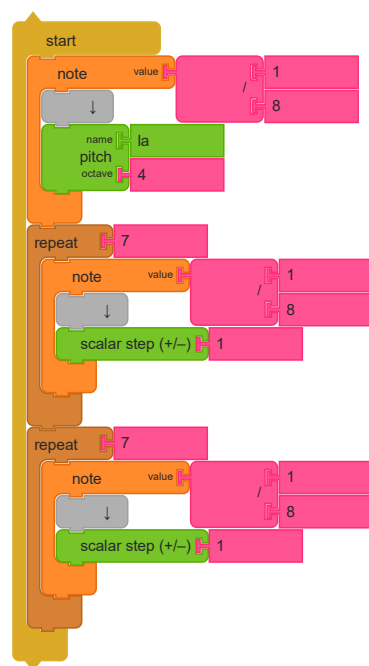


3.2 Pitch Transformations

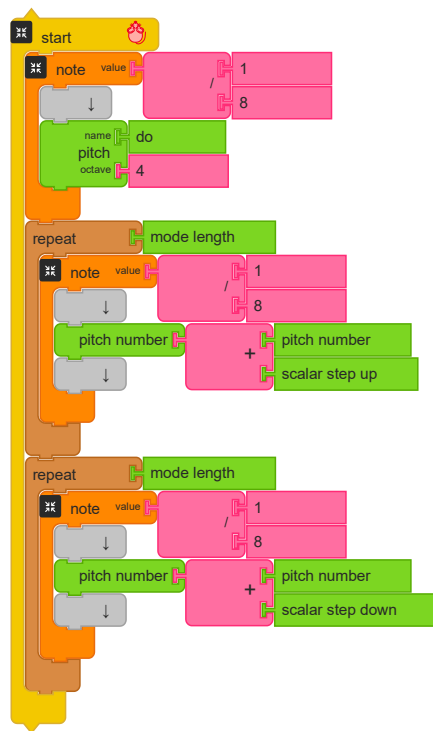
There are many ways to transform pitch, rhythm, and other sonic qualities.

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.

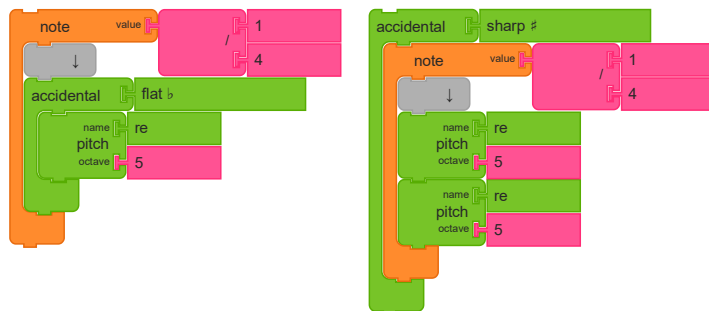
[RUN LIVE](#)



Another way to move up and down notes in a scale is to use the *Scalar Step Up* and *Scalar Step Down* blocks. These blocks calculate the number of half-steps to the next note in the current mode. (You can read more about [Musical Modes](#) below.) Note that the *Mouse Pitch Number* block returns the pitch number of the most recent note played.

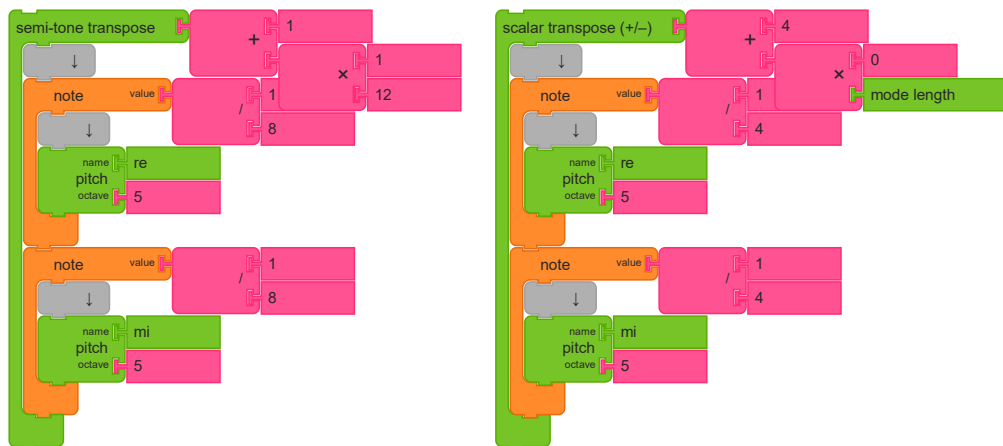
In this example, we are using the *Mode length* block, which returns the number of scalar steps in the current mode (7 for Major and Minor modes).

3.2.2 Sharps And Flats



The *Accidental* block can be wrapped around *Pitch* blocks, *Note value* blocks, or chunks of notes inside of *Action* blocks. 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 *re* is lowered by one half step; on the right, both *Pitch* blocks are raised by one half step. (You can also use a double-sharp or double-flat accidental.)

3.2.3 Adjusting Transposition



There are multiple ways to transpose a pitch: by semi-tone or scalar steps or by a ratio. The *Semi-tone-transposition* block (above left) 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 up an entire octave, transpose by 12 half-steps. -12 will shift down an entire octave.

