

FoxDot >> Musik kodieren

Teil 1: Einfuehrung fuer Anfaenger

Tutor: Jens Meisner
Computerkuenstler und Mediengestalter

Eine Einfuehrung >>

Was ist Live Coding?

>> **FoxDot**

Live coding with Python and SuperCollider

Eine Einfuehrung >>

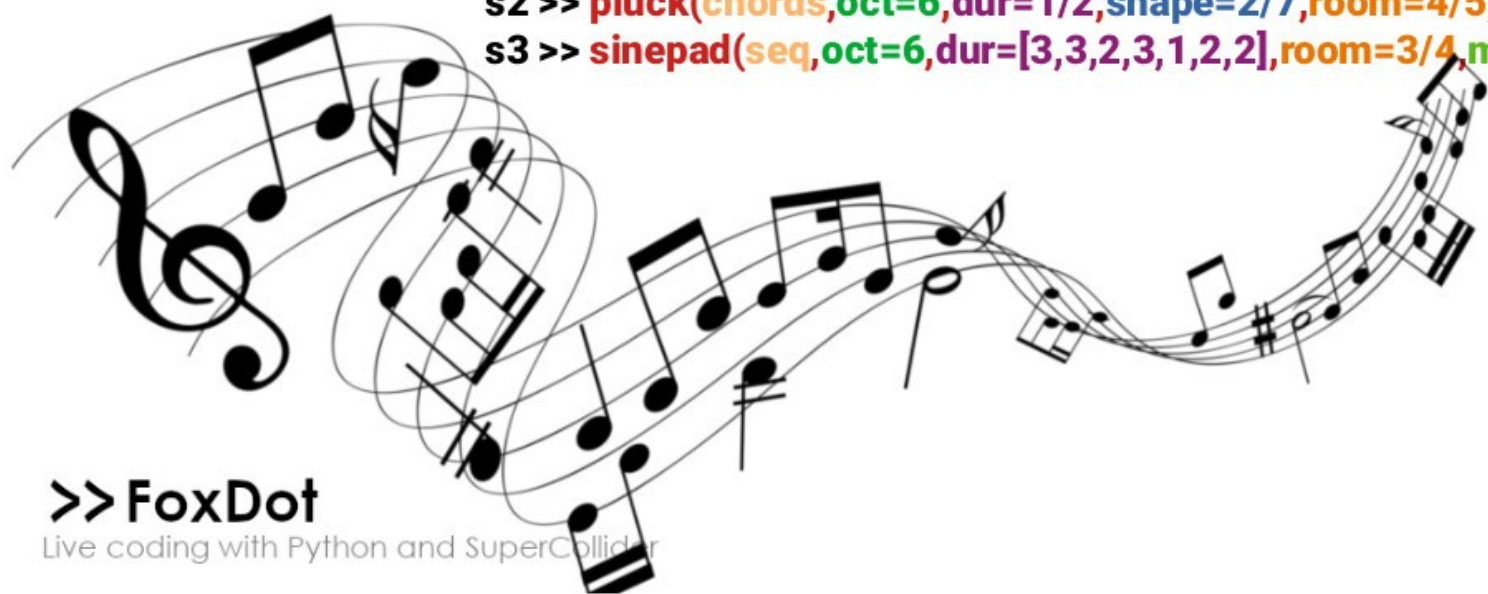
Warum code fuer Musizieren nutzen?

>> **FoxDot**

Live coding with Python and SuperCollider



```
s1 >> klank(bassline,oct=4,dur=[6,4,2,2,2],shape=2/5,amplify=3/5,amp=var([0,1],8))  
s2 >> pluck(chords,oct=6,dur=1/2,shape=2/7,room=4/5,mix=1/2,amplify=1/2,amp=1)  
s3 >> sinepad(seq,oct=6,dur=[3,3,2,3,1,2,2],room=3/4,mix=1/2,amplify=4/5,amp=1)
```



