Sonic Pi

Sam Aaron
Sam ist der Entwickler von Sonic Pi. Tagsüber ist er wissenschaftlicher Mitarbeiter am Cambridge Computer Laboratory; nachts schreibt er Code in Clubs.
sonic-pi.net

LIVE CODING IN DER BILDUNG

Die Scheinwerfer schneiden sich durch den Nebel, der Rhythmus bewegt die Menge, die Synthesizer heizen die Stimmung weiter auf. Wie auch immer, irgendetwas ist nicht ganz richtig. Über dem Darbieter bewegt sich in hellen Farben futuristischer Text, tanzend und blitzend. Das sind keine abgefahrenen Visuals; es ist eine Projektion von Sonic Pi auf einem Raspberry Pi. Da steht nicht ein DJ, mit Plattentellern oder CDs hantierend - Die Person schreibt, editiert und evaluiert Code. Live.
Das ist Live Coding.
Dies klingt wie eine Szene von einem zukünftigen Musikfestival, ist aber die Beschreibung einer Schülerin auf der Bühne ihrer Schule.
Der Ansatz Code zu manipulieren ist nicht nur für Live-Auftritte interessant, es ist auch ein neuer Weg Schülerinnen und Schüler für das Coden zu begeistern.

Dieser Artikel wurde aus dem Englischen übersetzt und in Teilen an die Bildungslandschaft der deutschsprachigen Schweiz angepasst. Der Originalartikel ist unter **sonic-pi.net** zu finden.
Übersetzung:
Christian Dietz, Nico Steinbach und Nando Stöcklin, 2017

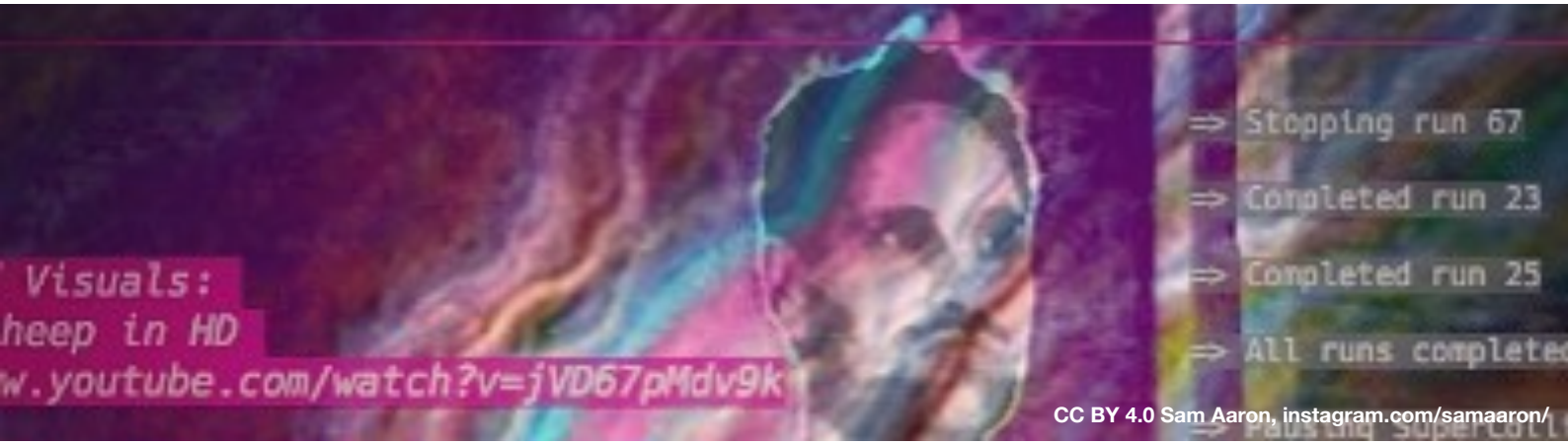
Begeisterung für's Coden

Die Bildungslandschaft ist sich in weiten Teilen einig, dass Medien und Informatik schon in der Volksschule ein integraler Bestandteil sein muss. Beispielsweise hat sich dies im neuen Lehrplan 21 der deutschschweizer Kantone im Modullehrplan Medien und Informatik niedergeschlagen.
Hingegen nicht einig ist man sich über "wie wollen wir Schülerinnen und Schüler denn überhaupt dafür begeistern?". Es scheiden sich die Geister, wenn es um die Vermittlung von Informatikkonzepten und Grundlagen geht.