The *Scalar-transposition* block (above right) shifts a pitch based on the current key and mode. For example, in C Major, a scalar transposition of 1 would transpose C to D (even though it is a transposition of 2 half steps). To transpose E to F is 1 scalar step (or 1 half step). To shift an entire octave, scalar transpose by the mode length up or down. (In major scales, the mode length is 7.)

The **Register** block provides an easy way to modify the register (octave) of the notes that follow it. In the example above it is first used to bump the $\text{mi } 4$ note up by one octave and then to bump the $\text{so1 } 4$ note down by one octave.

3.2.4 Summary of Pitch Movements

Representation	Pitch Movement	Properties
Scalar Step	scalar	0=no change
		1=next scalar pitch in current key and mode
		1=next scalar pitch in current key and mode
		-1=previous scalar pitch in current key and mode
		If the argument to scalar step is positive, it moves up the scale; if it is negative, it moves down the scale.

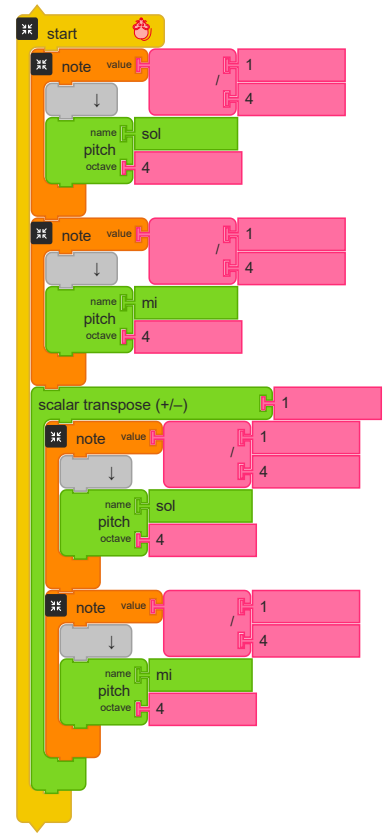
Music Blocks Code for Scalar Step

The example above demonstrates traveling up and down the major scale by moving an octave up from the starting note, do, one note at a time and then back down the same way.

Standard Notation with Scalar Step

Representation	Pitch Movement	Properties
Transposition	Semi-tone	Creates shifts in pitch by half-steps
		If the argument to transpose is positive, it will shift upwards in pitch; if it is negative, there will be a downwards shift.
		There are 12 half-steps shifts per octave.
		An argument of -12 will shift down one octave.
		An argument of zero will not change the pitch.

Music Blocks Code with Scalar Transpose



Standard Notation for Scalar Transpose

The musical notation shows a treble clef, a 4/4 time signature, and a sequence of four eighth notes: C4, D4, E4, and F4. This represents a scalar transposition of 4 steps (fifth) from the original key of C major.

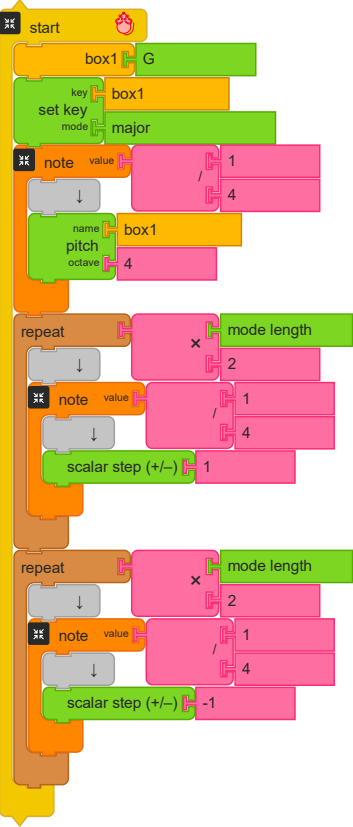
Representation	Pitch Movement	Properties
Transposition	Scalar	Shifts the pitch based on the current key and mode
		Each number represents a scalar step.
		Scalar transposition can transform your original key to a new key by counting the notes between the keys.
		For example: Transposing C-D-E-F by 4 (fifth) will give us G-A-B-C
		To transpose an octave: shift by the mode length (7 in major scales) up or down.

3.2.5 Set Key

The *Set key* block is used to change both the mode and key of the current scale. (The current scale is used to define the mapping of Solfege [when set Movable Do = True] to notes and also the number of half steps take by the the *Scalar step* block.) For example, by setting the key to C Major, the scale is defined by starting at C (or Do) and applying the pattern of half steps defined by a Major mode. In this case, the pattern of steps skips past all of the sharps and flats. (On a piano, C Major is just the white keys).

When using the *Set key* block, the mode argument is used to define the pattern of half steps and the key argument is used to define the starting position of the pattern. For example, when mode = "major" and key = "C", the pattern of half steps is 2 2 1 2 2 2 1 and the first note in the scale from which the pattern is applied is "C".

Set Key Example



Using the example above, one can modify the arguments to *Set key* in order to move up and down one octave in a scale. The example shows G Major scale, which has an F#, but it could be used for any combination of key and mode.

Standard Notation for Set Key Example

[RUN LIVE](#)

Various examples for Major modes are shown in the following table.

