

Humdrum Rhythm Tools

craig @ ccrma.stanford.edu

24 April 2014

Humdrum program documentation

`timebase -h` gives one-page summary of *mint* command (similar for all Humdrum Toolkit programs)
`beat --options` list options for *serialize* command (same for all Humdrum Extras programs)

Humdrum Toolkit man pages:

<http://www.humdrum.org/Humdrum/commands/timebase.html>

Humdrum Extras man pages:

<http://extras.humdrum.org/man/beat>

Chapter 23 in the Humdrum Users' Guide (Rhythm):

<http://www.humdrum.org/Humdrum/guide23.html>

List of various Humdrum resources:

<http://humdrum.ccarh.org>

Humdrum Users' Group (**HUG):

<https://groups.google.com/forum/?fromgroups#!forum/starstarhug>

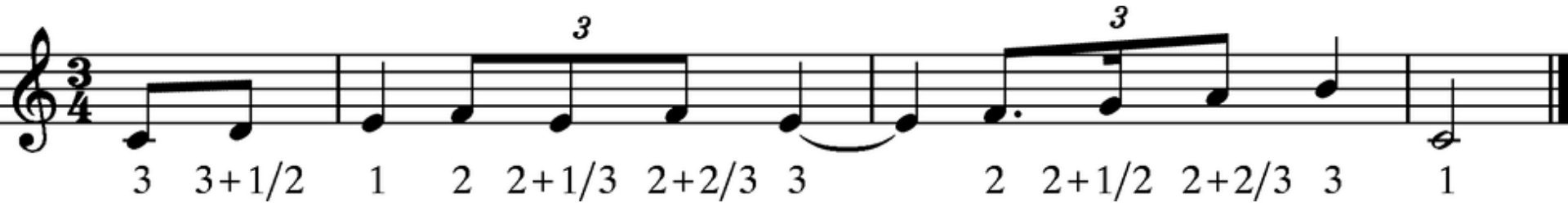
beat

<http://extras.humdrum.org/man/beat>

	beat	beat -f	beat -c	beat -d
**kern	**beat	**beat	**absb	**dur
*M3/4	*M3/4	*M3/4	*M3/4	*M3/4
*clefG2	*	*	*	*
8cL	3	3	0	0.5
8dJ	3.5	3+1/2	0.5	0.5
=1	=1	=1	=1	=1
4e	1	1	1	1
12fL	2	2	2	0.333333
12e	2.33333	2+1/3	2.33333	0.333333
12fJ	2.66667	2+2/3	2.66667	0.333333
[4e	3	3	3	2
=2	=2	=2	=2	=2
4e]	.	.	4	.
12.fL	2	2	5	0.5
24gk	2.5	2+1/2	5.5	0.166667
12aJ	2.66667	2+2/3	5.66667	0.333333
4b	3	3	6	1
=3	=3	=3	=3	=3
2c	1	1	7	2
==	==	==	==	==
*_	*_	*_	*_	*_

beat

```
beat -fa input.krn | hum2muse --text beat | muse2ps =z21j \  
| convert -density 300 -trim -resize 50% output.png
```



gettime

<http://extras.humdrum.org/man/gettime>

- Calculate the performance time in seconds/milliseconds for a score.

`gettime -T` :: calculate the total performance duration of the score(s)

`gettime -T h://chopin/preludes/prelude28-01.krn`

Total time: 25.5 seconds

`gettime -T h://chopin/preludes`

Total time: 31:05.83537 minutes

`thrux h://beethoven/sonatas | gettime -T`

Total time: 9:23:55.3539 hours

`thrux -v norep h://beethoven/sonatas | gettime -T`

Total time: 7:44:52.8678 hours

minrhy

<http://extras.humdrum.org/man/minrhy>

- Calculates the minimum rhythmic unit for which all rhythms in the score are integer multiples.

```
**kern
*M3/4
*clefG2
8cL
8dJ
=1
4e
12fL
12e
12fJ
[4e
=2
4e]
12.fL
24gk
12aJ
4b
=3
2c
==
*_
```