>> FoxDot

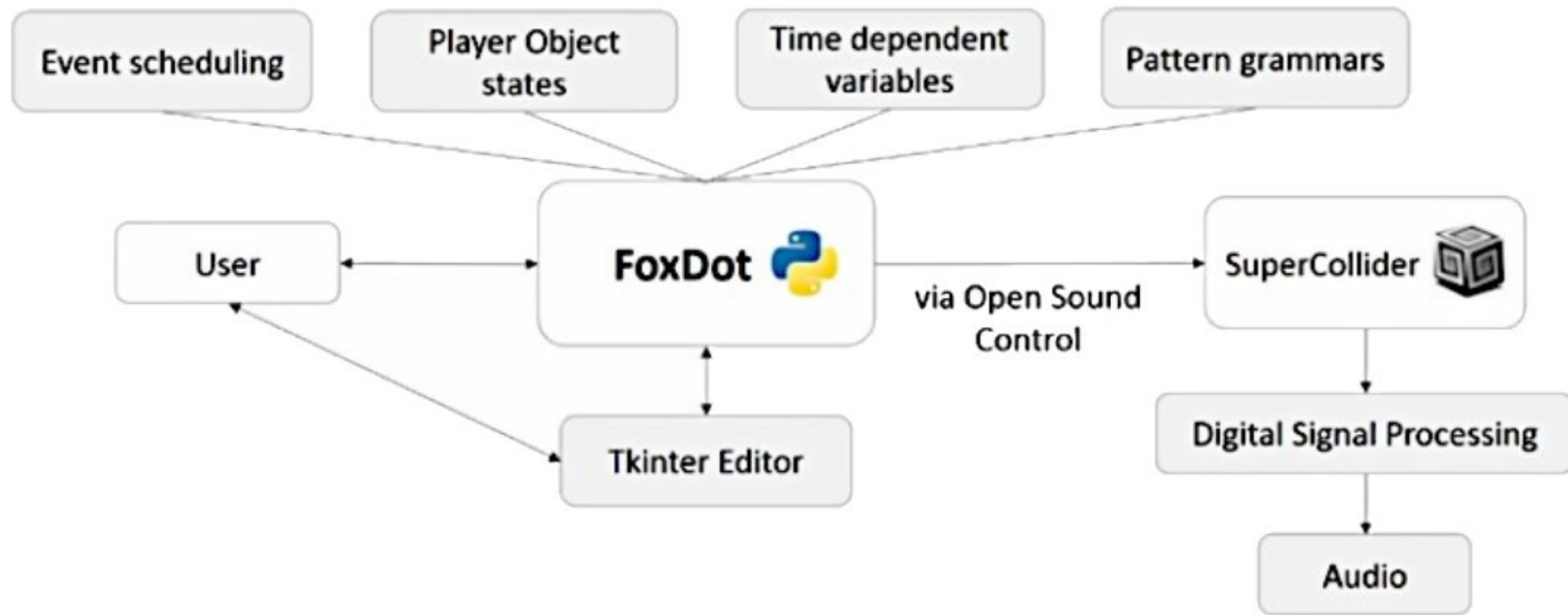
Live coding with Python and SuperCollider

Eine Einfuehrung >>

Was ist FoxDot?

>> **FoxDot**

Live coding with Python and SuperCollider



>> **FoxDot**

Live coding with Python and SuperCollider

```
T Troop - Ryan@localhost:57890
File Edit Code Constraints Help
1 # Whos here to dance?
2
3 d1 >> play("x", dur=PDur([2,2,3,5,7],8), sus=1, pan=PStep(7, P+(-1,1))) + 2
4
5 b1 >> bass(dur=16, sus=4, room=1, mix=1, shape=0.5, lpf=linvar([1000,8000],12)) + (0, PRand
  ([6,7,9]))
6
7 l1 >> play("shroom", rate=PRand([5,10]), shape=.1).every(6, "stutter", 4, pan=[-1,1])
8
9 l2 >> sitar(PWalk(), oct=[2,3], dur=[4,2,2], sus=[1,2,3,4,5], slide=0.2) + [1,3,5]
10 Ryan
11 z1 >> soprano(12.degree + (2,[4,6,7]), dur=[8,4,2,1], oct=4, sus=2, blur=4, amp=)
12 loz
13 Scale.default=("mixolydian")
14
loz : z1 >> soprano(12.degree + (2,[4,6,7]), dur=[8,4,2,1], oct=7, sus=2
, blur=4)
loz : z1 >> soprano(12.degree + (2,[4,6,7]), dur=[8,4,2,1], oct=5, sus=2
, blur=4)
Ryan : d1 >> play("x", dur=PDur([2,2,3,5,7],8), sus=1, pan=PStep(7, P+(-
1,1))) + 2
loz : z1 >> soprano(12.degree + (2,[4,6,7]), dur=[8,4,2,1], oct=4, sus=2
, blur=4)
```

>> FoxDot

Live coding with Python and SuperCollider

Erste Schritte mit FoxDot >>

Wenn man mehr ueber eine Funktion oder Klasse wissen moechte,
schreibe `help` + das Python Objekt in Klammern

`help(object)`

Zum Beispiel, wenn du mehr ueber das “pluck” Instrument wissen moechtest:

`help(pluck)`

Erste Schritte mit FoxDot >>

Spieler Objekte (Player objects)

“pluck” ist ein Python Objekt, welches eine Synthesizer Definition in SuperCollider names SynthDef representiert.

