

Guide to Programming with Music Blocks

Music Blocks is a programming environment for children interested in music and graphics. It expands upon Turtle Blocks by adding a collection of features relating to pitch and rhythm.

The Turtle Blocks guide is a good place to start learning about the basics. In this guide, we illustrate the musical features by walking the reader through numerous examples.

TABLE OF CONTENTS

- 1 [Getting Started](#)
- 2 [Making a Sound](#)
 - 2.1 [Note Value Blocks](#)
 - 2.2 [Pitch Blocks](#)
 - 2.3 [Chords](#)
 - 2.4 [Rests](#)
 - 2.5 [Drums](#)
- 3 [Programming with Music](#)
 - 3.1 [Chunks](#)
 - 3.2 [Musical Transformations](#)
 - 3.2.1 [Step Pitch Block](#)
 - 3.2.2 [Sharps and Flats](#)
 - 3.2.3 [Adjust-Transposition Block](#)
 - 3.2.4 [Dotted Notes](#)
 - 3.2.5 [Speeding Up and Slowing Down Notes via Mathematical Operations](#)
 - 3.2.6 [Repeating Notes](#)
 - 3.2.7 [Swinging Notes and Tied Notes](#)
 - 3.2.8 [Set Volume, Crescendo, Staccato, and Slur Blocks](#)
 - 3.2.9 [Intervals and Articulation](#)
 - 3.2.10 [Absolute Intervals](#)
 - 3.2.11 [Inversion](#)
 - 3.2.12 [Backwards](#)
 - 3.2.13 [Setting Voice and Keys](#)
 - 3.2.14 [Vibrato](#)
 - 3.3 [Voices](#)
 - 3.4 [Graphics](#)
 - 3.5 [Interactions](#)
- 4 [Widgets](#)
 - 4.1 [Monitoring status](#)
 - 4.2 [Generating chunks of notes](#)
 - 4.2.1 [Pitch-Time Matrix](#)
 - 4.2.2 [The Rhythm Block](#)
 - 4.2.3 [Creating Tuplets](#)
 - 4.2.4 [What is a Tuplet?](#)
 - 4.2.5 [Using Individual Notes in the Matrix](#)
 - 4.3 [Generating rhythms](#)
 - 4.4 [Musical Modes](#)
 - 4.5 [The Pitch-Drum Matrix](#)
 - 4.6 [Exploring musical proportions](#)
 - 4.7 [Generating arbitrary pitches](#)
 - 4.8 [Changing tempo](#)
- 5 [Beyond Music Blocks](#)

Many of the examples given in the guide have links to code you can run. Look for RUN LIVE links.

1. GETTING STARTED

[Back to Table of Contents](#) | [Next Section \(2. Making a sound\)](#)

Music Blocks is designed to run in a browser. Most of the development has been done in Chrome, but it should also work in Firefox (although you may need to disable hardware acceleration). You can run it from [github io](#) or by downloading a copy of the code and running a local copy directly from the file system of your computer.

For more details on how to use Music Blocks, see [Using Music Blocks](#). For more details on how to use Turtle Blocks, see [Using Turtle Blocks JS](#).

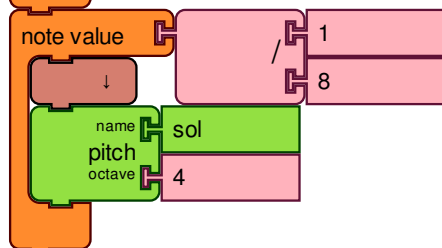
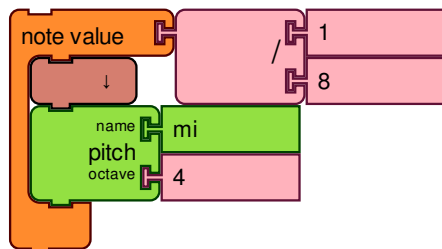
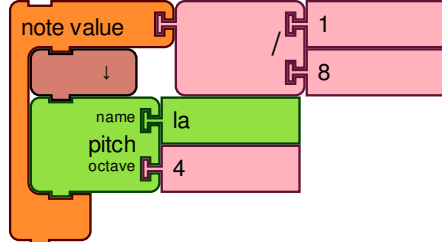
2. MAKING A SOUND

[Previous Section \(1. Getting Started\)](#) | [Back to Table of Contents](#) | [Next Section \(3. Programming with Music\)](#)

Music Blocks incorporates many common elements of music, such as [pitch](#), [rhythm](#), [volume](#), and, to some degree, [timbre and texture](#).

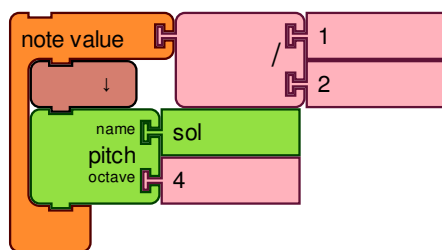
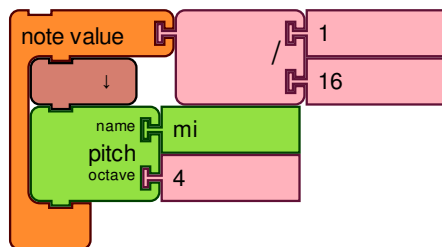
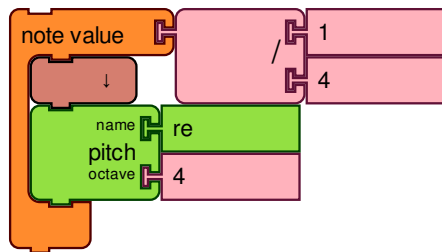
2.1 Note Value Blocks

At the heart of Music Blocks is the *Note value* block. The *Note value* block is a container for a [pitch block](#) that specifies the duration (note value) of the pitch.



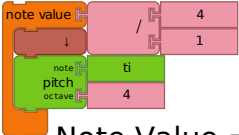

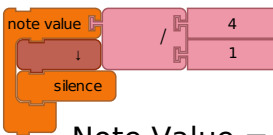
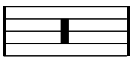
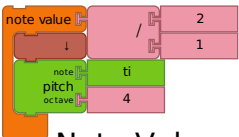

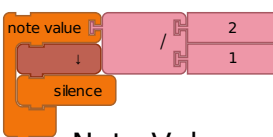

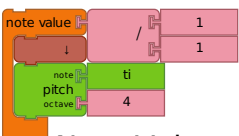
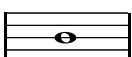
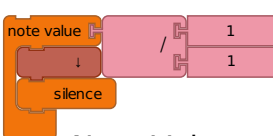
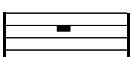
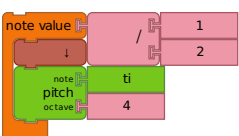
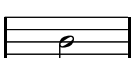
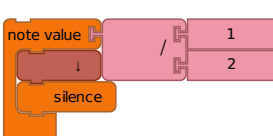

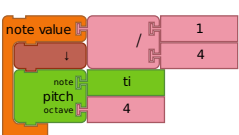
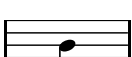
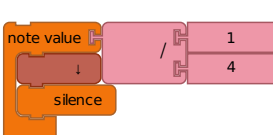

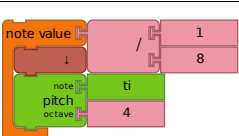

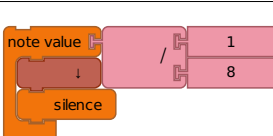
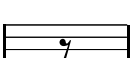
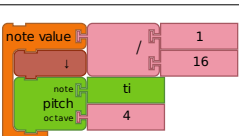
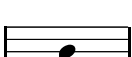
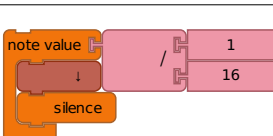
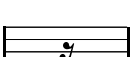
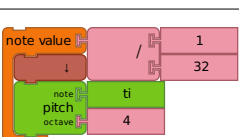
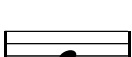
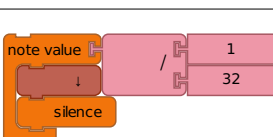
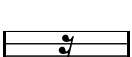
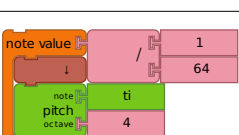
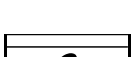
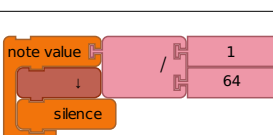

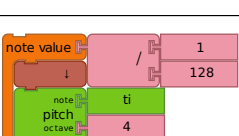

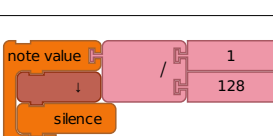

At the top of the example above, a single (detached) *Note value* block is shown. The $1/8$ is value of the note, which is, in this case, an eighth note.

At the bottom, two notes that are played consecutively are shown. They are both $1/8$ notes, making the duration of the entire sequence $1/4$.



In this example, different note values are shown. From top to bottom, they are: $1/4$ for an quarter note, $1/16$ for a sixteenth note, and $1/2$ for a half note.

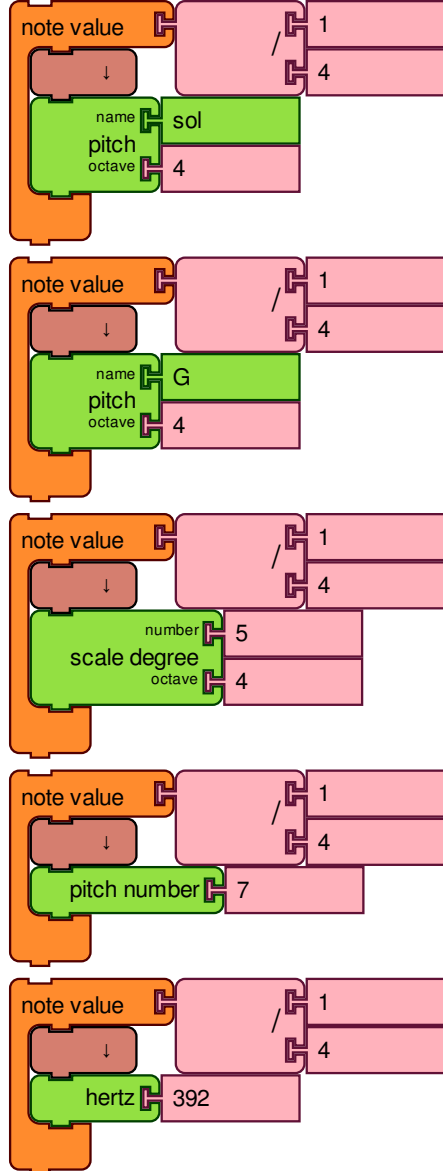
Note that any mathematical operations can be used as input to the *Note value*.

Note Value Blocks	Western Notation (Notes)	Silence Blocks	Western Notation (Rests)
 <p>Note Value = 4/1</p>	 <p>Longa Note</p>	 <p>Note Value = 4/1</p>	 <p>Longa Rest</p>
 <p>Note Value = 2/1</p>	 <p>Breve Note</p>	 <p>Note Value = 2/1</p>	 <p>Breve Rest</p>
 <p>Note Value = 1/1</p>	 <p>Whole Note</p>	 <p>Note Value = 1/1</p>	 <p>Whole Rest</p>
 <p>Note Value = 1/2</p>	 <p>Half Note</p>	 <p>Note Value = 1/2</p>	 <p>Half Rest</p>
 <p>Note Value = 1/4</p>	 <p>Quarter Note</p>	 <p>Note Value = 1/4</p>	 <p>Quarter Rest</p>
 <p>Note Value = 1/8</p>	 <p>Eighth Note</p>	 <p>Note Value = 1/8</p>	 <p>Eighth Rest</p>
 <p>Note Value = 1/16</p>	 <p>Sixteenth Note</p>	 <p>Note Value = 1/16</p>	 <p>Sixteenth Rest</p>
 <p>Note Value = 1/32</p>	 <p>Thirty-second Note</p>	 <p>Note Value = 1/32</p>	 <p>Thirty-second Rest</p>
 <p>Note Value = 1/64</p>	 <p>Sixty-fourth Note</p>	 <p>Note Value = 1/64</p>	 <p>Sixty-fourth Rest</p>
 <p>Note Value = 1/128</p>	 <p>Hundred twenty-eighth Note</p>	 <p>Note Value = 1/128</p>	 <p>Hundred twenty-eighth Rest</p>

Please refer to the above picture for a visual representation of note values.

2.2 Pitch Blocks

As we have seen, *Pitch* blocks are used inside the *Note value* blocks. The *Pitch* block specifies the pitch name and pitch octave of a note that in combination determines the frequency (and therefore pitch) at which the note is played.

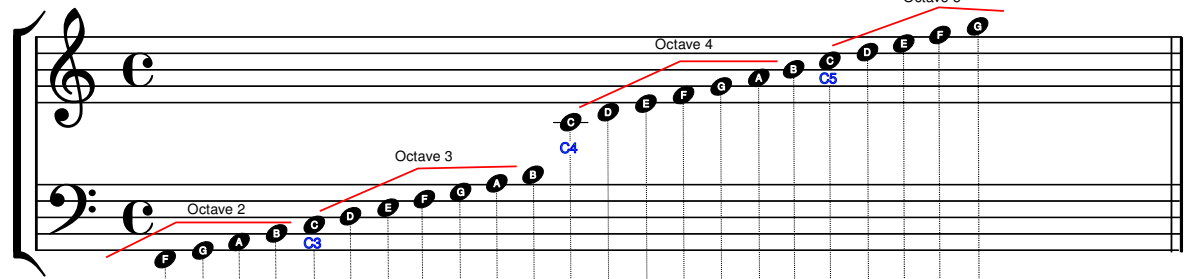


There are many systems you can use to specify a *pitch* block's name and octave. Some examples are shown above. The top pitch block is specified using a *Solfège* block (**Sol** in **Octave 6**), which contains the notes **Do Re Me Fa Sol La Ti** . The middle block is specified using a *Pitch-name* block (**B flat** in **Octave 4**), which contains the notes **C D E F G A B** . The last block is specified using the *Hertz* block in conjunction with a *Number* block (**440** Hertz) , which corresponds to the frequency of the sound made.

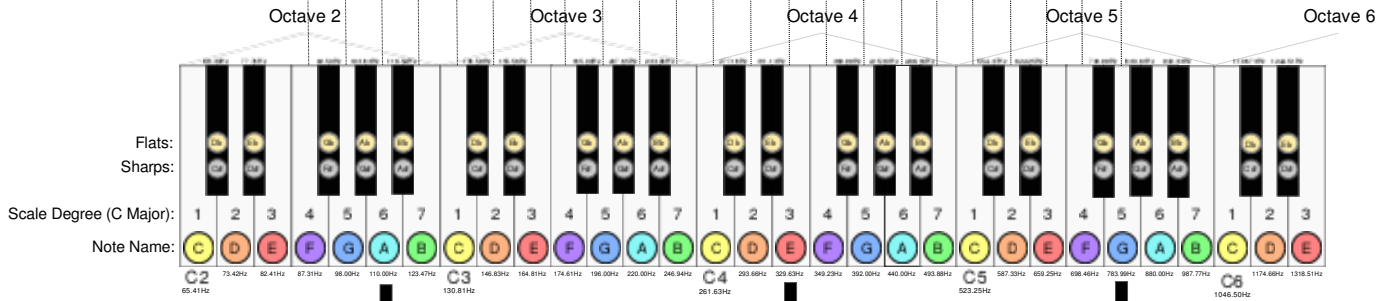
The octave is specified using a number block and is restricted to whole numbers. In the case where the pitch name is specified by frequency, the octave is ignored.

Note that the pitch name can also be specified using a *Text* block.

Staff Notation



Keyboard



Using the *Hertz Block*
1 Hertz = 1 Hz = 1 Cycle per second

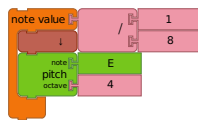


Using the *Pitch Block*

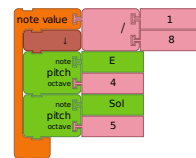


Using the *Solfege Block*

Use Individually to create a *Note*

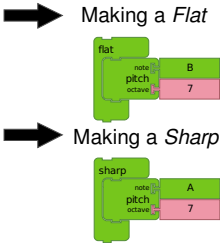
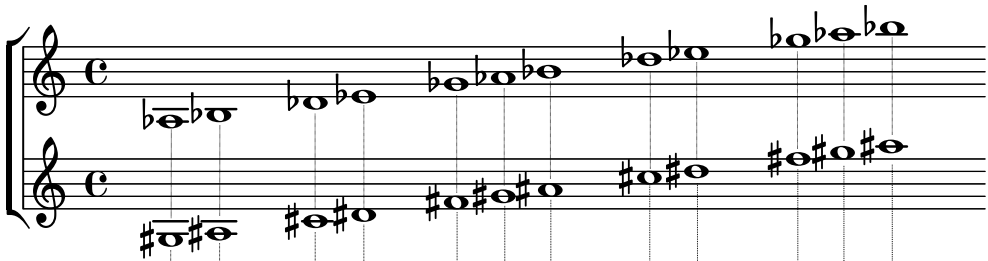


Combine them to create a *Chord*

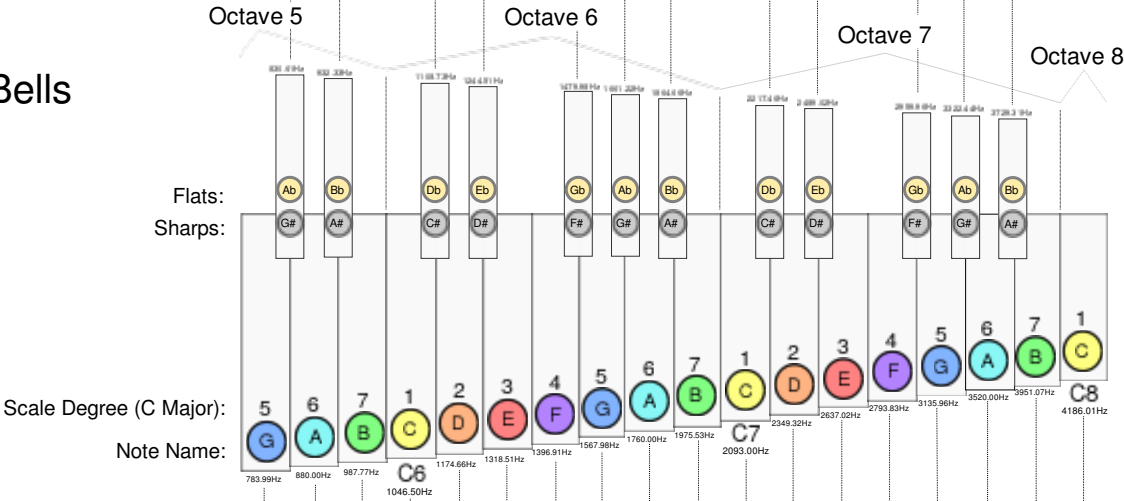


Staff Notation: Flats & Sharps

Mallet sounds are two octaves higher than written

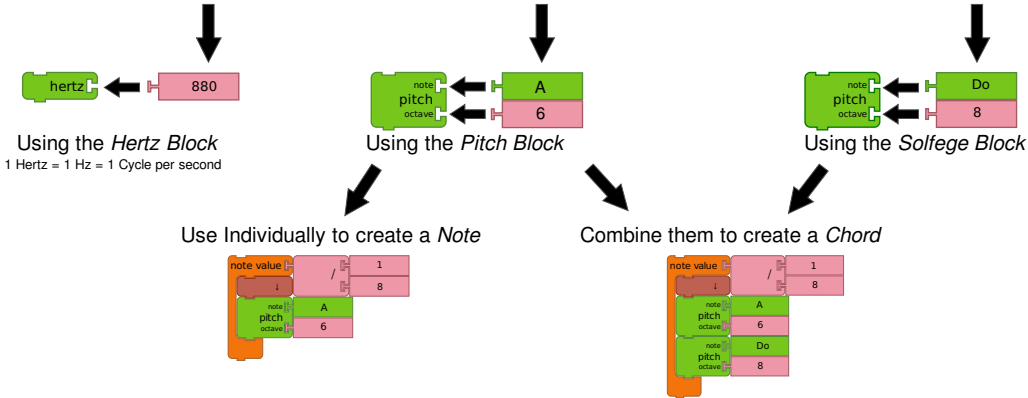
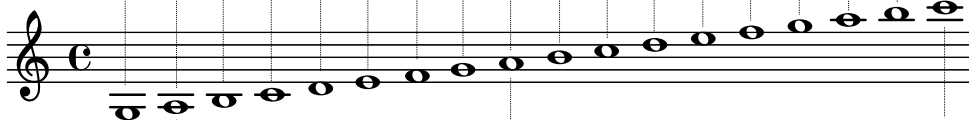


Mallet Bells



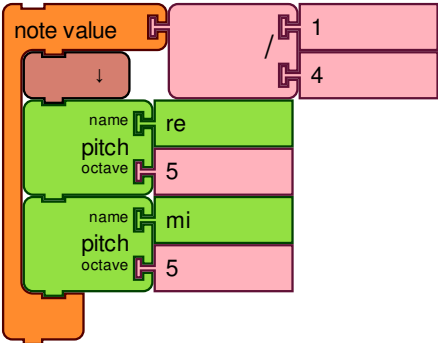
Staff Notation: Naturals

Mallet sounds are two octaves higher than written



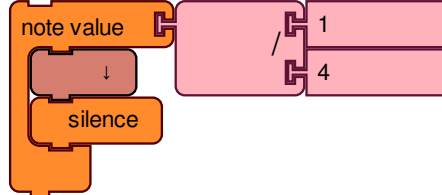
Please refer to the above charts for a visual representation of where notes are located on a keyboard or staff.

