FoxDot >> Musik kodieren

Teil 1: Einfuehrung fuer Anfaenger

Tutor: Jens Meisner
Computerkuenstler und Mediengestalter

Eine Einfuehrung >>

Was ist Live Coding?

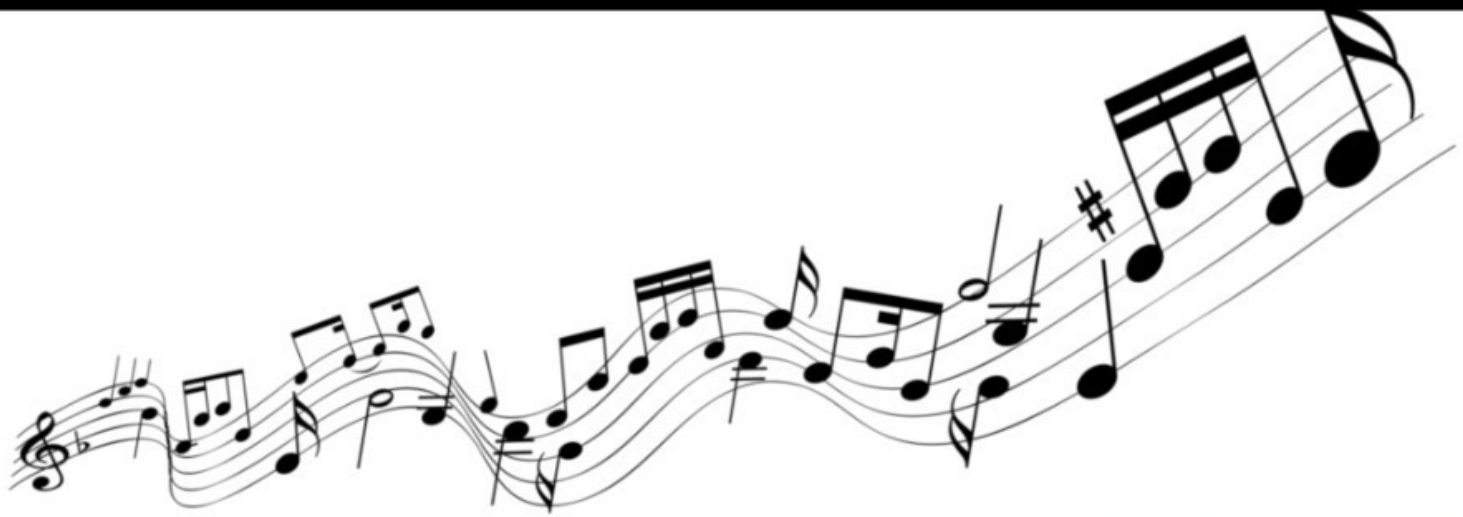
“Live Coding ist eine neue Richtung in der elektronischen Musik und Video, mit interessanten Entwicklungen...

Live Coders zeigen und verknuepfen die Innereien von Software, waehrend sie improvisierte Musik erzeugen.”
- toplap.org -

