Edisyn

A Java-based Synthesizer Patch Editor
Version 10
By Sean Luke
sean@cs.gmu.edu

Edisyn is a no-nonsense synthesizer patch editor designed to be extensible for many different kind of of synthesizers, and to have unusual programming assistance capabilities. It is not skewmorphic and not skinnable: it's design is very no-nonsense and consistent. Edisyn is free open source.

Edisyn at present has patch editors for the following synthesizers, which strangely enough are the very ones that I own!

- Kawai K4 and K4r (Single and Multimode)
- Waldorf Blofeld Desktop, Blofeld Desktop SL, and Blofeld Keyboard (Single and Multimode)
- Waldorf Microwave II, Microwave XT, and Microwave XTk (Single and Multimode)
- Yamaha TX81Z (Single and Multimode)
- Oberheim Matrix 1000 (Single)
- Preen FM2 (Single)

Edisyn does not support patches for global parameters, nor for wave or wavetable editing.

1 Starting Edisyn

If you're on a Mac, Edisyn will look like a standard application, just double-click on it. On other platforms, Edisyn comes as a single Java jar file. Just double-click on the jar file (you'll have to have Java installed) and Edisyn should launch.

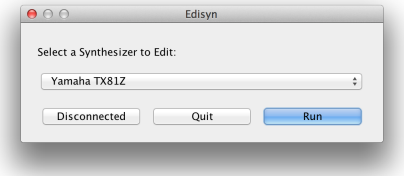

Figure 1: Initial Synthesizer Dialog

You'll first be presented with the dialog at right, asking you to choose a synthesizer patch editor. You can either connect to a synth then and there, or run in **Disconnected Mode**, where you're not attached to MIDI. You can also quit immediately.

Edisyn will now build a patch editor for you and display it. But unless you chose **Disconnected Mode**, it'll first ask you to set up MIDI for this editor. The dialog at right presents you with up to 6 fields (5 are shown here):

- The USB MIDI Device from which you will **Receive** MIDI data sent by the synthesizer. Here we are sending to a Tascam US-2x2 interface, which presents itself as a generic, nameless device.
- The USB MIDI Device to which you will **Send** MIDI to ultimately be sent to the synthesizer. Here we are sending to a Tascam US-2x2 interface, which presents itself as a generic, nameless device.

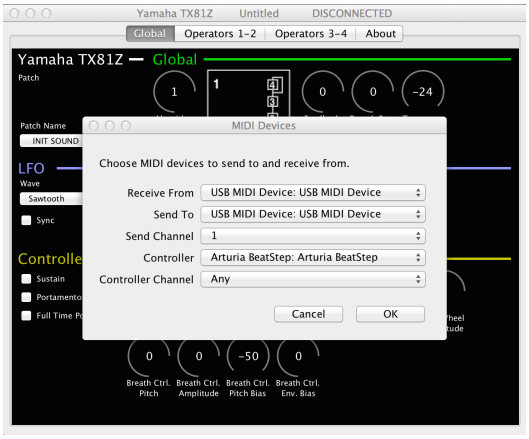

Figure 2: MIDI Dialog

- The **Channel** on which the synthesizer is listening. (Here, 1).
- (Not visible here) The optional **ID** of the synthesizer. Some synthesizers require a special ID embedded in their sysex so they can tell that the message is for them rather than another copy of the same synthesizer. (The Yamaha TX81Z doesn't have an ID, so it's not displayed in this example).
- The USB MIDI Device from which you will receive MIDI data sent by a **controller**. This may be a controller keyboard to play test notes on the synthesizer, or it may be a control surface to send CC data to the synthesizer or to Edisyn itself. Here we are receiving from an Arturia Beatstep.
- The **Channel** over which you will receive MIDI data sent by a **controller**. This can be any specific MIDI channel, or (in this example) "Any", meaning any channel or OMNI.

If you are not connected to MIDI, or if you cancel, then Edisyn will inform you that you must continue in **Disconnected Mode**.

