SUPERCOLLIDER

Einstieg & Übersicht

Tonseminar WS19

// Christian Feess

01 GLIEDERUNG

02 WAS IST SUPERCOLLIDER?

01 Programmiersprache & -plattform

Supercollider ist eine kostenfreie Programmiersprache und Programmierplattform. Entwickelt wurde diese von James McCartney. Veröffentlicht wurde Supercollider im Jahre 1996. Im Jahre 2002 wurde es dann kostenfrei im Internet zur Verfügung gestellt.

Gebaut wurde Supercollider für Echtzeit Audio Berechnungen und algorithmisches Audio. Die Sprache ist auf Audioapplikationen ausgerichtet. Im Vergleich zu anderen Programmiersprachen soll die Audioprogrammierung leichter fallen, dank spezifischer Audio Klassen und der Ausrichtung auf einen Audio-Workflow.

02 Merkmale

Ein entscheidender, prinzipieller Unterschied einer Programmiersprache für Audio zu herkömmlichen DAWs (Digital Audio Workstations) ist die Interaktion. DAWs bieten meist graphische, benutzerfreundliche Oberflächen und ein vorgefertigtes Set an Tools. Die Interaktion und Bedienung erfolgt dann über Mausklicks und Tastatureingaben (Shortcuts).

In Supercollider interagiert der/ die NutzerIn über die Programmiersprache. Das bedeutet jede Aktion wird als Code niedergeschrieben und später im Programmablauf ausgeführt. Eine bisuelle Einsicht in fast jegliche Parameter und Signalfllüsse ist somit gegeben. Diese Codezeilen können fast beliebig komplex werden, was dem/ der NutzerIn eine große Freiheit in der Umsetzung seier/ ihrer Ideen bietet. Supercollider und andere Programmiersrachen bieten so einen alternativen und zudem sehr flexiblen Weg, Ideen umzusetzen.

Ein weiteres Merkmal von Supercollider sind die Interaktivitätsmöglichkeiten. Durch externe Peripherie können jegliche Paramater angesteuert und verändert werden, in der Weise, wie es der/ die NutzerIn erwünscht und belegt hat.

03 KERNKOMPONENTEN

01 scide, sclang & scsynth

Supercollider setzt sich aus drei Kernkomponenten zusammen. Diese heißen: scide, sclang und scsynth:

SCIDE

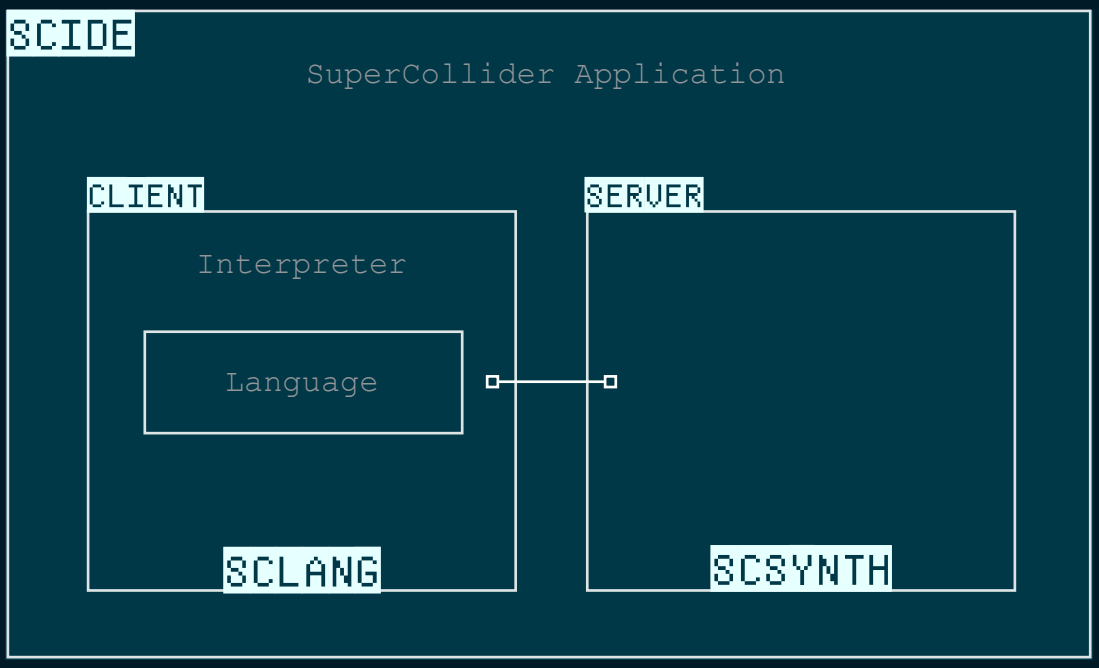
scide ist die Oberfläche und der Editor von Supercollider. Alles was der/ die NutzerIn sehen. Hiermiet wird interagiert.
// ide: "integrated development environment"

