overview.org

November 6, 2023

1 clog-gui-widgets

1.1 Overview

The package provides the infrastructure and clog guis for Hardware Midi Controllers. It needs `clog-dsp-widgets` and `cl-midictl` (including their dendencies) to work and consists of two parts:

- Midi Controllers

  These are defined in `cl-midictl` and establish a representation of the hardware controllers, their connection to the midi input output and their state as visible to the application.

  A Hardware midi controller instance is initialized with `add-midi-controller`, which adds the controller instance to the cl-midictl:*midi-controllers* hash-table.

  Midi input is handled by `generic-midi-handler`. The routine first stores the incoming values in `*midi-cc-state*` or `*midi-note-state*` and then calls the `handle-midi-in` method in all controllers registered in `*midi-controllers*`. This routine should take care of updating the slots of the midi-controllers according to the defined behaviour for the hardware controller. Generally this state is equivalent to the state in `*midi-cc-state*`, but -in the case of non-motor sliders or buttons- it can provide mechanisms to ”catch” values in case the state of the controller instance has changed through code rather than midi input, by comparing the `cc-state` of the controller instance to the incoming values stored in `*midi-cc-state*` and act accordingly, e.g. by only updating the controller state in case the incoming value is matching it.

- Controller Gui

  The controller gui class is defined in `clog-gui-widgets`. It contains slots for all the gui elements (being clog instances in the gui) and a slot containing the controller instance defined in cl-midictl. This seperation is necessary as there might be more than one Gui active (in case of multiple connections to the server), which all relate to the same controller instance. The methods of the gui instance maintain the gui state of all connected guis and the synchronization between the controller instance and the guis.

  The synchronization between the gui and the controller is done in the following way:

  The gui elements simply set the corresponding slots of the controller instance in their val-change-cb function. As the slots in the hardware controller instance are implemented as `cellctl:value-slots`, changing their value (using the `val` method) triggers the ref-set method of the controller slot instance.

  The `ref-set method` takes care of defining the code to synchronize all gui elements and the visual elements of the hardware controller (leds, etc.). It is set up for all controller slots in the `initialize-instance :after` method of the gui controller class.

1.2 Usage

In order to use a gui controller in an application, its `on-new-window` method needs to ensure, that the corresponding midi controller exists before instantiationg the gui instance (using `find-controller` or initializing a new hardware controller in case it doesn’t yet exist). Then it instantiates the gui controller instance, providing it with the controller instance using the :midi-controller keyword.

1.3 Defining new gui controllers

A new gui controller can be defined accordingly: In case it is just an alternative gui to an already existing hardware controller, the elements of the gui need to get defined in the class definition and its appearence in the `initialize-instance :after` method. In this method also the updating behaviour needs to be defined by setting the ref-set-hooks of the slots used in the hardware controller instance.