Key	Mode	Mode Pattern in Half Steps	Pitch Pattern
C	Major	2 2 1 2 2 2 1	C, D, E, F, G, A, B, C
G	Major	2 2 1 2 2 2 1	G, A, B, C, D, E, F#, G
D	Major	2 2 1 2 2 2 1	D, E, F#, G, A, B, C#, D
F	Major	2 2 1 2 2 2 1	F, G, A, B, C, D, E, F
B ^b	Major	2 2 1 2 2 2 1	B ^b , C, D, E ^b , F, G, A, B ^b

The next table is the same sets of various keys (starting pitches), but the mode is set to “Dorian” instead of Major.

Key	Mode	Mode Pattern in Half Steps	Pitch Pattern
C	Dorian	2 1 2 2 2 1 2	C, D, E ^b , F, G, A, B ^b , C
G	Dorian	2 1 2 2 2 1 2	G, A, B ^b , C, D, E, F, G
D	Dorian	2 1 2 2 2 1 2	D, E, F, G, A, B, C, D
F	Dorian	2 1 2 2 2 1 2	F, G, A ^b , B ^b , C, D, E ^b , F
B ^b	Dorian	2 1 2 2 2 1 2	B ^b , C, D ^b , E ^b , F, G, A ^b , B ^b

This last table is the same set of keys as the above two tables, but the mode is set to “Phrygian”.

Key	Mode	Mode Pattern in Half Steps	Pitch Pattern
C	Phrygian	1 2 2 2 1 2 2	C, D ^b , E ^b , F, G, A ^b , B ^b , C
G	Phrygian	1 2 2 2 1 2 2	G, A ^b , B ^b , C, D, E ^b , F, G
D	Phrygian	1 2 2 2 1 2 2	D, E ^b , F, G, A, B ^b , C, D
F	Phrygian	1 2 2 2 1 2 2	F, G ^b , A ^b , B ^b , C, D ^b , E ^b , F
B ^b	Phrygian	1 2 2 2 1 2 2	B ^b , C ^b , D ^b , E ^b , F, G ^b , A ^b , B ^b

Notice how in all the examples, the sets with the same mode results in the same “Mode Pattern of Half Steps”, but the resultant “Pitch Pattern” is different. Also, notice how G Dorian and F Major have the same set of pitches in “Pitch Pattern” (they both have B^b and no other sharps or flats). C Dorian, D Phrygian, and B^b Major all have the same set of pitches as well (all three have B^b and E^b).

If these lists were expanded further, there would be many more such examples. These are because these modes (Major, Dorian, and Phrygian) all have essentially the same modal pattern; the starting point is just shifted slightly for each: Dorian could be thought of starting from the second scale degree of Major and Phrygian from the third, for example. Not all modes have this relationship to Major. The ones that do are: Ionian (Major), Dorian, Phrygian, Lydian, Myxolydian, Aeolian (Minor), and Locrian.

Set Key & Scalar Step

The *Set key* block is used to select a subset of notes in the given temperament. (By default, Music Blocks uses equal temperament of 12 equal divisions of the octave. The key and mode determine which of these notes will be used.)

Set Key & Movable Do

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., F# , G# , A# , when in F# Major. It only impacts the mapping of solfege when the *movable Do* block is set to True.

You can find the *Set key* block on the *Intervals* palette.

Music Blocks for Set Key and Movable Do

start

note

value

1

4

name

sol

pitch

octave

4

note

value

1

4

name

mi

pitch

octave

4

scalar transpose (+/-)

4

note

value

1

4

name

sol

pitch

octave

4

note

value

1

4

name


mi

pitch

octave

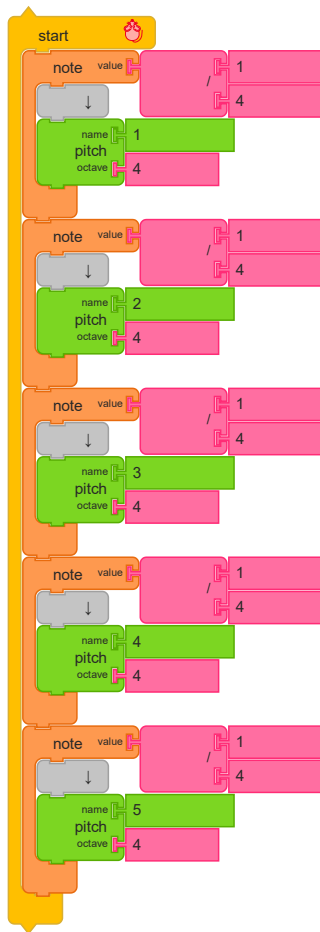
4

Standard Notation for Set Key and Movable Do



Representation	Pitch Movement	Properties
Scale Degree	Scalar	The key block sets the key and mode.
		The scale degree blocks indicate which position the pitch is taking in the scale relative to the tonic which is "scale degree 1".

