowest tone of the chord, for diagrams can represent inversions of the given chord, chord symbols never do. (For the moment I ignore the presentation of chord names that indicate a bass note with a slash beneath the root name, which in fact would be an inversion. An explanation follows at the very end). The second element (optional) is a bunch of special signs and/or digits to indicate layered and altered thirds. I call this the “remainder”. When renaming chords moved on the fretboard only the root name is affected. Here we have the problem that the root name can consist either of 1 character or of 2. Therefore, if we extract the second character from the chord name and it is a “#” or a “b”, it belongs to the root name. In all other cases it belongs to the remainder. The remainder, as complex and fancy as it may be, stays as is and is of no interest to our topic. Even if the second character is a “-“ or “m” or “minor” it belongs to the remainder, as well as a “+” or “$^{\emptyset}$” or “$^{o}$”. In the end, it means only altering the third low or altering the fifth high / altering the third and fifth low / altering the third, fifth, and twice the seventh low. “C$^{\Delta 7}$”, “C$^{j7}$” or “C$^{maj7}$” or “C$^{major7}$” are different ways to denote a major seventh. We all know that there is no compulsory standard for chord naming and have to live with that … The good news is that they all belong the remainder, too. Regarding layered and altered thirds as in “$^{b5}$, $^{\#5}$, $^{b9}$, $^{\#9}$, $^{\#11}$, $^{13}$”, they contain a “$^{b}$” or “$^{\#}$” but those do not conflict with the idea how to separate the root name from the remainder, because they never follow the root name directly

(where the sign altering the fifth or ninth could erroneously be taken for an accidental of the root name), i. e. they never occur in the second position of the chord's name. Always they would be preceded by a "$^{7}$". Even if "C$^{9}$" is a correct chord name although it contains the seventh as well, "C$^{b9}$" is not but "C$^{7b9}$" is! This covers even "C$^{b5}$", for there is no chord with a flattened fifth only. It always contains the seventh, too, and therefore is always written as "C$^{7b5}$" or "C$^{7\#11}$" or "C$^{j7b5}$" or even as "C$^{ø}$" or "C$^{o}$". Hopefully this covers all problematic naming patterns …
As for the worst-case length the "remainder" string could reach, add every possible layer of the additional thirds $^{5}$, $^{9}$, $^{11}$, $^{13}$, plus an accidental "$^{b}$" or "$^{\#}$" for each, plus the "$^{minor}$" for a minor third, plus the "$^{major}$" for the major seventh and the $^{7}$ itself. That makes 19. The "remainder" cannot get longer, not even reach this length, because those additional markings never all occur at the same time, the $^{13}$ hardly has any accidentals and "$^{minor}$" or "$^{major}$" are usually abbreviated. But with 19 you are absolutely on the safe side.

I assume that it is easy enough to retrieve whether the key of the piece or the last key signature before the diagram we move by frets consists of flats or sharps (when transposing, this has to be done in any case). Hence we can define in advance whether the new name of the symbol should use rather flats or sharps if an accidental had to be added. If the key consists of sharps a changed symbol name should contain a sharp as well – if necessary, but never a flat. If the key consists of flats the changed symbol name should only contain flats, if necessary. With C-Major where the "key" consist of no accidentals, sharps shall be added if an accidental would proof necessary. Keys with "mixed" accidentals, sharps *and* flats, do not occur! This method makes you independent from the way the diagram together with its symbol has been stored in the template section. Enharmonic renaming takes place automatically according to the key of the piece when moving the diagram to the intended place on the fretboard.
Another method could be to decide the form of the accidental by the initial root name of the symbol. If it already has a flat, then flats are kept during the move, if it has a sharp already, sharps are kept. If it neither has a sharp nor a flat but is a "plain" letter, the form of the accidental is decided by the direction of the move. Downward moves result in flat accidentals, upward moves in sharp accidentals. But the former method looks more elegant in my opinion.

To be able to decide between sharps or flats at will, you need, in fact, two arrays, one with note names using flats only and one with note names using sharps only. Which array to apply depends on what is parsed from the key of the piece or – using the other method – from the accidental found in the original symbol's name or from the direction of the move respectively. For C- Major no key signature exists and it cannot be decided which array to apply. Then the array with sharps shall be applied.

