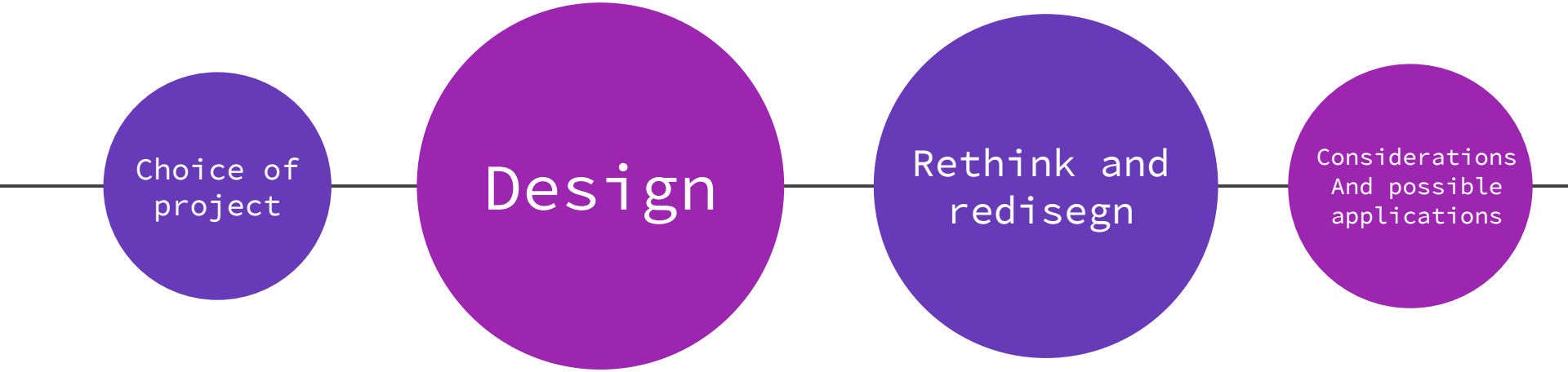
Development, Part 2c

Alessandro Favero

Task

Create an instrument created in a programming environment

Process


Choice of project
Design
Rethink and redisegn
Considerations And possible applications

Choice of Project

An instrument for everyone

Create different musical scenes that always work with each other within Sonic Pi.

Main principle: use musical skills so that non-musical players can not worry about and just enjoy the instrument, without the tough learning curve musicians go through.

Image from www.madeon.fr/adventuremachine/

Run Stop Rec Save Load Size Size Scope Info Help Prefs

```
in_thread do

  in_thread do
    live_loop :approach do

      with_fx :reverb, room: 1, mix: 0.5 do
        sample "/Users/alessandrofavero/Desktop/Sonic Pi Project/RainDropArp.wav", beat_stretch: 16, amp: 0.5, lpf: 80, lpf_release: 4, release: 16#, attack: 10 # FREE THIS LINE TO ENTER Th

      end
      sync :sixtbars
    end

    live_loop :rain do
      with_fx :reverb, room: 1 do
        sample "/Users/alessandrofavero/Desktop/Sonic Pi Project/CrispyRain.wav", beat_stretch: 17, release: 16, lpf: 100#, lpf_release: 4
      end
      sync :raincue
    end
  end

  in_thread do
    sync :raincue

    live_loop :chord2x do

      ##| 4.times do

      with_fx :echo,decay: 6 do
        with_fx :reverb, room:1, mix: 0.7 do
          play rrand(50, 95), amp: 0.1, pan: rrand(-1, 1)
        end
      end

      with_fx :reverb, room:1, amp: 0.7 do

        use_synth :fm
        play :c3, pan: 0, attack: 0.9, release: 1.5

        play :eb, pan: -0.7, attack: 0.9, release: 1.5

        in_thread do
          use_synth :sine
          play :g
        end

        use_synth :fm
        play :bb, pan: 0.8, amp: 0.5, attack: 0.9, release: 1.5

        sleep 2



        play :bb2, pan: 0, attack: 0.9, release: 1.5

        play :d, pan: -0.7, attack: 0.9, release: 1.5

        in_thread do
          use_synth :sine
          play :f
        end

        use_synth :fm
        play :bb, pan: 0.8, amp: 0.5, attack: 0.9, release: 1.5
```

Registro

```
=> Initialised Erlang OSC Scheduler

=> Initialised SuperCollider Audio Ser

=> Initialised MIDI subsystems

=> The only secret to mastering
   live coding is practice.
   Lots and lots of practice.

=> Let the Live Coding begin...

=> Has Sonic Pi made you smile?

   We need *your* help to fund furth

   We need at least 1000 supporters
   Currently we have 408 kind indivi

   Please consider becoming a Patreo
   and help us keep Sonic Pi alive:

   https://patreon.com/samaaron
```

Segnali di partenza

Buffer 0 Buffer 1 Buffer 2 Buffer 3 Buffer 4 Buffer 5 Buffer 6 Buffer 7 Buffer 8 Buffer 9

Sonic Pi v3.1 on Mac

Image from Spaceship Journey

Design

Sonic Pi’s dynamics are not suitable for the project

Sonic Pi is very limited when it comes to pre-made blocks and their management

Starting and stopping scenes cannot be done easily by means of a MIDI controller

Through actual coding, the instrument would never be easily playable by everyone

Rethink and Redesign

Using Sonic Pi's unique features

Other DAWs can do that kind of instrument better and more efficiently.

In order to make the project something that has reason to exist, it has to use the unique features of Sonic Pi:

- Changes can be made on the fly
- "Exposed wires" aesthetics
- Gives a sense of control, detail and freedom

Image from ask.com

The Code

The Coding of Spaceship Journey

The structure is built so that any player can easily transition from one scene to the next one.

If a player knows coding and/or decides to experiment using his or her intuition, parameters can be modified to “steer the ship”, and to adventure outside of the pre-designed journey

The two main categories of blocks used in Spaceship Journey are *sample* and *live_loop*, aka One Shots and Loops (infinite and finite upon pre-set conditions)

One Shots are mainly used to transition from one scene to another, whereas loops are used mainly to create the actual musical scene

The Code

Errors, mistakes and limitations

The player cannot easily jump from one scene to another unless it is the next one. In other words, the journey can only go forward. This is due to the interdependence of the buffers (which contain the scenes)

Not all the sync points are precise. This is a trade off between making it more manageable and making it more interesting. This way it is more interesting, though less precisely manageable

Demonstration

Considerations and Possible Applications

Considerations

Spaceship Journey works by itself and can be quite enjoyable, but does not represent an efficient general baseline for replication. This can be considered a partial failure for an instrument, but not necessarily for a piece of art.

Considerations and Possible Applications

Possible Applications

Spaceship Journey could become part of an art and/or science installation, like in a museum or an art exhibition like the Biennale.
Inside the replica of the cockpit of a space shuttle, the on-board computer could be an instance of Spaceship Journey to start an artistic representation of travelling through space

Development, Part 2c

Alessandro Favero