Music Blocks Code with Scale Degrees 1-5

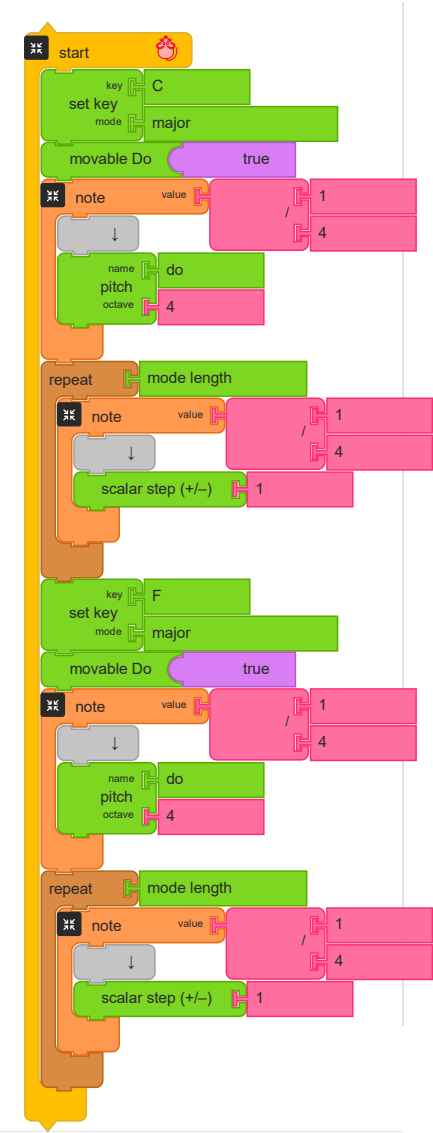


Standard Notation for Scale Degrees 1-5

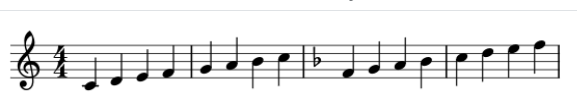


Representation	Pitch Movement	Properties
Movable "Do"	Advanced transposition by mode	Movable Do in combination with the Scale/Mode blocks will transpose sections of music in a nuanced way.
		The Set-key block allows you to change both the mode and key of how solfege is mapped to the notes.
		For example, in C major - Do is C, Re is D, Mi is E, etc.
		In F major - Do is F, Re is G, Mi is A

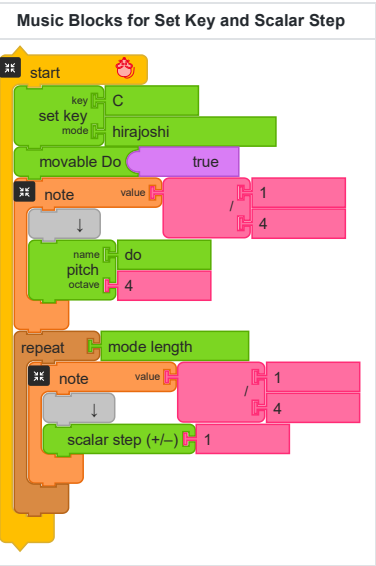
Music Blocks Code with Set Key and Movable Do



Standard Notation Code for Set Key and Movable Do



Representation	Pitch Movement	Properties
Movable "Do"	Advanced transposition by mode	You also have the option of changing the mode to Minor, Major, Chromatic, and many other exotic modes like hirajoshi, as shown in the example below.



Standard Notation with Set Key and Scalar Step



Fixed and Movable Systems

Fixed pitch systems represent pitches in an absolute way. Pitches in a fixed system do not change, regardless of a tonal context (such as key or mode). Movable systems, on the other hand, represent pitches in a relative way based on their tonal context.

An example of a fixed system is Alphabet Notation. Pitches are expressed as A, B, C, D, E, F, and G. Regardless of whether the key is C major or G minor, the pitch of G is the same. In Alphabet Notation, pitches are the same (“fixed”) regardless of the context.



An example of a movable system is Scale Degree. Pitches are expressed as 1, 2, 3, 4, 5, 6, and 7. For C major, these pitches are C, D, E, F, G, A, and B. For G (natural) minor, these pitches are G, A, B, C, D, E, and F. For D dorian, these pitches are D, E, F, G, A, B, and C. In all three examples, the pitches are determined by the tonal context.



Solfege is an example of a system that can be either fixed or movable; it can either be a fixed system (Fixed Solfege) or a movable system (Movable Solfege).

Fixed Solfege works like the alphabet system; La is A, Ti is B, Do is C, etc. Context does not affect the sounding pitch. Movable Solfege works like the Scale Degree system; for any major, Do is 1st scale degree, Re is 2nd, Mi is 3rd, Fa is 4th, etc. Hence, in Movable Solfege in the key of G (natural) minor, Do is G, Re is A, et al.



Music Blocks users can create and preview code in both Fixed Solfege and Movable Solfege. Teachers and learners may use either system (or both) to express their musical ideas as well as deepen their understanding of music.

Using Tonal Context with Movable Systems

For movable systems an important point of context is its key and mode. For “C Major”, the key is “C” and the mode is “Major” (also called Ionian). Key and mode are important as they define the tonal framework, i.e., which pitches are “in” and which are “out”. It also defines the function of the pitches within the framework. This is why for scale degrees 1, 2, 3, 4, and 5, the expected result for C major is C, D, E, F, and G (skipping any sharps/flats), while those same scale degrees for D major are D, E, F#, G, and A. The set of pitches that make up C major have no sharps or flats, so they are skipped. D major has two sharps, F# and C#. The F# is the 3rd scale degree for D major.

