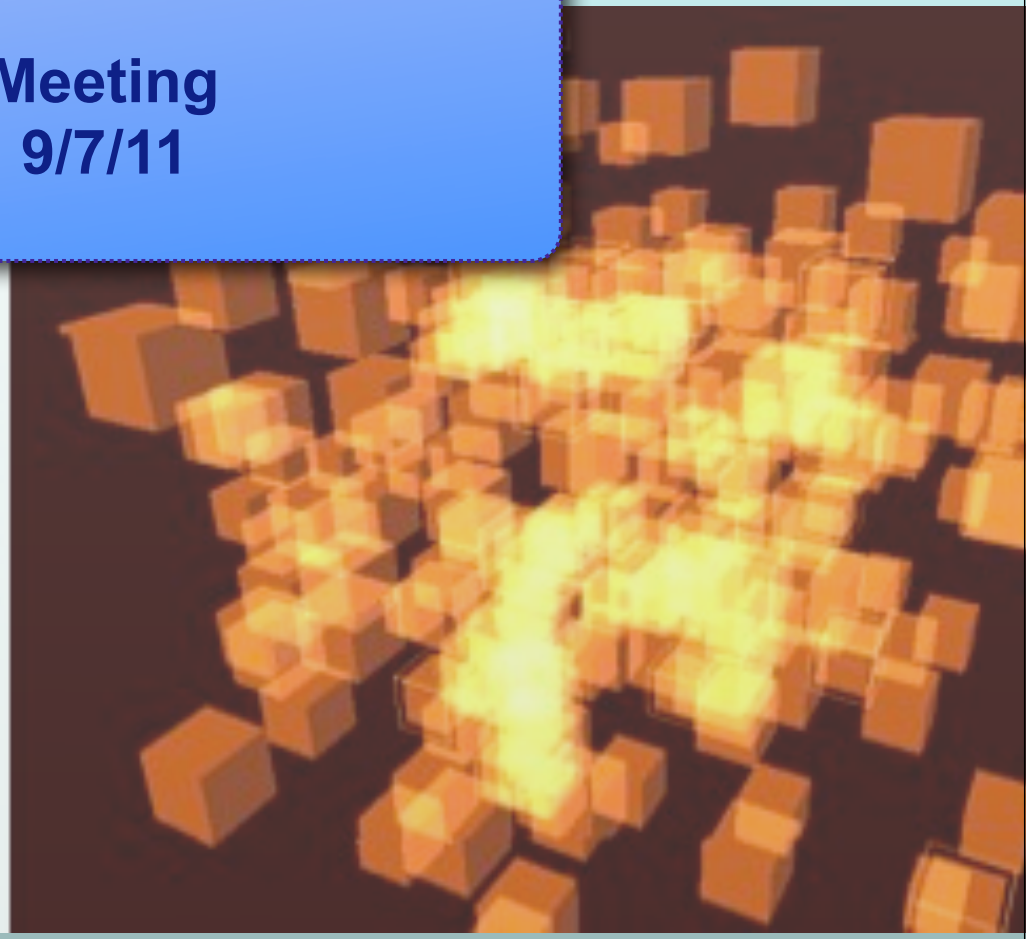
Introduction to the Impromptu livecoding environment

2011 Libre Software Meeting
Strasbourg, France, 9/7/11



Michele Pasin

Department of Digital Humanities

Kings College, London

michele.pasin@kcl.ac.uk
www.michelepasin.org/

Quick Facts #1

1. **OSx.** Impromptu is an OSX programming language and environment for composers, sound artists, VJ's and graphic artists with an interest in live or interactive programming (developed by Andrew Sorensen, since 2006-7)

2. **Scheme.** Impromptu is a Scheme language environment, a member of the Lisp family of languages.

3. **AudioUnits.** Impromptu is a programmable AudioUnit host. Musical material can be precisely scheduled for performance by any AudioUnit instrument node and parameters, program changes and presets can be programmatically changed on-the-fly as well as directly via the AU's user interface.

Quick Facts #2

4. **On-the-fly programming.** Write, modify, compile and hot swap custom signal processing code - an ability that allows you to write low level sound synthesis code during live performance. This functionality is designed to interoperate with the AudioUnit signal graph allowing you to freely mix your favorite AudioUnits with custom synthesis code.