Now for the wrapping problem: using a limited pair of array, say from "C, C$^{\#}$, D, D$^{\#}$, E, F, F$^{\#}$, G, G$^{\#}$, A, A$^{\#}$, B" and "C, D$^{b}$, D, E$^{b}$, E, F, G$^{b}$, G, A$^{b}$, A, B$^{b}$, B" respectively, makes it necessary to consider it as a circle and when counting upwards from "B" you should continue with "C" at the lower end. When counting downwards from "C" you should continue with "B" at the upper end. This method has the inconvenience of having additionally to calculate wrapping needs, but it makes it simple to identify the "starting point" of the counting within the array because the original root name occurs only once in that array.
If you want to avoid wrapping, though, the arrays have to consist of about 3 octaves:
"E, F, F$^{\#}$, G, G$^{\#}$, A, A$^{\#}$, B, C, C$^{\#}$, D, D$^{\#}$, E, F, F$^{\#}$, G, G$^{\#}$, A, A$^{\#}$, B, C, C$^{\#}$, D, D$^{\#}$" and "E, F, G$^{b}$, G, A$^{b}$, A, B$^{b}$, B, C, D$^{b}$, D, E$^{b}$, E, F, G$^{b}$, G, A$^{b}$, A, B$^{b}$, B, C, D$^{b}$, D, E$^{b}$, E, F, G$^{b}$, G, A$^{b}$, A, B$^{b}$, B, C, D$^{b}$, D, E$^{b}$". The

middle octave represents the real fretboard and starts best with "E" to "D$^{\#}$" or "E$^{b}$" respectively (lower E-string). Beneath that there is one more octave just for counting purposes. Moving a diagram from the 12th fret down then still leads to the appropriate root name even when identifying the starting point for counting at the first tone of the middle octave - and that without wrapping. The same is valid for the part of the array above the middle octave. Moving a diagram from the 1st fret up to the 12th while the root name is identified at the twelfth tone of the middle octave (B-string), as well leads to the correct root name. This method is convenient in not having to wrap, just count the steps for the pointer. However, you have to pay attention to where you identify the starting point for the counter, for every root name exists several times within the array – it has always to be the middle octave!