SCLANG

sclang ist die Programmiersprache. Sie interpretiert den Code, kontrolliert den scsynth, steuert diesen an.
// lang: "language"

SCSYNTH

scsynth ist der Echtzeit Audio Server von Supercollider. Hier finden alle (Signal-) Berechnungen statt.
// synth: "synthesizer"



04 HELLO SOUND!

01 Oberfläche & Bedienung

Die Oberfläche von Supercollider besteht standardmäßig aus drei verschiedenen Fenstern. Groß mittig befindet sich der Code-Editor. Darin wird der Code niedergeschrieben und später auch ausgeführt. Rechts oben befindet sich ein Hilfefenster, welches die Dokumentation von Supercollider beinhaltet. Alle Klassen und Funktionen von Supercollider sind darin dokumentiert, teils mit Erklärungen und Beispielen. Darunter befindet sich das ”Post”-Fenster. Hier werden unter anderem Code-Ausgaben und Fehler beim Ausführen des Codes angezeigt. Unter diesem Fenster befindet sich eine Statusanzeige, die Informationen über den Interpreter und Server gibt. Da die Funktionsweise von Supercollider auf einem Client-Server Prinzip basiert, muss ein (standardmäßig interner) Server gestartet werden. Dies ist notwendig um Berechnungen dort über den scsynth ausführen zu können.

// Um den Server zu starten kann man den Shortcut “Strg + B”
// nutzen oder den Menüpunkt “Server” -> “Boot Server”.

02 Codebeispiel “Hello Sound”

Um einen Einstieg in die Programmierung zu geben, folgt der typischen “Hello World” Code. Hier für Supercollider beispielhaft ein “Hello Sound” Programmcode.

IN CODE:

```
{
     SinOsc.ar(220) * XLine.kr(0.001, 1, 2, 1, 0, 1)
}.play;
```

In diesem Code Beispiel wird ein Sinuston mit 220 Hz Frequenz generiert. Die Amplitude des Tons wird mit einer exponentiellen Kurve multipliziert. Der Sinuston wird über 2 Sekunden hinweg von in seiner Lautstärke von sehr leise bis zu voller Lautstärke von der Kurve moduliert. Die Ausgabe des Signalflusses des SinOscs ist bei der Initiierung 1. Das bedeutet volle Aussteuerung des Signals. Ist der Wert auf 0, gibt es keine Aussteuerung.

05 UGENS: UNIT GENERATORS

UGens sind Klassen in der Supercollider Programmiersprache. Das Wort steht kurz für “Unit Generators”. UGens sind Klassen, die Signalberechnungen und -verarbeitungen auf scsynth vornehmen. Sie generieren Ausgangssignale in Signal-Blöcken, die jeweils 64 Sample groß sind. Zwei UGens haben wir im vorherigen Beispiel bereits kennengelernt; SinOsc() und XLine().

01 Hauptmethoden

Alle UGens haben drei Methoden. Diese werden durch einen Punkt und deren Benenung nach dem UGen angestuert. Diese drei Methoden heißen:

.AR

.ar steht für “audio rate”. Wenn diese Methode verwendet wird, wird der Signalfluss des UGens in einer höheren Auflösung berechnet. Es werden mehr Abtastpunkte verwendet. Dies ist wichtig für Audio-Signale die abgespielt werden sollen.

