armada

Partitursynthese-Software von Daniel Verasson mit Ausgabeformaten für CSOUND und FOMUS.

Installation:

armada basiert auf der Programmiersprache *Scheme* und nutzt den Interpreter *guile*. Es kann durch den Befehl *(load "/loadpath/armada.scm")* gestartet werden. Der Speicherort muss selbstverständlich angepasst werden. Möglich ist auch das standardmäßige Laden des Codes im Editor Emacs durch Einfügen des gleichen Befehls in die Boot-Datei von *guile* "boot-9.scm".
Damit *armada* sinvoll arbeitet, muss zusätzlich folgende Software installiert sein: Fomus, Lilypond und Csound.
Zusätzlich zu *armada* gibt es die Datei *tierchen_142803.scm*, in der über 60 kleinere Scheme-Prozeduren enthalten sind.

Erstellen einer .sco-Datei (**CSOUND**)

< armada (dateiname	*String*
	anzahl-zeilen	*ganze Zahl*
	startpunkt(p2)	*1 Element: Zahl oder Liste / mehrere Elemente: Liste*
	dauer(p3)	*1 Element: Zahl oder Liste / mehrere Elemente: Liste*
	…	*alle weiteren Parameter sind optional*
	pn	
	:instr	*optional (Erklärung s.u.)*
	:include) >	*optional (Erklärung s.u.)*

Beim Auswerten wird eine .sco-Datei erstellt.
Die ersten 4 Argumente sind obligatorisch, auch ihre Reihenfolge ist festgelegt; alle weiteren sind optional und folgen dem gleichen System wie die Argumente p2 und p3.
Hat ein Argument nur ein Element, wird es bei jedem neuen Zeilenaufruf verwendet, bei einer Liste aus mehreren Argumente rotiert die Liste bei jedem neuen Zeilenaufruf, so dass das erste Element an die letzte Stelle rückt usw.
Ist #f das letzte Argument, wird keine .sco-Datei erstellt, sondern die Gesamtdauer in Sekunden angegeben.

Optionale Argumente:

:instr	*Csound-Instrument-Nummer* *1 Element: Zahl oder Liste / mehrere Elemente: Liste* *default: 1*
:include	*Datei wird in die Score eingebunden (z.B. function-tables)* *String (Dateipfad)* *default:<leer>*

2 einfache Beispiele:

(armada "name.sco" 3 '(0 1 2) '(3 2 1))	*erstellt Datei "name.sco"*

(armada "name.sco" 3 '(0 1 2) '(3 2 1) #f)	*gibt die Gesamtdauer der Datei "name.sco" in Sekunden an*

Erstellen einer .fms-Datei (FOMUS)

< armada (dateiname	*String*
anzahl-toene	*ganze Zahl*
startpunkt	*1 Element: Zahl oder Liste / mehrere Elemente: Liste (♩ =1)*
dauer	*1 Element: Zahl oder Liste / mehrere Elemente: Liste (♩ =1)*
tonhoehe	*1 Element: Zahl oder Liste / mehrere Elemente: Liste MIDI-Notennummern (gerundet auf Vierteltoene)*
:staff	*optional (Erklärung s.u.)*
:voice	*optional (Erklärung s.u.)*
:dynamics	*optional (Erklärung s.u.)*
:articulation	*optional (Erklärung s.u.)*
:format	*optional (Erklärung s.u.)*
:measure	*optional (Erklärung s.u.)*
:tempo) >	*optional (Erklärung s.u.)*

Beim Auswerten wird eine .fms-Datei erstellt.
Die ersten 5 Argumente sind obligatorisch, auch ihre Reihenfolge ist festgelegt, alle weiteren sind optional und werden von dem entsprechenden Ausdruck gefolgt. Die Reihenfolge der optionalen Argumente spielt keine Rolle.
Hat ein Argument nur ein Element, wird es bei jedem neuen Ton verwendet, bei einer Liste aus mehreren Argumente rotiert die Liste bei jedem neuen Ton, so dass das erste Element an die letzte Stelle rückt usw. Ausnahmen bilden hier die Argumente für :measure und :tempo, deren Angaben nicht rotieren, sondern initial gesetzt werden.
Ist #f das letzte Argument, wird keine .fms-Datei erstellt, sondern die Gesamtdauer in Vierteln angegeben.

Optionale Argumente:

:staff	*Nummer des Notensystems* *1 Element: Zahl oder Liste / mehrere Elemente: Liste* *default: 1*
:voice	*Nummer der Stimme innerhalb des Notensystems* *1 Element: Zahl oder Liste / mehrere Elemente: Liste* *default: 1*
:dynamics	*Dynamik von 1 (ppp) bis 8 (fff)* *1 Element: Zahl oder Liste / mehrere Elemente: Liste* *default:<leer>*
:articulation	> / . ^ ! *1 Element: String oder Liste / mehrere Elemente: Liste* *default:<leer>*

:format *Ausgabeformate:*
"ly" (Lilypond)
"xml" (Sibelius oder Finale)
"mid" (MIDI)
"fms" (schreibt nur .fms-Datei - keine Auswertung !)
1 Element: String oder Liste / mehrere Elemente: Liste
default: "ly"

:measure *Taktart*
1 Element: Liste / mehrere Elemente: Liste aus Listen
Bsp.: 4/4-Takt bei Zeitpunkt 0
'(0 4 4) bzw. '((0 4 4))
Bsp.: 4/4-Takt bei Zeitpunkt 0, 5/8-Takt bei Zeitpunkt 4
'((0 4 4) (4 5 8))
default: '(0 4 4)

:tempo *Tempo*
1 Element: Liste / mehrere Elemente: Liste aus Listen
Bsp.: Viertel=60 bei Zeitpunkt 0
'(0 1/4 60) bzw. '((0 1/4 60))
Bsp.: Viertel=60 bei Zeitpunkt 0, Halbe=60 bei Zeitpunkt 4
'((0 1/4 60) (4 1/2 60))
default:<leer>

2 einfache Beispiele:

```
(armada "name.fms"
        3
        '(0 1 2)
        1
        '(60 64 67))    erstellt Datei "name.fms"

(armada "name.fms"
        3
        '(0 1 2)
        1
        '(60 64 67)
        #f )            gibt die Gesamtdauer der Datei "name.fms" in Vierteln an
```