2.3 Chords



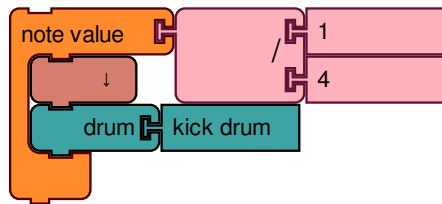
A chord (multiple, simultaneous pitches) can be specified by adding multiple Pitch blocks into a single Note value block, like the above example.

2.4 Rests

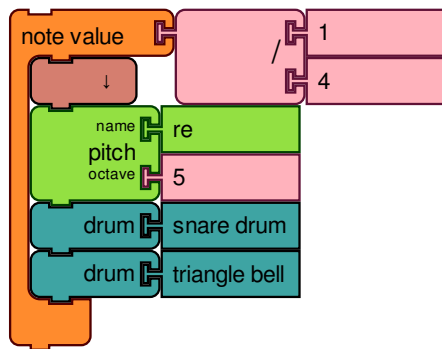


A rest of the specified note value duration can be constructed using a *Silence* block in place of a *pitch* block.

2.5 Drums



Anywhere a *Pitch* block can be used—e.g., inside of the matrix or a *Note value* block—a *Drum Sample* block can also be used instead. Currently there about two dozen different samples from which to choose. The default drum is a kick drum.



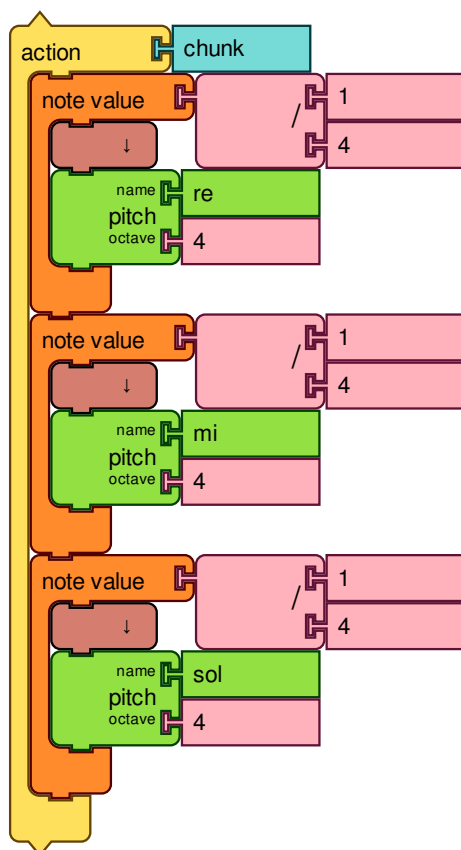
Just as in the [chord](#) example above, you can use multiple *Drum* blocks within a single *Note value* blocks, and combine them with *Pitch* blocks as well.

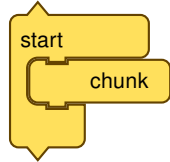
3. PROGRAMMING WITH MUSIC

[Previous Section \(2. Making a Sound\)](#) | [Back to Table of Contents](#) | [Next Section \(4. Widgets\)](#)

This section of the guide discusses how to use chunks of notes to rogram music. Note that you can program with chunks you create by hand or use the [Pitch-time Matrix](#) widget to help you get started.

3.1 Chunks



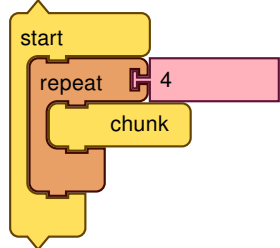
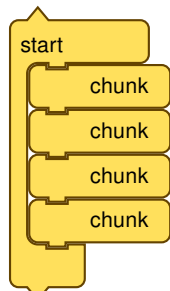


Every time you create a new *Action* stack, Music Blocks creates a new block specific to, and linked with, that stack. (The new block is found at the top of the *Block* palette, found on the left edge of the screen.) Clicking on and running this block is the same as clicking on your stack. By default, the new blocks are named *chunk*, *chunk1*, *chunk2* ... but you can rename them by editing the labels on the *Action* blocks.

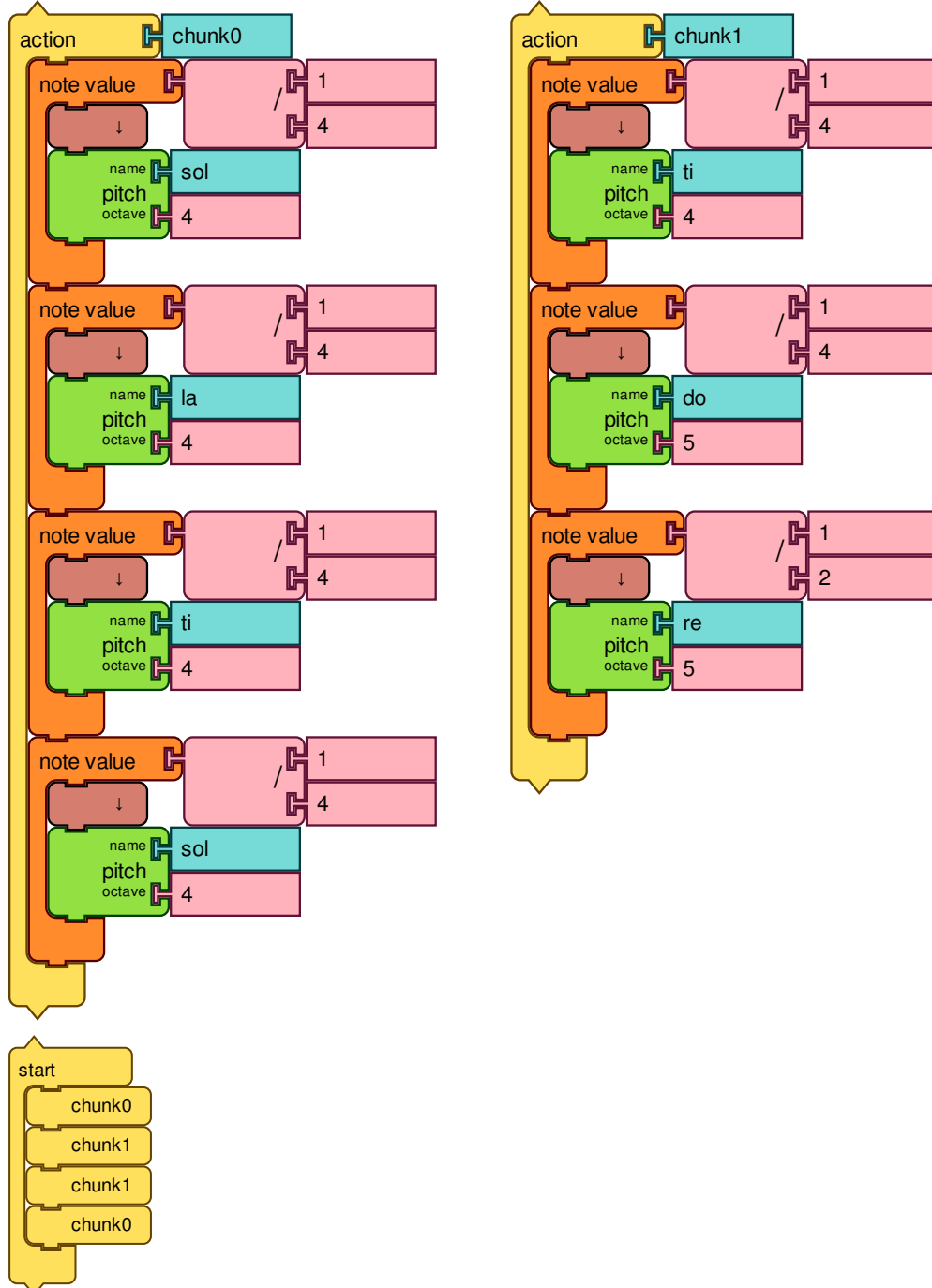
An *Action* block contains a sequence of actions that will only be executed when the block is referred to by something else, such as a start block. This is useful in orchestrating more complex programs of music.

A *Start* Block is a *chunk* that will automatically be executed once the start button is pressed. This is where most of your programs will begin at. There are many ways to *Run* a program: you can click on the *Run* button at the upper-left corner of the screen (the "rabbit") to run the music at a fast speed; click on the *Run Slow* button (the "turtle") to run it slower; and the *Step* button (the "snail"), to step through the program one block per button press.

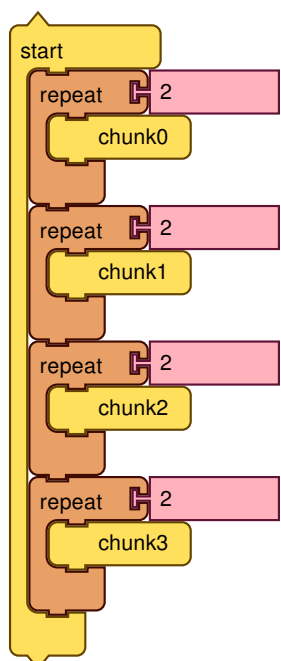
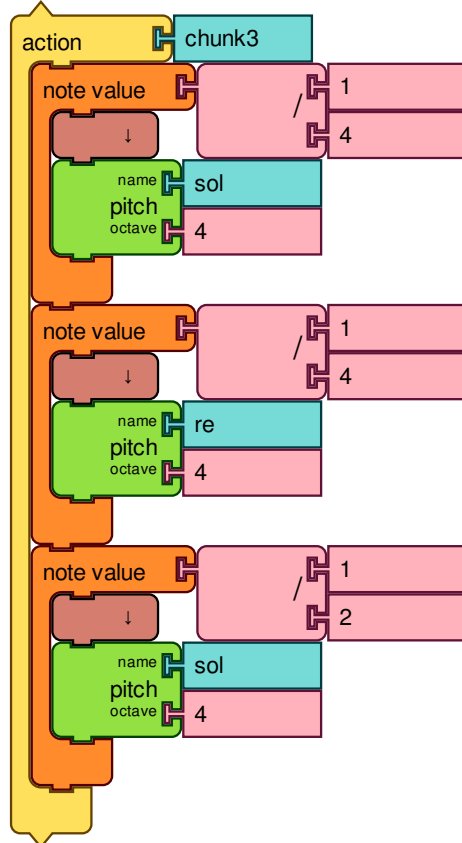
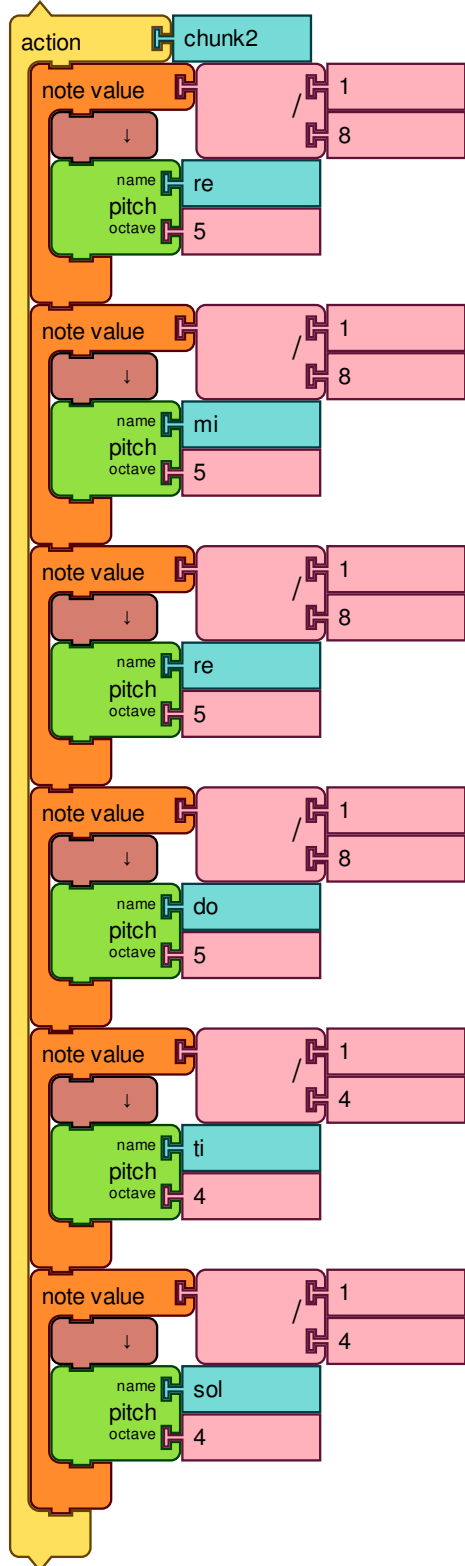
In the example above, the *Chunk* block is inside of a *Start* block, which means that when any of the start buttons is pressed, the code inside the *Start* block (the *Chunk* block) will be executed. You can add more chunks after this one inside the *Start* block to execute them sequentially.



You can [repeat](#) chunks either by using multiple *Chunk* blocks or using a *Repeat* block.



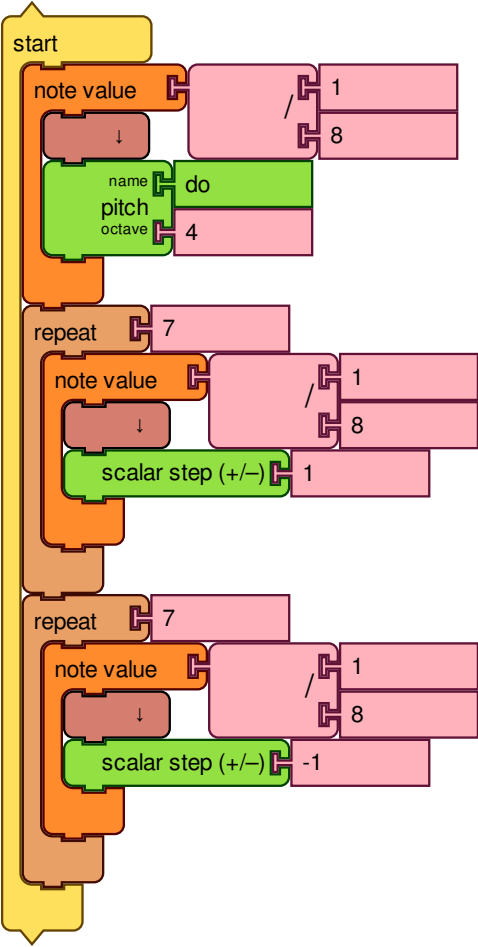
You can also mix and match chunks. Here we play the action block with name "chunk", followed by "chunk1" twice, and then "chunk" again.



3.2 Musical Transformations

There are many ways to transform pitch, rhythm, and other sonic qualities.

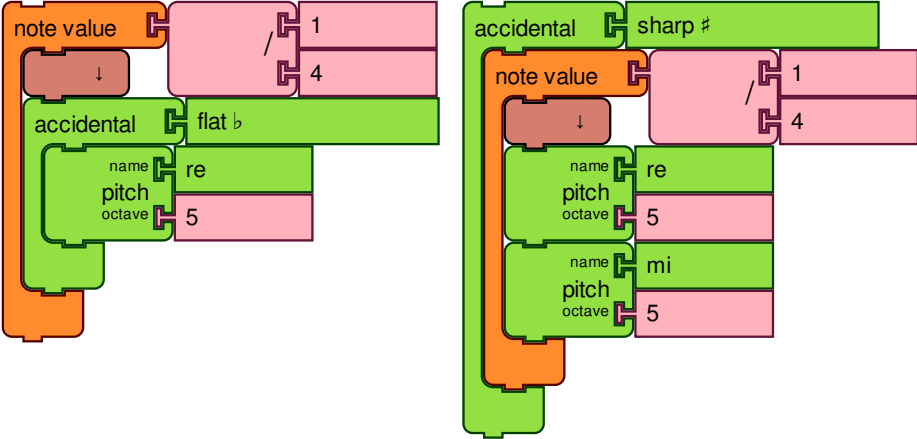
3.2.1 Step Pitch Block



The *Step Pitch* block will move up or down notes in a scale from the last played note. In the example above, *Step Pitch* blocks are used inside of *Repeat* blocks to repeat the code 7 times, playing up and down a scale.