.KR

.kr steht für “control rate”. Im Vergleich zu .ar wird bei der Verwendung von .kr eine kleinere Abtastrate verwendet. Wenn das Signal des UGens nicht abgehört werden soll und es zur Ansteuerung anderer Werte verwendet wird nutzt man .kr um weniger Leistung des Rechners zu fordern.

.IR

.ir steht für “initialization rate”. Im Vergleich zu den vorherigen Methoden ist .ir statisch. Der UGen wird hier mit einem Wert initiiert, der dann unverändert bleibt.

Im vorherigen Code-Beispiel wurde .ar und .kr schon verwendet. Der Sinusgenerator “SinOsc()” wurde mit .ar gerufen. Diesen wollen wir abhören. Die “XLine()” soll lediglich das Signal modulieren. In dem Fall die Lautstärke. Dementsprechend wird dieser UGen mit der Methode .kr gerufen.

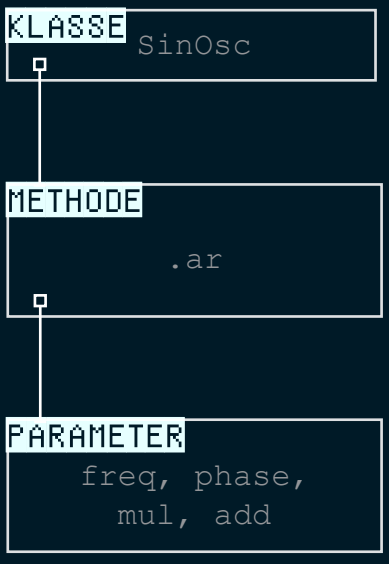
06 UGEN AM BEISPIEL: SINOSC

Um die Verwendung und Funktionsweise eines UGens besser darzustellen, wird hier der schon oben genannte SinOsc() exemplarisch erklärt.

SinOsc() ist eine UGen-Klasse in Supercollider. Diese Klasse hat diverse Methoden. Im folgenden Beispiel soll SinOsc() genutzt werden um einen Sinuston mit 440 Hz, voll ausgesteuert, zu generieren. Dafür wird die SinusOsc-Methode ".ar" gerufen. Im Code referenziert man auf diese indem ".ar()" hinten an SinOsc angehängt wird, so, dass sich "SinOsc.ar()" ergibt. Methoden haben meist Parameter, welche die Methode ansteuern. Die genutzte .ar Methode enthält die Parameter: freq, phase, mul, add. Diese stehen für: Frequenz, Phase, Multiplikator und Addition.

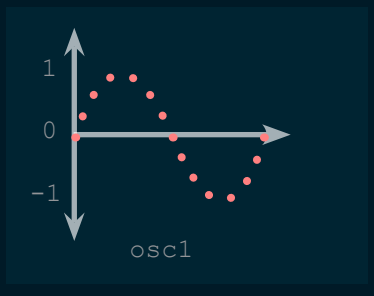
Die Parameter können nach Belieben mit numerischen Werten festgelegt werden. Parameter von Methoden sind nicht zwingend numerisch und unterscheiden sich von Methode zu Methode. Hier hilft die Dokumentation um herauszufinden, was die jeweiligen Parameter bezwecken. Wichtig beim Setzen der Parameter ist die Reihenfolge. Standardmäßig wird bei .ar folgende Reihenfolge in den Klammern ausgelsen: (freq, phase, mul, add).

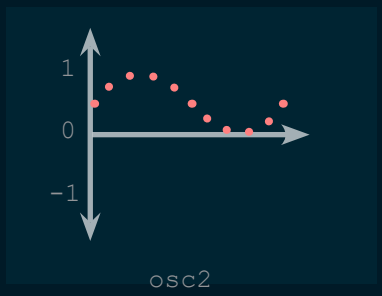
Daraus ergibt sich dann SinOsc.ar(440, 0, 1, 0).



IN CODE:

```
var osc1, osc2;
osc1 = SinOsc.ar(440, 0, 1, 0);
osc2 = SinOsc.ar(440, 0, 0.5, 0.5);
```





07 LITERATURVERZEICHNIS

Wilson, S., Cottle, D. & Collins, N. (2011), *The Supercollider Book*, MIT Press.

Supercollider Dokumentation:
https://doc.sccode.org/ (Abruf: Oktober 2019)