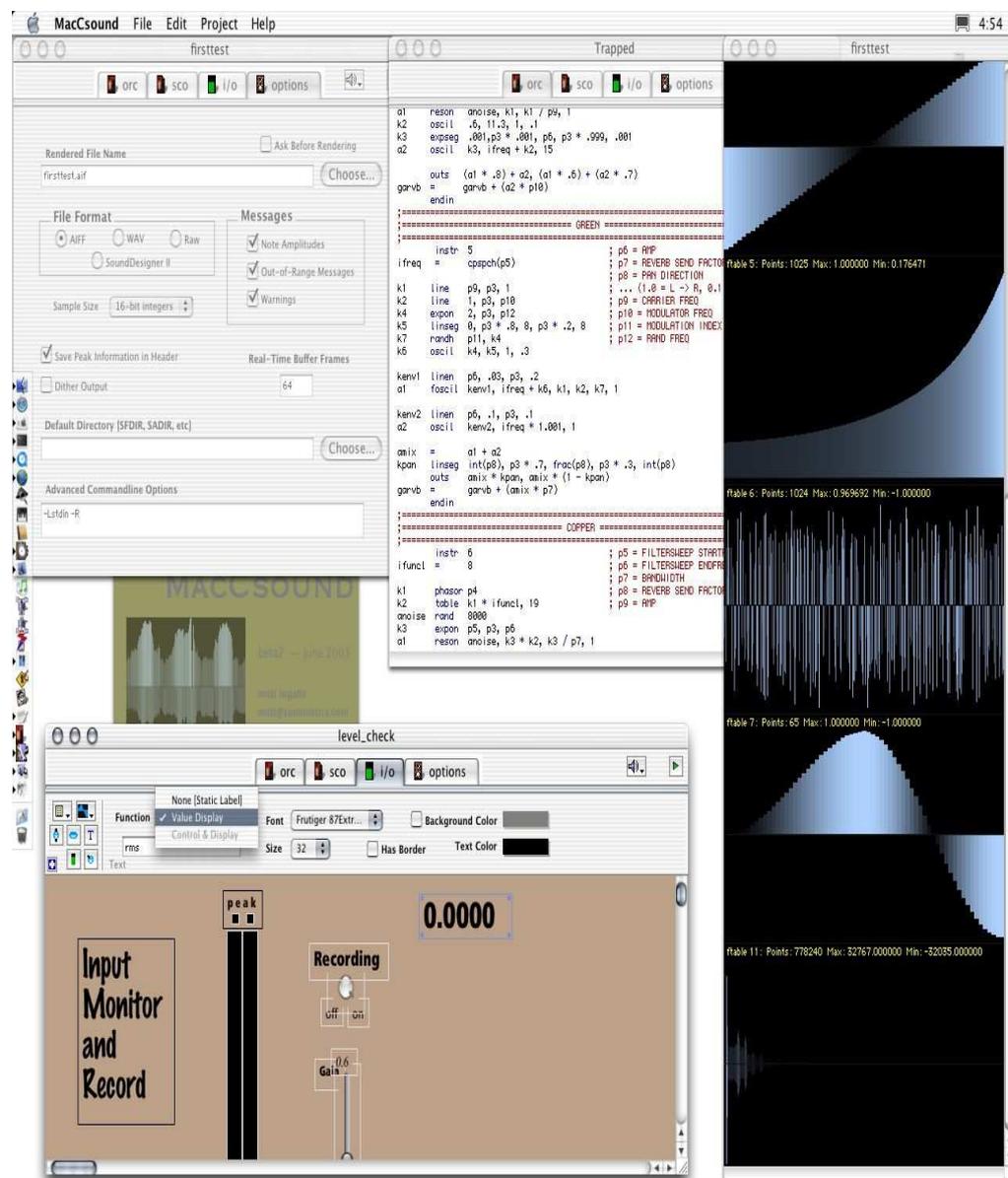


# Panoramica su Csound

Programmazione musicale, sintesi  
audio e live electronics

# Csound: un sintetizzatore virtuale

- Csound è un free software per l'elaborazione e la sintesi digitale del suono.
- Csound include al suo interno un numero altissimo di istruzioni (opcodes) per realizzare le più svariate forme di sintesi sonora



# Breve storia di Csound

**Linguaggio di programmazione** sviluppato specificatamente **per la sintesi audio** e la composizione musicale

Sviluppato inizialmente da Barry Vercoe al MIT nel 1984-1985, usando il linguaggio C (da cui il nome)

