mehackit

Creative technology for youth

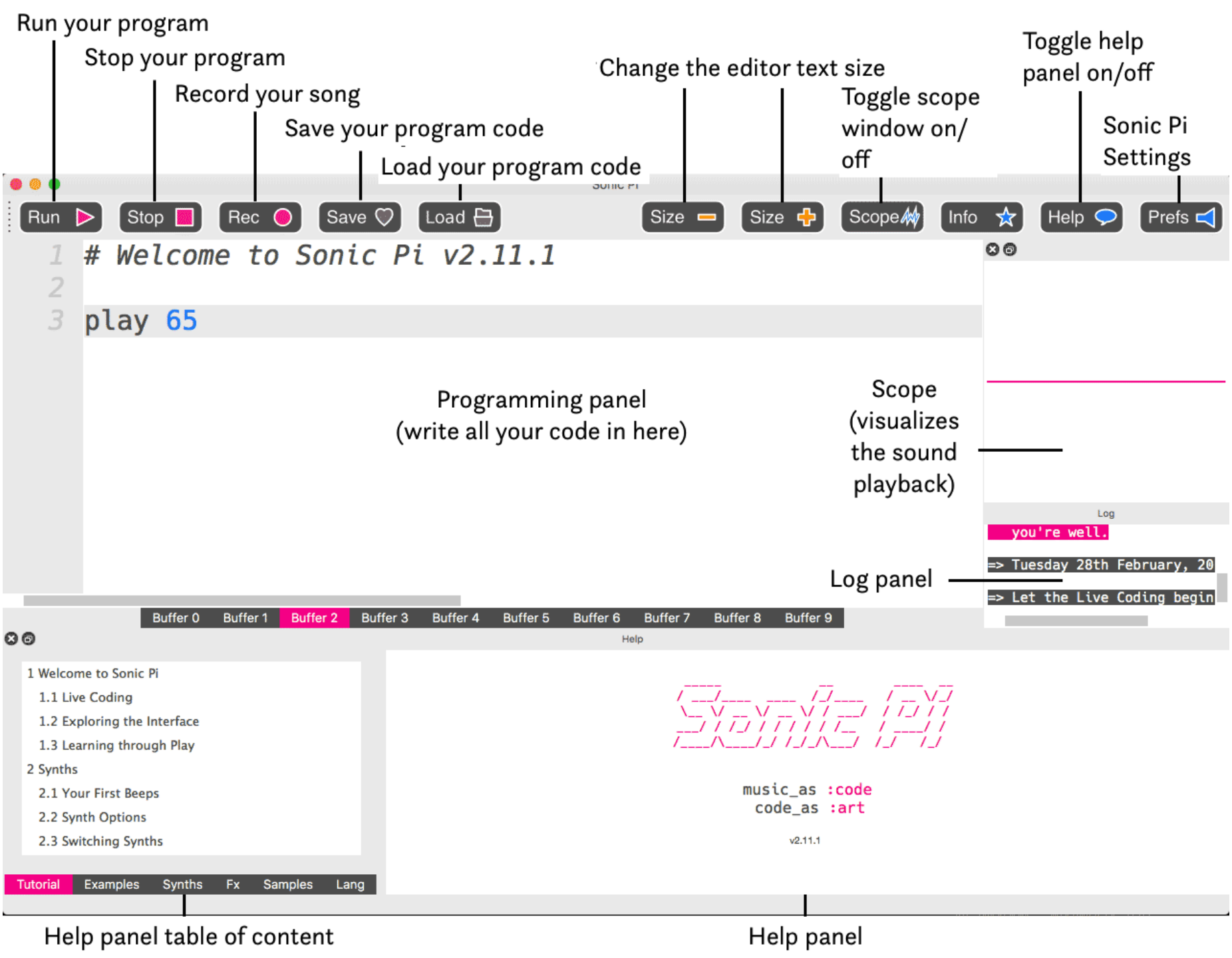
ARDUINO
Robotics & Electronics

PROCESSING
Visual Arts & Programming

SONIC PI
Music & Programming

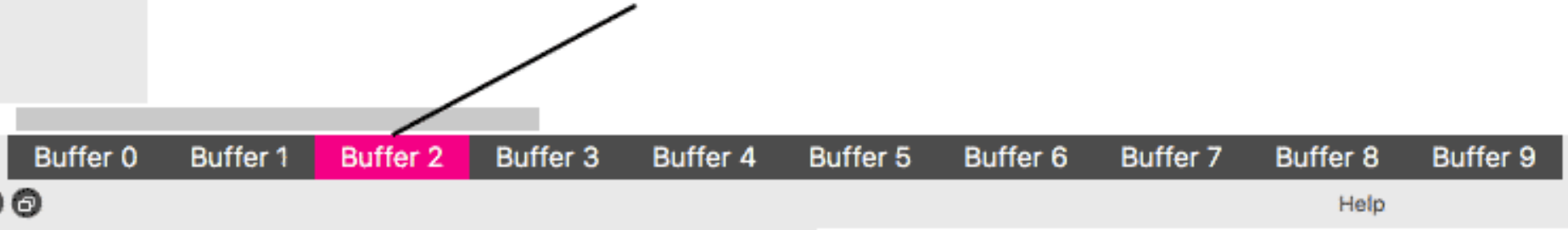
Download the application:
sonic-pi.net

Workshop materials:
sonic-pi.mehackit.org


Run your program
Stop your program
Record your song
Save your program code
Load your program code
Change the editor text size
Toggle scope window on/ off
Toggle help panel on/off
Sonic Pi Settings
Run
Stop
Rec
Save
Load
Size
Size
Scope
Info
Help
Prefs
1 # Welcome to Sonic Pi v2.11.1
2
3 play 65
Programming panel (write all your code in here)
Scope (visualizes the sound playback)
Log
you're well.
=> Tuesday 28th February, 20
=> Let the Live Coding begin
Log panel
Buffer 0
Buffer 1
Buffer 2
Buffer 3
Buffer 4
Buffer 5
Buffer 6
Buffer 7
Buffer 8
Buffer 9
Help
1 Welcome to Sonic Pi
1.1 Live Coding
1.2 Exploring the Interface
1.3 Learning through Play