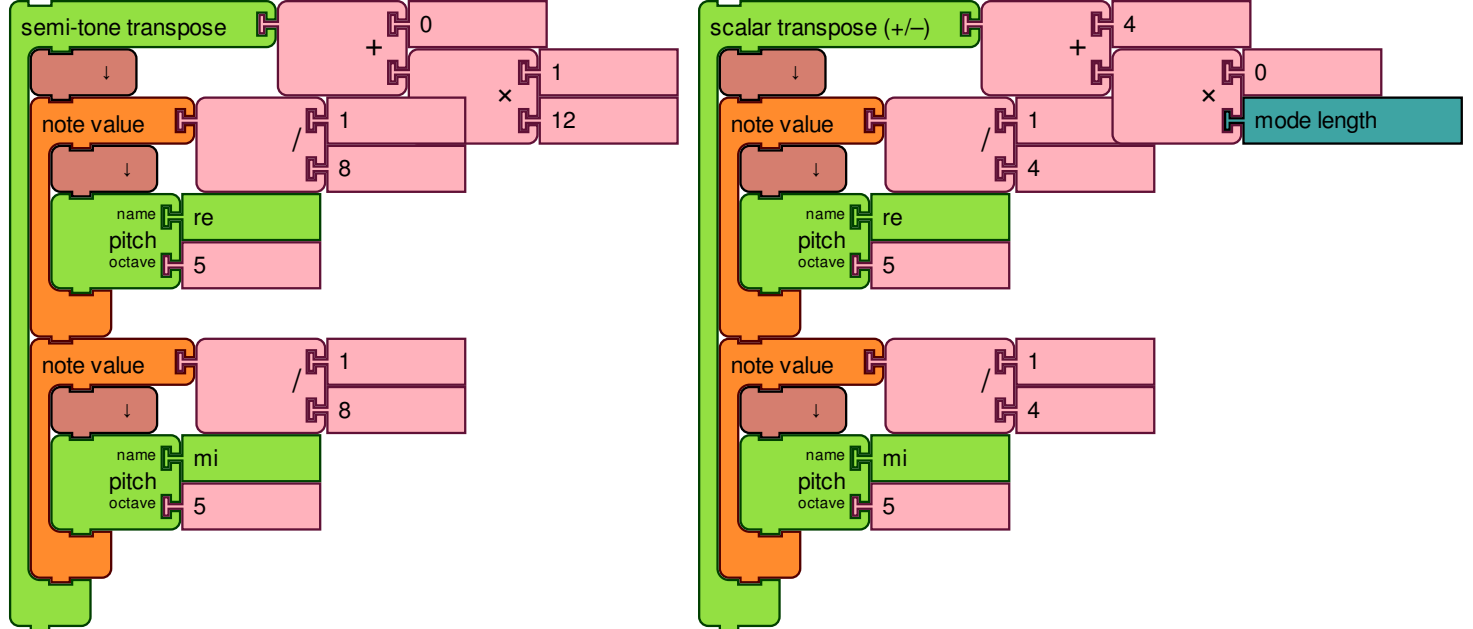
[RUN LIVE](#)

3.2.2 Sharps And Flats

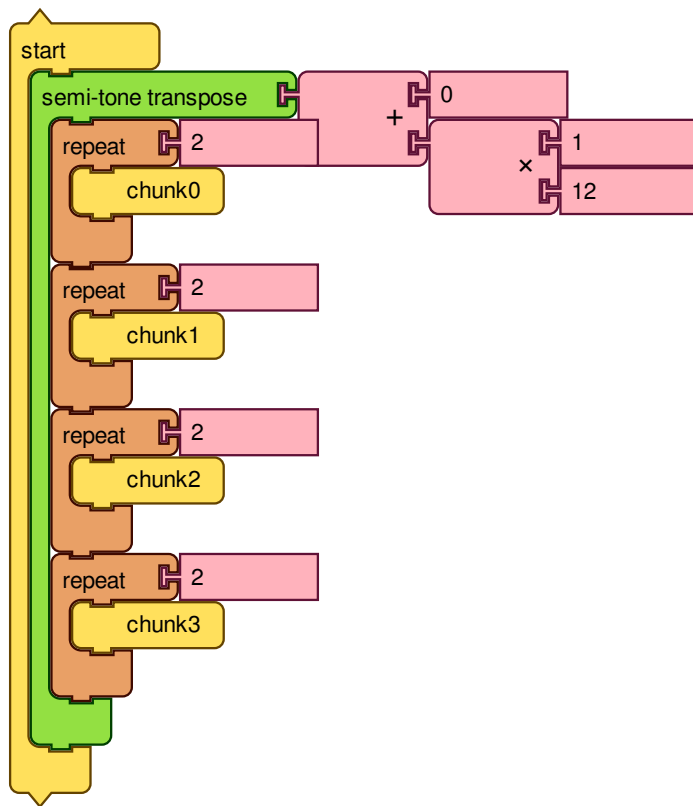


The *Sharp* and *Flat* blocks can be wrapped around *Pitch* blocks, *Note value* blocks, or *chunks*. A sharp will raise the pitch by one half step. A flat will lower by one half step. In the example, on the left, just the *Pitch* block **Mi** is lowered by one half step; on the right, both pitch blocks are raised by one half step.

3.2.3 Adjust-Transposition

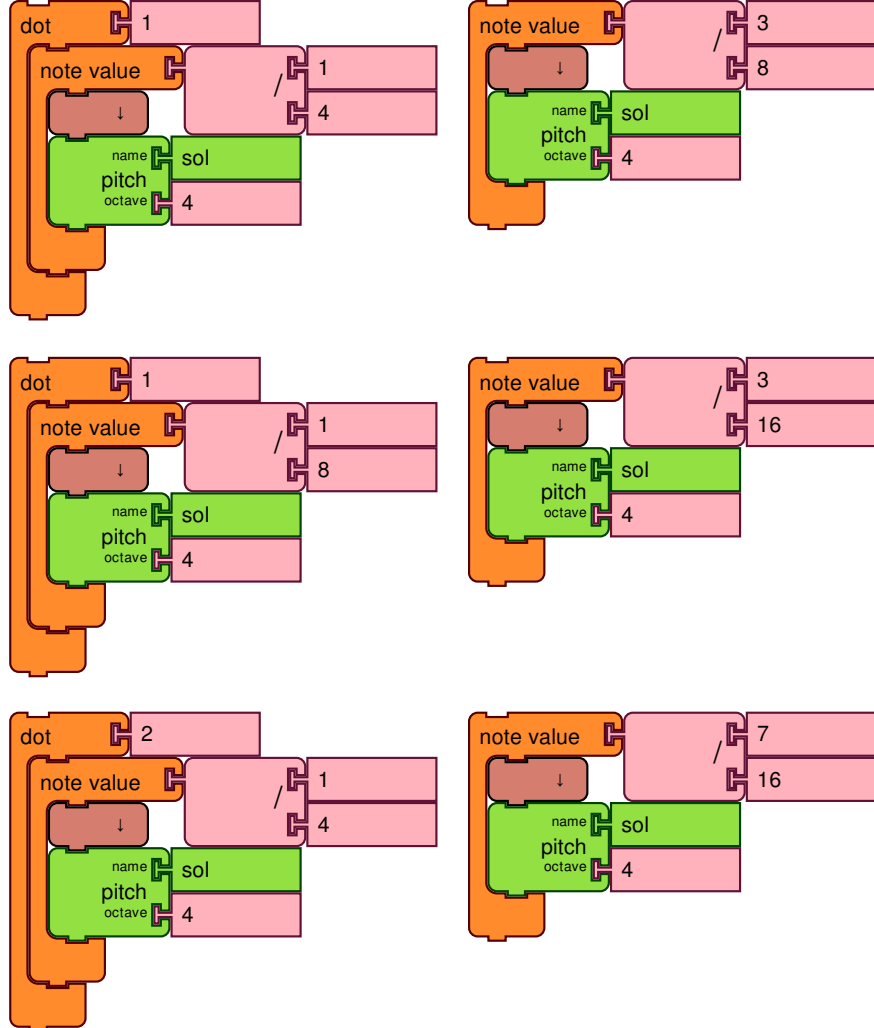


The *Adjust-transposition* block can be used to make larger shifts in pitch in half step units. A positive number shifts the pitch up and a negative number shifts the pitch down. The input must be a whole number. To shift an entire octave, transpose by 12 half-steps up. -12 will shift an octave down.



In the example above, we take the song we programmed previously and raise it by one octave.

3.2.4 Dotted Notes



You can "dot" notes using the *Dot* block. A dotted note extends the rhythmic duration of a note by 50%. E.g., a dotted quarter note will play for 3/8 ($1/4 + 1/8$) of a beat. A dotted eighth note will play for 3/16 ($1/8 + 1/16$) of a beat.

You can also simply change the note value to mimic a dotted note, for example indicating 3/8 instead of 1/4, for a dotted quarter note.


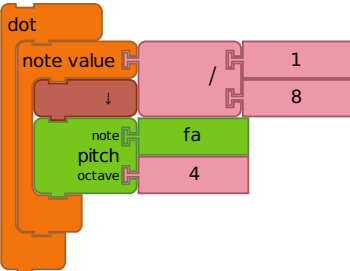
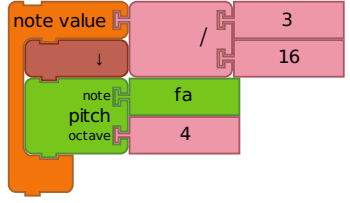
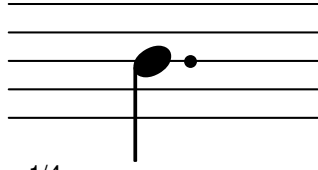
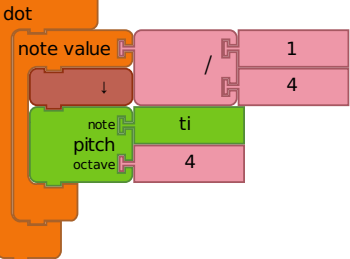
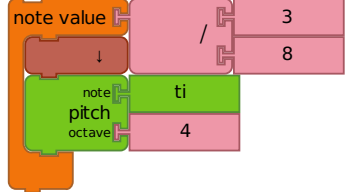
Using Dotted Notes

The dot increases the value of a note by half of its value.

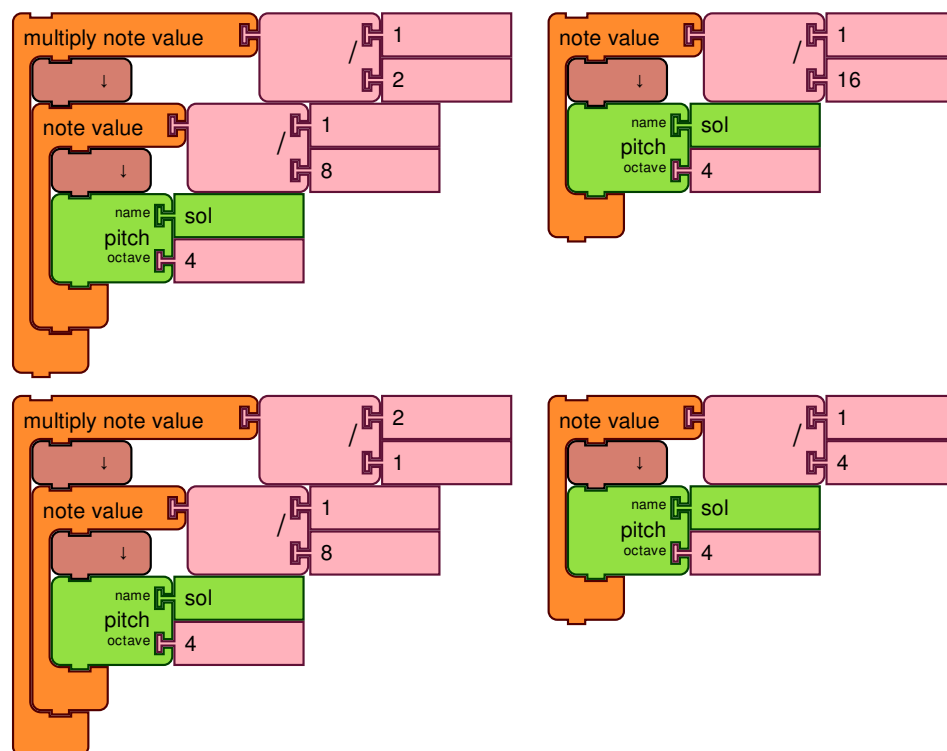
x = value of note

$$\text{Formula: } x + \frac{x}{2} = \text{value of dotted note}$$

Examples:

Western Notation	Music Blocks Notation with dot	Music Block Notation without dot
 <p>For x = 1/8,</p> $\frac{1}{8} + \frac{1}{(8 \cdot 2)} = \frac{1}{8} + \frac{1}{16} = \frac{2}{16} + \frac{1}{16} = \frac{3}{16}$		
 <p>For x = 1/4,</p> $\frac{1}{4} + \frac{1}{(4 \cdot 2)} = \frac{1}{4} + \frac{1}{8} = \frac{2}{8} + \frac{1}{8} = \frac{3}{8}$		

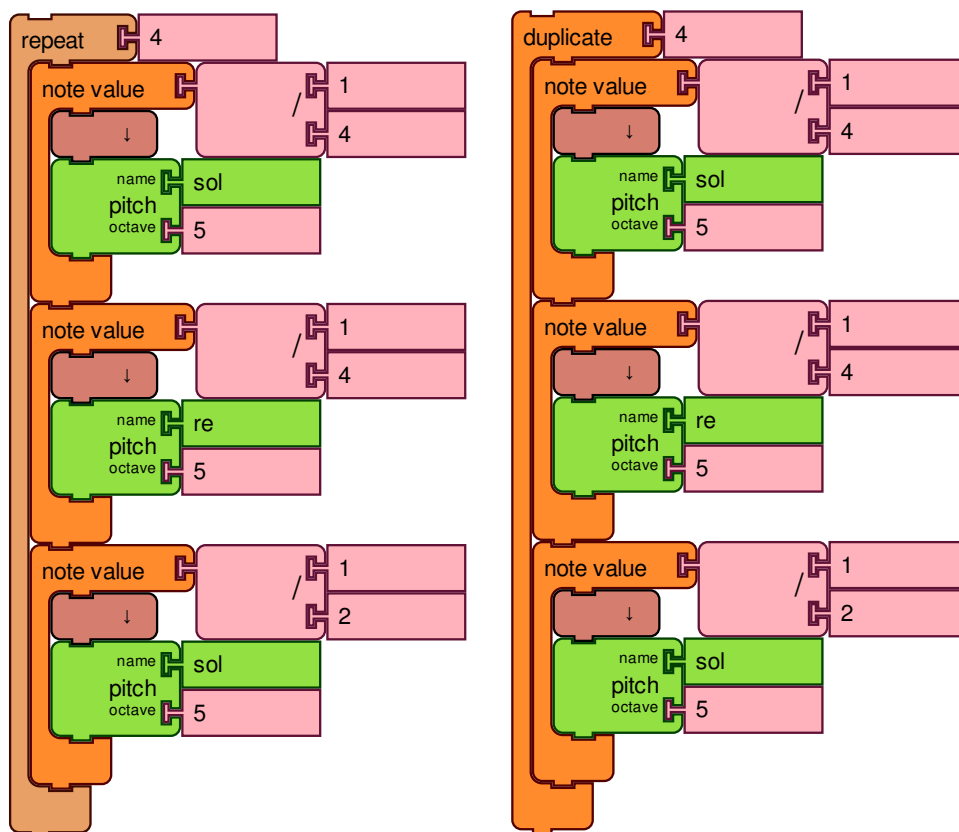
3.2.5 Speeding Up and Slowing Down Notes via Mathematical Operations



You can also multiply (or divide) the beat value, which will speed up or slowdown the notes. Multiplying the beat value of an 1/8 note by 2 is the equivalent of

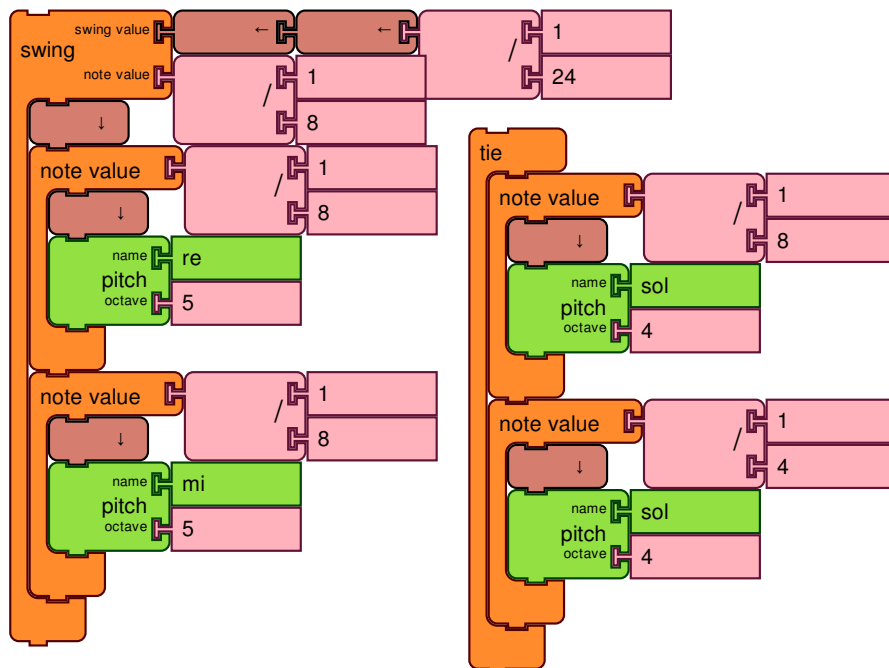
playing a $\frac{1}{16}$ note. Dividing the beat value of an $\frac{1}{8}$ note by '2' is the equivalent of playing a $\frac{1}{4}$ note.

3.2.6 Repeating Notes



There are several ways to repeat notes. The *Repeat* block will play a sequence of notes multiple times; the *Duplicate* block will repeat each note in a sequence. In the example, on the left, the result would be **Sol, Re, Sol, Sol, Re, Sol, Sol, Re, Sol, Sol, Re, Sol** ; on the right the result would be **Sol, Sol, Sol, Sol, Re, Re, Re, Re, Sol, Sol, Sol, Sol** .

3.2.7 Swinging Notes and Tied Notes



The *Swing* block works on pairs of notes (specified by note value), adding some duration (specified by swing value) to the first note and taking the same amount from the second note. Notes that do not match note value are unchanged. In the example, **re5** would be played as a $\frac{1}{6}$ note and **mi5** would be played as a $\frac{1}{12}$ note ($\frac{1}{8} + \frac{1}{24} === \frac{1}{6}$ and $\frac{1}{8} - \frac{1}{24} === \frac{1}{12}$). Observe that the total duration of the pair of notes is unchanged. Tie also works on pairs of notes, combining them into one note. (The notes must be identical in pitch, but can vary in rhythm.)