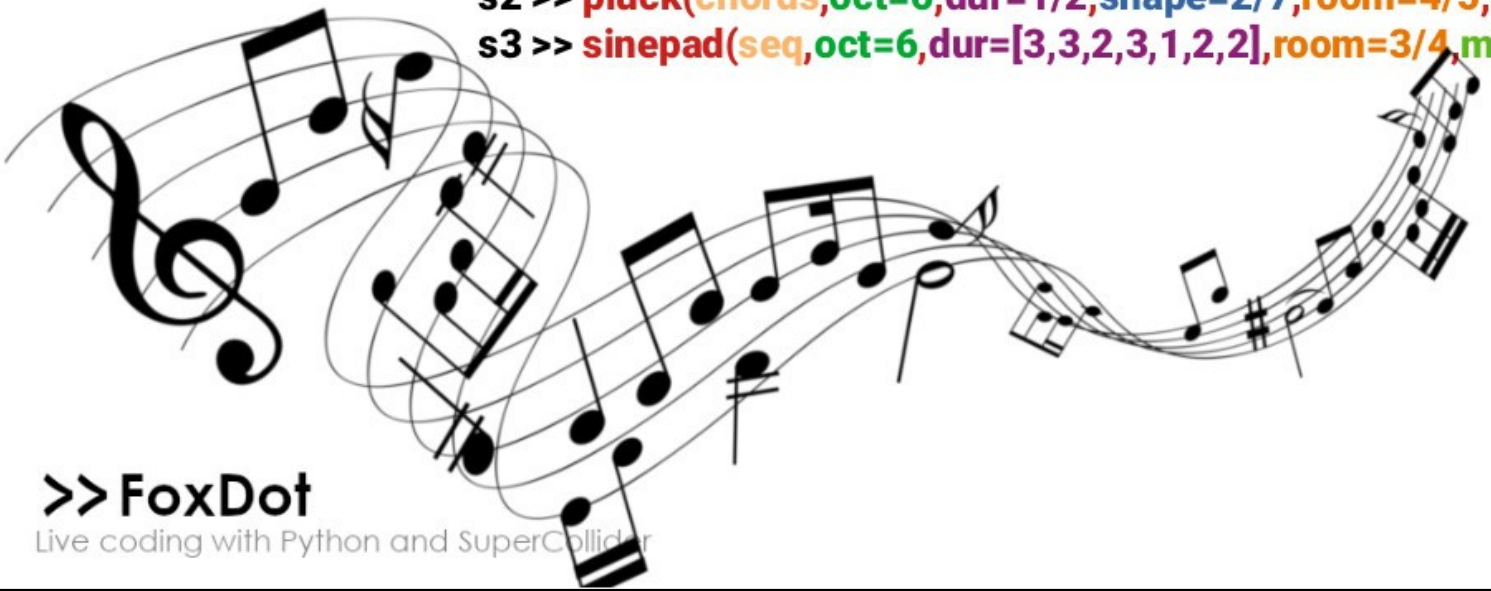
>>FoxDot
Live coding with Python and SuperCollider

Eine Einfuehrung >>

Warum code fuer Musizieren nutzen?



```
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)
```

