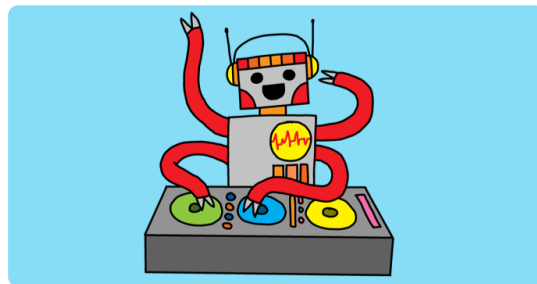




Projects

Live DJ

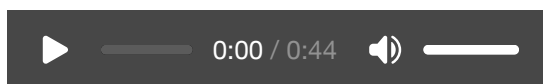
Learn how to code a live music performance.



Step 1 Introduction

In this project you will learn how to code a live music performance, that you can add to and edit without having to stop the music!

Press the play button below to hear how your music will sound:



Additional information for club leaders

If you need to print this project, please use the Printer friendly version (<https://projects.raspberrypi.org/en/projects/live-dj/print>).



Club leader notes

Introduction:

In this project, children will learn how to use `live_loop` to make multiple pieces of music play in time with each other. They will also learn that loops can be edited and synchronised without having to stop and restart the music.

Resources

The 'Project Materials' link for this project contains the following resources:

Club leader Resources

You can find a completed version of this project by clicking the 'Project Materials' link for this project, which contains:

- live-dj.txt
- live-dj.mp3

Learning Objectives

- Sonic Pi 'Live Loop'
- Playing random notes and samples

This project covers elements from the following strands of the Raspberry Pi Digital Making Curriculum (<http://rpf.io/curriculum>):

- Combine programming constructs to solve a problem. (<https://www.raspberrypi.org/curriculum/programming/builder>)

Challenges

- "Changing the drums" - editing the drum samples used;
- "Changing the sample" - editing the sample used;
- "Changing the bass" - editing the notes played;
- "Changing the effects" - editing the effects used.

Frequently Asked Questions

- To find samples available in Sonic Pi, learners can go to jumpto.cc/sonic-pi-samples (<http://jumpto.cc/sonic-pi-samples>). Alternatively, they can just type `sample [space]` and choose from the list that appears.



Project materials

Club leader resources

- Downloadable completed Sonic Pi project (<https://projects-static.raspberrypi.org/projects/live-dj/30395d3843bbf0d35dcab1ada0718f5134c300ee/en/resources/live-dj.txt>)
- Downloadable completed project mp3 file (<https://projects-static.raspberrypi.org/projects/live-dj/30395d3843bbf0d35dcab1ada0718f5134c300ee/en/resources/live-dj.mp3>)

Step 2 Drums

Let's start by creating a simple drum loop.

- Start by creating a `live_loop` called `:drums`.

```
live_loop :drums do  
  
end
```

Any code added to a `live_loop` will repeat until 'Stop' is pressed.

- Add an alternating drum and snare, that play for one beat each.

```
live_loop :drums do  
  sample :drum_heavy_kick  
  sleep 1  
  sample :sn_dolf  
  sleep 1  
end
```

- Press 'Run' to test your drum loop.



- If your drum loop is too fast/slow, you can change the beats per minute (bpm).

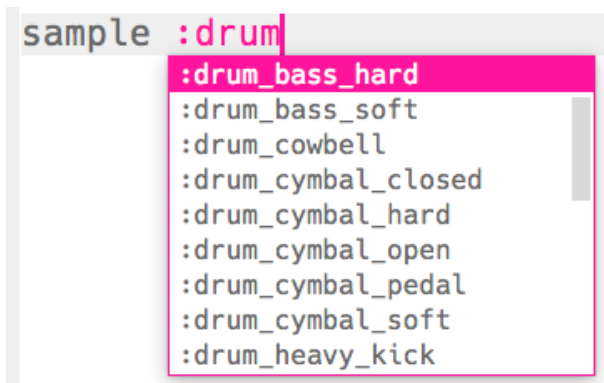
```
use_bpm 65  
  
live_loop :drums do  
  sample :drum_heavy_kick  
  sleep 1  
  sample :sn_dolf  
  sleep 1  
end
```

You'll need to stop and start your `live_loop` to speed it up or slow it down.