Um alle SynthDefs aufzulisten, benutze:

```
print(SynthDefs)
```

>> Bevor es beginnt, einige Basis Informationen >>

- Um eine einzelne Zeile auszuführen, presse ALT + ENTER
- Um einen Linienblock zu starten, presse CTRL + ENTER
- Um alle Soundevente zu stoppen, presse CTRL + .
- Wenn man eine einzelne Zeile stoppen möchte, addiere .stop() am Ende der Zeile, und presse ALT + ENTER:

Um ein funktionierendes Instrument zu erzeugen, muss man neben dem Namen des Instruments auch eine Bezeichnung geben. Diese kann lediglich aus einem Buchstaben und einer Ziffer bestehen:

```
s1 >> pluck()
```

Erste Schritte mit FoxDot >>

Muster (Patterns)

>> **FoxDot**

Live coding with Python and SuperCollider

Erste Schritte mit FoxDot >>

Zeitvariablen (TimeVars)

Erste Schritte mit FoxDot >>

“play” and “loop” Players

- Neben SynthDefs in SuperCollider gibt es 2 weitere Player, welche in FoxDot genutzt werden koennen
- “play” and “loop” sind eigentlich Audiosample Spieler
- “play” benutzt Buchstaben and Zeichen in Klammern, um die Samples zu rufen, welche durch die Standardinstallation von FoxDot verfuegbar sind
- z.B.:

b1 >> play("x-o-")

Benutze { } [] () <> wie: **b1 >> play("x-o[-{ox}]")**

{ } - Random, [] - All-in-one step, () - In turn, <> - Simultaneously

- Es gibt viele Variablen, die man zu den Playern addieren kann. Bei “play” werden diese nach den Anführungszeichen, getrennt mit einem Komma hinzugefügt. Wie folgendes Beispiel zeigt:

```
b1 >> play("<x-o-><..+.[+]>", pan=(-1, 1))
```

- Eines der Variablen ist “sample”.
- Jedes Zeichen und Charakter ist mit einem alphabetisch geordneten Verzeichnis verknüpft. Um eine bestimmte Datei (also Sample) in dem Verzeichnis aufzurufen, gebe “sample” mit der numerischen Position an, wie das folgende Beispiel zeigt:

```
b1 >> play("x-o-", sample=1)
```

- Die folgende Liste zeigt eine lose Kategorisierung der momentan vorhandenen Audiosamples