Scale Degree with *Set Key* is a very powerful tool for expression. It is also very common in music pedagogy. However, because the number values 1-7 are hard wired into this system, it is a tool that works best to express seven-pitch tonal frameworks (e.g. major, minor, and other common seven pitch scales). For musical ideas where a more purely mathematical form of expression is required, Music Blocks offers the user the *nth Modal Pitch* block.

nth Modal Pitch is similar to *Scale Degree* in that it is a movable system that uses numbers to express pitches. However, unlike *Scale Degree*, *nth Modal Pitch* starts at 0, allows for negative numbers, and is not restricted to a seven-pitch tonal framework. 0 is the first pitch of the mode, 1 is the next pitch, 2 is the pitch above that, etc. -1 is the pitch before the first pitch of the mode. This tool is especially helpful for expressing a musical idea that requires computation as you can run computations directly on the number value. It is also helpful if you are, for example, creating music in a whole tone (six note) pitch space. In the case of *Set Key* set to “whole tone”, 6 would be the octave above.

Pitch Number, MIDI, and Set Pitch Number Offset

Pitch Number is similar to *nth Modal Pitch* in that it is a zero-based, mathematical system to express pitches. However, unlike *nth Modal Pitch*, *Pitch Number* disregards any tonal framework. It is also chromatic by default, meaning that its pitch space includes the sharp/flat pitches (black keys on piano) as well as the natural pitches (white keys on piano). By default, middle C (C_4) is 0. The C major scale in the 4th octave is 0, 2, 4, 5, 7, 9, and 11. 12 is the C an octave above middle C (C_5). This system is useful mathematically, but because it disregards key, it is difficult to control when creating music. That being said, fretted instruments such as ukulele and guitar use this system to express pitch, so it is a good system for expressing how these instruments work.

MIDI also uses a similar system to *Pitch Number* to express pitches, but the 0 is offset from Music Blocks’ default. In order to change the sounding pitch of 0 for *Pitch Number*, use *set pitch number offset*. This makes *Pitch Number* blocks behave as a relative system as it transposes the pitches up or down accordingly (but has no effect on key).

Two Subsystems for Movable

For Movable Do, there exists yet two more systems. One system, which we call *Movable=Do*, allows the user to express solfege syllables in relation to the Major mode. Therefore, if a user were to specify A minor, then La would be A, the first scale degree in A Minor. The other system, which we call *Movable=La*, allows the user to express solfege in relation to the particular mode specified. Therefore, if a user were to specify A Minor, then Do would be A. *Scale Degree* works like *Movable=La* by default such that 1 is always the first pitch of a given mode.

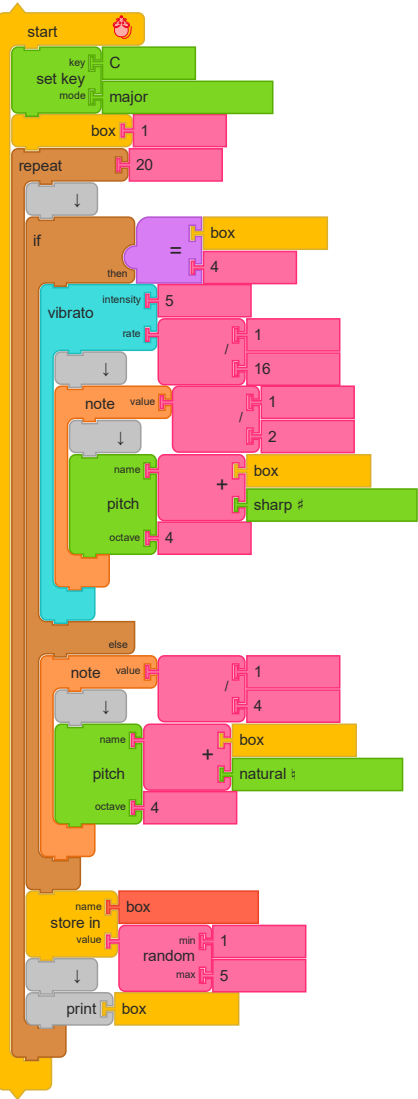
Because some users may want to explicitly spell out all of the pitches regardless of the chosen key, we allow them to use Scale Degree with the *Movable Do* block (remember, Scale Degree works like Movable=La by default). Please see [this code](#) as an example.

The following chart describes the behavior of different blocks depending on whether or not a *Movable Do* block is present.

Block(s)	Fixed or Movable? (Do or La?)	Set Key Transformation?
Alphabet Pitch	Fixed	No effect.
Solfege	Fixed by default	No effect.
Solfege and Movable=Do	Specified via "movable" block set to Do	Yes.
Solfege and Movable=La	Specified via "movable" block set to La	Yes. Works like Scale Degree.
n th modal pitch	Movable	Yes. Good for modes of any length.
Scale Degree	Movable	Yes. Most useful for 7 note systems. Works just like Movable=La for Solfege by default.
Scale Degree and Movable=Do	Movable	Yes. When preceded by Movable=Do, the user can be explicit in their spelling.
Scalar Step	Movable	Yes. Navigates up/down within <i>nth modal pitch</i> space.
Scalar Interval	Movable	Yes. Adds above/under within <i>nth modal pitch</i> space.
Scalar Inversion	Movable	Yes. Inversion around a specified axis within <i>nth modal pitch</i> space.
Pitch Number	Movable	No effect. Pitches can be transformed via Set Pitch Number Offset.