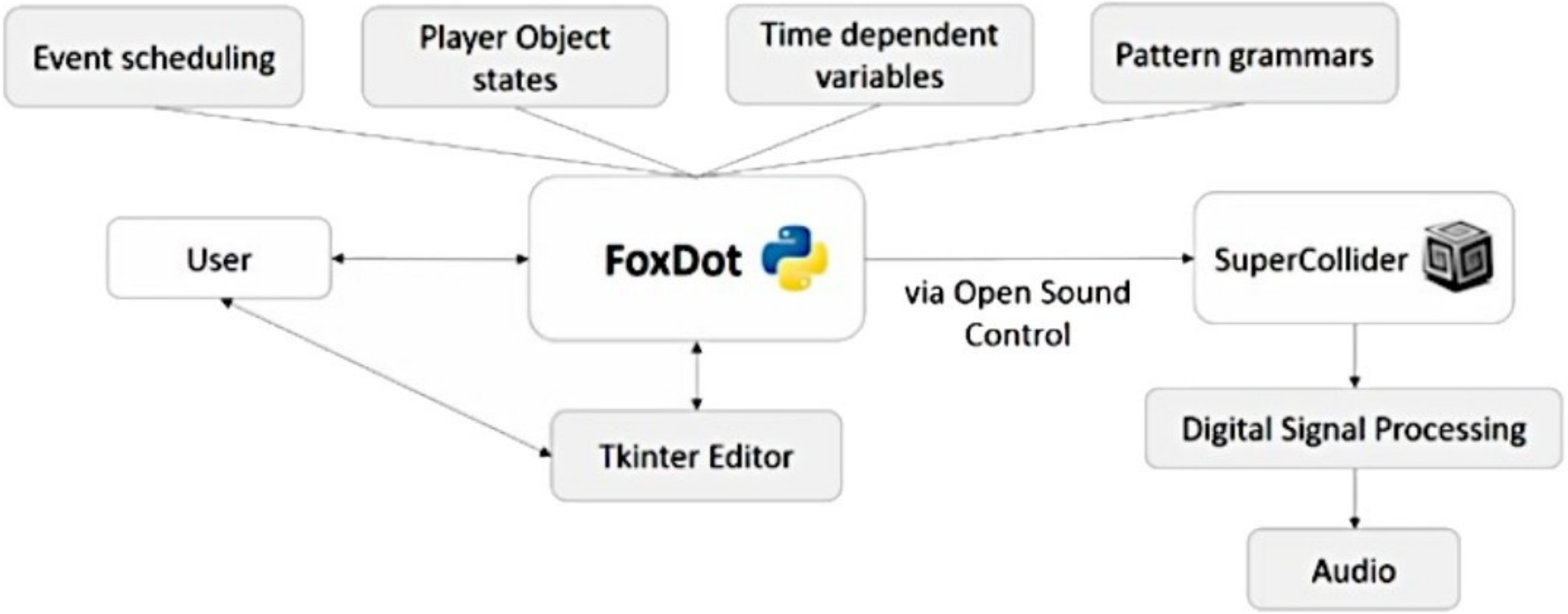


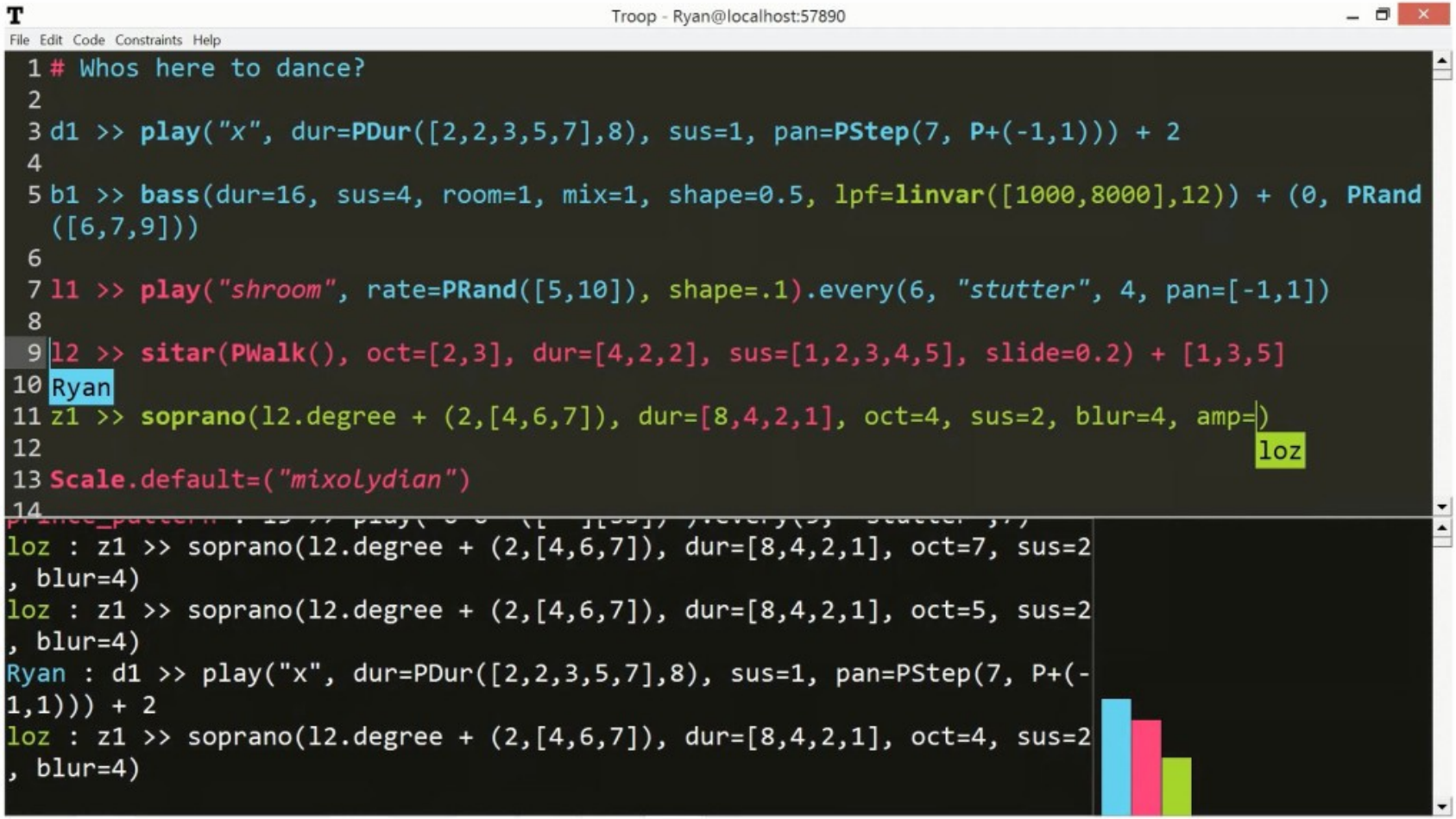
>> FoxDot

Live coding with Python and SuperCollider

Eine Einfuehrung >>

Was ist FoxDot?


Event scheduling
Player Object states
Time dependent variables
Pattern grammars
User
FoxDot
via Open Sound Control
SuperCollider
Tkinter Editor
Digital Signal Processing
Audio



>>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 auszufuehren, presse ALT + ENTER