5. **Graphics.** Graphics routines can be applied with the same temporal accuracy as audio material allowing artists to tightly integrate audio and visual components. Eg OpenGL, live video processing, vector drawing routines, image rendering, CoreImage filters, text rendering and quicktime movie support

6. **ObjC bridge.** Impromptu also includes a bidirectional ObjC-Bridge allowing Scheme to instantiate and call ObjC objects and ObjC objects to call back into the Scheme interpreter.

Useful Resources :: the website



Libraries Resources Downloads Gallery Tutorials Links/Acks

What Is Impromptu?

Impromptu is an OSX programming language and environment for composers, sound artists, VJ's and graphic artists with an interest in live or interactive programming. Impromptu is a Scheme language environment, a member of the Lisp family of languages. Impromptu is used by artist-programmers in livecoding performances around the globe.

Time plays a major role in the Impromptu environment allowing accurate real-time scheduling of events and code. Impromptu is a dynamic environment designed for the creation and manipulation of running programs in live performance.

Impromptu is a programmable AudioUnit host. A powerful environment for creating signal processing graphs of arbitrary complexity with precise programmatic control over individual AU nodes. Musical material can be precisely scheduled for performance by any AudioUnit instrument node and parameters, program changes and presets can be programmatically changed on-the-fly as well as directly via the AU's user interface.

Impromptu allows you to write, modify, compile and hot swap custom signal processing code - an ability that allows you to write low level sound synthesis code during live performance. This functionality is designed to interoperate with the AudioUnit signal graph allowing you to freely mix your favorite AudioUnits with custom synthesis code. All on-the-fly of course.

Graphics routines can be applied with the same temp
to tightly integrate audio and visual components. Ope
routines, image rendering, CoreImage filters, text rend
the graphics features available for artists to play with.

http://impromptu.moso.com.au/

Useful Resources :: other stuff

- Mailing List

http://lists.moso.com.au/pipermail/impromptu/

- Wiki

http://moso.com.au/wiki/index.php

- Google Group

http://groups.google.com/group/impromptu_lang

Extempore: opening up Impromptu

- Open Source - Open Platform: Extempore currently builds on both OSX and Linux platforms and should easily port to any UNIX environment. The project intends to also extend to the Windows platform at some point in the not to distant future.

- Very recent development..

get involved if interested!

http://groups.google.com/group/extemporelang

https://github.com/digego/extempore

Example: play a note

```
(play-note (now) synth c3 soft minute)
```

Example: a function that creates chords

```
(define  chords
   (lambda  (beat chord duration)
      (for-each (lambda (pitch)
                     (play synth pitch soft duration ))
                chord )
      (schedule (*metro* (+ beat duration)) chords
                (+  beat  duration )
                chord
                duration)))

(chords (*metro* 'get-beat 4) (list 60 64 67) 4)
```

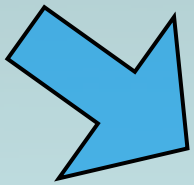
Example: chords function.. with a touch of randomness

```
(define  chords
   (lambda (beat chord duration)
      (for-each  (lambda (pitch)
                    (play synth pitch soft duration ))
                 chord )
      (schedule (*metro* (+ beat duration ))
                chords
                (+  beat  duration )
                (random (assoc chord '((Cmin7 Dmin7)
                                       (Dmin7 Cmin7))))
                duration)))

(chords (*metro* 'get-beat 4) Cmin7 4)
```

Example: generating high notes for Dmin7

```
(define  chords
   (lambda  (beat chord duration)
      (for-each (lambda (pitch)
                     (if (equal? chord Cmin7)
                         (play synth pitch soft duration )
                         (play synth (+ 24 pitch) soft duration )))
                chord )
      (schedule (*metro* (+ beat duration )) chords
                (+  beat  duration )
                (random (assoc chord '((Cmin7 Dmin7)
                                       (Dmin7 Cmin7))))
                duration)))

(chords (*metro* 'get-beat 4) Cmin7 4)
```

... thanks!



www.michelepasin.org

email me at: *michele.pasin@kcl.ac.uk*