Von Tabellenkalkulation über Schildkröten, bis hin zu fahrenden Kleinstrobotern. Die allermeisten Herangehensweisen sind in der Faszination einzelner Experten für Technik begründet.
Sonic Pi beschreitet einen andern Weg. Das Programm verwandelt geschriebenen Code in Musik und wird so zu einem neuartigen Musik-Instrument. Es ermöglicht so Schülerinnen und Schülern ihre eigene Musik und somit ihren eigenen Stil zu programmieren.
Das Musikmagazin Rolling Stone beschreibt den Sonic Pi - Auftritt am Moogfest USA, 2016:
"... it truly seemed less like a performance and more like an invitation to code your own adventure."

Leistungsstark aber simpel

Ein System, welches einen hohen Anspruch an Professionalität hat, ist oftmals schwierig zu erlernen, eignet sich somit nur sehr beschränkt für die Schule. Sonic Pi wurde deshalb in enger Kollaboration mit Lehrpersonen und unzähligen Unterrichtsstunden mit Schülerinnen und Schülern entwickelt.
Einfachheit im Design des Programms und dessen Bedienung war der Hauptanspruch in der Entwicklung. Ein zehnjähriges Kind kann es verstehen und bedienen.
Für den Einstieg in die Sonic Pi - Welt bedarf es lediglich zwei sehr einfache Anweisungen. Die `play`-Anweisung, welche verschiedene Noten spielt und die `sleep`-Anweisung, welche die Wartezeit bis zur nächsten Note definiert.



Mit `sample` wird die nächste Sonic Pi - Ebene erschlossen, welche aufgenommene Sounddateien (Samples) abspielen kann. Dies können ganze Drumloops oder einfach auch Geräusche sein. Natürlich können diese Sounddateien auch während dem Abspielen manipuliert werden.
Zusätzlich stehen der/dem Coder auch Studio-Effekte wie zum Beispiel Hall oder Verzerrung zur Verfügung. Für eine Gitarre mit Hall braucht es drei Zeilen Code:

```
with_fx :reverb do
   sample :guit_harmonics
end
```

Modullehrplan Medien und Informatik

Die allermeisten informatischen Kompetenzen des Modullehrplans Medien und Informatik, sowie weitere, auch komplexere, Informatikkonzepte lassen sich mit Sonic Pi aufzeigen.
Die Stärke von Sonic Pi liegt im Gegensatz zu anderen Umgebungen im auditiven Zugang, welcher stark an Empfindungen geknüpft ist.
Eine eindeutige Zuteilung von Sonic Pi in Fächer wie Musik oder Informatik / Mathematik ist von vornherein ausgeschlossen, da die direkte, auditive Rückmeldung sowie danach die Gleichwertigkeit des Empfundenen mit dem geschriebenen Code ganz im Vordergrund steht, was wiederum motivierend wirken kann.
Nebst den bereits erwähnten Anweisungen können Schleifen, Listen, Variablen, bedingte Anweisungen sowie Funktionen einfach programmiert werden.

Schleifen

Mittels den zwei einfachen Anweisungen `play` und `sleep` lassen sich schon etliche bekannte aber auch selbst erfundene Melodien abbilden. Damit nicht meterlanger Code geschrieben werden muss, bietet Sonic Pi eine einfach verständliche Schleife, analog den Wiederholungszeichen in der musikalischen Schreibweise, an.

```
2.times do
   play 60
   sleep 1
   play 62
   sleep 1
   play 64
   sleep 1
   play 60
   sleep 1
end
```

