Basics

make sound	d1 sound "bd" or s "bd"
different sample	bd:1 or # n "1" or $ n "0 2" # sound "arpy"
compile: one line	shift + enter
compile: multi-line	ctrl + enter
patterns	"bd sd" bd*4 bd/4 "[bd hh] sd" (can nest) or bd hh . sd (easier to type)
layering	[bd bd, hh sd hh]
different each cycle	"bd <arpy:0 arpy:3>"
make quiet	d1 silence solo $ d1 mapM ($ silence)[d1,d2,d3] hush
* / on groups	[bd sd]*2 cp if you don't get through a whole subpattern in a cycle, it will start where it left off with the next one. Imagine each group is a little tape loop that plays while it is its turn
polyrhythms	{bd hh cp}%4

Bjorkland	bd(5,8) or bd(5,8,1)
	bd([5,3]/2,8)
	e 5 8

Tempo	cps 1 cps (140/60/4)

Functions

Pattern Transformers

Unless otherwise stated, between $ ... $

Deterministic

brak	squash every other cycle to fit 1/2 a cycle, and offset it by a 1/4 of a cycle
fast, slow	fast 2
fit	
iter	
jux (and juxBy)	jux (rev) applies function to RH only. juxBy 0.5 brings 0.5 closer to centre
linger	linger 0.25 repeats first 1/4 cycle
Beat rotation	0.25 <~ shifts pattern 1/4 cycle
rev	rev
smash	
spread	
toScale	
trunc	trunc 0.75 plays first 3/4 of cycle
zoom	zoom (0.25, 0.75) plays the section from 0.25 to 0.75 of cycle over time period of original pattern

Stochastic

degradeBy	degradeBy 0.9 removes events from pattern 90%
scramble	scramble 2 sampling with replacement
shuffle	shuffle 2 sampling without replacement

Sample

loopAt	loopAt 4 makes sample fit the given number of cycles
gap	
chop	chop 16 granulator granualizes every sample in order
striate	striate 16 granulator interlaces all samples together striate' 32 (1/16)
striateL	striateL' 3 0.125 4 loops each sample chunk a number of times (2nd arg), loop count is 3rd arg
stut	stut 4 0.5 0.2 delay 4 echos, each one 50% quieter than the last, with 1/5th of a cycle between them
stut'	stut' 2 (1/3) (# vowel "a e i o u%2") generalised stut for different functions

Conditional

someCyclesBy	someCyclesBy 0.25 (fast 2)
foldEvery	foldEvery 3, 4] (fast 2) equiv. to every 3 (fast 2) $ every 4 (fast 2)
ifp	ifp ((== 0).(flip mod 2)) (striate 4) (# coarse "24 48") striate on even, coarse on odd
mask	
every	$ every 3 (fast 2) $
every'	every' 3 1 (fast 2) d1 $ sound (every 4 (fast 4) "bd*2 [bd [sn sn*2 sn] sn]")
sometimesBy	sometimesBy 0.25 (fast 2)
swingBy	
when	
whenmod	when remainder ≥
within	

Compositions

seqP	
cat	cat [s "bd*2", s "arpy jvbass*2"] concatenates a list of patterns into a new pattern; each pattern in the list will maintain its original duration
fastcat	new patternâĂŹs length will be a single cycle
interlace	
randcat	cat but picks patterns at random
append	
spin	
stack	d1 $ stack [sound "bd bd*2", sound "hh*2 [sn cp] cp future*4"]
superimpose	
weave	applies pattern to list and is offset for each pattern d1 $ weave 4 (pan sine) sound "[jazz:0 hh jazz:0 hh, sn]", sound "casio casio:1"
wedge	

Transitions

anticipateIn	t1 (anticipateIn 4) $ sound "jvbass(5,8)"
clutchIn	
histpan	
jump	
jumpIn	
jumpIn'	
jumpMod	
mortal	
superwash	
wait	
wash	
xfade	

Operators

# equiv. |=|

Synth Parameters (or effects)

given by #

Sound

sustain
speed
endSpeed
begin
end
loop
pan

Envelope

attack
hold
release

hpf

hcutoff
hresonance

lpf

cutoff
resonance

bpf

bandf
bandq

delay

delay
delayfeedback
delaytime

reverb

room	// size

tremolo

tremolorate	// tremolodepth

phaser

phaserrate	// phaserdepth

The Rest

coarse	
crush	
cut	
accelerate	
legato	
loop	
nudge	(Should this be here? Or is this pattern)
shape	
sound	
unit	
vowel	

Continuous Patterns

between 0,1. sine,saw,tri,square
slow 2 $ scale 0 2 $ sine

Operators

Utility

choose	choose[2...8] or choose[2,3..8]
irand	irand n generates a pattern random integers 0 to n-1 "amencutup*8" # n (irand 8)
pequal	
rand	
run	
scale	
up	

Combine transforms

[.](# speed "0.5") . rev

Super Dirt

load own samples	~ dirt.loadSoundFiles("/path/to/samples/*")
change speaker output	

Dirt Samples

808 808bd 808cy 808hc 808ht 808lc 808lt 808mc 808mt 808oh 808sd 909 ab ade ades2 ades3 ades4 alex alphabet amencutup armora arp arpy auto baa baa2 bass bass0 bass1 bass2 bass3 bassdm bassfoo battles bd bend bev bin birds birds3 bleep blip blue bottle breaks125 breaks152 breaks157 breaks165 breath bubble can casio cb cc chin chink circus clak click clubkick co control cosmicg cp cr crow d db diphone diphone2 dist dork2 dorkbot dr dr2 dr55 dr_few drum drumtraks e east electro1 erk f feel feelfx fest fire flick fm foo future gab gabba gabbaloud gabbalouder glasstap glitch glitch2 gretsch gtr h hand hardcore hardkick haw hc hh hh27 hit hmm ho hoover house ht if ifdrums incoming industrial insect invaders jazz jungbass jungle juno jvbass kicklinn koy kurt latibro led less lighter linnhats lt made made2 mash mash2 metal miniyeah moan monsterb moog mouth mp3 msg mt mute newnotes noise noise2 notes numbers oc odx off outdoor pad padlong pebbles perc peri pluck popkick print proc procshort psr rave rave2 ravemono realclaps reverbkick rm rs sax sd seawolf sequential sf sheffield short sid sine sitar sn space speakspell speech speechless speedupdown stab stomp subroc3d sugar sundance tabla tabla2 tablex tacscan tech techno tink tok toys trump ul ulgab uxay v voodoo wind wobble world xmas yeah

Nice snippets

linger "<1 0.5 0.25 0.125>"
d1 $ spread ($) [fast 2, rev, slow 2, striate 3, (# speed "0.8")] $ sound "[bd*2 [bd]] [sn future]*2 cp jvbass*4"
d1 $ loopAt 4 $ chop 32 $ sound "breaks125"