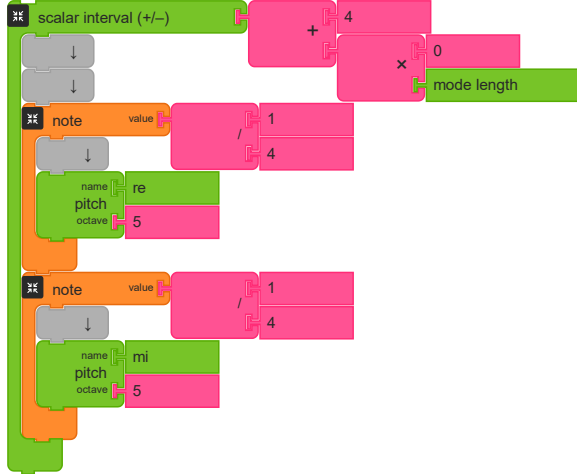
Illustrative example:

The following example demonstrates how the Scale Degree functionality combines math and musical modifiers. When combining numbers and accidentals, it recreates the same functionality as the *Scale Degree* block.



Scale Degree Improv

3.2.7 Intervals



The *Scalar interval* block calculates a relative interval based on the current mode, skipping all notes outside of the mode. For example, a *fifth*, and adds the additional pitches to a note's playback. In the figure, we add *La* to *Re* and *Ti* to *Mi*.

As a convenience, a number of standard scalar intervals are provided on the *Intervals* palette: *Unison*, *Second*, *Third*, ..., *Seventh*, *Down third*, and *Down sixth*.

The *Scalar interval measure* block can be used to measure the number of scalar steps between two pitched.

3.2.7.1 Absolute Intervals

Absolute (or semi-tone) intervals are based on half-steps.



The *Augmented* block calculates an absolute interval (in half-steps), 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*.

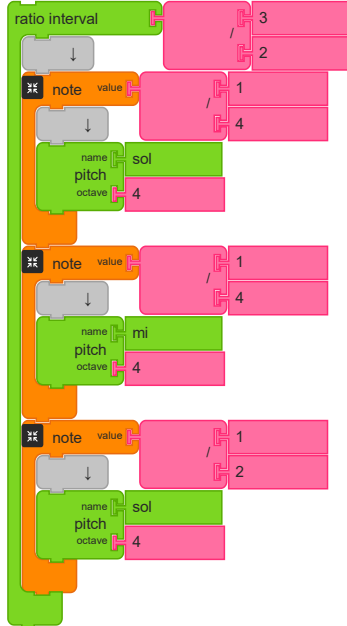
As a convenience, a number of standard absolute intervals are provided on the *Intervals* palette: *Major 2*, *Minor 3*, *Perfect 4*, *Augmented 6*, *Diminished 8*, et al.

The *Doubly* block can be used to create a double augmentation or double diminishment.

The *Semi-tone interval measure* block can be used to measure the number of half-steps between two pitches.

3.2.7.2 Ratio Intervals

Another way to think about intervals is in terms of ratios. For example, a ratio of 2:1 would be an octave shift up; 1:2 would be an octave shift down; 3/2 would be a fifth.

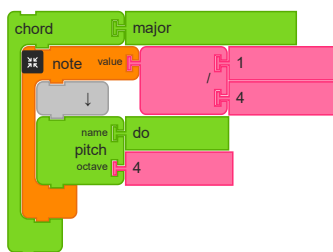


The *Ratio Interval* block lets you generate an interval based on a ratio.

3.2.8 Chords

A chord is a group of notes that are played together (often used for harmony in music). There are triads (three notes), tetrachords (four notes), and even five-, six-, and seven-note chords.

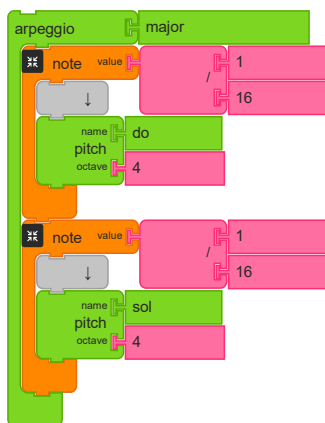
The *Chord* block builds a chord from a base note.



We support many basic chords:

chord	intervals	example
major	1 4 7	C major C - E - G
minor	1 3 7	C minor C - E \flat - G
dominant 7	1 4 7 10	C7 C - E - G - B \flat
minor 7	1 3 7 10	Cmin7 C - E \flat - G - B \flat
major 7	1 4 7 11	Cmaj7 C - E - G - B

The *Arpeggio* block also builds a chord from a base note, but rather than playing all of the pitches at once, each pitch is played in sequence.



In the example above, since the Major chord intervals are 1 4 7, the notes played are do, mi, sol, sol, ti, mi.

3.2.9 Inversion

The *Invert* block will rotate a series of notes around a target note. There are three different modes of the *Invert* block: *even*, *odd*, and *scalar*. In *even* and *odd* modes, the rotation is based on half-steps. In *even* and *scalar* mode, the point of rotation is the given note. In *odd* mode, the point of rotation is shifted up by a $\frac{1}{4}$ step, enabling rotation around a point between two notes. In "scalar" mode, the scalar interval is preserved around the point of rotation.