- Um einen Linienblock zu starten, presse CTRL + ENTER
- Um alle Soundevente zu stoppen, presse CTRL + .
- Wenn man eine einzelne Zeile stoppen moechte, 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)

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

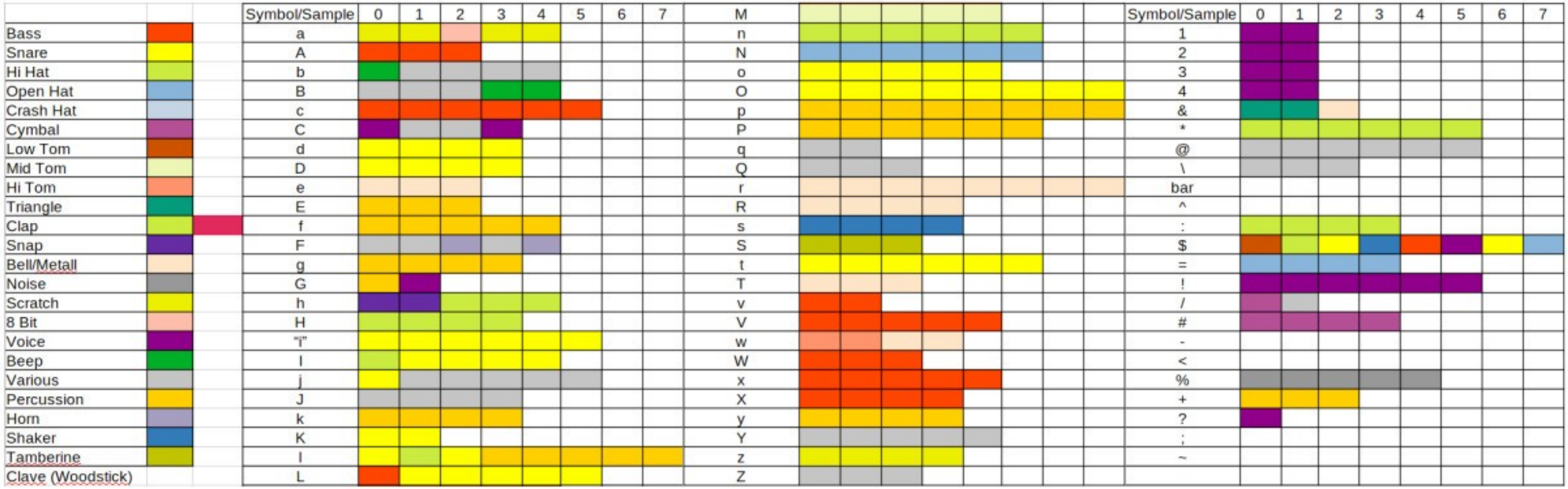
- Es gibt viele Variablen, die man zu den Playern addieren kann. Bei “play” werden diese nach den Anfuehrungszeichen, getrennt mit einem Komma hinzugefuegt. Wie folgendes Beispiel zeigt:

