***DRFX**: Dynamic Routing for FX*

[DRFX] is an abstraction that automatically creates a signal routing system and associated controls based on inputs and effect modules that you specify. This allows you to make any type of series or parallel connection chain between your inputs and effects, and change routing on the fly. To use this system, you need to create your effect modules as sub-patches or abstractions that follow some basic conventions, which are described below. Effect modules will be referred to as *plugins* in the rest of this document, but have no relationship to VST, AAX, or AU plugins used in other digital audio workstation software.

Creating FX Plugins

The file PLUGIN-SHELL.pd in the plugins directory illustrates how to make a plugin called `fxname`. The best way to understand it is to look at the patch itself, but the most important things it contains are:

- a subpatch to do effect processing
- a [catch~] with a receive name argument `fxname-in`
- a [send~] with a send name argument `fxname-out`
- a [throw~] with a send name argument of your choice (for the main audio output bus to the speakers).
- GUI controls for output gain, tap output gain, and all parameters needed for your effect
    - notice that these controls have send/receive names that follow specific conventions, like: `fxname-out-gain` and `fxname-out-gain-set`. These conventions must be followed exactly to enable [DRFX]'s system for storing data and exporting presets.

You don't need to make your plugin an abstraction, but in most cases it makes sense to do so. That way, you can re-use it in various projects, or even within the same project if you need multiple plugins that do a specific type of processing. The PLUGIN-SHELL.pd example is intended to be an abstraction, and incorporates a symbolic creation argument into its send/receive names.

Specifying Inputs

Once you have created plugins, you can tell `[DRFX]` about your input signals. The `input-list` message to `[DRFX]` must specify the send/receive names of all input signals you intend to send to the routing system. Input signals must be broadcast via [send~] objects. The `[DRFX]` help patch broadcasts two input signals, named `sine-pulse` and `noise-pulse`, so the `input-list` message is:

```
input-list sine-pulse noise-pulse
```

Specifying FX

The `fx-list` message to `[DRFX]` must specify the names of all effect plugins. The `[DRFX]` help patch uses 6 plugins in the `[pd FX-plugins]` subpatch, named `echodel-A echodel-B phaser-A pitch-shift-A pulser-A reverb-A ring-mod-A`. The appropriate `fx-list` message is therefore:

```
fx-list echodel-A echodel-B phaser-A pitch-shift-A pulser-A reverb-A
ring-mod-A
```

Specifying FX Parameters

The `fx-params` message to `[DRFX]` must specify the send/receive names of all effect parameters. The values associated with these send/receive names are monitored and stored internally so that routing and effect settings can be automatically exported as preset files. Any number of parameters can be listed, and there is no restriction on naming. In the `[DRFX]` help patch, there are a total of 8 parameters for the 6 effect plugins, and the `fx-params` message is:

```
fx-params echodel-A-deltime echodel-A-fdbk echodel-B-deltime echodel-
B-fdbk phaser-A-rate pitch-shift-A-transpo pulser-A-rate reverb-A-rvb
ring-mod-A-freq
```

Creating the Matrix

When the `input-list`, `fx-list`, and `fx-params` messages have been sent, a matrix can be created with the `create` method. This method initiates a dynamic patching process that creates all the routing objects, GUI controls, and internal storage needed for the specified system. The `destroy` method clears all dynamically created content.

The Routing Matrix

Once a routing system is created, the `matrix` message will bring up a subpatch containing the GUI routing matrix composed of toggles. This is automatically created based on your input and FX lists. You can manually click on routing matrix toggles to try out various connections; however, the patch automatically creates send/receive names in the properties of all the toggles, which means that you can compose FX chains directly with messages.

For instance, to activate the toggle for a connection between `sine-pulse` and `ring-mod-A`, you would send the message:

```
sine-pulse-ring-mod-A-switch-set 1;
```

You can deactivate the connection by sending a `0` instead. The fade time of this connection is 40ms by default, but can be changed globally for all routing switches via the `global-ramp-time` method. It is also possible to change the ramp time of any individual switch in the matrix with broadcast messages, e.g.:

```
sine-pulse-reverb-A-switch-ramp-time 15000;
```

A full routing chain command should also address the FX `out-gain` (which is the level of the signal sent out of the FX plugin before going to the next plugin in the chain) and the FX `tap-out-gain` (which is the level of the signal sent to the main audio bus). An example of a basic chain involving parallel and serial connections would be:

```
sine-pulse-ring-mod-A-switch-set 1;
ring-mod-A-out-gain-set 100;
ring-mod-A-reverb-A-switch-set 1;
reverb-A-out-gain-set 100;
reverb-A-tap-out-gain-set 96;
;
sine-pulse-echodel-A-switch-set 1;
echodel-A-out-gain-set 100;
echodel-A-tap-out-gain-set 92;
```

Such a command could be in a message box connected to preset controls, but it could also be in a cue list text file loaded by the `[qlist]` object, which makes it possible to actualize very specific combinations of effects that change cue-by-cue in performance.

Exporting Preset Files

When using the routing matrix manually in order to explore various FX combinations, you can take a snapshot of all routing, gain, and FX parameter settings by using the `export` method. This method writes a text file containing the broadcast messages needed to recreate all settings. The file can be loaded directly with `[qlist]` in order to restore settings, or its text can be copied into message boxes directly within your patch.

Additional Settings

Several convenience methods exist for clearing routing and muting outputs. These are specified in the `[DRFX]` help patch.