Using Notes with Ties

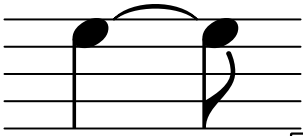
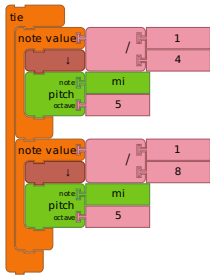
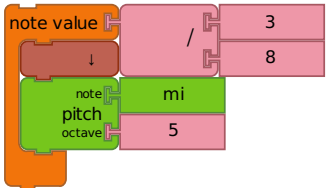

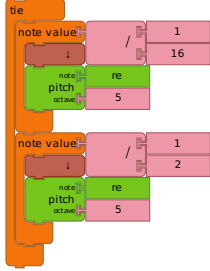
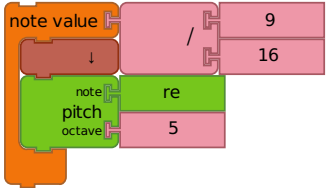
A tie connects two notes of the same pitch* and indicates that they are to be played as the sum of the two notes.

x = value of note 1

y = value of note 2

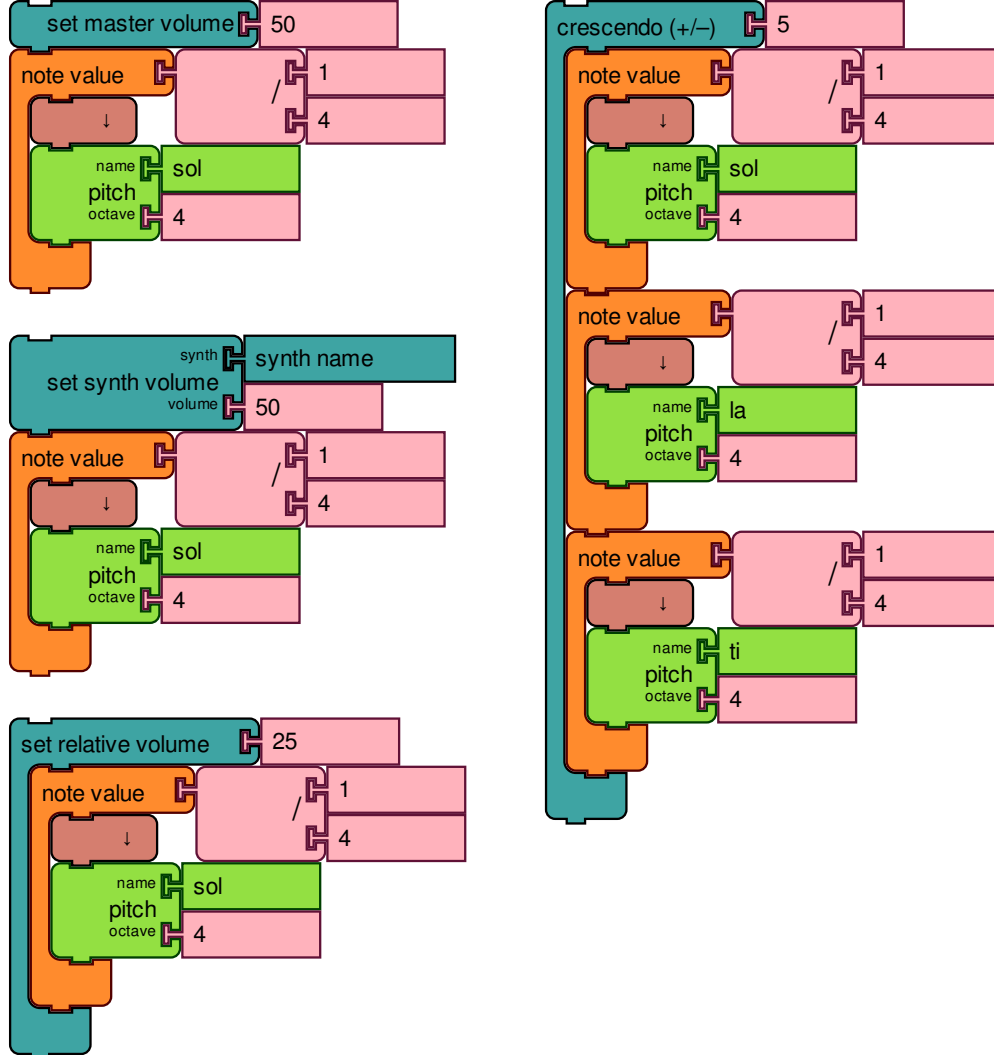
Formula: $x + y =$ total value of notes contained within tie

Examples:

Western Notation	Music Blocks Notation with tie	Music Block Notation without tie
 $\frac{1}{4} + \frac{1}{8} = \frac{3}{8}$ <p>Find common denominator: $x = \frac{1}{4}$ $y = \frac{1}{8}$ $2 * \frac{1}{4} = \frac{2}{8} + \frac{1}{8} = \frac{3}{8}$</p>		
 $\frac{1}{16} + \frac{1}{2} = \frac{9}{16}$ <p>Find common denominator: $x = \frac{1}{16}$ $y = \frac{1}{2}$ $8 * \frac{1}{2} = \frac{8}{16} + \frac{1}{16} = \frac{9}{16}$</p>		

* Ties affect rhythm, not pitch. For tie to work, both pitches must be exactly the same. If not, it will be considered a slur.

3.2.8 Set Volume, Crescendo, Staccato, and Slur



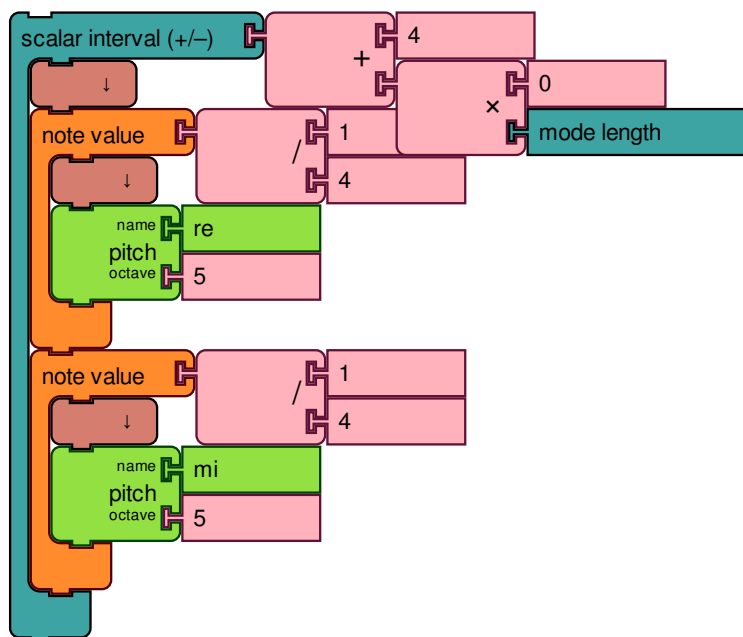
The *Set volume* block will change the volume of the notes. The default is 50; the range is 0 (silence) to 100 (full volume).

The *Crescendo* block will increase (or decrease) the volume of the contained notes by a specified amount for every note played. For example, if you have 3 notes in sequence contained in a *Crescendo* block with a value of 5, the final note will be at 15% more than the original value for volume.

The *Staccato* block shortens the length of the actual note—making them tighter bursts—while maintaining the specified rhythmic value of the notes.

The *Slur* block lengthens the sustain of notes—running longer than the noted duration and blending it into the next note—while maintaining the specified rhythmic value of the notes.

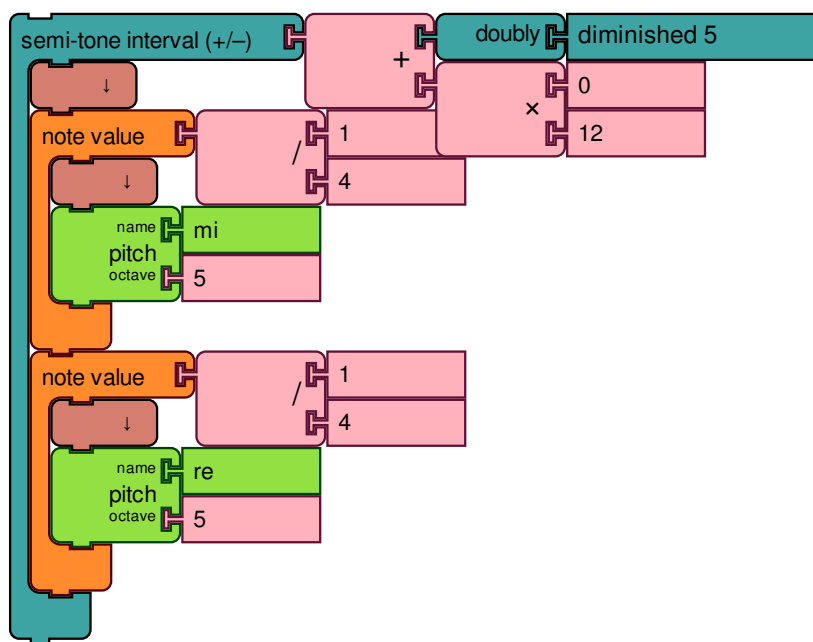
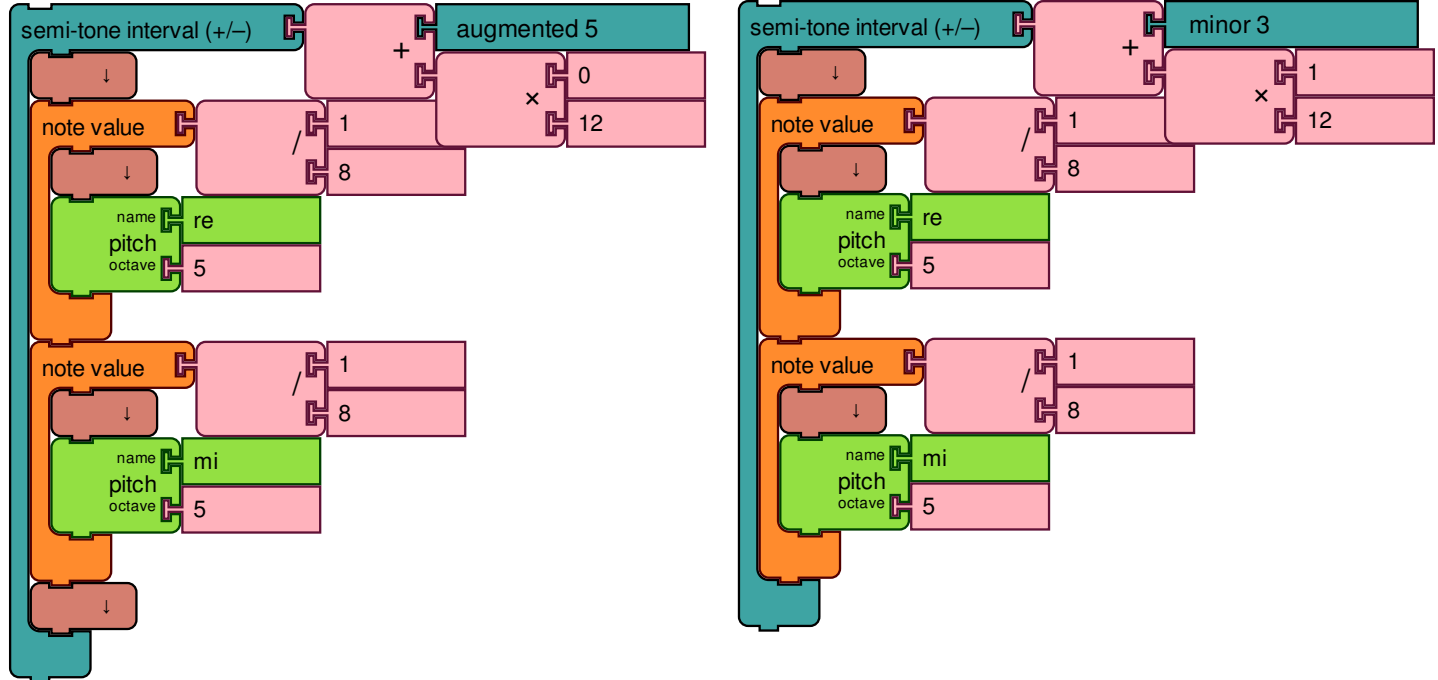
3.2.9 Intervals and Articulation



The *Interval* block calculates a relative interval, e.g., a fifth, and adds the additional pitches to a note's playback. In the figure, we add **La** to **Re** and **Ti** to **Mi**.

The *Articulation* block changes the volume of a group of notes without affecting the master volume for the rest of the user's Music Blocks code.

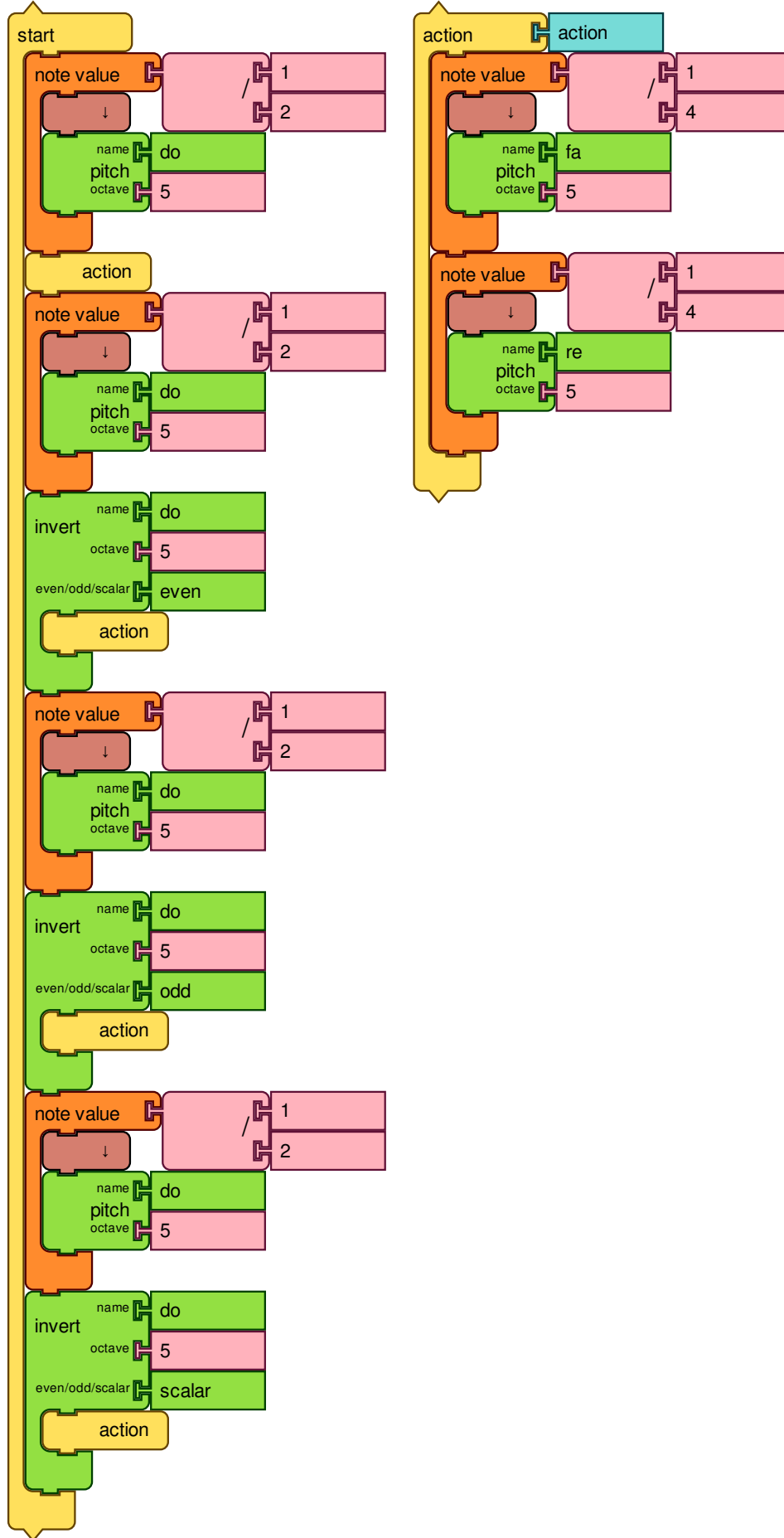
3.2.10 Absolute Intervals



The *Augmented* block calculates an absolute interval, e.g., an augmented fifth, and adds the additional pitches to a note. Similarly, the *Minor* block calculates an absolute interval, e.g., a minor third. Other absolute intervals include *Perfect*, *Diminished*, and *Major*.

In the augmented fifth example above, a chord of D5 and A5 are played, followed by a chord of E5 and C5. In the minor third example, which includes a shift of one octave, first a chord of D5 and F5 is played, followed by chord of E5 and G6.

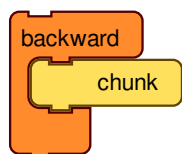
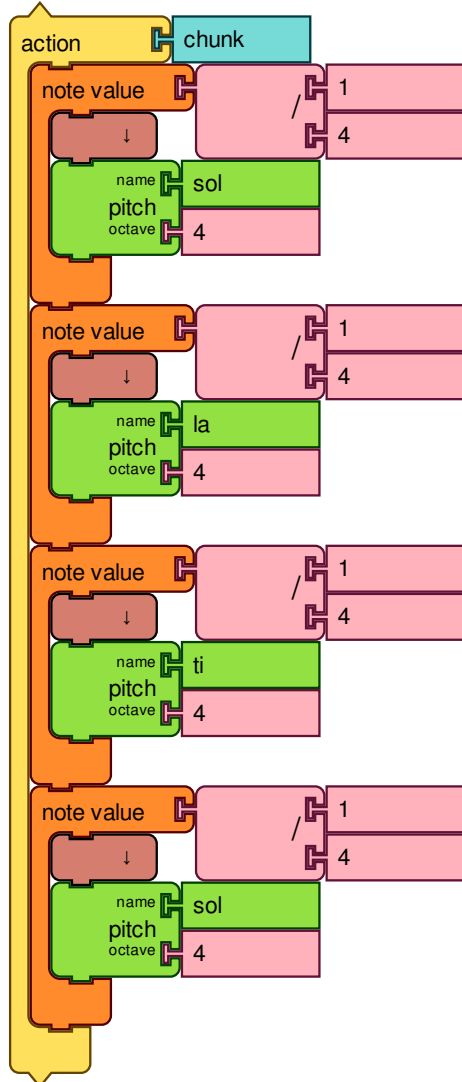
3.2.11 Inversion



The *Invert* block will invert a series of notes around a target note. There are two different modes of the *Invert* block: *odd* and *even*, the latter shifts the point of rotation up by a $\frac{1}{4}$ step, enabling rotation around a point between two notes.

In the *invert (even)* example, **D4** is inverted around **G4**, resulting in a **C5**. In the *invert (odd)* example, **D4** is inverted around a point midway between **G4** and **G#4** resulting in a **C#5**

3.2.12 Backwards

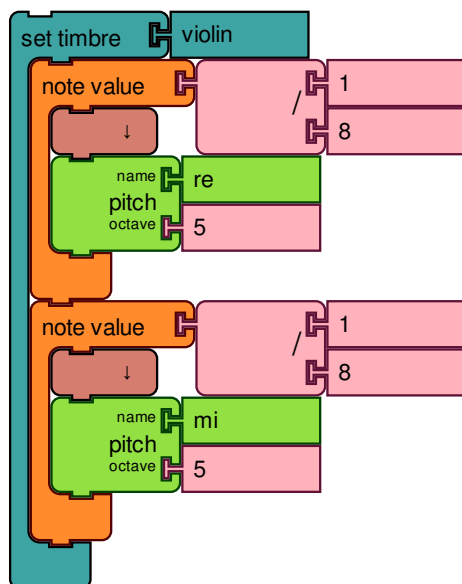


The *Backward* block will play the contained notes in reverse order (retrograde). In the example above, the notes in *Chunk* are played as Sol , Ti , La , Sol , i.e., from the bottom to the top of the stack.