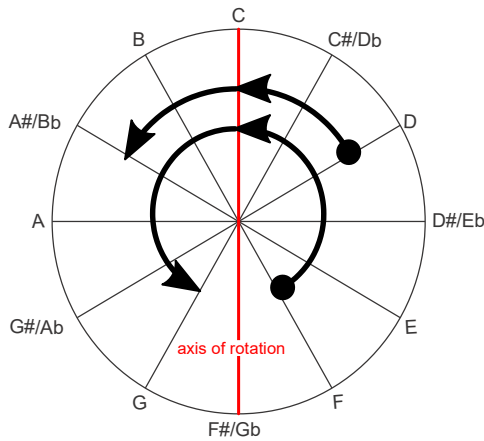
NOTE: The initial `c5` pitch (as a half note) remains unchanged (in all of the examples) as it is outside of the invert block.

The above example code has an *even* inversion for two notes `f5` and `d5` around the reference pitch of `c5`. We would expect the following results:

Even inversion

Starting pitch	Distance from <code>c5</code>	Inverse distance from <code>c5</code>	Ending pitch
<code>f5</code>	5 half steps <i>above</i>	5 half steps <i>below</i>	<code>g4</code>
<code>d5</code>	2 half steps <i>above</i>	2 half steps <i>below</i>	<code>b4</code>

This operation can also be visualized on a pitch clock. The arrows on the following diagram point from the starting pitch, around the axis of the reference pitch, to its destination ending pitch.



In standard notation the result of this *even* inversion operation is depicted in the second measure of the following example. The first measure is the original reference.

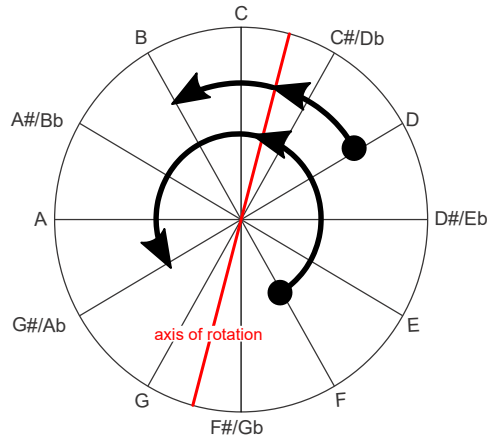


Underneath the *even* inversion in the example code is an *odd* inversion for the same two notes of `f5` and `d5` around the same reference pitch of `c5`. We would expect the following results:

Odd inversion

Starting pitch	Distance from midway-point between <code>c5</code> and <code>c#5</code>	Inverse distance from midway-point between <code>c5</code> and <code>c#5</code>	Ending pitch
<code>f5</code>	4.5 half steps <i>above</i>	4.5 half steps <i>below</i>	<code>A#4</code>
<code>d5</code>	1.5 half steps <i>below</i>	1.5 half steps <i>above</i>	<code>B4</code>

This operation can be visualized on a pitch clock similar to *even* inversion except offset in-between `c5` and `c#5` (i.e. quarter step *above* `c5`).



In standard notation the result of this *odd* inversion operation is depicted in second measure of the following example. The first measure is the original reference. NOTE: The `c5` pitch remains unchanged as it is not operated upon in the example block code (above). If it were contained in the operation it would be changed to `c#5` (i.e. `c5` is 0.5 half steps *below* the axis of rotation, so the result of an inversion around `c5` and *odd* would be 0.5 half steps *above* the axis of rotation).

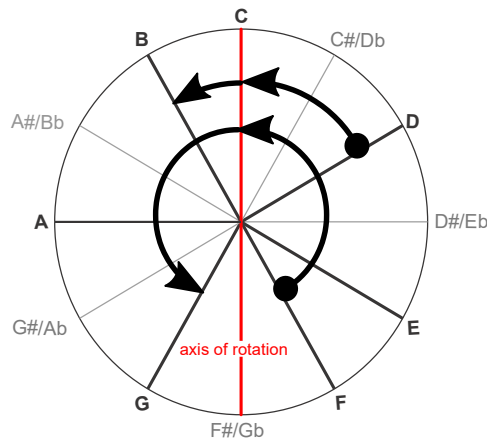


Scalar inversion

Underneath the *even* and *odd* inversion blocks in the example code is an inversion block set to *scalar*. We would expect the following results:

Starting pitch	Scalar distance from <code>c5</code> (in steps)	Inverse scalar distance from <code>c5</code> (in steps)	Ending pitch
<code>f5</code>	3 above (C5 → D5 → E5 → F5)	3 below (C5 → B4 → A4 → G4)	<code>g4</code>
<code>d5</code>	1 above (C5 → D5)	1 below (C5 → B4)	<code>b4</code>

This operation can be visualized on a pitch clock similar to *odd* and *even* except that all non-scalar pitches (i.e. pitches outside the chosen key) are skipped. NOTE: The scalar pitches are shown in bold in the following pitch clock diagram.



In standard notation the result of *scalar* inversion operation is depicted in the second measure of the following example. The first measure is the original reference.