minrhy
24

- Smallest rhythm value not necessarily present:

```
**kern
4c
8d
8d
12e
12e
12e
*_
```

minrhy

24

timebase

<http://www.humdrum.org/Humdrum/commands/timebase.html>

- Insert null-tokens lines to make all data lines represent the same duration.
- Use minrhy to calculate the time unit to use with timebase
- Data loss if timebase rhythm is too large
- Grace notes are lots (should be fixed by converting to reversible comments)

	minrhy	timebase -t 8	timebase -t 16
**kern	8	**kern	**kern
2c		*tb8	*tb16
8d		2c	2c
4e		.	.
8f		.	.
4g		.	.
*_		.	.
		8d	8d
		4e	4e
		.	.
		8f	8f
		4g	4g
		.	.
		*_	*_

gettime (2)

```
**kern
*M4/4
*MM60
=1-
4c
4d
4e
4f
=2
*MM120
4c
4d
4e
4f
=3
*MM144
4c
4d
4e
4f
==
*
```

gettime -s

```
**time
*u=sec
*M4/4
*MM60
=1-
0
1
2
3
=2
*MM120
4
4.5
5
5.5
=3
*MM144
6
6.41667
6.83333
7.25
==
*
```

gettime -as | gettime -at

**kern	**time	**tempo
*	*u=sec	*
*M4/4	*M4/4	*M4/4
*MM60	*MM60	*MM60
=1-	=1-	=1-
4c	0	60
4d	1	60
4e	2	60
4f	3	60
=2	=2	=2
*MM120	*MM120	*MM120
4c	4	120
4d	4.5	120
4e	5	120
4f	5.5	120
=3	=3	=3
*MM144	*MM144	*MM144
4c	6	144
4d	6.41667	144
4e	6.83333	144
4f	7.25	144
==	==	==
*	*	*

assemble

- Reverses the process of extract command, joining parts into full score
1. Calculate the minimum rhythmic unit (least common multiple of all rhythms in the input to assemble).
 2. Apply timebase at that minimum rhythm (warning about grace notes)
 3. Run assemble command
 4. Remove null-token lines

file1.krn

**kern

*M3/4

=1-

4c/

8aL/

8aL/

4c/

==

*_

file2.krn

**kern

*M3/4

=1-

4cc\

12ccL\

12cc\

12ccJ\

4cc\

==

*_

minrhy file1.krn file2.krn

file1.krn: 8

file2.krn: 12

all: 24

timebase -t 24 file1.krn > t1.krn

timebase -t 24 file2.krn > t2.krn

assemble t2.krn t1.krn | rid -d > score.krn

partjoin

- timebase command eats grace notes (**kern notes which contain “q” character)
- partjoin script automates minrhy/timebase/assemble & handles grace notes

```
#!/usr/bin/perl
use strict;
my @files = @ARGV;
die "Usage: $0 file1 file2 ...\n" if @files < 2;
my $filelist = join(" ", @files);
my $minrhy = `minrhy $filelist`;
$minrhy =~ /all:\s*(\d+)/;
$minrhy = $1;
my @assemblefiles;
foreach my $file (@files) {
    my $contents = getContents($file);
    open (PROCESS, "| timebase -t $minrhy > $file-timebase$minrhy") or die;
    print PROCESS $contents;
    close PROCESS;
    $assemblefiles[@assemblefiles] = "$file-timebase$minrhy";
}
my $assemblelist = join(" ", reverse @assemblefiles);
my $result = `assemble $assemblelist | rid -d | grep -v "^*tb"`;
printResult($result);
rm -f $assemblelist;
exit(0);
```

```
sub getContents {
    my ($file) = @_ ;
    open (FILE, $file) or die;
    my @contents = <FILE>;
    for (my $i=0; $i<@contents; $i++) {
        my $line = $contents[$i];
        next if $line =~ /^!/;
        next if $line =~ /\^*/;
        if ($line =~ /q/) {
            my @fields = split(/\t/, $line);
            for (my $j=0; $j<@fields; $j++) {
                $fields[$j] = "!QqQ$fields[$j]";
            }
            $contents[$i] = join("\t", @fields);
        }
    }
    return join("", @contents);
}

sub printResult {
    my ($result) = @_ ;
    my @output = split(/\n/, $result);
    my $line;
    foreach $line (@output) {
        if ($line !~ /!QqQ/) {
            print "$line\n";
            next;
        }
        my @fields = split(/\t/, $line);
        for (my $j=0; $j<@fields; $j++) {
            my $token = $fields[$j];
            if ($token eq "!") {
                print ".";
            } else {
                $token =~ s/^!QqQ//;
                print $token;
            }
            print "\t" if $j < $#fields;
        }
        print "\n";
    }
}
```