2 Synths
2.1 Your First Beeps
2.2 Synth Options
2.3 Switching Synths
Tutorial
Examples
Synths
Fx
Samples
Lang
music_as :code
code_as :art
v2.11.1
Help panel table of content
Help panel

```
3 play 65
4
5
6
7
8
9
10
```

Buffers (0-9) can be used to store your Sonic Pi songs.
They can also be used to quickly test different ideas!



Your First Beep!

Write the following command:

```
play 60
```

…and hit ”RUN”

Playing a melody

```
play 60
sleep 1
play 64
sleep 1
play 67
```

Altering the rhythm

```
play 60
sleep 1.5
play 64
sleep 0.5
play 67
```

Changing the tempo

Add the following command to the beginning of your program:

```
use_bpm 120
```

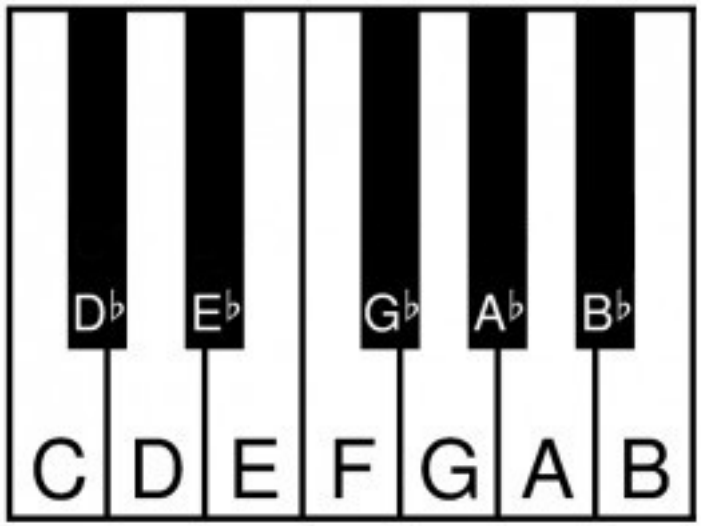
Now what happens with values like 400 or 80?

Playing a melody

You can use numbers between 0 and 127 as notes with the play command.

The numbers represent actual notes from piano. If you're familiar with the traditional musical notation, you can also use following...