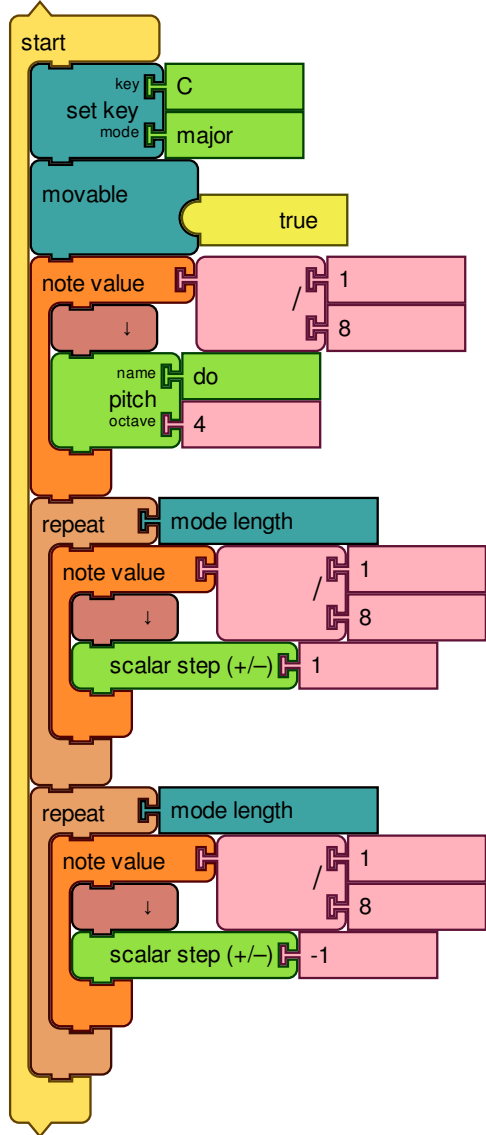
[RUN LIVE](#)

Note that all of the blocks inside a *Backward* block are reverse, so use this feature with caution if you include logic intermixed with notes.

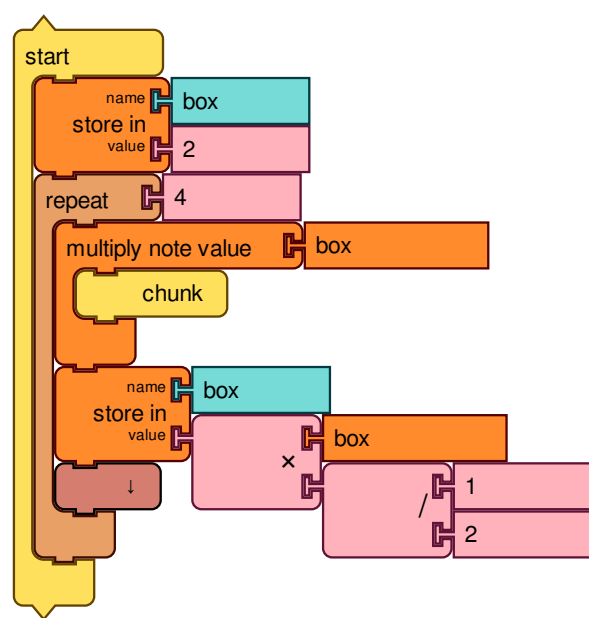
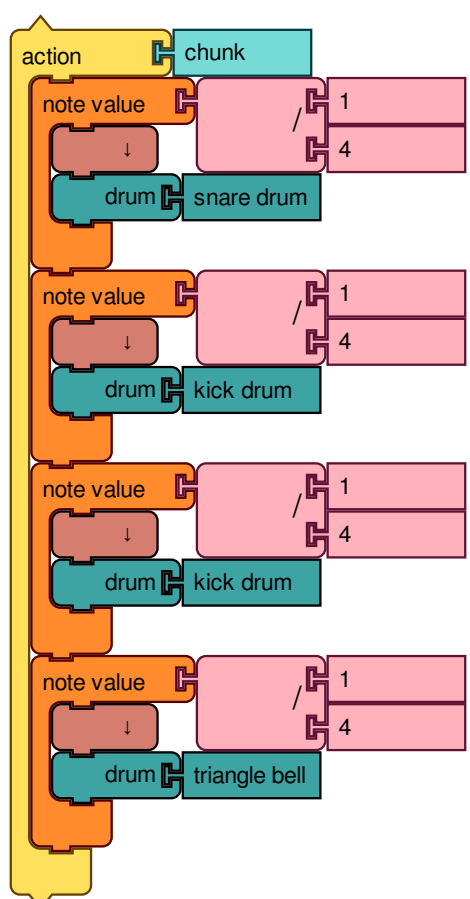
3.2.13 Setting Voice and Keys



The *Set Voice* block selects a [voice](#) for the synthesizer for any contained blocks, e.g., violin or cello.

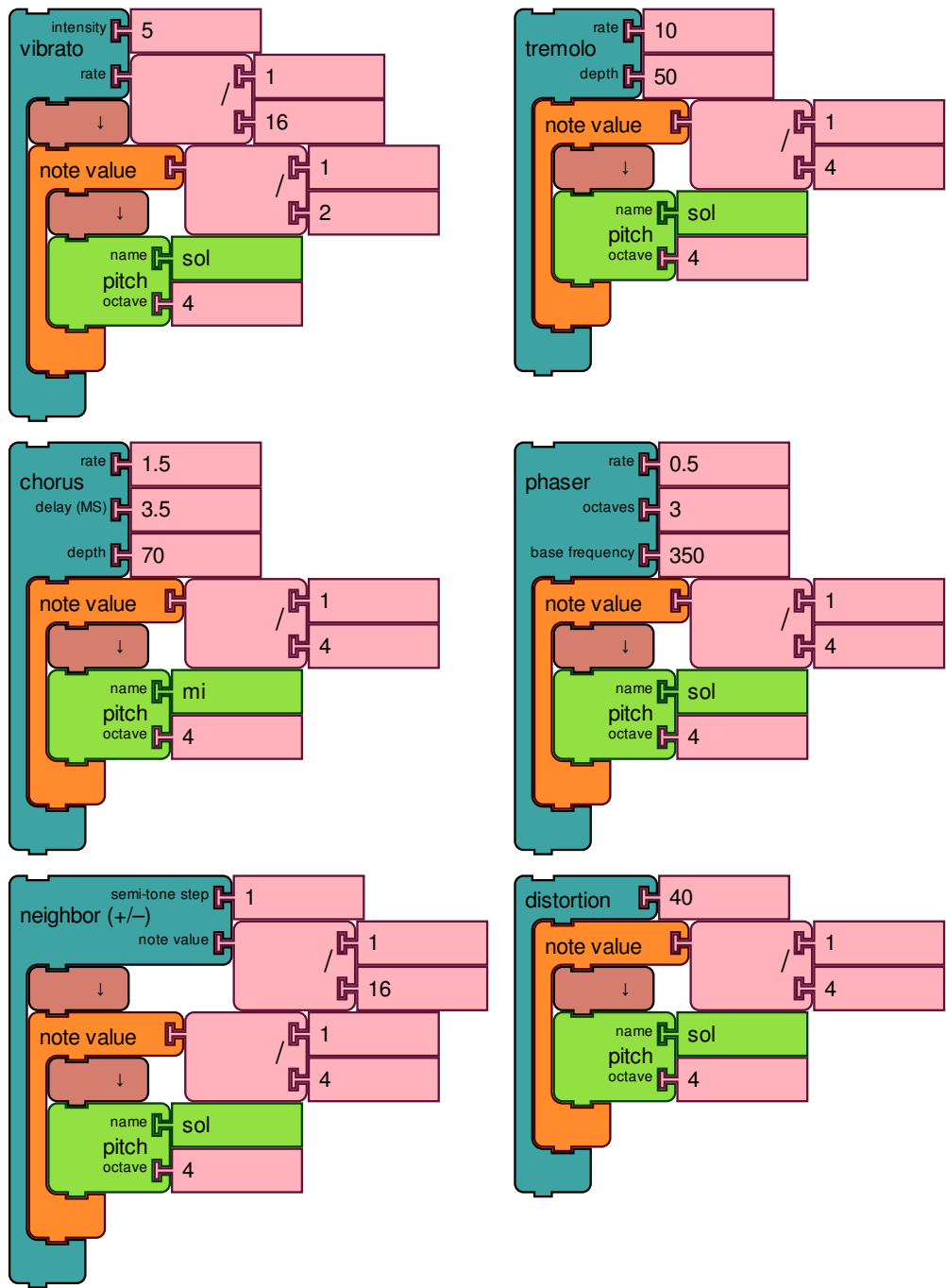


The **Set Key** block will change the key and mode of the mapping between solfege, e.g., **Do**, **Re**, **Mi**, to note names, e.g., **C**, **D**, **E**, when in C Major. Modes include Major and Minor, Chromatic, and a number of more exotic modes, such as Bebop, Geez, Maqam, etc. This block allows users to access "movable Do" within Music Blocks, where the mapping of solfege to particular pitch changes depending on the user's specified tonality.



In the above example, the sequence of **drum** beats is increased over time.

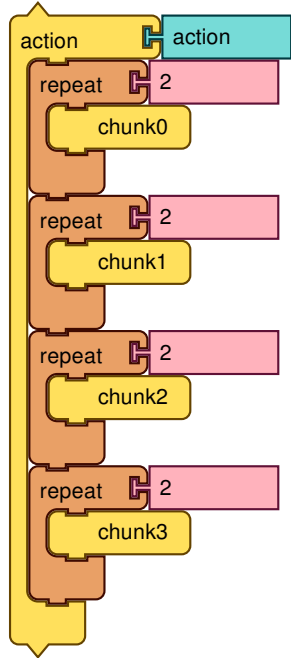
3.2.14 Vibrato



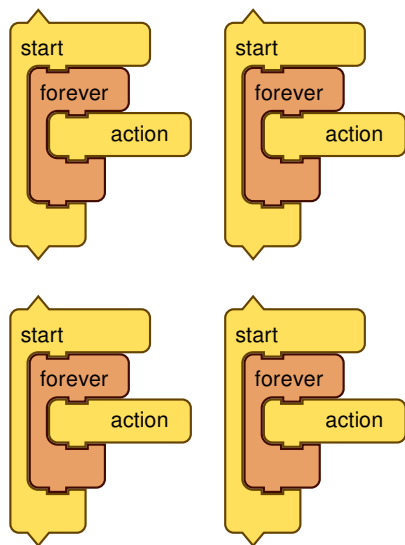
The *Vibrato* Block adds a rapid variation in pitch to any contained notes. The intensity of the variation ranges from 1 to 100 (cents), e.g. plus or minus up to one half step. The rate argument determines the rate of the variation.

3.3 Voices

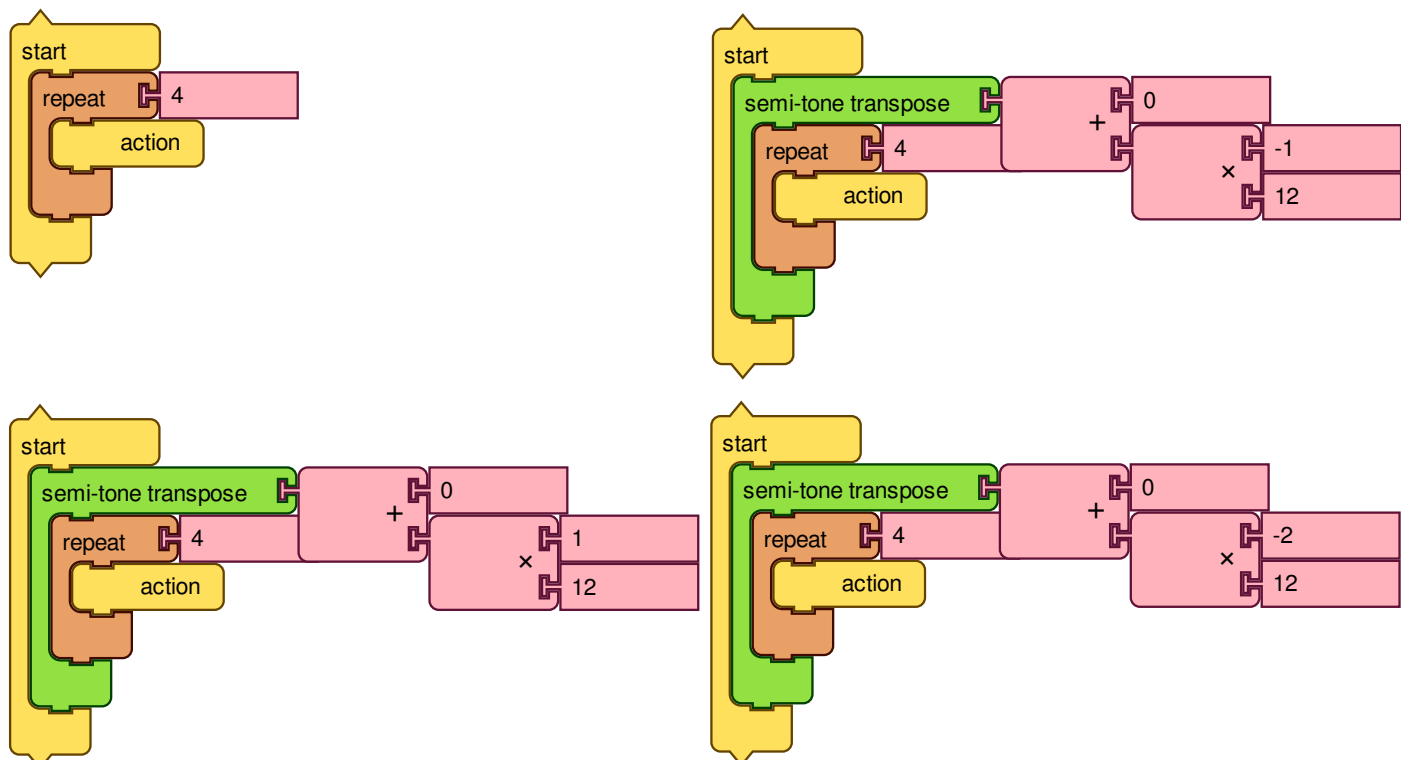
Each *Start* block runs as a separate voice in Music Blocks. (When you click on the Run button, all of the *Start* blocks are run concurrently.)



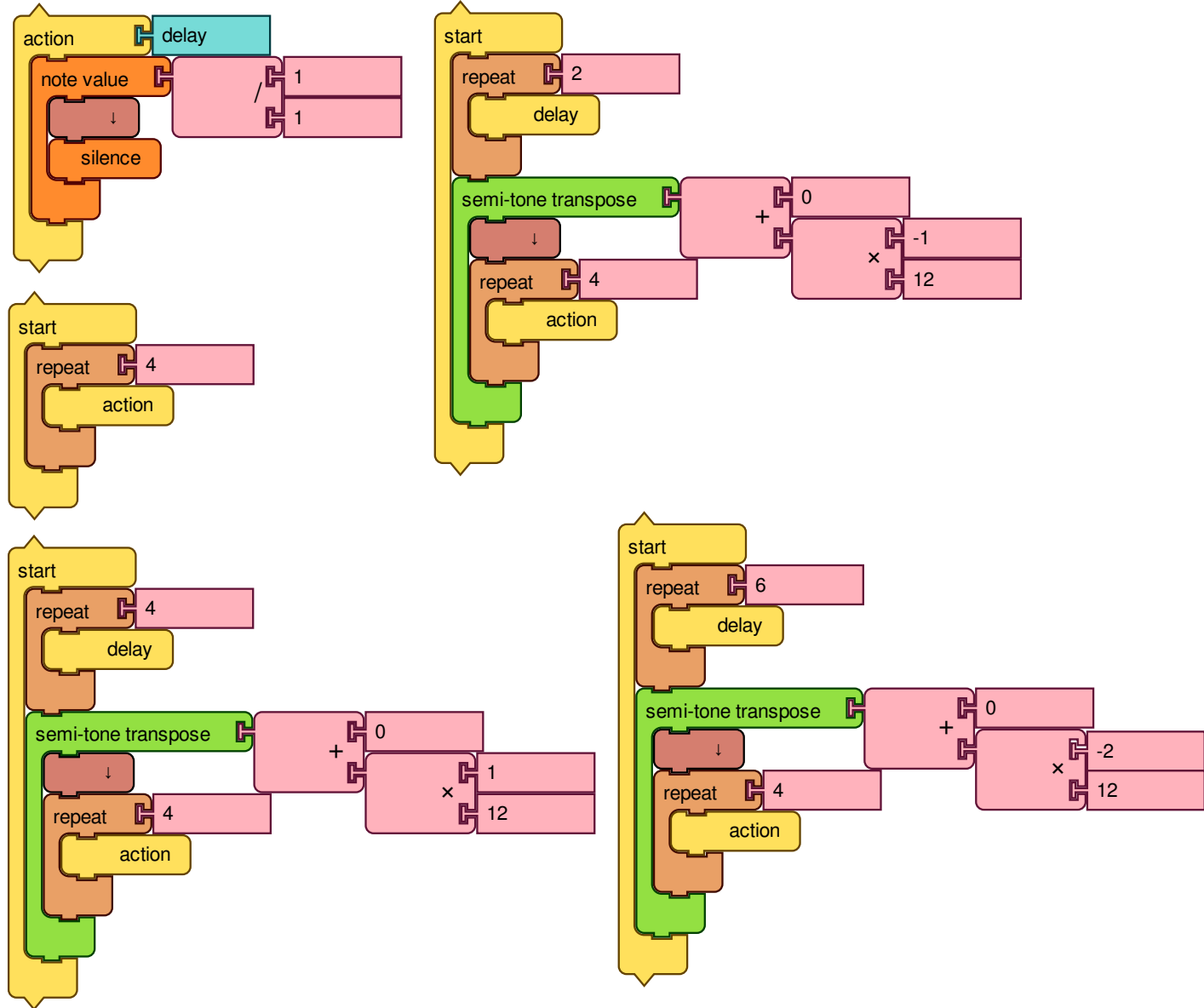
If we put our song into an action...



...we can run it from multiple *Start* blocks.

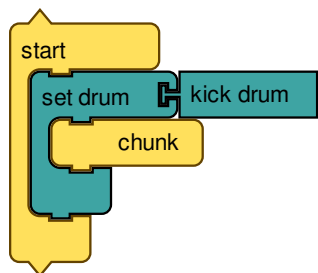


It gets more interesting if we shift up and down octaves.