Summary: use two long arrays of 3 octaves, with complete tone names, one for flats, one for sharps. Parse the character of the key signature - flat or sharp, which decides which array to use. Parse the actual Root name of the chord symbol combined with the diagram. Calculate the number of frets moved and the direction of the movement from the comparison of the fret number box before and after the keyboard/mouse action ("after" minus "before"). Identify the root name within the chosen array's middle octave (above the element 12) and place the pointer on it. Move the pointer in the array by the calculated number of steps into the recognized direction (negative means moving downwards). Retrieve the new root letter from the array's element where the pointer now points to and replace the old root name in the symbol with it. Done.

```
SYMBOL_NAME = get_chord_symbol_name();                          # cord symbol exists
if SYMBOL_NAME = “” then goto end_processing;                   # chord symbol doesn’t exist
ARRAY_SHARP = “E, F, F#, G, G#, A, A#, B, C, C#, D, D#, E, F, F#, G, G#, A, A#, B, C, C#, D, D#, E, F, F#, G, G#, A, A#, B, C, C#, D, D#”;
ARRAY_FLAT = “E, F, Gb, G, Ab, A, Bb, B, C, Db, D, Eb, E, F, Gb, G, Ab, A, Bb, B, C, Db, D, Eb, E, F, Gb, G, Ab, A, Bb, B, C, Db, D, Eb”;
:start_processing;                                              # another mouse/keb. action?
set FLAT = false; set SHARP = false;                            # defaults
set FRETNUMBER = 0; set MOVED_FRET_NUMBER = 0; set OFFSET = 0;  set POINTER = 0;    # defaults
set SOURCE_NAME = “”; set REMAINDER = “”                        # defaults
KEY_SIGNATURE = get_key();                                      # retrieve last key signature
if pos(“b”, KEY_SIGNATURE) > 0 then FLAT = true                 # consists of flats only
else SHARP = true                                               # empty or consist of sharps
end                                                             #
FRET_NUMBER = get_fret_number();                                # initial fret of the diagram
If ((SYMBOL_NAME[2] = “b”) or (SYMBOL_NAME[2] = “#”)) THEN      # of root name contains accidtl.
   SOURCE_NAME = SYMBOL_NAME[1,2];                              # separate root name … (2 chars)
   REMAINDER = SYMBOL_NAME[3,19]                                # … from remainder (can’t surely …
else                                                            # … be more than 19 chars)
   SOURCE_NAME = SYMBOL_NAME[1];                                # ditto (but only 1 char)
   REMAINDER = SYMBOL_NAME[2,19]                                # ditto
end;                                                            #
If (fretbox_mouse action() or keyboard_achtion()) then          # mouse or keyboard activity?
   MOVED_FRET_NUMBER = get_fret_number();                       # final fret of the diagram
   OFFSET = (MOVED_FRET_NUMBER – FRET_NUMBER);                  # number of frets and direction
   if focus_is_not_on_ diagram() then OFFSET = 0;               # another object marked
   if OFFSET = 0 then goto end_processing;                      # done
   if FLAT then                                                 # Use array with flats
```

```
    POINTER = (pos(SOURCE_NAME, ARRAY_FLAT[13,12]) + 12);      # identify source root name
    POINTER = (POINTER + OFFSET);                              # identify target root name
    TARGET_NAME = ARRAY_FLAT[POINTER]                          # retrieve target root name
  else                                                         # use array with sharps
    POINTER = (pos(SOURCE_NAME, ARRAY_SHARP[13,12] + 12);      # identify source root name
    POINTER = (POINTER + OFFSET);                              # identify target root name
    TARGET_NAME = ARRAY_SHARP[POINTER]                         # retrieve target root name
  end;                                                         #
  TARGET_NAME = concat(TARGET_NAME, REMAINDER)                 # complete target chord symbol
  push_target_name_to_chord_symbol();                          # replace name of "chord symbol"
  SYMBOL_NAME = TARGET_NAME;                                   # new "chord symbol" name in case
end;                                                           # another action takes place with
goto start_processing                                          # fret of the same diagram
: end_processing                                               # all done
```

I'm not familiar with the programming style and the actually available functions of MuseScore. This Pseudo code is just to illustrate the steps that need to be taken. Syntax is of course not correct but should be understandable. The procedure is triggered only by a mouse or keyboard action on the fret number box and is dropped as soon as the diagram loses focus, i. e. as soon as another object is marked – or none.

I am convinced, after some brain juice has been spilled, the solution will look comparably simple, even elegant.

Limitations: I don't deny, that there may be situations when a user wants a certain diagram to be named in another way than the proposed procedure will result in. Then manual editing would be unavoidable. But these situations will be too rare, that they could challenge the advantage of the described method.

The described method ignores so far the behavior of indicating a certain note beneath the root name explicitly, separated from it by a slash and often subscript, as in "$C_{/B}^{9}$". Including explicitly indicated bass notes into the automatic calculation of a new root name when moving the respective diagram might be considered a pain in the neck by some programmer! That's why I completely ignored it at first. He would have to separately identify the indicated bass note and to twice calculate the shifting. Fortunately, this bass note – if indicated at all – follows the root name immediately before the remainder containing the alterations of thirds. Otherwise, it would prove more cumbersome to retrieve it from the remainder. I spent quite a while thinking over this, though. Here is what I have found, but that's getting a little polemical.

Bass note indication is not a regular means. It occurs only if explicitly intended by the composer and therefore explicitly included in the chord symbol. Fretboard diagrams often constitute an inversion of the chord and at the lowest point they have a note, which is not the root of the chord. In return chord names always give the basic naming regardless of the true structure of the chord shown in the diagram. If a bass note is indicated explicitly, it denotes often the third of the chord, for thirds are normally avoided in bass position. Of course, other notes can also appear there in order to form kinda "bass melody" together with preceding and subsequent chords. Bass indication, however, only makes sense if the composer fears that the guitarist interpreting the chord name would apply it in a different inversion than the composer intends – or if that note is normally not an element of the