Important Note At present, due to bugs in the MIDI library Edisyn relies on, if you connect a device to your computer *after* you have fired up Edisyn, Edisyn will not be able to see it. You'll need to restart Edisyn if you want to use that device. So connect your devices first.

2 Edisyn Patch Editors

An Edisyn patch editor is a single window with multiple tabbed panes. You can switch tabs by clicking on them or via shortcuts (see the **Tabs Menu**). The far-right tab is the **About Tab**. It gives you information about the eccentricities of the synthesizer that require custom behavior in Edisyn (they all do!). You should read it carefully to understand how Edisyn will interact with your synthesizer.

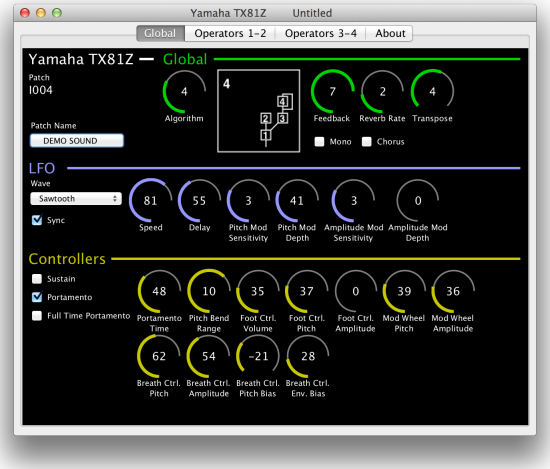

Figure 3: Typical Patch Editor Panel (TX81Z)

Categories At right is a typical tab pane. You'll note that various widgets are grouped together in regions (called **Categories**). There are four categories shown here. Three are various random categories for this synthesizer: "Global", "LFO", and "Controllers". They're in various colors to differentiate them. Other categories will be found in other tab panes. But one category is special: the **Synthesizer Category**, always shown in white, here named "Yamaha TX81Z". It normally contains the patch name and bank/patch number.

Widgets Edisyn has a number of widgets. Here are some:

- The **Patch Display**, currently showing patch "I004". Sometimes this display will be inaccurate, particularly if you manually change the patch on the synthesizer while Edisyn is running; or if Edisyn has no idea what the patch should be (it'll usually display a default value like, in this case, "A001".
- The **Patch Name Button**, currently showing "DEMO SOUND". Click on this button to change the name of your patch. A dialog will pop up to let you change the sound, with an additional **Rules** button to explain the constraints the synthesizer places on patch names.

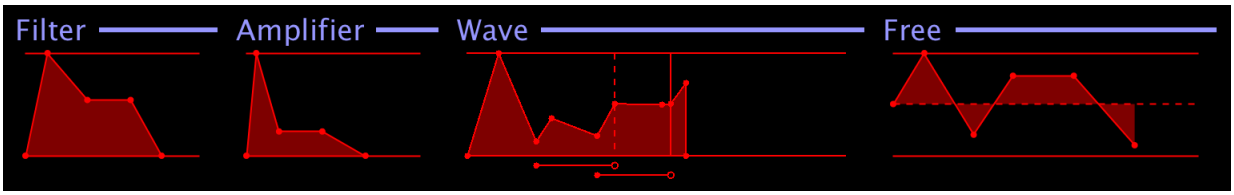

Figure 4: Envelope Displays of the Waldorf Microwave II, XT, and XTk.

- Various **Dials**.[1] These are semicircles in gray, partly in some other color, with a value in the center. You change these values by clicking on the dial and dragging vertically. You can also double-click on a dial to reset it to a default value (often zero). Finally, you can two-finger drag (on the Mac), or spin the mouse wheel to change the dial more subtly.

  Dials vary in orientation. Most look sort of like a "C", with the zero point at the bottom center. Other dials are symmetric, such as the "Breath Ctrl. Pitch Bias" dial (bottom row, second from right in the Figure), and have zero point at center top. Occasionally dials have other orientations: the goal is to keep the zero point centered (at top of bottom).