**Progettato a partire dai primi linguaggi di elaborazione audio (Music-N)** sviluppati da Max Mathews al Bell Labs (Music-1 1957)



# Breve storia di Csound

- Fino ai primi anni 90, i computer di media fascia, senza hardware dedicato, erano troppo lenti per generare audio complesso in tempo reale. Prima di poter ascoltare un'elaborazione audio (soprattutto se particolarmente complessa), essa doveva venire prima salvata su file.
  - Csound nasce quindi come **programma di 'rendering audio'**
  - Oggi Csound può essere utilizzato per elaborazioni **audio in tempo reale**

# Installazione

- Il sito ufficiale dove trovare tutte le indicazioni per installare e lavorare con Csound è: <http://www.csounds.com/> ,
- Sorgenti: <http://csound.github.io/>
- [http://www.csounds.com/community/\*\*csound-journal\*\*](http://www.csounds.com/community/csound-journal/)
- Ubuntu 14.04: Csound version 6.02.0

# CsoundQt

- CsoundQT è attualmente il frontend di riferimento per Csound
- E' particolarmente utile anche per imparare i primi rudimenti di Csound
- Esempi > Getting Started > Fondamentali > Hello World
- Canonical Csound Reference Manual:  
[/usr/share/doc/csound-doc/html](http://usr/share/doc/csound-doc/html)

# Scrivere codice Csound

- Il codice Csound è composto da due elementi fondamentali
  - **L'orchestra**, che contiene gli strumenti definiti dall'utente
  - **Lo spartito (score)**, che contiene gli eventi che gli strumenti dell'orchestra eseguiranno nel tempo

# File .csd

- Sorgenti di Csound
- File a marcatori
  - <CsoundSynthesizer>
    - <CsOptions> [command line flags]
      - -odac ;realtime audio out
      - -iadc ;realtime audio input
    - <CsInstruments>
    - <CsScore>

# Strumenti ed esecuzione

```
<CsInstruments>
```

```
sr=44100
```

```
ksmps=10 ;num campioni audio generati ad ogni k ciclo
```

```
nchnls=2
```

```
0dbfs=1
```

```
/* commento su più righe
```

```
*/
```

```
instr 123 ; Il nostro primo strumento
```

```
  aSin oscils 0dbfs/4, 440, 0 ; sine oscillator
```

```
  out aSin
```

```
endin
```

```
</CsInstruments>
```

```
<CsScore>
```

```
i 123 0 1
```

```
e
```

```
</CsScore>
```

# Csound come strumento di sintesi

- *sintesi additiva*
- *sintesi sottrattiva*
- *sintesi granulare*
- *sintesi per formanti*
- *sintesi FM*
- *sintesi a modelli fisici*

# Sintesi additiva

sr = 44100

ksmps = 32

nchnls = 2

Odbfs = 1

instr 1

kamp = .6

kcps = 440

ifn = p4

asig oscil kamp, kcps, ifn

outs asig,asig

endin

</CsInstruments>

<CsScore>

f1 0 16384 10 1

; Sine

f2 0 16384 10 1 0.5 0.3 0.25 0.2 0.167

0.14 0.125 .111 ; Sawtooth

f3 0 16384 10 1 0 0.3 0 0.2 0 0.14

0 .111 ; Square

f4 0 16384 10 1 1 1 1 0.7 0.5 0.3

0.1 ; Pulse

i 1 0 2 1

i 1 3 2 2

i 1 6 2 3

i 1 9 2 4

e

</CsScore>

</CsoundSynthesizer>

# Sintesi sottrattiva

```
instr 1
; Generate a sine waveform.
kamp init 1
kcps init 440
knh init 3
ifn = 1
asine buzz kamp, kcps, knh, ifn
```

```
kfco line 300, p3, 3000
```

```
kres init 0.8
```

```
kdist = p4
```

```
ivol = p5
```

```
aout lpf18 asine, kfco, kres, kdist
```

```
; filtro passa bassa
```

```
; frequenza di taglio – risonanza - distorsione
```

```
out aout * ivol
```

```
endin
```

```
</CsInstruments>
```

```
<CsScore>
```

```
; Table #1, a sine wave.
```

```
f 1 0 16384 10 1
```

```
; different distortion and volumes to  
compensate
```

```
i 1 0 4 0.2 0.7
```

```
i 1 4.5 4 0.9 0.6
```

```
e
```