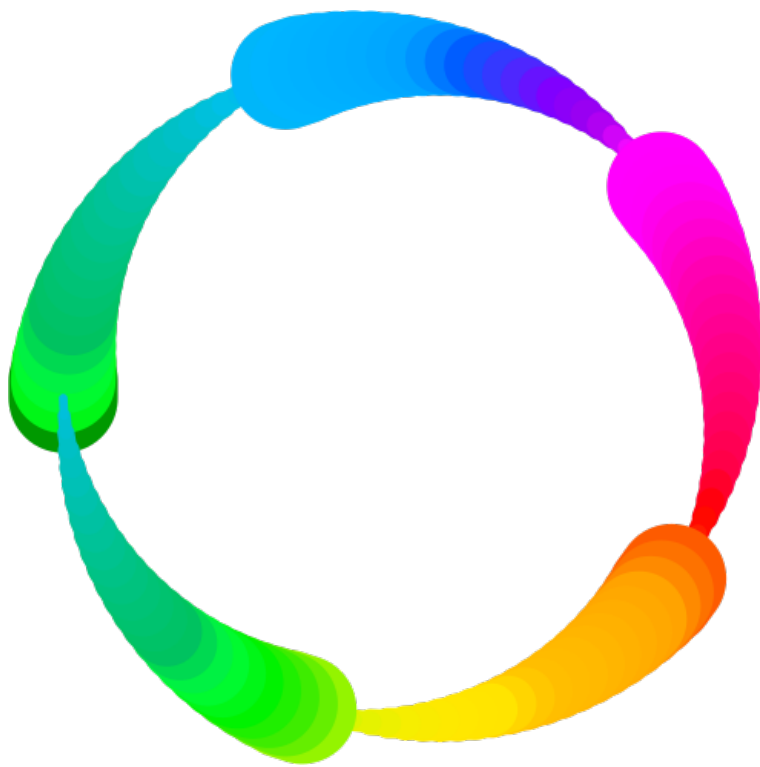
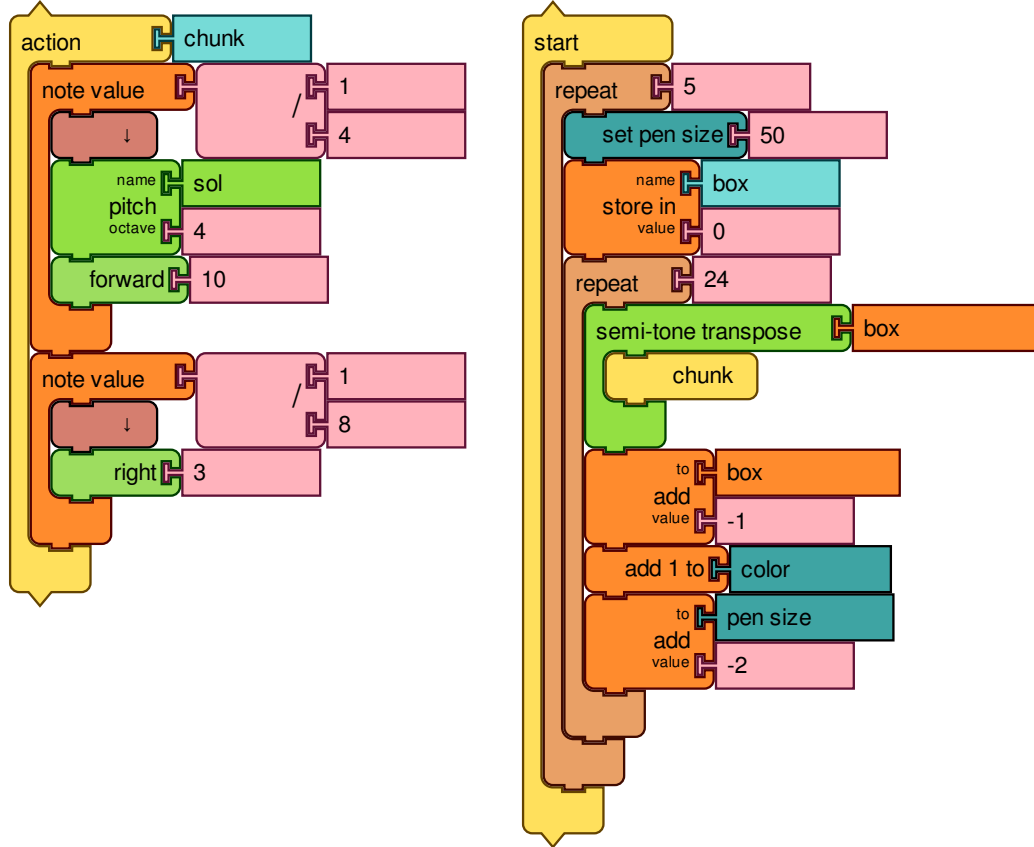
And even more interesting if we bring the various voices offset in time.

[RUN LIVE](#)

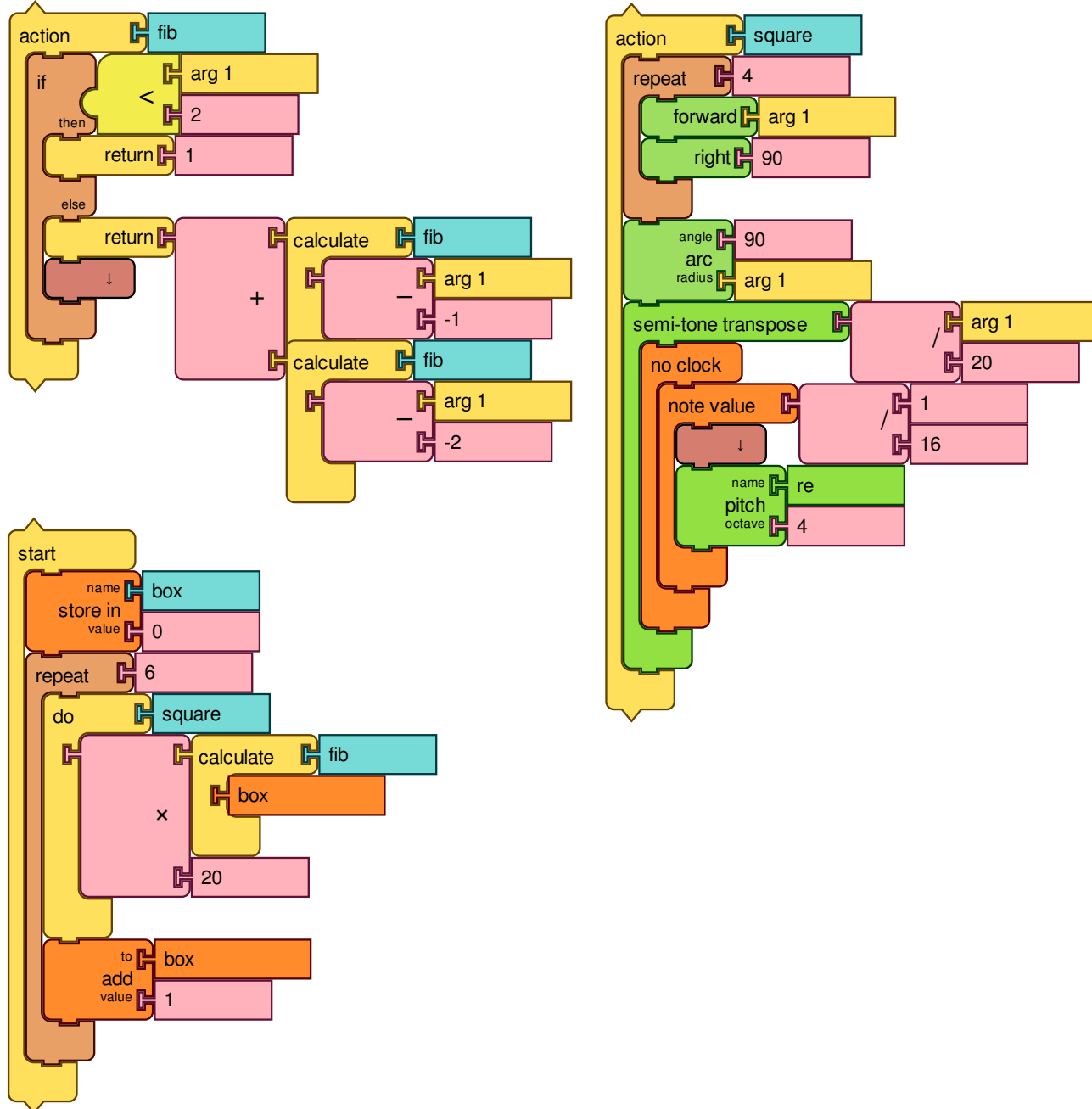


A special "drum" version of the *Start* block is available for laying down a drum track. Any pitch blocks encountered while starting from a drum will be played as **C2** with the default drum sample. In the example above, all of the notes in **chunk** will be played with a kick drum.

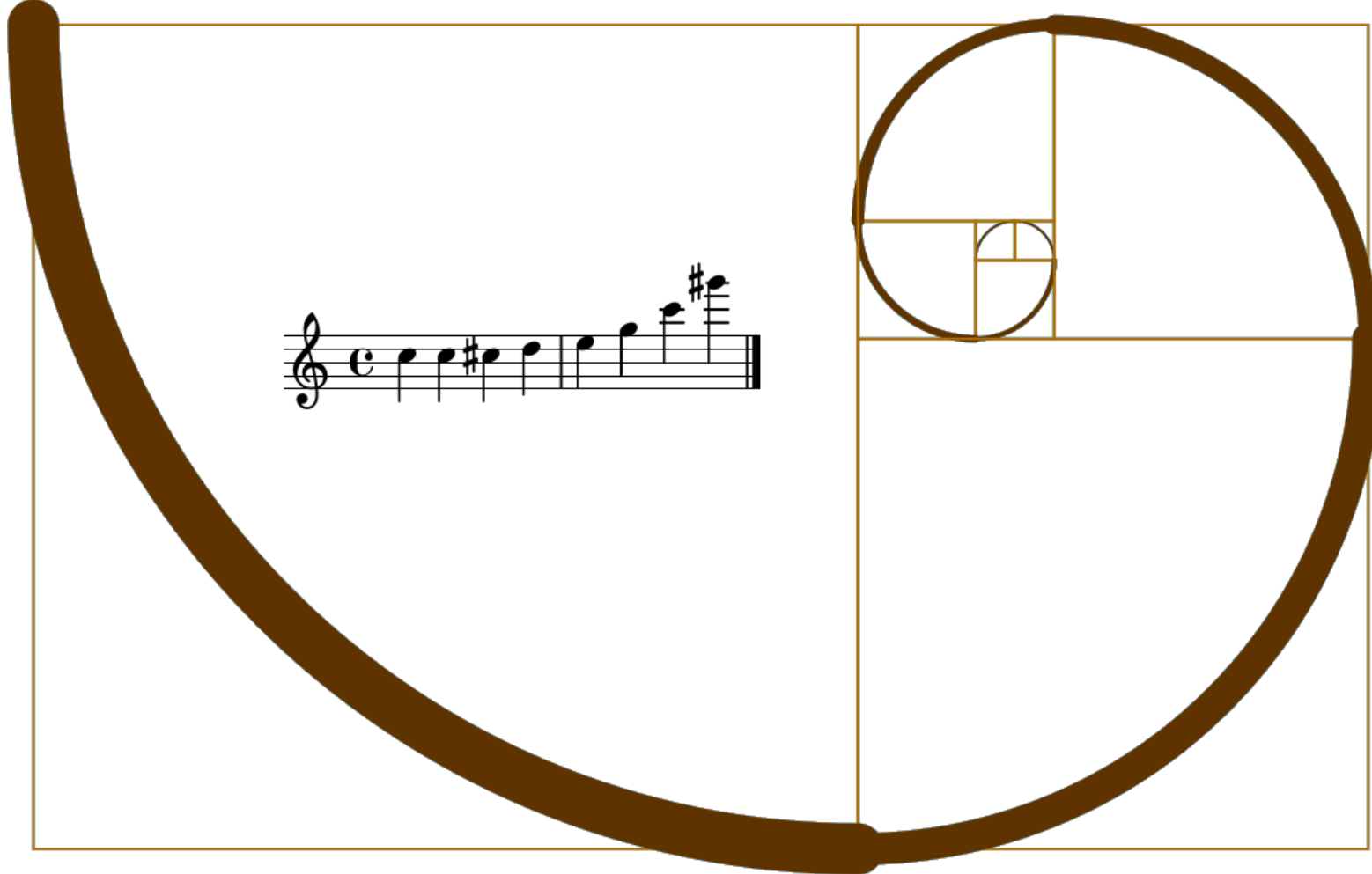
3.4 Adding graphics



Turtle graphics can be combined with the music blocks. By placing graphics blocks, e.g., *Forward* and *Right*, inside of *Note value* blocks, the graphics stay in sync with the music. In this example, the turtle moves forward each time a quarter note is played. It turns right during the eighth note. The pitch is raised by one half step, the pen size decreases, and the pen color increases at each step in the inner repeat loop.

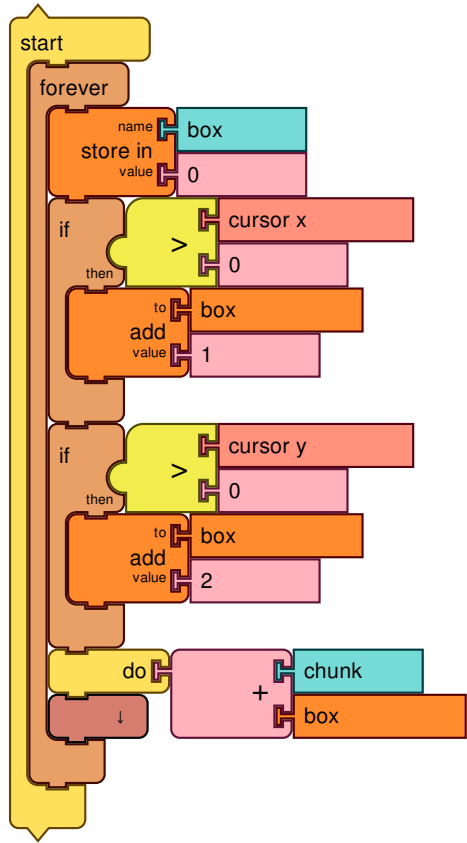


In this example, because the computation and graphics are more complex, a *Free-time* block is used to decouple the graphics from the master clock. The "Free-time" block prioritizes the sequence of actions over the specified rhythm.



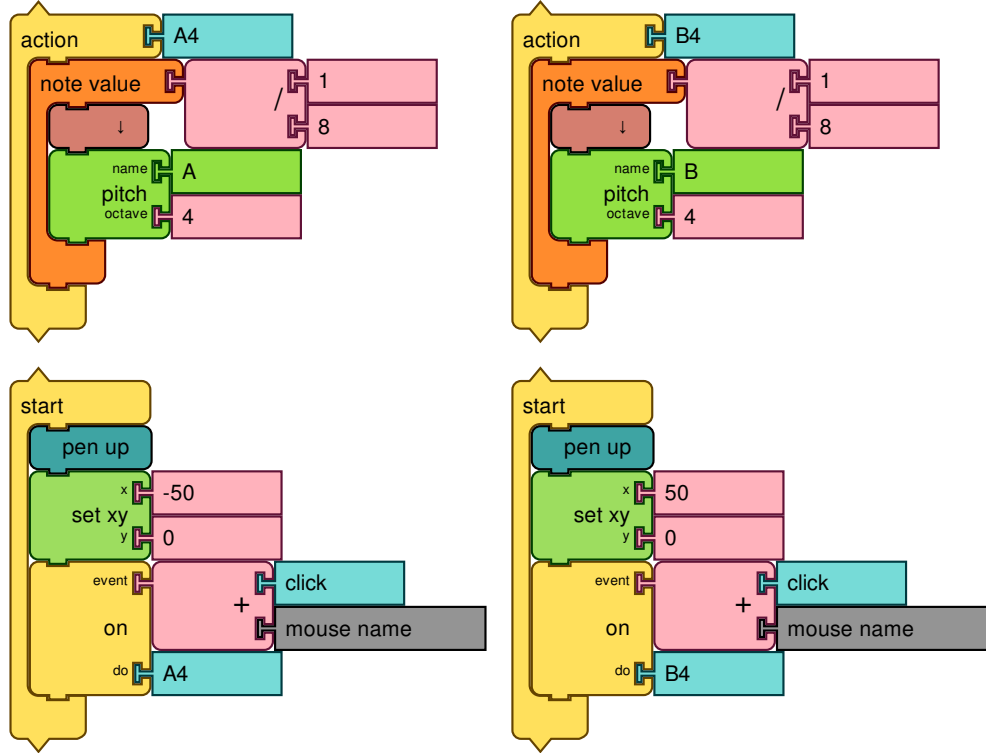
3.5 Interactions

There are many ways to interactive with Music Blocks, including tracking the mouse position to impact some aspect of the music.



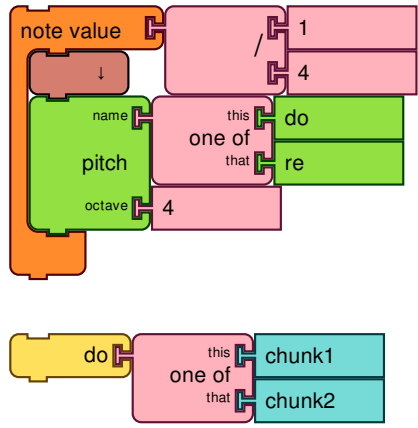
For example, we can launch the phrases (chunks) interactively. When the mouse is in the lower-left quadrant, `chunk` is played; lower-right quadrant, `chunk1` ; upper-left quadrant, `chunk2` ; and upper-right quadrant, `chunk3` .

[RUN LIVE](#)



In the example above, a simple two-key piano is created by associating *click* events on two different turtles with individual notes. Can you make an 8-key piano?

[RUN LIVE](#)



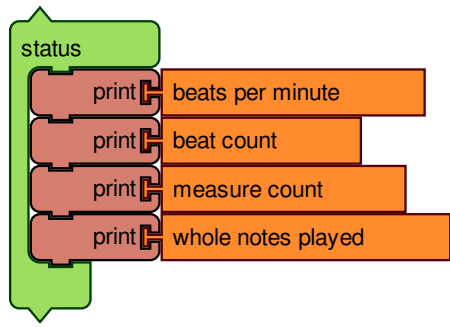
You can also add a bit of randomness to your music. In the top example above, the *One-of* block is used to randomly assign either **Do** or **Re** each time the *Note value* block is played. In the bottom example above, the *One-of* block is used to randomly select between **chunk1** and **chunk2**.

4. WIDGETS

[Previous Section \(3. Programming with Music\)](#) | [Back to Table of Contents](#) | [Next Section \(5. Beyond Music Blocks\)](#)

This section of the guide will talk about the various Widgets that can be added to Music Blocks to enhance your experience.

4.1 Status

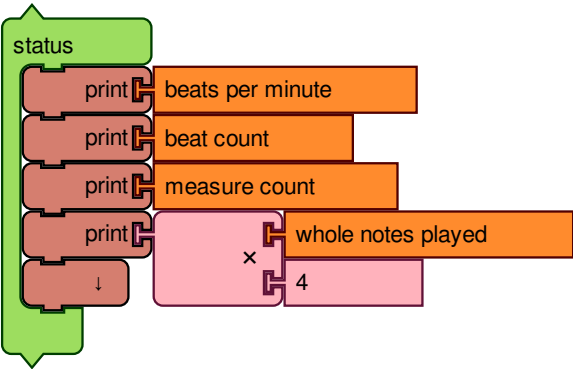


	beats per minute	beat count	measure count	whole notes played	note
	90	1	2	2	C4 8
	90	1	2	2	A5 4

The *Status widget* is a tool for inspecting the status of Music Blocks as it is running. By default, the key, BPM, and volume are displayed. Also, each note is

displayed as it is played. There is one row per voice in the status table.

Additional *Print* blocks can be added to the *Status* widget to display additional music factors, e.g., duplicate, transposition, skip, [staccato](#), [slur](#), and [graphics](#) factors, e.g., x, y, heading, color, shade, grey, and pensize.

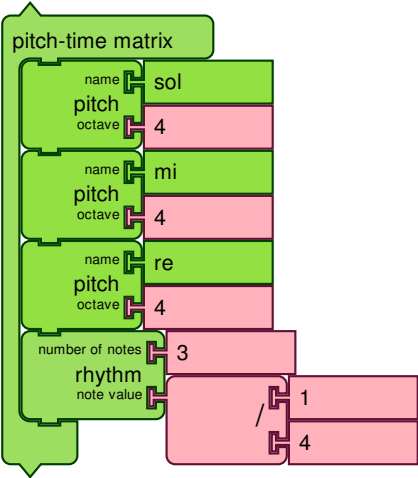


You can do additional programming within the status block. In the example above, the volume is divided by 10 before being displayed.