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

- Eines der Variablen ist “sample”.
- Jedes Zeichen und Charakter ist mit einem alphabetisch geordneten Verzeichnis verknuepft. 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



>> FoxDot

Live coding with Python and SuperCollider

- 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

- 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

Akkorde

Melodien – Von Akkord zur Melodie

Melodien – Von der Melodie zum Akkord

Melodien – Addiere eine Gegenmelodie (Arpeggio)

Melodien – Von Akkordprogression zu Basslinien

>> FoxDot

Live coding with Python and SuperCollider

Melodien – Von der Basslinie zur Akkordprogression

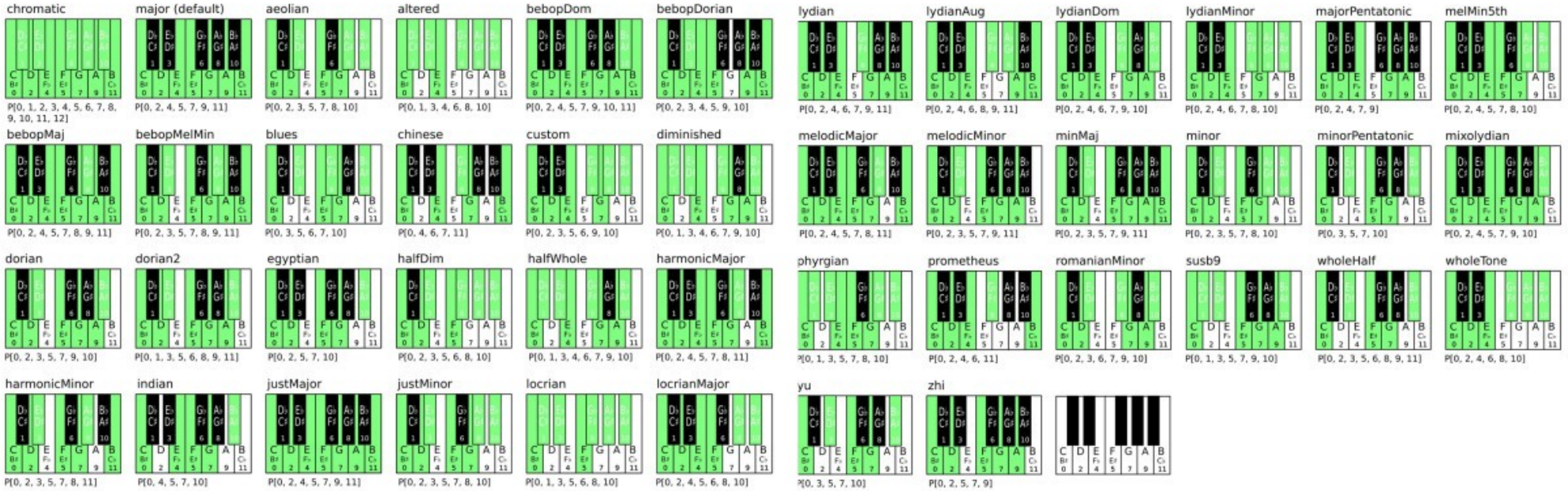
Zu Beginn ein wenig Musiktheorie >>

Skalen und Modes

Benutze Skalen

Benutze Modes

- Diese Schematik zeigt die momentan verfuegbaren Skalen in FoxDot



>>FoxDot

Live coding with Python and SuperCollider

Zu Beginn ein wenig Musiktheorie >>

BeatBox

Live Jam >> Gemeinsam musizieren

>>FoxDot
Live coding with Python and SuperCollider

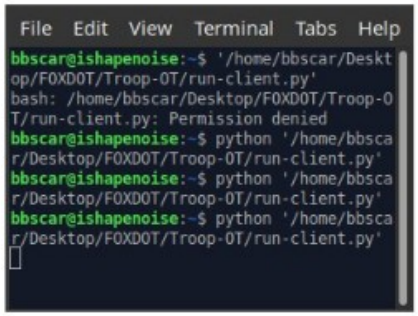
Live Jam >>

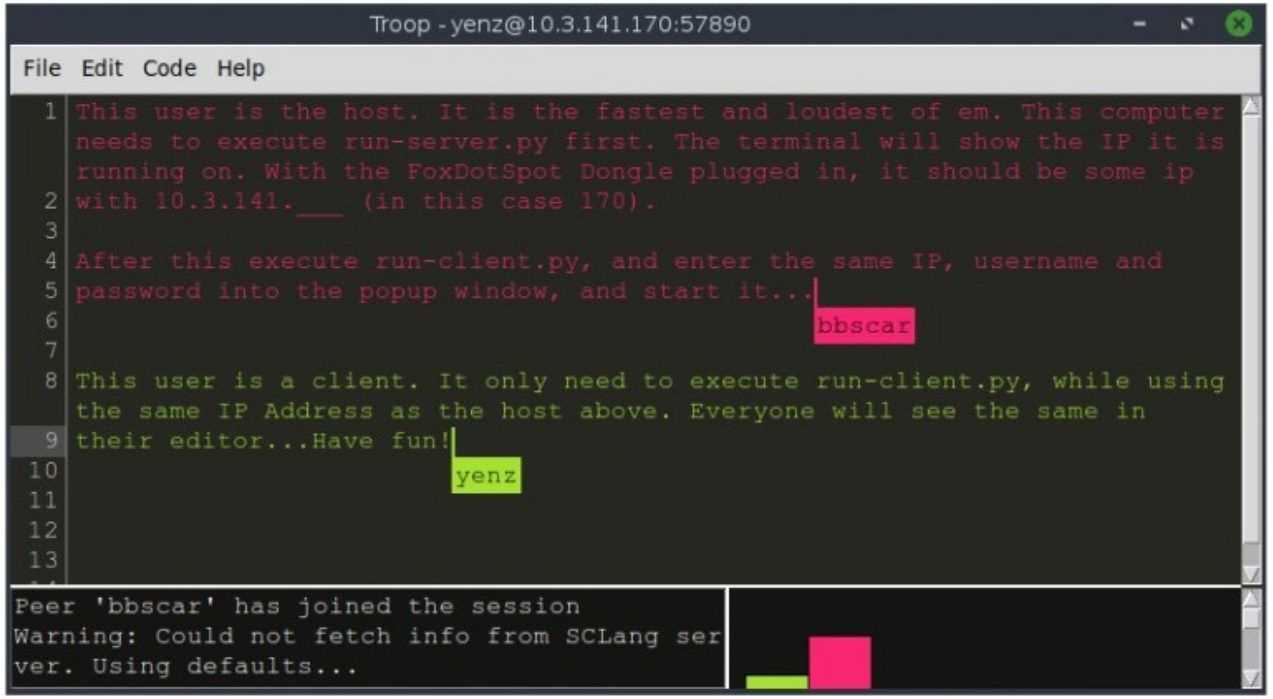
Verbinde und starte Troop

Verbindung zum Wifi Network

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

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





>>FoxDot
Live coding with Python and SuperCollider