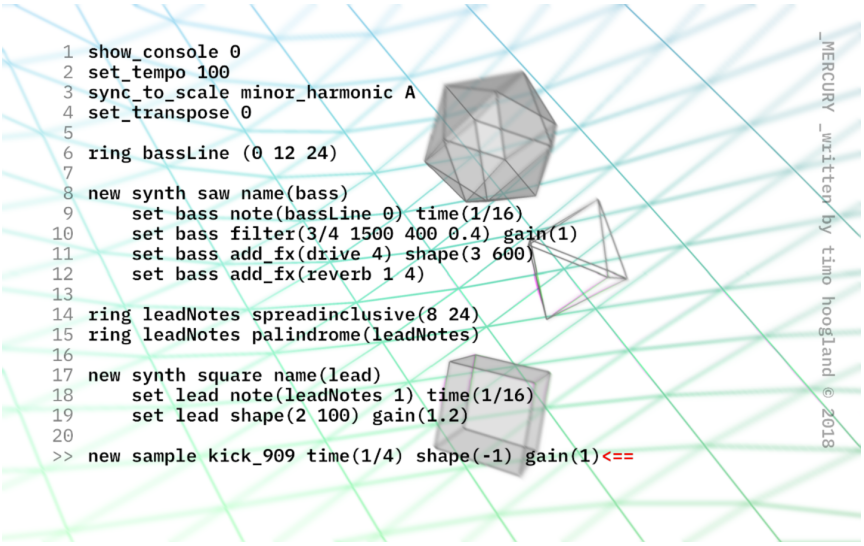
_MERCURY

: a live coding environment focussed on quick expression for composing, performing and communicating.

Timo Hoogland, info@timohoogland.com
HKU University of Arts Utrecht, Netherlands

This is Mercury. A live coding environment that is designed with the focus on quick and hands-on composing, performing and communicating. Mercury provides the performer with a highly abstracted programming language for music and sound complemented with visuals. By allowing for a higher level of abstraction, the coder does not need to write large amounts of code and the comprehension by the audience is greatly improved. Complementing the sound with visual elements also adds to this understanding and adds value to the sound.



Quick expression

Mercury has a library of functionalities like pre-designed synths, a sampler, automatic sequencing, (algorithmic) composition tools and audioprocessing effects.

- A loop is always running in the background. All instruments have an internal counter incrementing every trigger. The count is used as index for list lookup. The lists are circular arrays called rings.

```
ring aMelody (0 7 12)
new synth sine time(1/16) note(aMelody 1)

result: 48 55 60 48 55 60 48 55 etc.
```

- The time interval between triggering of an instrument is determined by the `time()` function. Fractional numbers are also allowed to create polyrhythmic patterns.

```
new sample kick_909 time(3/8)
new sample hat_909 time(2/8)

result: x - - x - - x - - x - - x - - x - -
        x - x - x - x - x - x - x - x - x -
```

- The `beat()` function takes a ring of values between 0 and 1 as probability per beat to apply fuzzy logic.

```
ring aBeat (1 0.5 0.25)
new sample hat_808 time(1/16) beat(aBeat)

result: x x - x - - x - - x x - x x x x x -
```

- Pitch is described as relative intervals in a coördinate system. The x-axis is the interval in semitones (0 to 11), and the y-axis is the octave (-3 to 7). A scale and tonic is set with the `sync_to_scale` function and maps all values to the closest correct scale note. By default `note(0 2)` corresponds to midi-value 60, middle C. This allows for using the same list for multiple instruments.

```
ring bassMelody (0 -1 0 7 12)
ring leadMelody clone(leadMelody 0 12 7)

new synth saw note(bassMelody 0) time(1/8)
new synth saw note(leadMelody 1) time(1/16)

bass: 36 -  35 -  36 -  43 -  48 -  36 -  35 -  36
lead: 48 47 48 55 60 60 59 60 67 72 55 54 55 62 67
```

Transparency towards the audience

Mercury aims to provide more transparency towards the audience by:

- Using words for parameters, settings and functions. No abbreviations.
- Automatic text resizing to fit the screen.
- Limiting the amount of text lines to 30. Forcing the performer to also delete lines of code during performance.
- Complementing audio with visual objects closely related to the sound.

Algorithmic composition toolset

Ring functions are sorted in two categories:

- Generating: `spread`, `spreadinclusive`, `fill` and `random`.
- Transformational: `shuffle`, `mirror`, `flip`, `duplicate`, `rotate`, `clone`, `palindrome`, `join`, `unique` and `merge`.

```
set_random_seed 122
ring start random(3 0 12)
      // result: 7 0 5
ring flipped flip(start)
      // result: 5 12 7
ring combined join(start flipped)
      // result: 7 0 5 5 12 7
ring theme palindrome(combined)
      // result: 7 0 5 5 12 7 7 12 5 5 0 7
ring phrase clone(theme 0 7 3 12)
      // result: 7 0 5 5 12 7 7 12 5 5 0 7 14 7
      //         12 12 19 14 14 19 12 12 6 14
```