4.2 Generating Chunks of Notes

Using the Pitch-Time Matrix, it is possible to generate chunks of notes at a much faster speed.

4.2.1 The Pitch-Time Matrix



Music Blocks provides a widget, the *Pitch-time Matrix*, as a scaffold for getting started.

Once you've launched Music Blocks in your browser, start by clicking on the *Pitch-time Matrix* stack that appears in the middle of the screen. (For the moment, ignore the *Start* block.) You'll see a grid organized vertically by pitch and horizontally by rhythm.

Solfa							
sol							
mi							
re							
rhythmic note values	1/4	1/4	1/4				

The matrix in the figure above has three *Pitch* blocks and one *Rhythm* block, which is used to create a 3 x 3 grid of pitch and time.

Note that the default matrix has five *Pitch* blocks, hence, you will see five rows, one for each pitch. (A sixth row at the bottom is used for specifying the rhythms associated with each note.) Also by default, there are two *Rhythm* blocks, which specifies six quarter notes followed by one half note. Since the *Rhythm* blocks are inside of a *Repeat* block, there are fourteen (2 x 7) columns for selecting notes.

Solfa							
sol							
mi							
re							
rhythmic note values	1/4	1/4	1/4				

By clicking on individual cells in the grid, you should hear individual notes (or chords if you click on more than one cell in a column). In the figure, three quarter notes are selected (black cells). First **Re 4**, followed by **Mi 4**, followed by **Sol 4**.



If you click on the *Play* button (found in the top row of the grid), you will hear a sequence of notes played (from left to right): **Re 4**, **Mi 4**, **Sol 4**.



Once you have a group of notes (a "chunk") that you like, click on the *Save* button (just to the right of the *Play* button). This will create a stack of blocks that can be used to play these same notes programmatically. (More on that below.)

You can rearrange the selected notes in the grid and save other chunks as well.



The *Sort* button will reorder the pitches in the matrix from highest to lowest and eliminate any duplicate *Pitch* blocks.



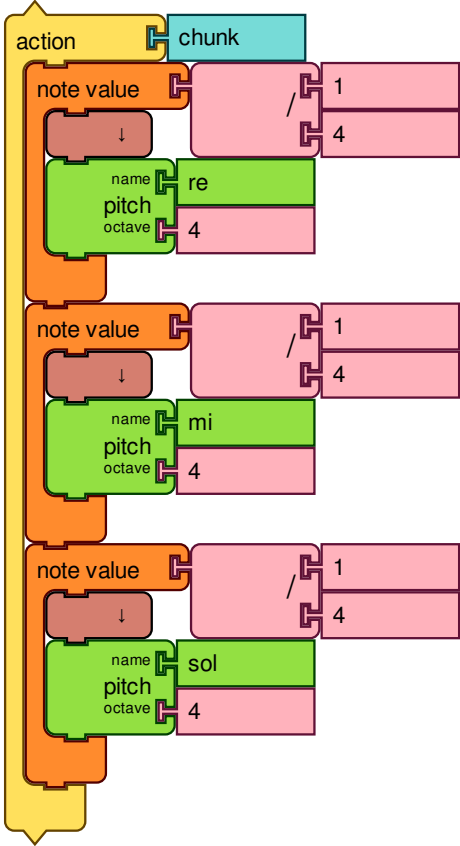
You can hide the matrix by clicking on the *Close* button (the right-most button in the top row of the grid.)



There is also an Erase button that will clear the grid.

Don't worry. You can reopen the matrix at anytime (it will remember its previous state) and since you can define as many chunks as you want, feel free to experiment.

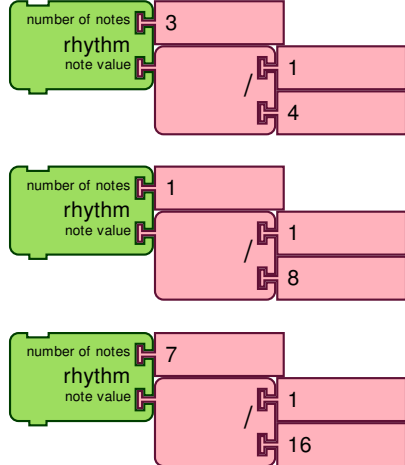
Tip: You can put a chunk inside a *Pitch-time Matrix* block to generate the matrix to corresponds to that chunk.



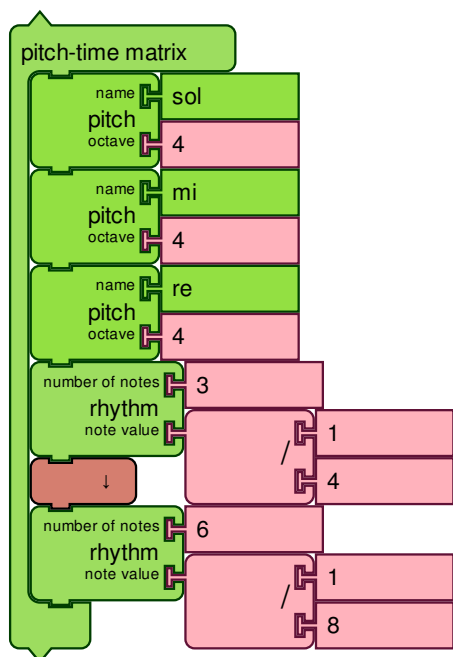
The chunk created when you click on the matrix is a stack of blocks. The blocks are nested: an *Action* block contains three *Note value* blocks, each of which contains a *Pitch* block. The *Action* block has a name automatically generated by the matrix, in this case, chunk. (You can rename the action by clicking on the name.). Each note has a duration (in this case 4, which represents a quarter note). Try putting different numbers in and see (hear) what happens. Each note block also has a pitch block (if it were a chord, there would be multiple *Pitch* blocks nested inside the Note block's clamp). Each pitch block has a pitch name (*Re* , *Mi* , and *Sol*), and a pitch octave; in this example, the octave is 4 for each pitch. (Try changing the pitch names and the pitch octaves.)

To play the chuck, simply click on the action block (on the word action). You should hear the notes play, ordered from top to bottom.

4.2.2 The Rhythm Block



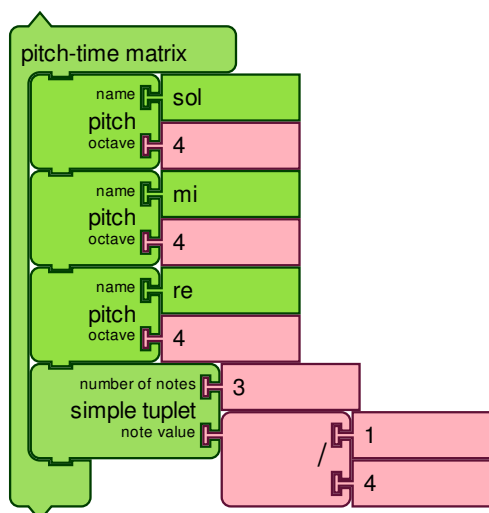
Rhythm blocks are used to generate rhythm patterns in the *Pitch-time Matrix* block. The top argument to the *Rhythm* block is the number of notes. The bottom argument is the duration of the note. In the top example above, three columns for quarter notes would be generated in the matrix. In the middle example, one column for an eighth note would be generated. In the bottom example, seven columns for 16th notes would be generated.



Solfa							
sol							
mi							
re							
rhythmic note values	1/4	1/4	1/4	1/8	1/8	1/8	1/8

You can use as many *Rhythm* blocks as you'd like inside the *Pitch-time Matrix* block. In the above example, two *Rhythm* blocks are used, resulting in three quarter notes and six eighth notes.

4.2.3 Creating Tuplets



Solfa						
sol						
mi						
re						
tuplet note values	1/12	1/12	1/12			
tuplet value	3					
rhythmic note values	1/4					

Tuplets are a collection of notes that get scaled to a specific duration. Using tuplets makes it easy to create groups of notes that are not based on a power of 2. In the example above, three quarter notes—defined in the *Rhythm* block—are played in the time of a single quarter note. The result is three twelfth notes.

You can mix and match *Rhythm* blocks and *Tuplet* blocks when defining your matrix.

4.2.4 What is a tuplet?

Using Tuplets

A tuplet is a specific group of notes played in a condensed amount of time.


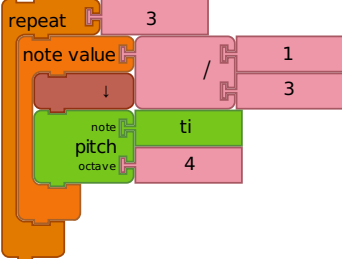
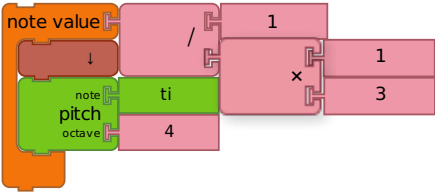
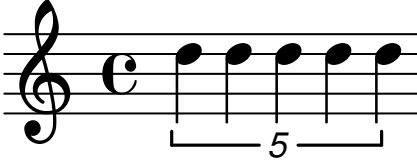
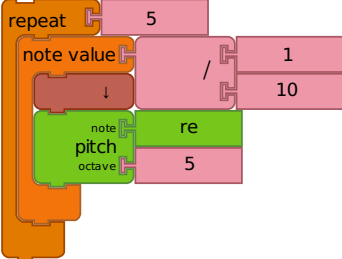
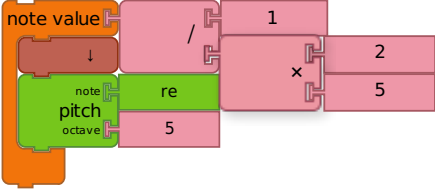
x= power of the note*

y= tuplet value

Formula:

$$\frac{1}{2^{x*} y} = \text{resulting note value**}$$

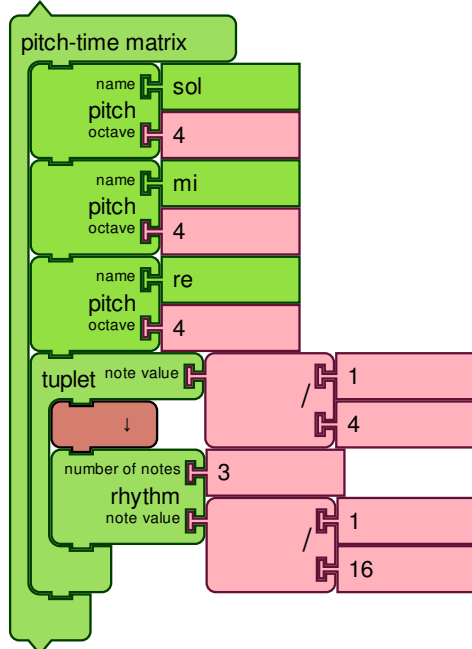
Examples:

Western Notation	Music Blocks Tuplet	Music Block Math for Tuplet
 <p>x= 0 and y=3</p> $\frac{1}{2^0 \times 3} = \frac{1}{1 \times 3} = \frac{1}{3}$		
 <p>x= 1 and y=5</p> $\frac{1}{2^1 \times 5} = \frac{1}{2 \times 5} = \frac{1}{10}$		

*The power of the note occurs as follows:
longa= -3, breve= -2, whole= -1, half=0, quarter=1, eighth=2, sixteenth=3, thirty-second=4,
and continues in this pattern.

**Different tuplet values produce different rhythmic qualities when mixed with note values of different tuplet values.

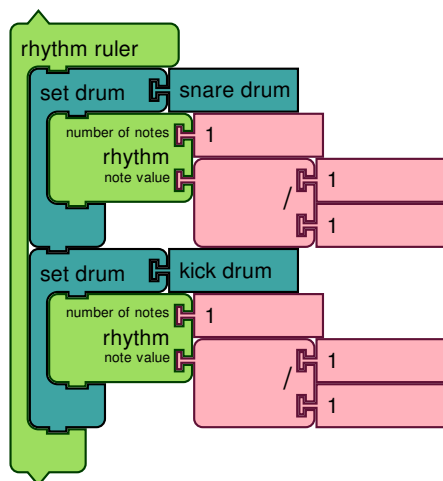
Power of Two	Music Blocks Tuplet	Tuplet in Western Notation
$\frac{1}{2^{-1} \times 3} = \frac{2}{3}$		
$\frac{1}{2^0 \times 3} = \frac{1}{3}$		
$\frac{1}{2^1 \times 3} = \frac{1}{6}$		
$\frac{1}{2^2 \times 3} = \frac{1}{12}$		
$\frac{1}{2^3 \times 3} = \frac{1}{24}$		



You can also use individual notes when defining the grid. These blocks will expand into *Rhythm* blocks with the corresponding values.

4.3 Generating Rhythms

The *Rhythm Ruler* block is used to launch a widget similar to the *Pitch-time Matrix* block. The widget can be used to generate rhythmic patterns.

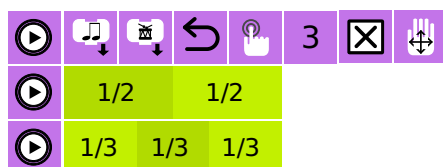


The argument to the *Rhythm Ruler* block specifies the duration that will be subdivided to generate a rhythmic pattern. By default, it is 1 / 1, e.g., a whole note.

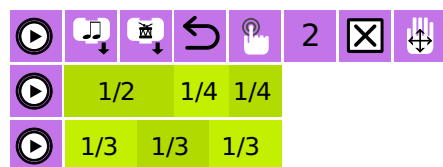
The *Set Drum* blocks contained in the clamp of the *Rhythm Ruler* block indicates the number of rhythms to be defined simultaneously. By default, two rhythms are defined. The embedded *Rhythm* blocks define the initial subdivision of each rhythm ruler.



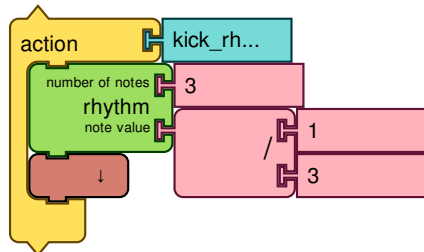
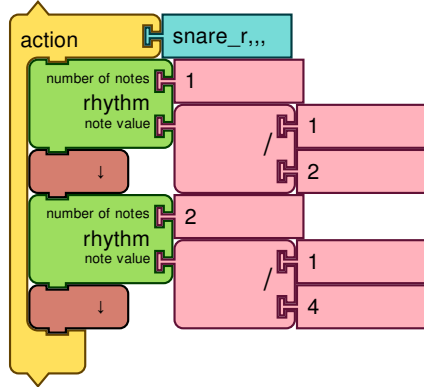
When the *Rhythm Ruler* block is clicked, the *Rhythm Ruler* widget is opened. It contains a row for each rhythm ruler. An input in the top row of the widget is used to specify how many subdivisions will be created within a cell when it is clicked. By default, 2 subdivisions are created.



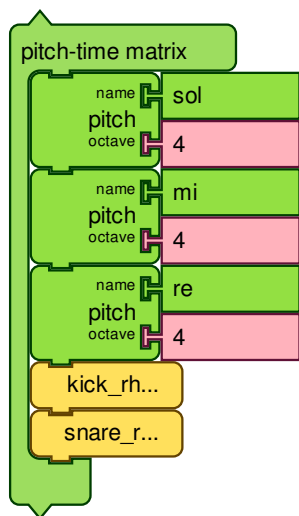
As shown in the above figure, the top rhythm ruler has been divided into two half-notes and the bottom rhythm ruler has been divided into three third-notes. Clicking on the *Play* button to the left of each row will playback the rhythm using a drum for each beat. The *Play-all* button on the upper-left of the widget will play back all rhythms simultaneously.



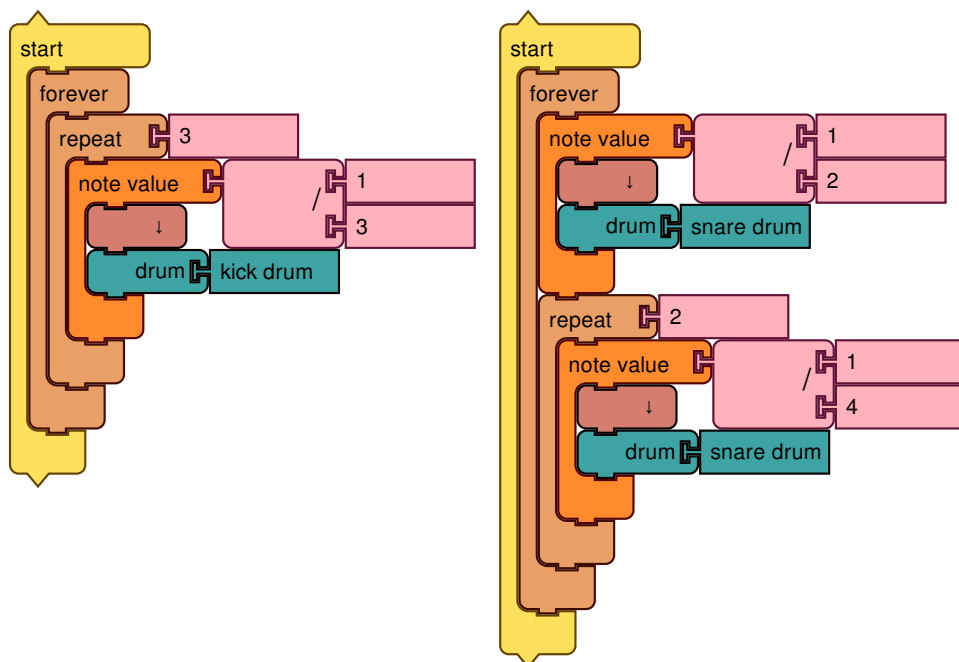
The rhythm can be further subdivided by clicking in individual cells. In the example above, two quarter-notes have been created by clicking on one of the half-notes.



The *Save stack* button will export rhythm stacks.



These stacks of rhythms can be used to define rhythmic patterns used with the *Pitch-time Matrix* block.



The *Save drum machine* button will export *Start* stacks that will play the rhythms as drum machines.

4.4 Musical Modes

Musical modes are used to specify the relationship between [intervals](#) (or steps) in a scale. Since Western music is based on 12 half-steps per octave, modes specify how many half steps there are between each note in a scale.

By default, Music Blocks uses the *Major* mode, which, in the *Key* of C, maps to the white keys on a piano. The intervals in the *Major* mode are 2, 2, 1, 2, 2, 2, 1. Many other common modes are built into Music Blocks, including, of course, *Minor* mode, which uses 2, 1, 2, 2, 1, 2, 2 as its intervals.

Note that not every mode uses 7 intervals per octave. For example, the *Chromatic* mode uses 11 intervals: 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1. The *Japanese* mode uses only 5 intervals: 1, 4, 2, 3, 2]. What is important is that the sum of the intervals in an octave is 12 half-steps.

custom mode

key

C

set key

mode

major

The *Mode* widget lets you explore modes and generate custom modes. You invoke the widget with the *Custom mode* block. The mode specified in the *Set key* block will be the default mode when the widget launches.

mode

0

1

2

3

4

5

6

7

8

9

10

11

In the above example, the widget has been launched with *Major* mode (the default). Note that the notes included in the mode are indicated by the black boxes, which are arrayed in a circular pattern of twelve half-steps to complete the octave.

Since the intervals in the *Major* mode are 2, 2, 1, 2, 2, 2, 1, the notes are 0, 2, 4, 5, 7, 9, 11, and 12 (one octave above 0).