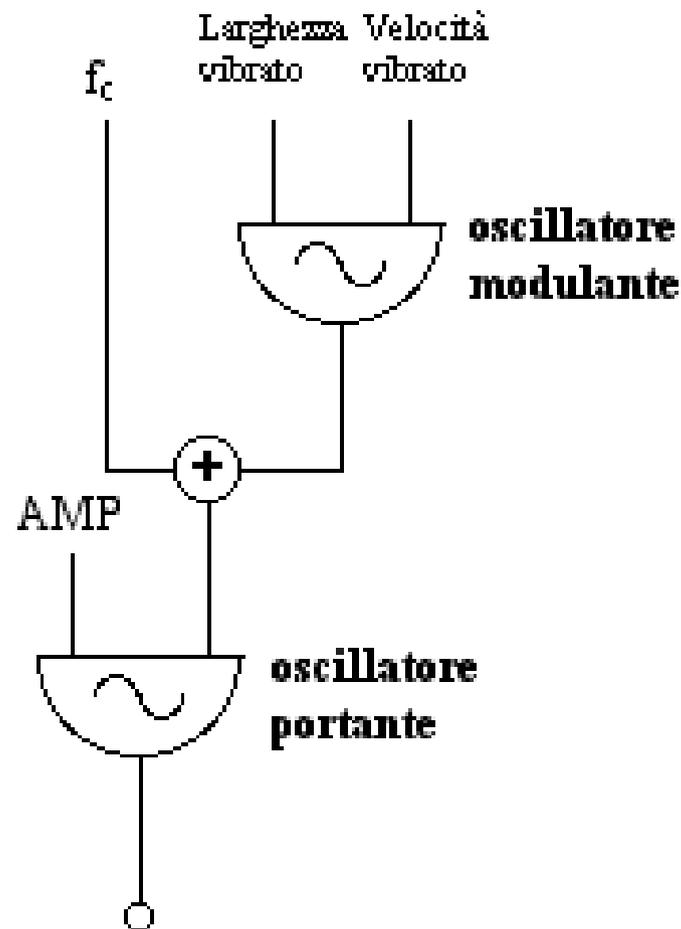
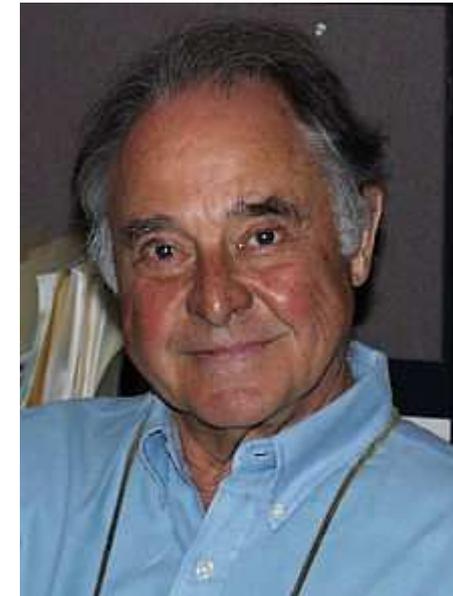
```
</CsScore>
```

```
</CsSoundSynthesizer>
```

<http://anatrofobia.bandcamp.com/track/con-fine-di-mezzo>

<http://anatrofobia.bandcamp.com/track/ff1>

# Sintesi per modulazione di frequenza (FM)



Se il modulatore oscilla ad una frequenza bassa (sotto i 20 Hz), il risultato della frequenza è un vibrato

# Sintesi granulare

Nella sintesi granulare un suono è costruito mettendo insieme centinaia o migliaia di piccoli grani. I parametri principali sono:

- la forma d'onda usata per i grani
- la lunghezza dei grani
- l'altezza dei grani
- l'inviluppo dei grani
- Il tempo che separa un grano dall'altro

<http://anatrofobia.bandcamp.com/track/caduti-in-libert>

# Live Events con CsoundQt

The screenshot displays the QuteCsoundQt application window. The main editor shows a Csound score with the following code:

```
<CsoundSynthesizer>
<CsOptions>
</CsOptions>
<CsInstruments>

sr = 44100
ksmps = 128
nchnls = 2
Odbfs = 1

; This file introduces the Live Event Panels of QuteC
; First Run this file!
; Then open the Live Event Controller window from the
; (If it's not currently at the front, you may need t
; You can play the contents of the panels by clicking
; Alternatively, you can show the panels, to control
; Right click there on the cells to see actions.
; Try the "Send events" action when standing over one

instr 1 ;simple sine wave
ifreq = 440
```

The Live Event Controller window is open, showing a table with columns: Show, Play, Loop, Sync, Name, Loop length, Loop Range, and Tempo. It contains two entries: Demo 1 and Demo 2.