Alles was zwischen `do` und `end` steht wird wiederholt. Im Beispiel oben die ersten zwei Takte von Bruder Jakob. Natürlich kann zwischen `do` und `end` auch anderer Code sowie mehrere Schleifen hintereinander geschrieben werden:

```
5.times do
   sample :bd_haus
   sleep 0.5
end

3.times do
   sample :drum_cymbal_open
   sleep 1
end
```

Listen

Das Beispiel von Bruder Jakob lässt sich auch mit einem Ringbuffer (Ring-Liste) darstellen:

```
8.times do
   play (ring 60, 62, 64, 60).tick
   sleep 1
end
```

Bei jedem Durchgang in der Schleife wird die nächste Zahl der Klammer zurückgegeben und somit abgespielt. `tick` zählt einfach bei jedem Aufruf eins weiter. Beim fünften Aufruf wird wieder die erste Zahl (60) ausgegeben; deshalb Ring-Liste.

Variablen

Anstelle einer Liste mit definierten Werten kann der Computer natürlich auch die Zahlen selber berechnen. Dies geschieht am besten mit einer frei zu definierenden Variable.

Im Beispiel unten wird diese `ton` genannt und am Ende der Schleife jeweils um eins erhöht.

```
ton = 30
60.times do
   play ton
   sleep 0.06125
   ton = ton + 1
end
```

Die Kombination von Schleifen, Listen und Variablen erlauben auch rhythmische Muster darzustellen:

```
rythmus = (ring 1, 0, 1, 0, 1, 0, 1, 1)
32.times do
   sample :bd_haus, amp: rythmus.tick
   sleep 0.5
end
```

Die Lautstärke des Pauken-Samples `:bd_haus` wird bei jedem Durchgang durch die Variable `rythmus` definiert. `amp:` steht für Verstärker (amplifier), was bei `0` einer musikalischen Pause entspricht.

Bedingte Anweisungen

Oftmals will man etwas nur unter bestimmten Umständen hörbar machen, oder dies auch dem Zufall überlassen.

```
loop do
  if one_in 2
   sample :drum_cowbell
  else
   sample :drum_snare_hard
  end
 sleep 0.125
end
```

`loop` ist die Endlosschleife, welche erst mit dem Klicken des Stop Buttons beendet wird. In dieser Endlosschleife wird mit `one_in 2` mit einer 50 zu 50 prozentigen Wahrscheinlichkeit entweder der Sample `:drum_cowbell` oder der Sample `:drum_snare_hard` abgespielt.

Funktionen

Ähnliche oder gleiche Aufgaben wie zum Beispiel eine bestimmte Tonabfolge mit veränderbaren Parametern (Tonsprung, Zeit ...) lassen sich in Funktionen "auslagern":

```
define :kleinesProgramm do |wieOft, ton, pause|
  wieOft.times do
    play ton
    sleep pause
    ton = ton + 1
  end
end

kleinesProgramm 50, 30, 0.015
```

Die Werte `50, 30, 0.015` nach dem Namen der Funktion sind die Werte für die Variablen `wieOft`, `ton` und `pause`. Einmal das Unterprogramm definiert, kann es nun leicht mit einer Zeile Code auch mehrfach aufgerufen werden.

```
kleinesProgramm 60, 30, 0.01
kleinesProgramm 10, 40, 0.25
```

Der live loop

Der `live_loop` ist eines der wichtigsten Elemente in Sonic Pi. Das Element vereint zwei wichtige Funktionen von Sonic Pi und kann als Herzstück bezeichnet werden.
Die erste Funktion beinhaltet analog zu `loop do #irgend ein Code end` die Endlosschleife, wie sie auch in Scratch mit "wiederhole fortlaufend" zu finden ist. Die zweite und elementar wichtige Funktion in Sonic Pi ist die Mehrstimmigkeit. Jeder `live_loop` hat seine eigene Abarbeitung, welche auch einen Namen braucht.
Im folgenden Beispiel `:bassdrum`.

```
live_loop :bassdrum do
   sample :bd_haus
   sleep 0.5
end
```