In the *invert (even)* example above, notes are inverted around `c5` . In the *invert (odd)* example, notes are inverted around a point midway between `c5` and `c#5` . In the *invert (scalar)* example, notes are inverted around `c5` , by scalar steps rather than half-steps.

3.2.10 Converters

Converters are used to transform one form of inputs into other, more usable form of outputs. This section of the guide will talk about the various conversion options Music Blocks has to offer.

Generalized shape of a converter is:



where the right argument is converted accordingly, and output is received on the left side.

Note: Before an introduction of the different types of converters, a little introduction on Y staff in Music Blocks. Staff is a set of horizontal lines and spaces and different positions along Y axis represents different notes. [C, D, E, F, G, A, B]



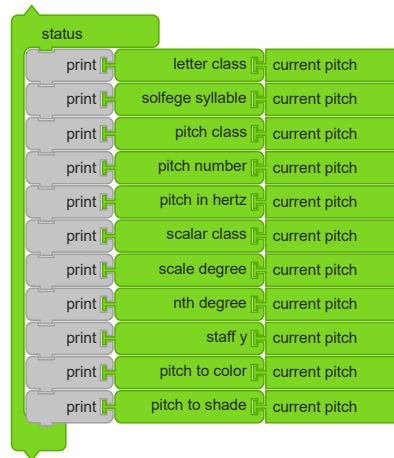
3.2.9.1 Y to Pitch



This converter takes input in the form of a number that represents Staff Y position in pixels, and processes the value such that it can be used with certain pitch blocks (pitch number, nth modal pitch, pitch) to produce notes corresponding to given Staff Y position as an argument.

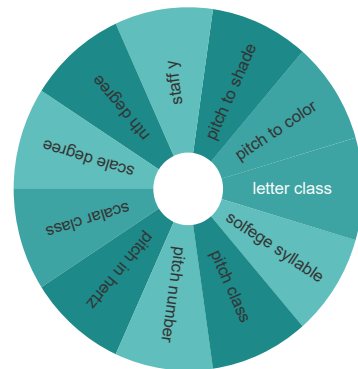
Additionally, the block can be plugged into a print block to view the converted note value.

3.2.9.2 Pitch converter



Pitch converter offers a range of options through a pie-menu based interface and it can potentially convert or extract info out of the current playing pitch using the current pitch block as an input. It can also take custom input in form of solfege, hertz, pitch number etc.

All these options are provided in the form of a pie-menu which can be accessed simply by clicking on the converter.



Below explained is the utility of every conversion option:

0. Alphabet:

Prints the alphabet data of the note being played e.g A, B, C, D, E, F, G, including accidentals.

1. Alphabet class:

Prints the alphabet data of the note being played e.g A, B, C, D, E, F, G. It doesn't print any info regarding accidentals.

2. Solfege Syllable:

Similar to Alphabet class, returns the data in form of solfege e.g do, re, mi. It too, gives no info regarding accidentals.

3. Pitch class:

Returns a number between 0 to 11, corresponding to the note played, where C is 0 and B is 11. Each increase in the number signifies an increase by one semitone.

4. Scalar class:

Returns a number between 1-7 corresponding to the scale degree of the note being played, with reference to the chosen mode. Provides no info regarding accidentals.

5. Scale Degree:

Intuitively, returns the scale degree of the note being played with reference to the chosen mode. It can also be thought of as Scalar class with accidentals.

6. N^th Degree:

Zero-based index of the degree of note being played in the chosen mode.

7. Pitch in Hertz:

Returns the value in hertz of the pitch of the note being currently played.

8. Pitch Number:

Value of the pitch of the note currently being played. It is different from Pitch class in the way that it can go below 0 and above 11 depending upon the octave.

9. Staff Y:

Returns the Y staff position of the note being played according to staff dimensions. It takes into account only the alphabet class, no accidental info is processed.

3.2.9.3 Number to Octave



This converter takes a numeric value which denotes pitch number and returns the octave corresponding to that pitch number.

3.2.9.4 Number to Pitch



This converter takes a numeric value which denotes pitch number and returns the pitch name corresponding to that pitch number. No octave is inferred.

Converter Name	Description
alphabet	Converts pitch to letter (as defined above). G maps to G. G# maps to G#.
alphabet class	Converts pitch to letter (as defined above). G maps to G. G# also maps to G.
solfege syllable	Converts pitch to solfege (as defined above). G maps to sol.
solfege class	Converts pitch to solfege class (as defined above). G maps to sol. G# maps to sol.
pitch class	Converts pitch to pitch class (as defined above). G maps to 7.
scalar class	Converts pitch to scalar class (as defined above). G maps to 5.
scale degree	Converts pitch to scale degree (as defined above). G maps to 5.
nth degree	Converts pitch to nth degree (as defined above). G maps to 4.
staff y	Maps the current pitch to a y value that corresponds to a position on the staff. G4 maps to 50.
pitch number	Converts pitch to pitch number (as defined above). G maps to 7.
pitch in hertz	Converts pitch to hertz. G4 maps to 392Hz.
pitch to color	Converts pitch to a color value (0-100). C maps to 0, G maps to 58.3, etc.
pitch to shade	Coverts the octave value of the current pitch to a shade. Octave 4 maps to 50.