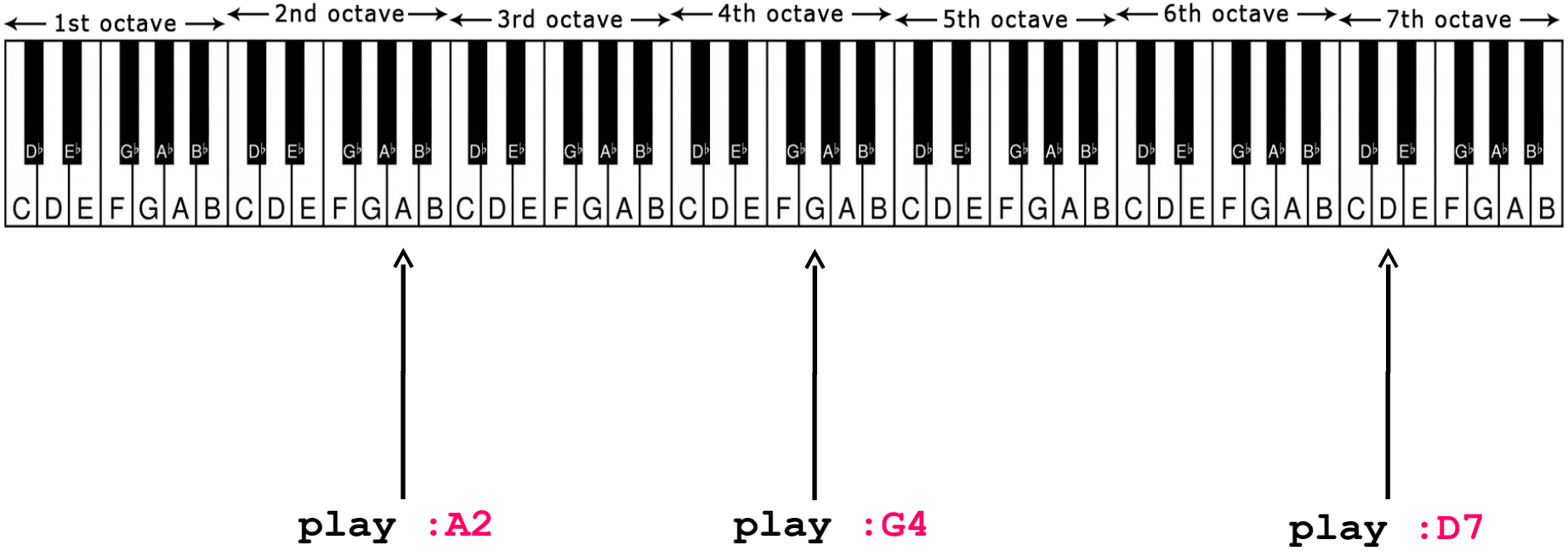
Notation in Sonic Pi



The following ”note symbols” can be used with the play command:

:C, :Db, :D, :Eb, :E, :F, :Gb, :G, :Ab, :A, :Bb, :B

Playing notes from different octaves



Just add a number after the note symbol! For example, play :C4

Using notation instead of numbers

```
play :C3
sleep 1
play :E4
sleep 1
play :G5
```

”Practice your play”

Write a program with Sonic Pi that plays a melody of at least 8 notes.

Repeating phrases

```
play :C4
sleep 1
4.times do
    play :E4
    sleep 0.5
    play :G4
    sleep 0.5
end
```

Intended areas are so called ”code blocks”

Changing your synth sound

```
use_synth :blade
play :C4
sleep 0.25
use_synth :pulse
play :C2
sleep 0.25
use_synth :chiplead
play :G3
sleep 0.25
```

Playing samples

For example:

```
sample :bd_fat
sample :ambi_piano
sample :ambi_choir
```

Controlling the volume of your synths and samples

For example:

```
play :C4, amp: 0.5
sample :bd_haus, amp: 2
```

”5 min break”

Let's spend 5 minutes exploring and getting to know the samples and synthesizers!

Looping and playing sounds concurrently

”Inf nite looping” – live_loop

```
live_loop :rummut do
    sample  :bd_haus, amp: 1.5
    sleep 1
    sample  :sn_dolf
    sleep 1
end

live_loop :hihat do
    sample  :drum_cymbal_closed
    sleep 0.25
end
```

”Inf nite looping” – live_loop

- You can have multiple live_loops running simultaneously
- They make it possible to have multiple synchonized threads of code running in Sonic Pi
- Every live_loop needs an unique **:name** and at least one sleep command

Commenting code

You can comment a line of code by adding # character to the beginning of the line. When you press ”Run”, commented lines of code won't be executed.