- Some **Checkboxes** (such as "Portamento") and **Pop-Up Choosers** or ComboBoxes (such as "Wave", set to "Sawtooth"). These are should be straightforward.

- Various **Pictorial Displays**. Here, changing the "Algorithm" dial will modify the Algorithm Display immediately to the right of it.

- Various **Envelope Displays**. Edisyn can draw envelopes using a variety of procedures. Consider the Waldorf Microwave envelopes in Figure **??** above, for example. The first two envelopes are ADSR envelopes, but the third is the Microwave's famous "Wave Envelope", an eight-stage envelope with two different looping intervals (shown below it), and with two special end times marked with vertical lines (here, the dashed line is where optional sustain occurs, and the solid line is the end of the wave). The last envelope is the Microwave's "Free Envelope", a four-stage envelope unusual in that it can have both positive and negative values: the dashed line is the axis.

  At present Edisyn can only draw envelopes: you can't drag the dots.

- **Action Buttons**. Some patch editors have buttons on them which perform actions rather than edit or display values. For example many multimode-patch editors have buttons that pop up single patch editors for the various individual patches.

If you are connected to a synthesizer over MIDI, then changing a widget will modify the underlying patch parameter in real time, if the synthesizer supports this. Also, if you modify a parameter on the synthesizer, then Edisyn will update the corresponding widget or widgets (again, if the synthesizer supports this).

3 Creating and Setting Up Additional Patch Editors

A patch editor is created by selecting one of the various **New...** menu options in the **File** menu. You have to create a new a patch editor before you can start loading a patch from a file or from the synthesizer. You can also **Duplicate** an existing patch editor (in the **File** menu). This will exactly duplicate the existing patch as well.

Whenever you create a new patch editor or duplicate one, you will once again be asked to set up MIDI as discussed in Section **??**, or to run in Disconnected Mode.

[1] Notably absent are **Scrollers**. Edisyn has software support for scrollers but to be honest they've not proven useful yet.

3.1 Persistence and Preferences

Now would be a good time to mention an Edisyn feature you may never notice: many things are **persistent**. For example, if you choose "Arturia Beatstep" as the controller for your Blofeld patch, the next time you call up a Blofeld patch editor, "Arturia Beatstep" will be presented as the default choice in the MIDI Devices window, assuming your Arturia Beatstep is plugged in. This goes for everything in the MIDI Devices window.

Furthermore if you pop a new patch editor for a synthesizer you have never edited before, the Arturia Beatstep will be the default option for that one too (until you change it one time). And these options are per synthesizer type.

Persistence appears in other places too. For example, the Initial Synthesizer dialog will default to the last synth you chose in that dialog. And certain menu choices are persistent as well.

4 Loading and Saving Files

You can save your edited patch via the **Save** and **Save As...** options in the **File** menu, and you can load a patch via the **Load...** option. This is called *Load* and not *Open* because you can only load a file into an existing patch editor: you cannot create a new patch editor automatically on opening a file.

Most patch editor files are sysex dumps ending in the extension `.syx`. These files are usually exactly the same sysex data that you'd normally dump to your synthesizer using a patch librarian software program. There are exceptions however. For example, some synthesizers, like the PreenFM2, have no sysex to speak of at all: they exchange parameters entirely over NRPN. In this situation, Edisyn has invented a sysex file just for the PreenFM2. It obviously won't work in your librarian software.

5 Communicating with the Synthesizer

First things first: if you're working in Disconnected mode, you'll need to set up MIDI before you can communicate with your synthesizer. This is done by selecting **Change MIDI** in the **MIDI** menu. (By the way, you can go Disconnected by selecting **Disconnect MIDI** in the **MIDI** menu as well). Remember that you have to connect USB devices to your computer *before* starting up Edisyn, or it won't see them, due to a bug in the MIDI subsystem.

Now that you're up and running, if you change widgets in the patch editor, many (not all) synthesizers will automatically update themselves. The opposite happens as well: changing a parameter on the synthesizer will update it in Edisyn. See the About pane to determine if your synthesizer can't do this.