Programmcode zur Laufzeit ändern = live coding

Noch während dem Abspielen des Codes lässt dieser sich schon wieder verändern, was unter dem Begriff „live coding" verstanden wird.
Mit dem folgenden Beispiel lässt sich dies schön verdeutlichen.

```
sample :ambi_glass_hum, rate: 0.7
```

Mittels `rate:` lässt sich die Abspielgeschwindigkeit eines Samples verändern. Ähnlich bei einem Regler für Tempo bei einem

alten Kassettengerät oder bei Plattenspieler, den man mit der Hand bremst. Je tiefer der Wert, umso tiefer und langsamer das Sample. Nach dem Starten (**Run**) des Programms kann jetzt der Wert von `rate:` verändert werden. Um den abgeänderten Code hörbar zu machen, klickt man erneut auf **Run**.
Mit nur einer Zeile Code können so Stimmungen erzeugt werden, welche zum Beispiel in einem Scratch Game, einer Theateraufführung oder einem Schulradio Verwendung finden.

Schlafen ist wichtig

Eine wichtige Anweisung bei der Verwendung eines `live_loop` ist die `sleep` Funktion. Sonic Pi stoppt einen Loop ohne Angabe sofort und gibt einen Warnhinweis aus. Dies aber ohne die restlichen `live_loop`s zu stoppen.
Was würde denn geschehen, wenn ich den folgenden Code ausführen würde?

```
live_loop :endlosGrosserChor do
   sample :ambi_choir
end
```

Der Computer würde wohl abstürzen; zumindest würde Sonic Pi nicht mehr reagieren.

Parallele Abarbeitung

Musik findet interaktiv und gleichzeitig statt. Das Schlagzeug zusammen mit dem Bass, mit Gesang, mit Gitarre etc. In der Informatik wird dies mit Nebenläufigkeit umschrieben. Der `live_loop` in Sonic Pi macht neben dem Loopen genau dies. Mit mehreren `live_loop`s lassen sich so gleichzeitig Sounds oder verschiedene Stimmen programmieren.

```
live_loop :schlagzeug do
   sample :bd_tek
   with_fx :echo, phase: 0.125, mix:  0.4 do
      sample :drum_cymbal_soft,sustain:0,
        release: 0.1
   end
   sleep 0.5
end

live_loop :bass do
   use_synth :tb303
   play :e1, release: 4, cutoff: 120,
     cutoff_attack: 1
   sleep 4
end
```

Der erste, kürzere `live_loop` spielt zwei Samples gleichzeitig ab, wobei der zweite Sample `:drum_cymbal_soft` mit einem kurzen Echo (Delay) in der vierfachen Geschwindigkeit des `live_loop`s abgespielt wird. Musikalisch ergibt dies eine Hihat im Sechzehntel-Raster.

Der zweite, längere `live_loop` spielt den Ton E mit einem legendären Roland TB-303 Synthesizer Sound. Die Filterbewegung erstreckt sich über einen Schlag, Bei Tempo = 60 bpm entspricht dies einer Sekunde. Es ist sogar möglich in verschiedenen `live_loop`s verschiedene Tempi zu verwenden. Somit lassen sich sehr komplexe musikalische Muster generieren.

Sonic Pi in der Schulklasse einsetzen

Musik ist Teil der Lebenswelt der Jugendlichen. Sonic Pi eignet sich also hervorragend als Einführung in eine Programmiersprache und vermittelt die wichtigsten Grundelemente einer Programmiersprache.
Das Programm kann unter sonic-pi.net für OS X, Windows und Linux kostenlos heruntergeladen werden. Nebst den integrierten Tutorials existieren weitere frei downloadbare Anleitungen wie zum Beispiel "`Code music with Sonic Pi - Sonic Pi Essentials`" von Sam Aaron, dem Entwickler von Sonic Pi.

Lizenz

Sam Aaron, Text Live Coding Education
`sonic-pi.net/files/articles/Live-Coding-Education.pdf`
Christian Dietz, Nico Steinbach, Nando Stöcklin
Übersetzung und Anpassung an Schweizer LP21