		Symbol/Sample	0	1	2	3	4	5	6	7	M							Symbol/Sample	0	1	2	3	4	5	6	7
Bass	Orange	a									n							1								
Snare	Yellow	A	Orange	Orange	Orange						N	Blue	Blue	Blue	Blue	Blue		2	Purple	Purple						
Hi Hat	Light Green	b	Green								o	Yellow	Yellow	Yellow	Yellow			3	Purple	Purple						
Open Hat	Light Blue	B	Grey	Grey	Green	Green					O	Yellow	Yellow	Yellow	Yellow	Yellow		4	Teal	Teal	Orange					
Crash Hat	Light Blue	c	Orange	Orange	Orange	Orange	Orange	Orange			p	Orange	Orange	Orange	Orange	Orange		&	Teal	Teal	Orange					
Cymbal	Purple	C	Purple				Purple				P	Orange	Orange	Orange	Orange	Orange		*	Light Green	Light Green	Light Green	Light Green	Light Green			
Low Tom	Brown	d	Yellow	Yellow	Yellow	Yellow					q	Grey	Grey	Grey	Grey		@	Grey	Grey	Grey	Grey	Grey	Grey			
Mid Tom	Light Green	D	Yellow	Yellow	Yellow	Yellow					Q	Grey	Grey	Grey	Grey		\	Grey	Grey	Grey	Grey	Grey	Grey			
Hi Tom	Light Green	e	Orange	Orange	Orange	Orange					r	Orange	Orange	Orange	Orange	Orange		bar	Orange	Orange	Orange	Orange	Orange	Orange		
Triangle	Teal	E	Orange	Orange	Orange	Orange					R	Orange	Orange	Orange	Orange	Orange		^	Orange	Orange	Orange	Orange	Orange	Orange		
Clap	Light Green	f	Orange	Orange	Orange	Orange	Orange	Orange			s	Blue	Blue	Blue	Blue		:	Light Green	Light Green	Light Green	Light Green	Light Green	Light Green			
Snap	Purple	F	Grey	Grey	Grey	Grey	Grey	Grey			S	Light Green	Light Green	Light Green	Light Green		\$	Brown	Light Green	Yellow	Blue	Orange	Purple	Yellow	Blue	
Bell/Metall	Light Green	g	Orange	Orange	Orange	Orange					t	Orange	Orange	Orange	Orange	Orange		=	Light Blue	Light Blue	Light Blue	Light Blue	Light Blue	Light Blue		
Noise	Grey	G	Orange	Purple							T	Orange	Orange	Orange	Orange	Orange		!	Purple	Purple	Purple	Purple	Purple	Purple		
Scratch	Yellow	h	Purple	Purple	Light Green	Light Green	Light Green				v	Orange	Orange	Orange	Orange	Orange		/	Purple	Purple	Purple	Purple	Purple	Purple		
8 Bit	Light Green	H	Light Green	Light Green	Light Green	Light Green					V	Orange	Orange	Orange	Orange	Orange		#	Purple	Purple	Purple	Purple	Purple	Purple		
Voice	Purple	T	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow			w	Orange	Orange	Orange	Orange	Orange		-	Orange	Orange	Orange	Orange	Orange	Orange		
Beep	Green	I	Light Green	Light Green	Light Green	Light Green	Light Green	Light Green			W	Orange	Orange	Orange	Orange	Orange		<	Orange	Orange	Orange	Orange	Orange	Orange		
Various	Grey	j	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow			x	Orange	Orange	Orange	Orange	Orange		%	Grey	Grey	Grey	Grey	Grey	Grey		
Percussion	Orange	J	Grey	Grey	Grey	Grey	Grey	Grey			X	Orange	Orange	Orange	Orange	Orange		+	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow		
Hom	Light Blue	k	Orange	Orange	Orange	Orange	Orange				y	Orange	Orange	Orange	Orange	Orange		?	Purple	Purple	Purple	Purple	Purple	Purple		
Shaker	Blue	K	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow			Y	Grey	Grey	Grey	Grey	Grey		:	Grey	Grey	Grey	Grey	Grey	Grey		
Tambrine	Light Green	l	Yellow	Light Green	Light Green	Light Green	Light Green	Light Green	Light Green	Light Green	z	Yellow	Yellow	Yellow	Yellow		~	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow		
Clave (Woodstick)	Orange	L	Orange	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow	Z	Grey	Grey	Grey	Grey	Grey			Grey	Grey	Grey	Grey	Grey	Grey		

- Der “loop” Player/Spieler kann benutzt werden, um eigene Samples, wie z.B. Gesang, die Aufnahme eines analogen Instruments, oder eigenen Schlagzeugsamples zu integrieren
- Die Variable dur wird dabei genutzt, um die Laenge des Loops festzulegen

l1 >> loop("path/to/my/file.wav", dur=32)

- Eigene permanente Samples kann man in das Verzeichnis FoxDot/snd/_loop_ ablegen, das durch “Help & Settings/Open Samples Folder” im Menue geoeffnet werden kann,
- Dadurch braucht man nicht den Pfad und Dateiendung angeben, wenn man die Datei aufruft:

l1 >> loop("my_file", dur=4)

Zu Beginn ein wenig Musiktheorie >>

Liedstruktur

>> **FoxDot**

Live coding with Python and SuperCollider

- Eine allgegenwertige Liedstruktur ist folgende:

Intro (4 Bars) - **Verse** (8 -16 Bars) + **Prechorus** (Optional) - **Chorus**(8 - 16 Bars) -
2nd Verse (8 – 16 Bars) + **Prechorus** (Optional) - **Chorus** (8-16 Bars)

- Die naechste Liedstruktur ist typisch fuer elektronische Musik:

Intro	Break	Buildup	Drop	Break	Buildup	Drop	Outro
16 Bars	16 Bars	4/8/16 Bars	16 Bars	16 Bars	4/8/16 Bars	16 Bars	16 Bars

Zu Beginn ein wenig Musiktheorie >>

Akkorde und Noten

>> **FoxDot**

Live coding with Python and SuperCollider

Akkorde

Melodien – Von Akkord zur Melodie

Melodien – Von der Melodie zum Akkord

Melodien – Addiere eine Gegenmelodie (Arpeggio)