chord name. Actually, adding a non-chord note to the “normal” chord (whatever is considered normal at this moment) makes the chord simply a different chord and hence the name of the chord should be changed to include this note, too. In fact, there is no such thing as a chord with a “wrong” note. Still, incorporating the indicated bass note into the proper name may result in chord names difficult to decode and mislead the guitarist to a “wrong” rendering. But hey, that’s why we added the diagram, isn’t it? Then there isn’t really a need for explicitly indicating the bass note at all – it is shown in the diagram. Only without a diagram, the explicit indication of bass notes in special cases is indispensable.
Concluding from all this pondering I claim, if a bass note should explicitly be indicated in the symbol, it can be considered part of what I defined as the "remainder", and the solution described above can stay unchanged. Of course, such a bass note is not changed when moving the diagram and becomes “wrong”. This, however, is only a problem when for a diagram already existing in the stave and having a symbol with an explicit bass note the fret will be changed anew. Then manual editing is necessary. In my humble opinion this also falls into the category of those rare problems that can legitimately be made the responsibility of the user. Moreover, it is not certain whether the bass note should still have the same relative relationship to the root note after shifting the diagram. In any case, the regular situation will be that the user selects a diagram from the sample section and attaches it to a note and that this diagram brings along its own chord symbol. Possibly he would adapt the fret of the diagram yet and only after this he would add an explicit bass note. This always means editing the chord symbol. I cannot imagine that anybody would store diagrams with symbols containing explicit bass notes into the template section!

If one should be keen on automating even this issue, however, the crux is to separate the chord name into 3 parts. Between the “root name” and the “remainder” there comes the “bass section”, yet. Therefore we would need some other variables BASS_PART etc. for our code that default to “” (null string), too. Identifying the root name works as described. The second character has just to be evaluated whether it contains a “b” or a “#”. All other characters belong to the “remainder”. But then the “remainder” once more has to be parsed for its first 2 characters separately. If the first character of the “remainder” is NOT a “/”, the remainder stays as is and the “bass section” stays empty. Done. If, however, the first character of the “remainder” should be a “/” then the second character has to be changed with exactly the same method and using the same arrays as for the root name itself. The first 2 characters have to be stripped off the “remainder” and the “bass section” has to be filled with the “/” followed by the newfound bass tone. Concatenation then includes “root note” + “bass section” + “remainder”. Done. As I realize by now, that would inflate the code of the procedure not so much. Maybe this supplement should be added after all?

In our pseudo code example the 3 variables
`set BASS_NOTE = “”; set TARGET_BASS_NOTE = “”; set BASS_PART = “”; # defaults`
have to be added and initialized in the defaults section. The assumed maximum length of the remainder then should be considered prolonged by 2 to be on the safe side. That is 14 and not 12 in
`… REMAINDER = SYMBOL_NAME[3,19]` ⇨ `… REMAINDER = SYMBOL_NAME[3,21]`
`… REMAINDER = SYMBOL_NAME[2,19]` ⇨ `… REMAINDER = SYMBOL_NAME[2,21]`

The line
`TARGET_NAME = concat(TARGET_NAME, REMAINDER);`
should be replaced by the following lines:

```
if pos("/", REMAINDER[1]) > 0 then                          # explicit bass indication detected
   BASS_PART = REMAINDER[1]; BASS_NOTE = REMAINDER[2]       # separate bass note name
   if FLAT then                                             # Use array with flats
      POINTER = (pos(BASS_NOTE, ARRAY_FLAT[13,12]) + 12);   # identify bass name
      POINTER = (POINTER + OFFSET);                         # identify target bass note
      TARGET_BASS_NOTE = ARRAY_FLAT[POINTER]                # retrieve target bass note
   else                                                     # use array with sharps
      POINTER = (pos(BASS_NOTE, ARRAY_SHARP[13,12]) + 12);  # identify bass name
      POINTER = (POINTER + OFFSET);                         # identify target bass note
      TARGET_BASS_NOTE = ARRAY_SHARP[POINTER]               # retrieve target bass note
   end;                                                     #
   REMAINDER = REMAINDER [3,12];                            # strip bass note from remainder
   BASS_PART = concat(BASS_PART, TARGET_BASS_NOTE)          # complete target bass part
end;                                                        #
TARGET_NAME = concat(TARGET_NAME, BASS_PART, REMAINDER)  # complete target chord symbol
```

Best regards Michael, Gimmel