Step 3 Challenge: Changing the drum loop

Can you use different samples in your drum loop?

To see what samples are available, you can go to jumpto.cc/sonic-pi-samples (<http://jumpto.cc/sonic-pi-samples>), or just type `sample :drum` and choose from the list that appears.



Step 4 Adding a sample

Let's add a looping sample over the basic drum loop.

- To play a sample in time with your drums, create another `live_loop` called `:sample`.

```
use_bpm 65

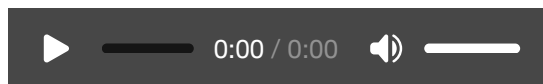
live_loop :drums do
  sample :drum_heavy_kick
  sleep 1
  sample :sn_dolf
  sleep 1
end

live_loop :sample do
end
```

- Add the sample `:loop_compus`, making it play every 8 beats.

```
live_loop :sample do
  sample :loop_compus
  sleep 8
end
```

- If you test your sample, you'll notice that it doesn't match the drums at all!



- The first thing you'll need to do is `sync` your sample with the drum beat.

```
live_loop :sample do
  sync :drums
  sample :loop_compus
  sleep 8
end
```

- This still doesn't sound right! Add code to print the duration of the sample:

```

10 live_loop :sample do
11   sync :drums
12   puts sample_duration(:loop_compus)
13   sample :loop_compus
14   sleep 8
15 end

```

- If you scroll back through the log, you'll see that although the sample is repeating every 8 beats, the sample doesn't quite last 8 beats.

```

Log
{run: 29, time: 0.0}
└─ cue :sample
   └─ sync :drums

{run: 29, time: 0.0}
└─ synced :drums (Run 29)
   └─ 7.027025699168555
      sample "/Applications/Sonic Pi.app/etc/samples",
          "loop_compus.flac"

```

(You can now remove the code to print the sample duration.)

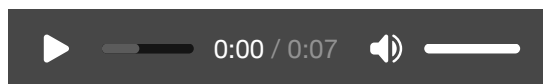
- To match your sample with the drums you'll need to stretch the sample so that it lasts exactly 8 beats as well.

```

live_loop :sample do
  sync :drums
  sample :loop_compus, beat_stretch: 8
  sleep 8
end

```

- Test your code by pressing 'Run' again – you don't need to stop and restart the music! You should now hear that your sample plays in time with your drum beat.



Step 5 Challenge: Changing the sample

Can you change the sample used?

To see what loop samples are available, you can go to jumpto.cc/sonic-pi-samples (<http://jumpto.cc/sonic-pi-samples>), or just type `sample :loop` and choose from the list that appears.

```
live_loop :sample do
  sync :drums
  sample :loop, beat_stretch: 8
  sleep 8 :loop_amen
end
      :loop_amen_full
      :loop_breakbeat
      :loop_compus
```

You might also need to change the numbers in your code for different samples. You can use the following code to find out the duration of the sample:

```
puts sample_duration(:sample_name)
```

To loop a sample without a gap, make sure that both numbers match.

```
live_loop :sample do
  sync :drums
  sample :loop_compus, beat_stretch: 8
  sleep 8
end
```


Step 6 Adding bass

Now let's add some bass notes to your music.

- Start by creating a new `live_loop` called `:bass`. This new loop should also `sync` with the drums.

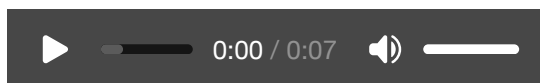
```
sync :drums
sample :loop_compus,
sleep 8
end

live_loop :bass do
  sync :drums
end
```

- Add code to play a single note every 8 beats. The note played uses the `:chipbass` synth.

```
live_loop :bass do
  sync :drums
  use_synth :chipbass
  play 36
  sleep 8
end
```

- Press 'Run' (no need to stop and restart your music). You should hear a note play every 8 beats.



