2014

Sam McCormack

[MIDI INTERFACE USER MANUAL]

A quick start guide to integrating the C# MIDI Interface with .NET compatible applications including a rundown of the methods required. Note that the interface is currently designed to work with the BCF2000 alone and any additional MIDI devices may cause unexpected behaviour.

1. FORM BASED SETUP

The interface can be initialised by creating a new instance of **InterfaceForm** and calling the **Show()** method, from there the majority of the parameters can be changed from within the form itself.

The form can be minimised to the system tray and subsequently maximised by clicking on its icon in the tray.

2. FORM–LESS SETUP

When attempting to integrate the interface without showing the form, a number of methods need to be called, these methods are dependent on whether the system requires static or non–static accessors:

- **HardwareSetup.Init()** – for static initialisation
- **initialise()** – within the HardwareSetup class for non–static initialisation

3. RELEASING RESOURCES

All resources for the Form based method are released when the form is closed, with the form–less setup however the following method needs to be called to release all resources before closing the application:

- **HardwareSetup.Rel()** – for static release
- **release()** – from within the HardwareSetup class for non–static release

4. GET AND SET PARAMETERS

All parameters are stored in the **parameters** class. The buttons, faders and knobs are stored in the following public arrays:

- **Control** stores all faders and knobs without scaling in float format, faders are stored in indexes 0–7 and knobs are stored in indexes 8–15
- **Button** stores all buttons on the interface in Boolean format
- **Faders** stores all scaled fader/knob pair values

These arrays can be accessed by calling the following methods:

- **parameters.returnAllX()** – returns the entire Control array
- **parameters.returnScaledX()** – returns the entire Faders array
- **parameters.returnY()** – returns the entire Button array

To set a specific parameter the following methods are used:

- **parameters.setX(int index, float val)** – sets the value of Control[index] to val
- **parameters.setY(int index, bool val)** – Changes Button[index] to val

Scaling is applied through some internal arrays, the scaling values can be adjusted on the form or alternatively through use of the following methods:

- **parameters.loadScale()** – load scaling parameters from Properties.Settings
- **parameters.saveScale()** – save any changed scaling parameters to Properties.Settings
- **parameters.setScale(int pair, float low, float, hi)** – sets the upper (hi) and lower (lo) scaling bounds for a fader/knob pair
- **parameters.setSensitivity(int pair, float value)** – Sets the sensitivity of the fader for a fader/knob pair

The **loadScale()** method is called when the form is created and **saveScale()** is called when the form is closed, hence it would only be necessary to call these methods when using the interface without the form.