By selecting **Request Current Patch**, you can also ask your synthesizer to send you a dump of whatever patch it is currently running. It is often the case that synthesizers respond in such a way that Edisyn cannot tell what the patch number or bank is. In these cases Edisyn will reset the patch number to some default (like A001).

Request Patch... will ask the synthesizer to send Edisyn a specific patch that you specify. Edisyn often (not always) does this by first asking the synthesizer to change to that patch and bank, and then requesting the current patch.

Send to Current Patch will dump Edisyn's current patch to the synthesizer, instructing it to only update its local working memory, and not to store the patch in permanent memory. This operation is not used all that often because most updates (changing widgets, doing bulk edits) do it automatically. But you could use it to sync the synthesizer up.

Send to Patch... will ask the synthesizer to change to a new patch and bank which you specify, then dump Edisyn's current patch to the synthesizer in its working (not permanent) memory. This also isn't used all that much: but some synthesizers (like the PreenFM2 or TX81Z) cannot be permanently written to remotely. Instead you send to a patch, then store the patch manually on the synthesizer itself.

Write to Patch... will ask the synthesizer to change to a new patch and bank which you specify, then dump Edisyn's current patch to the synthesizer to its permanent memory.

Note that various synthesizers cannot do one or another of these tasks: some cannot write to permanent memory (it must be done manually). Others cannot send to the current patch, only to a given patch. Still others cannot request a current patch. Still others cannot update individual parameters, and so on. When this happens, that feature will generally be disabled in the menu. As always, read the About Tab to learn more about what's going on with that synthesizer model.

Finally, if you don't have a controller keyboard, you can send a test note to your synthesizer by choosing **Send Test Note**. You can also toggle whether Edisyn constantly sends a stream of test notes by choosing **Send Test Notes...** And you can shut off all sound on the synthesizer with **Send All Sounds Off**.

6 Communicating with the Controller

The MIDI Dialog (Section **??**) also lets you choose a device and MIDI channel for incoming messages from a control surface or controller keyboard. Using this keyboard you can:

- Play the synthesizer (through Edisyn).
- Control the synthesizer (CC and Program Change messages, etc.)
- Control widgets in Edisyn

6.1 Controlling your Synthesizer

If you play a note, do a pitch bend, etc., on your control surface, Edisyn will route all of those MIDI messages directly to your synthesizer (changing the messages' channel to the one that Edisyn is using to talk to the synthesizer). You can also pass through Program Change messages, MIDI clock, etc. Control Change (CC) messages from your control surface are passed through only if you have toggled **Pass Through All CCs** in the **Map** menu.

6.2 Controlling Edisyn

Edisyn is capable of *mapping* Control Change (CC) messages or NRPN messages from your control surface to specific parameters in your patch editor. Each patch editor type can learn its own unique set of CC and NRPN mappings.

Mapping a Parameter Mapping a parameter is easy:

1. Choose one of three MIDI mapping menu options discussed next. The title bar will say "LEARNING".
2. Select the widget you want to map, and modify it slightly. The title bar will change to "LEARNING *parameter*[*range*]", where *parameter* is Edisyn's name for the synthesizer parameter in question. The title bar might also tell you what the *previous* mapping was.
3. Press or spin the knob/button on your controller. You're now mapped!
4. If you have chosen an absolute mapping, you'll want to change your controller's range to $(0...range - 1)$.

Edisyn accepts any of the following MIDI Control commands. Note that you are not permitted to map CC numbers 6, 38, 98, 99, 100, or 101, or Edisyn will think you're sending NRPN. So you only have 121 CCs to play with.

- **Absolute CC** The value of the CC sent is exactly what the parameter will be set to (between 0...127). To map, choose **Map CC/NRPN** in the **Map** menu. This style is particularly useful for potentiometers or sliders.

- **Relative CC "64"** Here, the CC value you send indicates how much to *add to* or *subtract from* the existing parameter value. In this form of Relative CC, 64 means 0 (add nothing), a value $x < 64$ means to subtract $64 - x$ from the current value, and a value $x > 64$ means to add $x - 64$ to the current value. This style is supported by a number of controllers and is useful for encoders. To map, choose **Map Relative CC[64]** in the **Map** menu.