- A chord is a group of notes played together.



Instead of playing the same note every 8 beats, you can `choose` a random note from a chord. In this case, the chord is C Minor.

```
live_loop :bass do
  sync :drums
  use_synth :chipbass
  play chord(:c, :minor).choose
  sleep 8
end
```

- 'Middle' C is actually `:c4`. To play lower bass notes, add a number lower than 4 after the chord name.

```
live_loop :bass do
  sync :drums
  use_synth :chipbass
  play chord(:c2, :minor).choose
  sleep 8
end
```

- Use a `sustain` to choose how many beats the note is held for.

```
live_loop :bass do
  sync :drums
  use_synth :chipbass
  play chord(:c2, :minor).choose, sustain: 7
  sleep 8
end
```

- You can also use `amp` to choose the loudness of the bass. A number lower than 1 will be quieter, and higher than 1 will be louder.

```
live_loop :bass do
  sync :drums
  use_synth :chipbass
  play chord(:c2, :minor).choose, sustain: 7, amp: 0.7
  sleep 8
end
```

- You can also add a (louder) sample to play at the start of each note.

```
live_loop :bass do
  sync :drums
  use_synth :chipbass
  sample :bd_sone, amp: 3
  play chord(:c2, :minor).choose, sustain: 7, amp: 0.5
  sleep 8
end
```

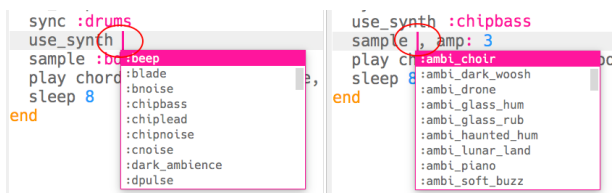
- Press 'Run' to test your code. There's no need to stop and restart your music.



Step 7 Challenge: Changing the bass

Can you change the bass notes in your music. You could change:

- The name of the chord played, e.g. f2 instead of c2
- The type of chord, e.g. :major instead of :minor
- The synth used
- The sample played



```
sync :drums
use_synth 1
sample :dc :chipbass
play chord
sleep 8
end

use_synth :chipbass
sample 1, amp: 3
play chord :ambt_choir
sleep 8
end
```

Step 8 Adding sound effects

Finally, let's add some sound effects to your music.

- Add another `live_loop` called `:effects`, which `syncs` with the drums.

```
play chord(:c2, :m3)
sleep 8
end

live_loop :effects do
  sync :drums
end
```

- Add this code to play the `:elec_blip2` sample every 2 beats.

```
live_loop :effects do
  sync :drums
  sample :elec_blip2
  sleep 2
end
```

- Click 'Run' to test your code (there's no need to stop and restart your music). You should hear a beep effect every 2 beats.



- Instead of playing the same effect each time, you could instead choose randomly from a list of 2 effects.

```
live_loop :effects do
  sync :drums
  sample choose([:elec_blip2, :elec_twip])
  sleep 2
end
```

- Click 'Run' to test your random effects (there's no need to stop and restart your music).

Step 9 Challenge: Changing the effects

Can you add even more sound effects to your music?

```
live_loop :effects do
  sync :drums
  sample choose([:elec_blip2, :elec_twip, :elec_beep, :elec_ping])
  sleep 2
end
```

Step 10 Challenge: Show off your DJ skills!

Use everything you've learnt to be a DJ for your friends! Remember that you can add to your music, as well as change notes and samples without having to stop the music.

Published by Raspberry Pi Foundation (<https://www.raspberrypi.org>) under a Creative Commons license (<https://creativecommons.org/licenses/by-sa/4.0/>).

View project & license on GitHub (<https://github.com/RaspberryPiLearning/live-dj>)