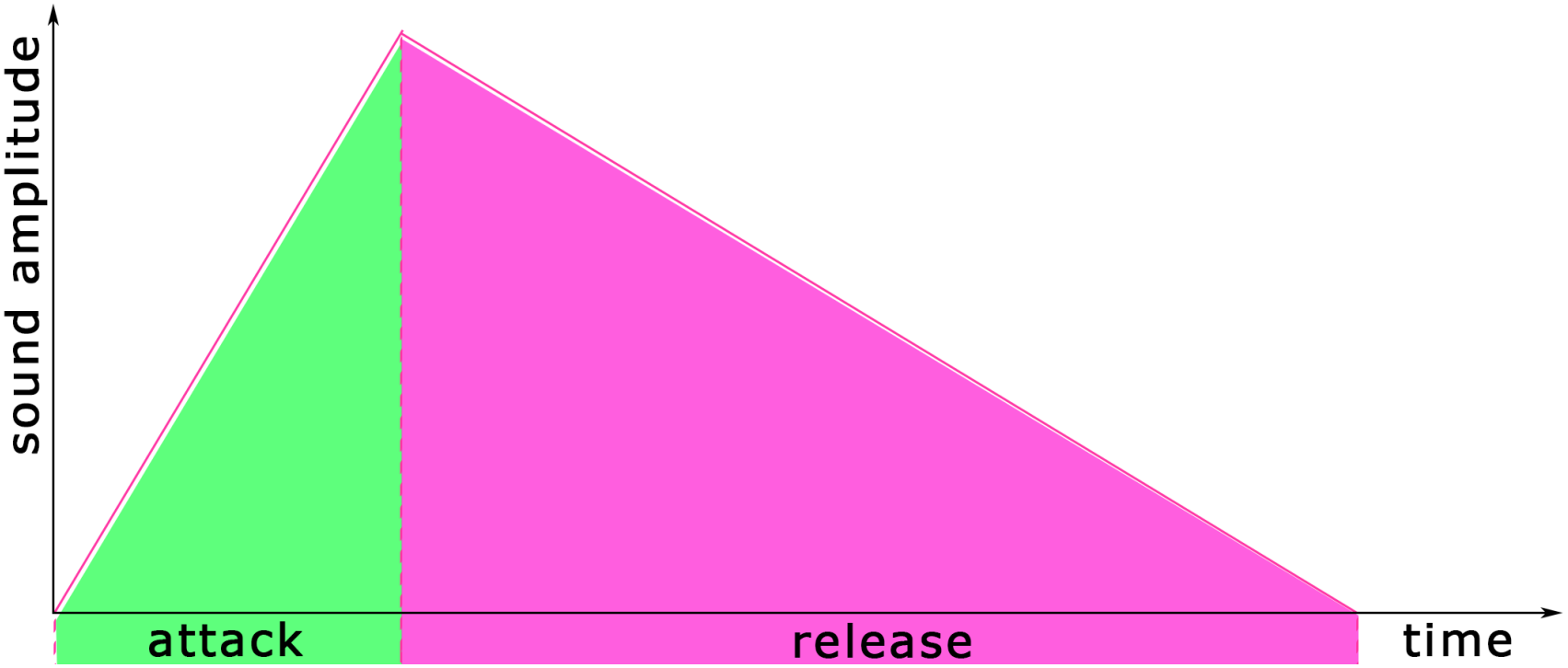
```
live_loop :rummut do
    #sample  :bd_haus, amp: 1.5
    sleep 1
    sample  :sn_dolf
    sleep 1
end
```

”Looping exercise”

Create a program with Sonic Pi that has at least two live_loops playing at the same time!

Some more advanced Sonic Pi topics

Duration of a note



For example:

```
play :C4, attack: 1, release: 2
```

Playing chords

For example:

```
play (chord :C4, :major)
```

or

```
play [:C4, :E4, :G4]
```

This kind of structure is called ”table” in coding

Randomization (1/2)

```
live_loop :randomMelodia do
    use_synth :chipbass
    play [:C3, :Eb5, :G4, :Bb4].choose
    sleep 0.25
end

live_loop :randomSleep do
    sample :elec_blip, amp: 2
    sleep [0.25, 0.5, 0.75].choose
end
```

Randomization (2/2)

```
live_loop :trance do
   use_synth :tb303
   play [:C2, :C3].choose, cutoff: rrand(50, 120), release: 0.25
   sleep 0.25
end

live_loop :hihat do
   sample :drum_cymbal_closed, amp: rrand(0,2)
   sleep 0.25
end
```

Effects

```
with_fx :reverb do
    ...
end

with_fx :echo do
    ...
end

with_fx :distortion do
    ...
end
```

play_pattern_timed

```
play :c2
sleep 0.5
play :d2
sleep 0.25
play :e2
sleep 0.75
play :d2
sleep 0.5
```

You can save many lines of code

```
play_pattern_timed [:c2, :d2, :e2, :d2], [0.5, 0.25, 0.75, 0.5]
```

Note sequencer

```
live_loop :bassline do
    use_synth :tb303
    notes = [:C2, :C2, :Eb2, :Bb2].ring.tick
    play notes, release: 0.25
    sleep 0.25
end
```

Note sequencer + random cutoff

```
live_loop :bassline do
    use_synth :tb303
    notes = [:C2, :C2, :Eb2, :Bb2].ring.tick
    play notes, release: 0.25, cutoff: rrand(60, 130)
    sleep 0.25
end
```

Tempo in electronic music

Ambient 50–100 BPM

Hip-hop 70–95 BPM

Deep house 110–130 BPM

Trance / Techno 130–145 BPM

Hard dance/hardcore 145–170 BPM

Drum and bass 160–180 BPM

Final exercise: Make a short looping song!

It can be, for example, a song made of four live_loops. One live_loop for each instrument: drums, bass, synth melody and funny samples!