The official Raspberry Pi magazine

Issue 32   Apr 2015   raspberrypi.org/magpi

BUILD A STROBE LIGHT
Get the party started with a Pi

ANIMATE GRAPHICS IN PYTHON
How to make your Pygame projects move

RASPBERRY PI WEATHER STATION
The Foundation & Oracle team up for giveaway

FACIAL RECOGNITION MADE EASY
Using OpenCV and the camera module

CROWDFUNDING’S GREATEST HITS
The biggest projects powered by Raspberry Pi & funded by you

> UNICEF PUTS PIS IN LEBANON
> CONQUER THE COMMAND LINE
> MAKE AN ELECTRONIC DOOR LOCK
> MORE OF YOUR BRILLIANT PROJECTS

SONIC PI: MAKE AMAZING MUSIC WITH OUR ESSENTIAL TIPS & TRICKS

Get closer to nature with this awesome wildlife camera kit
THERE ARE NO MISTAKES

This is the most important lesson. The best way to learn is to just try. Try lots of different things; stop worrying whether your code sounds good and start experimenting with as many different synths, notes, FX, and parameters as possible. You’ll discover a lot of things that make you laugh, because they sound awful, and some real gems that sound truly amazing. Just drop the things you don’t like and keep the things you do. The more ‘mistakes’ you allow yourself to make, the quicker you’ll learn and discover your own sound.

USE THE FX

Once you’ve mastered the basics of making sounds with sample and play, you might be wondering what’s next. Did you know Sonic Pi supports over 27 studio FX to change the sound of your code? FX are like fancy image filters in drawing programs, except that instead of blurring or making something black and white, you can add things like reverb, distortion, and echo to your sound. Think of it like plugging the cable from your guitar into an effects pedal of your choice and then into the amplifier, but Sonic Pi makes it much easier! All you need to do is to choose which section of your code you’d like the FX added to and wrap it with the FX code.

```ruby
with_fx :flanger do
  sample :loop_garzul
  16.times do
    sample :bd_haus
    sleep 0.5
  end
end
```

If you wanted to add FX to the `:loop_garzul` sample, you’d just tuck it inside a `with_fx` block, like this:

```ruby
with_fx :echo do
  16.times do
    sample :bd_haus
    sleep 0.5
  end
end
```

Now, if you wanted to add FX to the bass drum, go and wrap that with `with_fx`, too:

```ruby
with_fx :echo do
  16.times do
    sample :bd_haus
    sleep 0.5
  end
end
```

Remember, you can wrap any code within `with_fx` and any sounds created will pass through that FX.
In order to really discover your coding sound, you’ll soon want to know how to modify and control synths and FX. For example, you might want to change the duration of a note, add more reverb, or change the time between echoes. Sonic Pi gives you enough control to do exactly this, with special things called optional parameters. Copy this code into a workspace and hit Run:

```sonic-pi
sample :guit_em9
```

It’s a great guitar sound. Let’s change its rate:

```sonic-pi
sample :guit_em9, rate: 0.5
```

What’s that rate: 0.5 bit we just added at the end? That’s called a parameter. All of Sonic Pi’s synth supports them and there’s loads to play around with. They’re also available for FX:

```sonic-pi
with_fx :flanger, feedback: 0.6 do
  sample :guit_em9, rate: 0.5
end
```

Now, try increasing that feedback to 1 to hear some crazy sounds! Read the docs for full details on all the many parameters available to you.

The best way to quickly experiment and explore Sonic Pi is to live-code. This allows you to start off some code and continually change and tweak it while it’s still playing. So, if you don’t know what the cutoff parameter does to a sample, just play around with it. Copy this code into one of your Sonic Pi workspaces:

```sonic-pi
live_loop :experiment do
  sample :loop_amen, cutoff: 70
  sleep 1.75
end
```

Now, hit Run and you’ll hear a slightly muffled drum break. Now, change the cutoff: value to 80 and hit Run again. Can you hear the difference? Once you get the hang of using live_loop, you’ll never go back. If you were ever to do a live coding gig in the future, you’d find yourself relying on live_loops as much as a drummer relies on their sticks. To learn more about live coding, check out Section 9 of Sonic Pi’s built-in tutorial.

One thing that’s really fun to try is cheat by getting Sonic Pi to compose things for you. A really great way to do this is using randomisation. It might sound complicated, but it really isn’t. Try this:

```sonic-pi
live_loop :rand_surfer do
  use_synth :dsaw
  notes = (scale :e2, :minor_pentatonic, num_octaves: 2)
  16.times do
    play notes.choose, release: 0.1, cutoff: rrand(70, 120)
    sleep 0.125
  end
end
```

When you play this, you will hear a constant stream of random notes from the scale :e2 :minor_pentatonic played with the :dsaw synth. It might not sound like a melody, but that’s the first part of the trick: every time we go round the

“...You can explore as many melodic combinations as you can imagine...”

live_loop, we can tell Sonic Pi to reset the random stream to a known point. It’s like going back to a particular point in time and space with the TARDIS. Let’s try it. Add the line use_random_seed 1 to the live_loop:

```sonic-pi
live_loop :rand_surfer do
  use_random_seed 1
  use_synth :dsaw
  notes = (scale :e2, :minor_pentatonic, num_octaves: 2)
  16.times do
    play notes.choose, release: 0.1, cutoff: rrand(70, 120)
    sleep 0.125
  end
end
```

Now, every time the live_loop loops around, the random stream is reset. This means it chooses the same 16 notes every time, giving you an instant melody. Here’s the really exciting bit: change the seed value from 1 to another number – 4923, say – and it will give you another melody. So, just by changing one number (the random seed), you can explore as many melodic combinations as you can imagine, and that’s the magic of code.