Melodien – Von Akkordprogression zu Basslinien

Melodien – Von der Basslinie zur Akkordprogression

Zu Beginn ein wenig Musiktheorie >>

Skalen und Modes

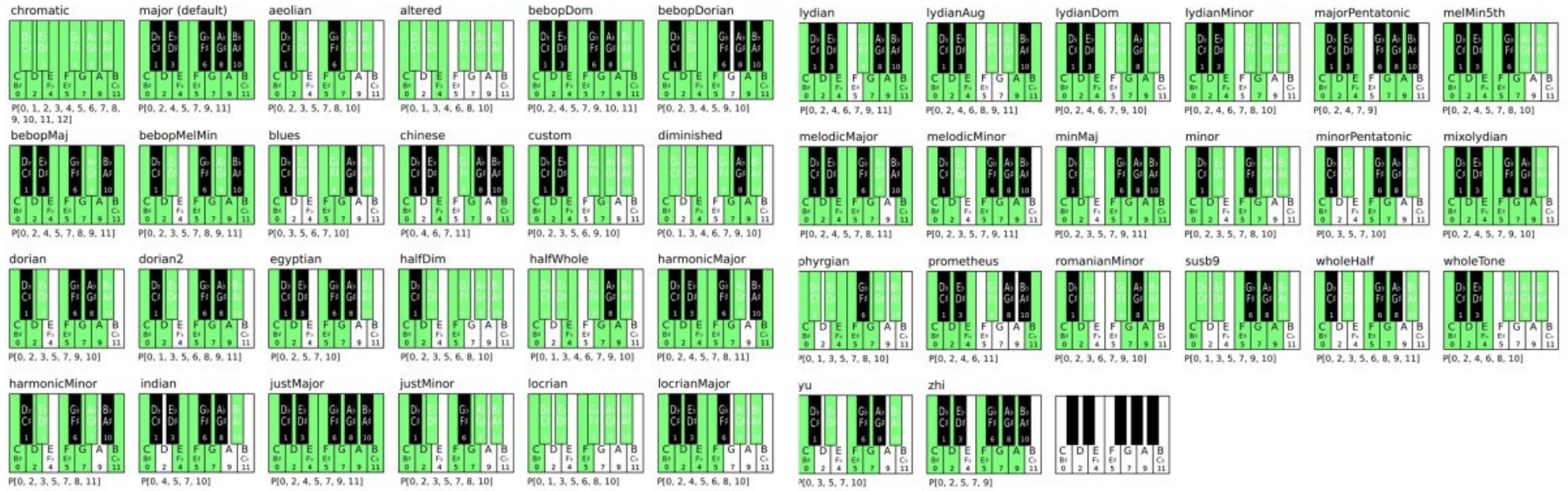
>> **FoxDot**

Live coding with Python and SuperCollider

Benutze Skalen

Benutze Modes

- Diese Schematik zeigt die momentan verfügbaren Skalen in FoxDot



Zu Beginn ein wenig Musiktheorie >>

BeatBox

>> **FoxDot**

Live coding with Python and SuperCollider

Live Jam >> Gemeinsam musizieren



>> FoxDot

Live coding with Python and SuperCollider

Live Jam >>

Verbinde und starte Troop

>> **FoxDot**

Live coding with Python and SuperCollider

Verbindung zum Wifi Network

SSID: **Wifi4DS**
Passwort: **25M4Ei7[89**

Oeffne den Terminal, fuehre “python /Path/To/Script/run-client.py” aus, fuehle die notwendigen Daten ein, und presse “Ok”

```
File Edit View Terminal Tabs Help
bbscar@ishapenoise:~$ python /home/bbscar/Desktop/FOXDOT/Troop-0T/run-client.py
bash: /home/bbscar/Desktop/FOXDOT/Troop-0T/run-client.py: Permission denied
bbscar@ishapenoise:~$ python /home/bbscar/Desktop/FOXDOT/Troop-0T/run-client.py
bbscar@ishapenoise:~$ python /home/bbscar/Desktop/FOXDOT/Troop-0T/run-client.py
bbscar@ishapenoise:~$ python /home/bbscar/Desktop/FOXDOT/Troop-0T/run-client.py
```

Troop v0.9.5	
Host:	10.3.141.170
Port:	57890
Name:	bbscar
Password:	
Language:	FoxDot
Ok	

```
Troop - yenz@10.3.141.170:57890
File Edit Code Help
1 This user is the host. It is the fastest and loudest of em. This computer
  needs to execute run-server.py first. The terminal will show the IP it is
  running on. With the FoxDotSpot Dongle plugged in, it should be some ip
  with 10.3.141.____ (in this case 170).
2
3
4 After this execute run-client.py, and enter the same IP, username and
  password into the popup window, and start it...
5
6 bbscar
7
8 This user is a client. It only need to execute run-client.py, while using
  the same IP Address as the host above. Everyone will see the same in
  their editor...Have fun!
9
10 yenz
11
12
13
Peer 'bbscar' has joined the session
Warning: Could not fetch info from SCLang server. Using defaults...
```

>> FoxDot

Live coding with Python and SuperCollider