Tutorial: Execute Commands and cmdutils

Running a command against bars or objects in a selection

Bob Zawalich July 10, 2021

When you run a command in Sibelius or in **Execute Commands**, you will typically find that it either processes all the objects in the selection or just the first one. I was running the command Bar Number Change, and I wanted to apply it to each bar in the selection. The command will only work on one bar, and that is the first bar in the selection.

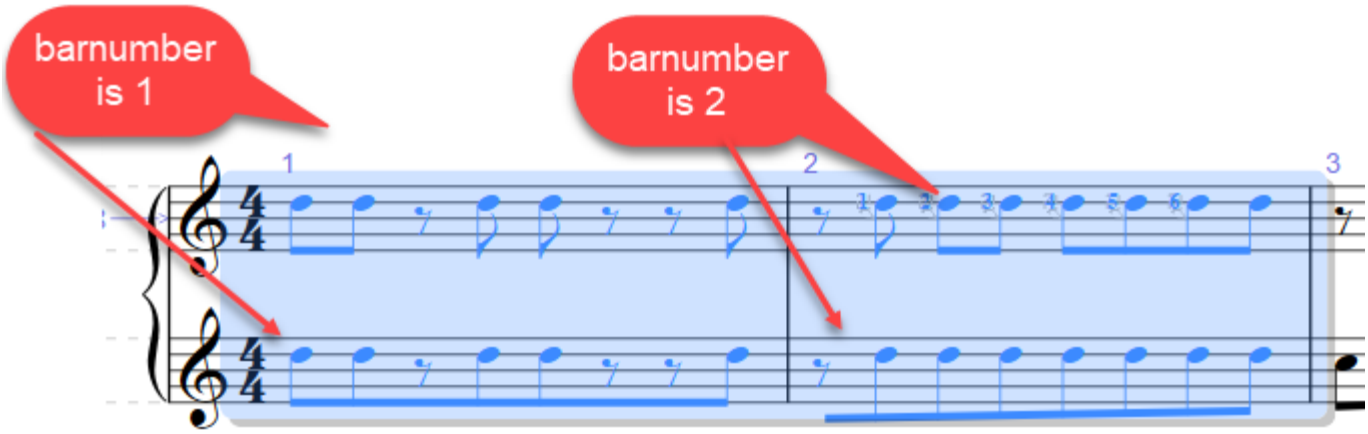
To do what I wanted to do, I could not write a macro, but I had to go into ManuScript and set things up so I could run the command on each bar in the selection.

I broke the problem into 2 parts. The first was to get a list of the bars I wanted to process.

This itself could be complicated, especially is I started with a non-passage selection, because I might have multiple selected objects in the same bar, and I might encounter the same bar in multiple staves. In some cases I would want to process each such bar separately, but for a bar number change, I really only want to apply the change once per bar number.

Also: the bar objects I want are held in the selection, but in order to run the command on each bar, I will have to select each bar, and that will trash the selection. So I can't just process each bar while looping through the selection. I will have to collect all the objects I want first, then save off the selection, trash it to select each bar, and then restore the selection. A pretty common scenario, actually.

Let's say I started with this:



If I wrote the code

```
for each Bar barCur in selection
{
        trace(barCur.BarNumber);
}
```

I would see

```
1
2
1
2
```

I really only want to process each barnumber once.

To avoid this kind of issue with duplicate bar numbers, I find it handy to read my objects into a **Dictionary**, whose key is the **bar number,** because a dictionary by nature overwrites duplicates. It is also one of the few structures that can hold Bar Objects and bars. So I started with this, filling the dictionary with selected Bar objects:

```
dict = CreateDictionary();
for each Bar b in selection
{
	dict[b.BarNumber] = b;
}
```

If I trace the keys (names) after it was done

```
for each Name name in dict
{
	trace(name);
}
```

I see
1
2

which is what I want. I second bar 1 overwrote the first in the dictionary, leaving me with a single entry. I have to be careful that the bar I store with the statement

```
	dict[b.BarNumber] = b;
```

will work for what I want, but I know that in this case it will.

See the tutorial on Dictionaries in the ManuScript Reference if that does not make sense to you.

Once I have the bars in the dictionary, I can trash the selection (which I would save first, and later restore), and this is what I wrote:

```
// I want to have bn changes that use the option, which is unavailable to plugins,
// so just bring up the BNC Command warning user which option to pick
// the command works on the selection, so I have to select each bar, then run the command

selection.StoreCurrentSelection();

for each Name name in dict
{
	barNew = dict[name];
	selection.SelectPassage(barNew.BarNumber);
	Sibelius.Execute("bar_number_change");
}

selection.RestoreSelection();
```

The first thing I did was save the original selection using **selection.StoreCurrentSelection();**
Then I extracted each bar from the dictionary using

```
for each Name name in dict
{
	barNew = dict[name];
```

At this point, barNew is a Bar object. I now want to make a passage selection of only that Bar object, so I wrote:

```
	selection.SelectPassage(barNew.BarNumber);
```

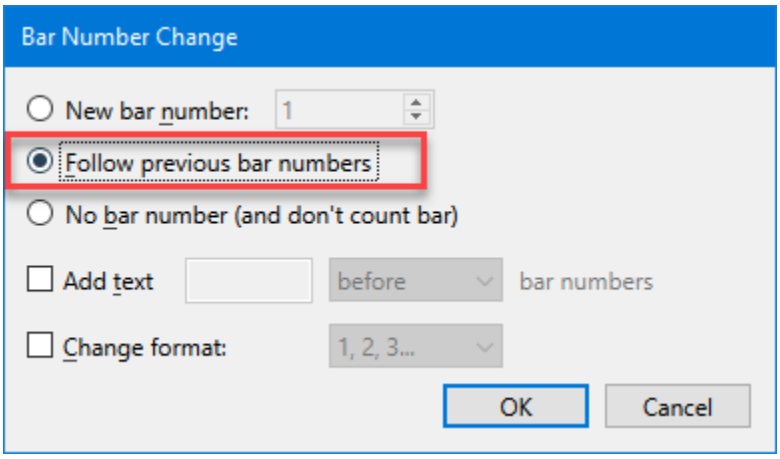
and then I can run the command, which will act on the first (and only) bar of the selection:

```
Sibelius.Execute("bar_number_change");
```

Finally, I restore the original selection so that everything looks the same as when I started

```
selection.RestoreSelection();
```

In my example, I will add a bar number change to bars 1 and 2. The way the **Bar Number Change** command works is that it will bring up a dialog with several options, and I will need to tell it what I want. In this case I am using the command, rather than creating a bar number change directly, because I need to use a feature unavailable to ManuScript, **Follow previous bar numbers**:



I could have used a ManuScript call to add a new bar number, but in this case all I wanted to do was make the bar numbers visible. If I use **Follow previous bar numbers**, these bars will adjust their bar numbers if someone adds a bar number change earlier in the score some time in the future. If I had added actual bar numbers, they would remain unchanged if someone made a change earlier in the score, and it might go undetected. I try to avoid things like that when possible.

The takeaway from this tutorial is that you often need to process a number of selected objects and then need to change the selection. The model shown here works well for that:

- Save the selection
- Gather objects into a dictionary or sparse array
- Process the objects, changing the selection as needed
- Restore the selection

Since Sibelius Commands and cmdutils Commands tend to work on the current selection, you may find yourself in this situation fairly often. You can't really do this sort of thing in a command sequence/macro, but it requires only a fairly minimal amount of ManuScript code.