partjoin (2)



input1.krn

```
**kern
*M3/4
=1-
4c
g#q
8a
g#q
8a
4c
==
*_
```

input2.krn

```
**kern
*M3/4
=1-
4cc\
bq
12ccL\
bq
12cc\
bq
12ccJ\
4cc\
==
*_
```

partjoin input1.krn input2.krn > output.krn

output.krn

```
**kern      **kern
*M3/4        *M3/4
=1-          =1-
4cc\         4c
bq           g#q
12ccL\       8a
bq           .
12cc\        .
.            g#q
.            8a
bq           .
12ccJ\       .
4cc\         4c
==           ==
*_           *_
```

Search for particular sonority

- How many times and where does the sonority “A B C” occur in a repertoire?



**kern

a b c

a b- c

*_

tntype

**tnt

3-2B

3-2A

*_

ABC sonority

```
tntype -at h://jrp/Jos | ditto -p | hgrep -mbH 3-2 > abc.txt
```

```
os0301a-Missa_Ave_maris_stella-Kyrie.krn:measure 22:beat 7:4E\2d\ (2.F/) (Or) 3-2BT02
Jos0301a-Missa_Ave_maris_stella-Kyrie.krn:measure 30:beat 7:2E\ 2G/ ([1f) 4ee\ 3-2AT04
Jos0301b-Missa_Ave_maris_stella-Gloria.krn:measure 47:beat 7:(1A) (1c) 4B-\ (Or) 3-2AT09
Jos0301b-Missa_Ave_maris_stella-Gloria.krn:measure 63:beat 4:4E\ (Of) (1d) (Or) 3-2BT02
Jos0301b-Missa_Ave_maris_stella-Gloria.krn:measure 71:beat 6:(0D) ([0d) 4e\ (2.ff\ 3-2BT02
Jos0301c-Missa_Ave_maris_stella-Credo.krn:measure 36:beat 12:(2E\ (2G/) 4f\ (2g/) 3-2AT04
Jos0301c-Missa_Ave_maris_stella-Credo.krn:measure 41:beat 10:(2BB-/ 4A/ (2B-\) 4c/ 3-2AT09
Jos0301c-Missa_Ave_maris_stella-Credo.krn:measure 144:beat 7:(0r) (0G) 4A/ (1b-) 3-2BT07
Jos0301c-Missa_Ave_maris_stella-Credo.krn:measure 156:beat 1:2G\ 2B-\ 2r 4a/] 3-2BT07
Jos0301d-Missa_Ave_maris_stella-Sanctus.krn:measure 5:beat 4:(1d) (0.d) 8e\L (1.ff) 3-2BT02
...
```

```
wc -l abc.txt
```

```
1574
```

Count 3-attack: `grep -v '[]()' abc.txt | wc -l`

Classify by transposition: `sed 's/.*3-2/3-2/' abc.txt | sort | uniq -c`

Classify by metric position: `sed 's/.*beat //; s/.*//' abc.txt | sort -n | uniq -c`