Show	Play	Loop	Sync	Name	Loop length	Loop Range	Tempo
<input checked="" type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	Demo 1	8	1-8	60
<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	Demo 2	8	1-1	60

A context menu is open over a table in the background, listing actions such as "Send Events", "Send Events without offset", "Loop Selection", "Mark Loop", "Stop Events", "Subtract", "Add", "Multiply", "Divide", "Randomize", "Reverse", "Shuffle", "Rotate", "Fill Cells", "Python Scripts", and "Stop running script".

Event	...	...	...	...	...	...	...
1							
2	i	1	0	1	440	0.2	0.3
3	i	1	0	1	880	0.2	0.3
4	i	1	0	1	220	0.2	0.3
5	; You can label columns						
6	i	1	0	1	220	0.2	0.3
7	i	1	1	1	220	0.2	0.3
8	i	1	2	1	220	0.2	0.3
9	; Have a look at the text view						

At the bottom of the main window, a console window shows the following output:

```
l1event: 1 91.915 11 91.915 M: 0.19895 0.19895
r1event: T103.100 TT103.100 M: 0.44957 0.44957
could not find playing instr 1.000000
```

# Elaborazione audio in tempo reale

<https://impattosonoro.bandcamp.com/track/anatrofobia-trasfigurazione>

<http://anatrofobia.bandcamp.com/album/brevi-momenti-di-presenza>

# Widgets

Widget Channel Name

```
instr 1
```

```
  kfreq invalue "freq" ; Widget Channel Name
```

```
  kamp invalue "amp"
```

```
  asig oscil kamp, kfreq, 1
```

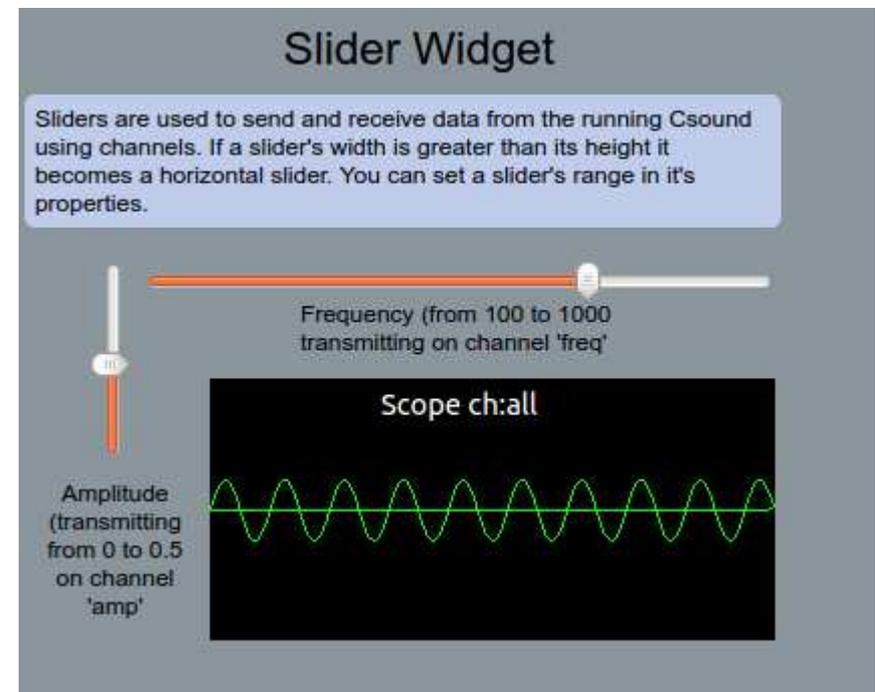
```
  outs asig, asig
```

```
endin
```

- </CsInstruments>
- <CsScore>
- f 1 0 4096 10 1 ;Sine wave
- i 1 0 1000
- </CsScore>
- </CsoundSynthesizer>

## Slider Widget

Sliders are used to send and receive data from the running Csound using channels. If a slider's width is greater than its height it becomes a horizontal slider. You can set a slider's range in its properties.



Frequency (from 100 to 1000 transmitting on channel 'freq')

Scope ch:all

Amplitude (transmitting from 0 to 0.5 on channel 'amp')

# Csound e computer music

Csound è uno strumento particolarmente versatile, in grado di colloquiare con la maggior parte dei software e dei protocolli musicali

- MIDI
- VST puglins
- Open Sound Control
- Pd (Pure Data)
- Python
- .....
- ...

# Perchè Csound?

- Flessibilità
- Potenza



# Limiti

Necessità di competenze tecniche, non soltanto musicali

ma la competenza tecnica aiuta ad essere più liberi?