- **Relative CC "0"** Here, the CC value you send again indicates how much to *add to* or *subtract from* the existing parameter value. In this form of Relative CC, 0 means 0 (add nothing), a high value $64 < x < 128$ means to subtract $128 - x$ from the current value, and a low value $0 > x \geq 64$ means to add x to the current value. This style is also supported by a number of controllers and is useful for encoders. To map, choose **Map Relative CC[0]** in the **Map** menu.

- **NRPN** You are permitted to map any NRPN parameter at all. The value of the CC sent is exactly what the parameter will be set to: all 14 bits. If your controller can only send 7-bit NRPN, then you should configure it to send "Fine" or "LSB-only". Edisyn also supports the NRPN Increment and Decrement options, though those are rare. To map, choose **Map CC/NRPN** in the **Map** menu.

Mapping by Panel or by MIDI Channel In Edisyn, each tab in a patch editor can have its own unique set of mappings: for example, the Oscillators tab might use CC#1 to change the Start Wave parameter, but the Envelopes tab might use CC#1 to change the attack of Envelope 1. CCs on MIDI channels which do not match Edisyn's controller channel are passed through to the synthesizer.

Alternatively you can turn off this behavior and instead send all CCs to Edisyn regardless of MIDI Channel: now two panels can both use CC#1 only if it's on different MIDI channels. This is a common mapping approach used in DAWs. For example, CC#1 on MIDI Channel 1 might change the Start Wave parameter , while CC#1 on MIDI Channel 2 changes the Detune parameter (in the same tab!) and CC#1 on MIDI Channel 3 changes the Envelope 1 Attack (in the Envelope panel).

If your controller can only send a few CCs (it only has a few knobs and buttons) I would use the first option (per-panel mapping). If your controller can send a vast number of CCs, or you're comfortable with it from experience with your DAW, you might use the second option.

You turn on per-MIDI-channel CCs by toggling **Do Per-Channel CCs** in the **Map** menu (and conversely choose per-Panel CCs by toggling it off).

7 Creative Programming

Edisyn has a number of facilities to help you program your synthesizer, including tools to help you wander through the possible space of patches to hunt for the sound you want. Here's what you can do:

Undo Edisyn has infinite levels of undo and redo. When you change a parameter or do a wholesale modification, this can be undone, as can patch dumps and merges from the synthesizer. Individual parameter changes made manually on the synthesizer are not undoable even if they're reflected in Edisyn (it'd be too many). Loading and saving patches is not undoable. See the **Edit** menu.

Reset You can reset the patch editor to its "init patch". Just choose **Reset** in the **MIDI** menu.

Randomize (by some amount) You can add some randomness your patch parameters. Try a small value: values $\geq 50\%$ are essentially full randomization. See the **Randomize** submenu in the **MIDI** menu. Because it's so common to randomize, then undo and try again, you can also do undo-and-randomize-again as a single task: select **Undo and Randomize Again** in the **Randomize** submenu of the **MIDI** menu. See below for a discussion of how randomization (called **mutation**) works in Edisyn.

Nudge The nudge facility lets you push your patch to sound more and more like one of four other target patches you have chosen. You can use this, plus randomize, to wander about in the patch space. Before you can nudge, you have to first select patches to nudge towards. You can pick up to four patches by first setting up or loading the patch in your patch editor, then selecting one of **Set 1** ... **Set 4** in the **Nudge** submenu of the **MIDI** menu. You don't have to ultimately select all four.

Above the **Set** options are four **Towards** options, also in the **Nudge** submenu of the **MIDI** menu. When you set a patch, its current name will appear in the equivalent Towards option. The patch name is just a helpful reminder — it's entirely possible for four completely different patches to have the same name.

Now when you chose any of **Towards 1:...** through **Towards 4:...**, your current patch will get **recombined** with the target patch, currently by 50%, to move it towards that target.