The widget controls run along the toolbar at the top. From left to right are:

- Play all*, which will play a scale using the current mode;
- Save*, which will save the current mode as the *Custom mode* and save a stack of *Pitch* blocks that can be used with the *Pitch-time Matrix* block;
- Rotate counter-clockwise*, which will rotate the mode counter-clockwise (See the example below);
- Rotate clockwise*, which will rotate the mode clockwise (See the example below);
- Invert*, which will invert the mode (See the example below);
- Undo*, which will restore the mode to the previous version; and
- Close*, which will close the widget.

You can also click on individual notes to activate or deactivate them.

Note that the mode inside the *Custom mode* block is updated whenever the mode is changed inside the widget.

mode

0

1

2

3

4

5

6

7

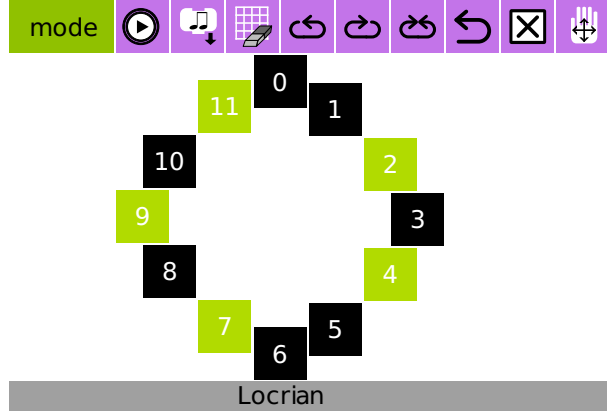
8

9

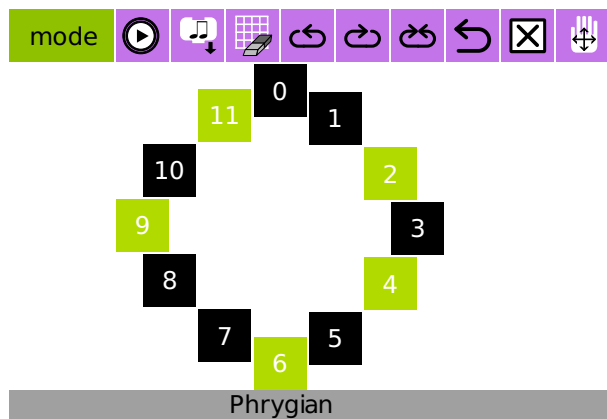
10

11

In the above example, the *Major* mode has been rotated clockwise, transforming it into *Dorian*.

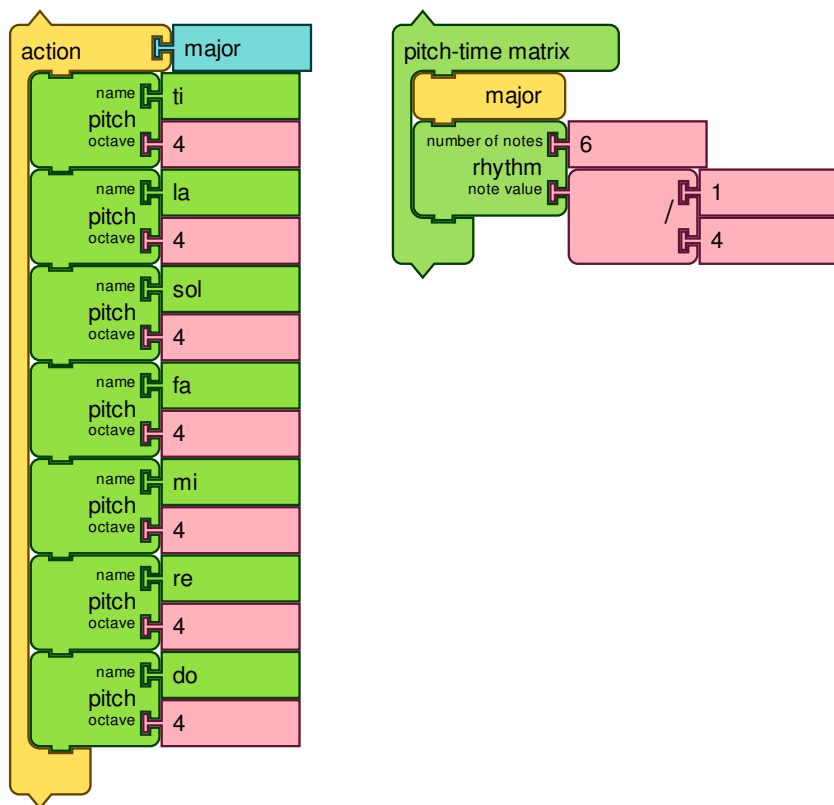


In the above example, the *Major* mode has been rotated counter-clockwise, transforming it into *Locrian*.



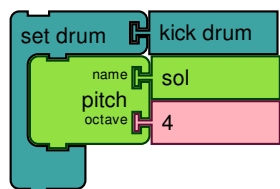
In the above example, the *Major* mode has been inverted, transforming it into *Phrygian*.

Note: The build-in modes in Music Blocks can be found in [musicutils.js#L68](https://musicutils.js.org/#L68).

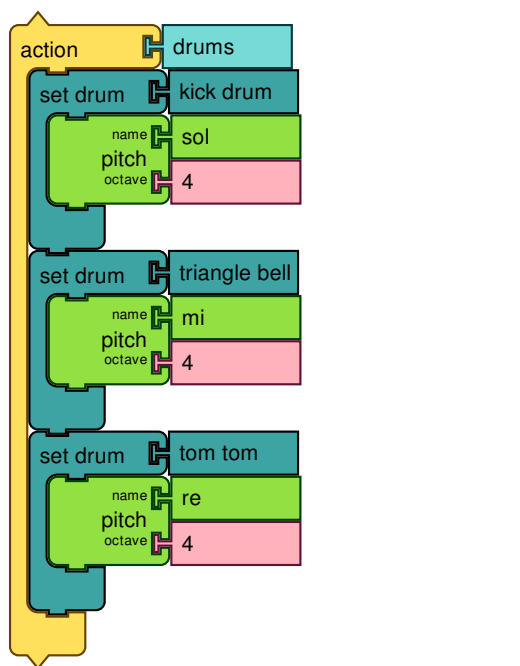
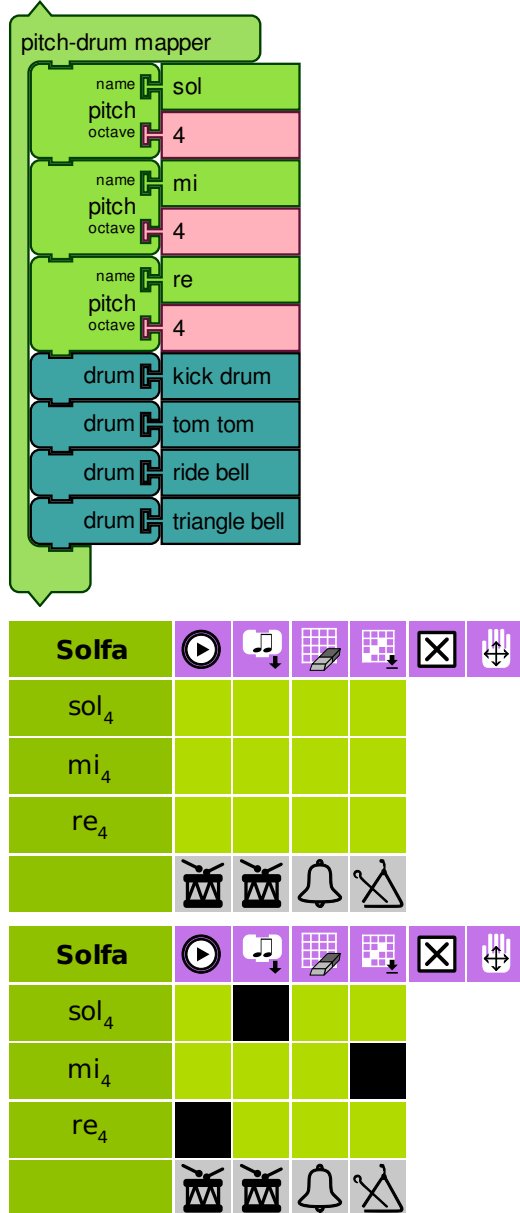


The *Save* button exports a stack of blocks representing the mode that can be used inside the *Pitch-time Matrix* block.

4.5 The Pitch-Drum Matrix



The *Set Drum* block is used to map the enclosed pitches into drum sounds. Drum sounds are played in a monopitch using the specified drum sample. In the example above, a *kick drum* will be substituted for each occurrence of a *Re* 4.

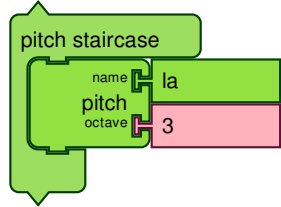


As an expedience for creating mapping with the *Set Drum* block, we provide the *Drum-Pitch Matrix*. You use it to map between pitches and drums. The output is a stack of *Set Drum* blocks.

4.6 Generating Arbitrary Pitches

The *Pitch Staircase* block is used to launch a widget similar to the *Pitch-time Matrix*, which can be used to generate different pitches using a given pitch and musical proportion.

The *Pitch* blocks contained in the clamp of the *Pitch Staircase* block define the pitches to be initialized simultaneously. By default, one pitch is defined and it have default note "la" and octave "3".

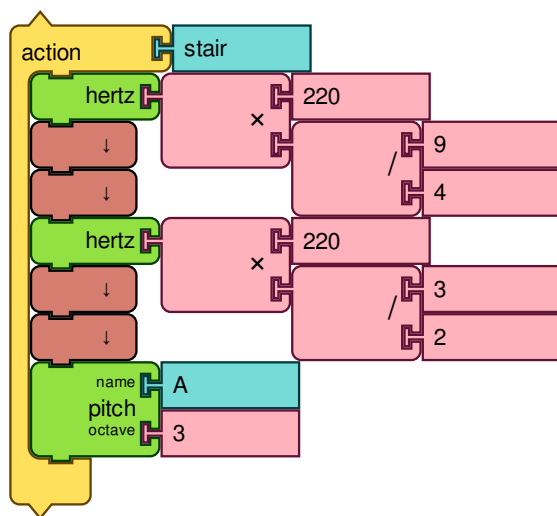


When *Pitch Staircase* block is clicked, the *Pitch Staircase* widget is initialized. The widget contains row for every *Pitch* block contained in the clamp of the *Pitch Staircase* block. The input fields in the top row of the widget specify the musical proportions used to create new pitches in the staircase. The inputs correspond to the numerator and denominator in the proportion respectively. By default the proportion is 3:2.

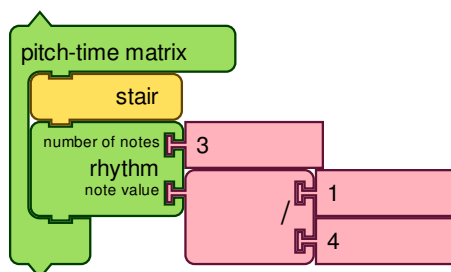


Clicking on the *Play* button to the left of each row will playback the notes associated with that step in the stairs. The *Play-all* button on the upper-left of the widget will play back all the pitch steps simultaneously. A second *Play-all* button to the right of the stair plays in increasing order of frequency first, then in decreasing order of frequency as well, completing a scale.

The *Save stack* button will export pitch stacks. For example, in the above configuration, the output from pressing the *Save stack* button is shown below:



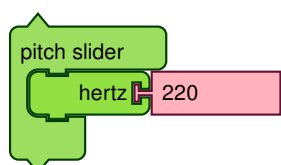
These stacks can be used with the *Pitch-time Matrix* block to define the rows in the matrix.



4.7 Generating Arbitrary Pitches

The *Pitch Slider* block is used to launch a widget that is used to generate arbitrary pitches. It differs from the *Pitch Staircase* widget in that it is used to create frequencies that vary continuously within the range of a specified octave.

Each *Sine* block contained within the clamp of the *Pitch Slider* block defines the initial pitch for an octave.



✕

⌕

220

▲

▼

When the *Pitch Slider* block is clicked, the *Pitch Slider* widget is initialized. The widget will have one column for each *Sine* block in the clamp. Every column has a slider that can be used to move up or down in frequency, continuously or in intervals of 1/12th of the starting frequency. The mouse is used to move the frequency up and down continuously. Buttons are used for intervals. Arrow keys can also be used to move up and down, or between columns.

pitch slider

hertz

220

hertz

440

✕

⌕

220

440

▲

▼

▲

▼

Clicking in a column will extract the corresponding *Note* blocks, for example:

✕

⌕

373

880

▲

▼

▲

▼

note value

↓

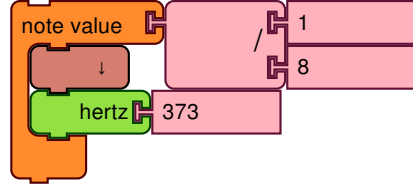
hertz

880

/

1

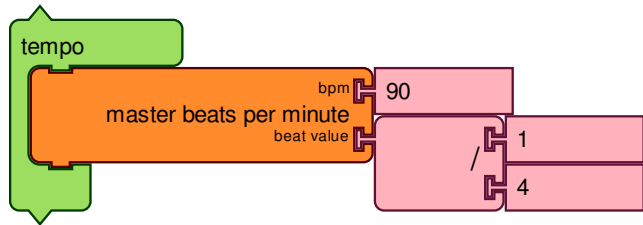
8



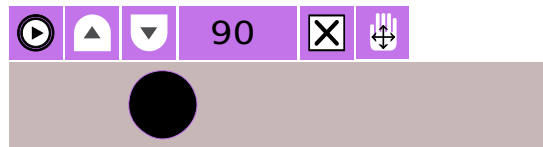
4.8 Changing Tempo

The *Tempo* block is used to launch a widget that enables the user to visualize Tempo, defined in beats per minute (BPM). When the *Tempo* block is clicked, the *Tempo* widget is initialized.

The *Master Beats per Minute* block contained in the clamp of the *Tempo* block sets the initial tempo used by the widget. This determines the speed at which the ball in the widget moves back and forth. If BPM is 60, then it will take one second for the ball to move across the widget. A round-trip would take two seconds.



The top row of the widget holds the *Play/pause* button, the *Speed up* and *Slow down* buttons, and an input field for updating the Tempo.

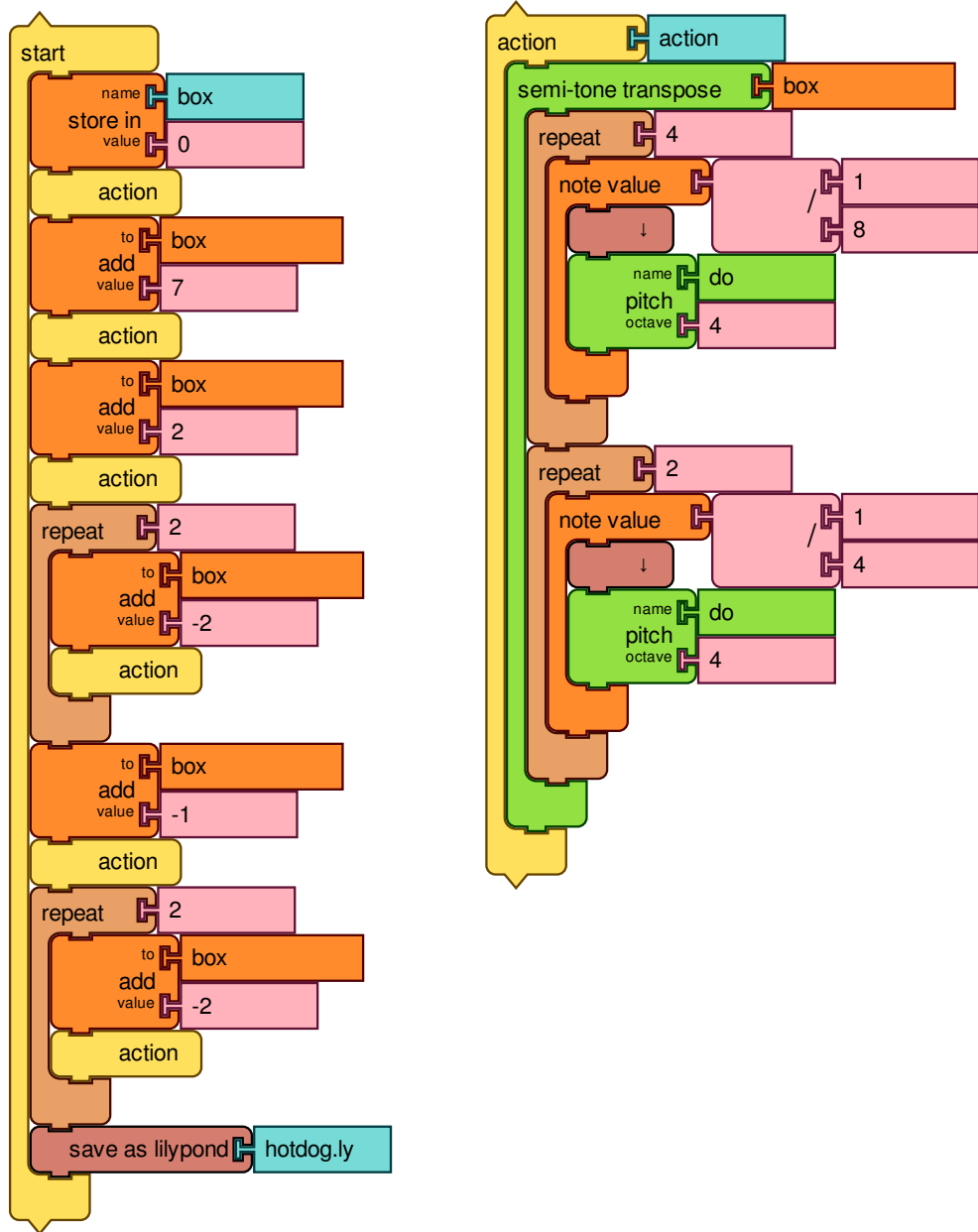


You can also update the tempo by clicking twice in spaced succession in the widget: the new BPM is determined as the time between the two clicks. For example, if there is 1/2 seconds between clicks, the new BPM will be set as 120.

5. BEYOND MUSIC BLOCKS

[Previous Section \(4. Widgets\)](#) | [Back to Table of Contents](#)

Music Blocks is a waypoint, not a destination. One of the goals is to point the learner towards other powerful tools. One such tool is [Lilypond](#), a music engraving program.



The *Save as Lilypond* block will transcribe your composition. The output of the program above is saved to [Downloads/hotdog.ly](#) . There is also a *Save as Lilypond* button on the secondary toolbar.



```
\version "2.18.2"

mouse = {
c'8 c'8 c'8 c'8 c'4 c'4 g'8 g'8 g'8 g'8 g'4 g'4 a'8 a'8 a'8 a'4
a'4 g'8 g'8 g'8 g'8 g'4 g'4 f'8 f'8 f'8 f'4 f'4 e'8 e'8 e'8 e'8
e'4 e'4 d'8 d'8 d'8 d'8 d'4 d'4 c'8 c'8 c'8 c'8 c'4 c'4
}

\score {
<<
\new Staff = "treble" {
\clef "treble"
\set Staff.instrumentName = #"mouse" \mouse
}
>>
\layout {}
}
```



RUN LIVE