A hint. It's a good idea to select target patches which don't have some radical difference creating a nonlinearity in the space between them: for example, if you were doing FM, I'd pick patches which all used the same operator Algorithm. See below for a discussion of how recombination works in Edisyn.

Merge (by some amount) Merging is a lot like nudging. But instead of nudging towards a predefined target patch, you are asking your synthesizer to load a given patch, which Edisyn will then directly **recombine** with your current patch to form a randomly merged patch. You specify the degree as a percentage: see the options in the **Request Merge** submenu of the **MIDI** menu.

Some patch editors may not be able to perform merges because the synthesizers can't load specific patches: if your synth can't do **Request Patch...**, it probably can't do a merge either.

7.1 How Recombination and Mutation Work

One unusual feature of Edisyn is its support for various ways to recombine (merge) and mutate (partially randomize) patches. These are used in a variety of tools designed to help you hunt for new and unusual sounds. Many systems let you randomize sounds, but Edisyn goes a lot further. Just so you know, here are the recombination and mutation procedures:

Recombination When your current patch **A** is recombined with another patch **B**, you produce a patch which is somewhat in-between the two. You will specify the degree to which this should be done as a percentage 0...100%. This number specifies what percentage of parameters in **A** may (randomly) get merged with their compatriots in **B**. Different kinds of parameters merge in different ways.

- A *metric parameter*, such as a number range from 0...127, will merge with another by picking a random number somewhere between the two. Many dials are metric parameters, for example.
- A *non-metric parameter* is just a set of arbitrary values, such as the Filter types HP, LP, BP, and Bit Crush. A non-metric parameter will merge with another by choosing one or the other of them with a 50/50 chance. Choosers and checkboxes are generally non-metric parameters.
- Some parameters have both metric and non-metric regions: for example, a MIDI Channel dial might have the values 1...16, plus *off* and *omni*. Here the values 1...16 are metric, and off and omni are non-metric. If both **A** and **B** have parameter values which are in the metric region, then they are merged in the metric way. If they're both non-metric, or one is metric and the other is not, then one or the other value is selected.

- Occasional parameters are *immutable*: they will refuse to be merged. These are typically things like patch names.

0% recombination means nothing changes in patch **A**. 100% recombination means that every parameter is recombined: but this doesn't imply that patch **A** is entirely replaced by patch **B**: just that the two are fully merged together.

Mutation When a patch **A** is *mutated*, it is modified with some degree of random noise. Again, you specify a percentage 0...100%, but this means something different than in recombination. Every single parameter will potentially get mutated, but percentage specifies the *degree* to which noise will impact on a parameter. Specifically:

- If your parameter is *metric*, then the percentage specifies the width of a uniform distribution centered at the current value. For example, if your parameter range is 0...127 (and so has size 128), and your current value is 30, and your percentage is 25%, then we will select a new random value within the region $[30 - 128 \times 0.25, 30 + 128 \times 0.25]$, which reduces to $[-2, 62]$. Obviously, -1 and -2 are not valid, so if we choose them, we try again repeatedly until we get a valid number.

  0% mutation means none at all. 100% mutation is guaranteed to be complete randomization over the entire range. But other values may be counterintuitive: for example, 25% sounds small but it's not: it's potentially selecting numbers over half of the range. Generally if you intuitively want to do a mutation of D%, cut your D down by half.

- If your parameter is *non-metric*, the percentage simply specifies the probability that it will be entirely randomized to a new value.

- If your parameter is has both metric and non-metric regions, then we first decide whether to jump the parameter value from metric to non-metric (or vice versa). This is half of the percentage. So if your percentage is 25%, then this happens with a $0.25 \times 0.5 = 0.125$ likelihood. If the value jumps then we pick a totally new random value for it (in its new metric or non-metric region). If the value doesn't jump, then it is mutated just like elements in its region (metric or non-metric) as above.

- Occasional parameters are *immutable*: they will refuse to be mutated. These are typically things like patch names.