



# Teaching a Computer to Sing

University of Massachusetts Lowell  
Bartlett Community Partnership School

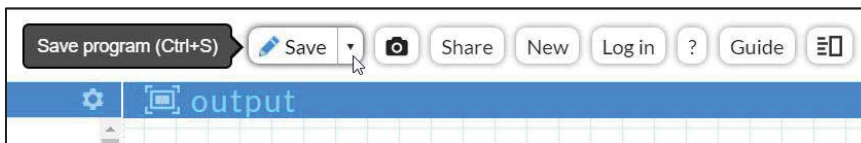


## Using TACTS Pencil Code Functions

Version 2.1, updated March 5, 2017

Mr. Jesse has simplified the way to use his **sing** and **singsay** functions and added new functions to display and animate your own pictures. All of these functions can now be loaded with a single statement in your Pencil Code programs. This document explains how to do that and the *parameters* (additional information) you need to pass to the functions to make them work.

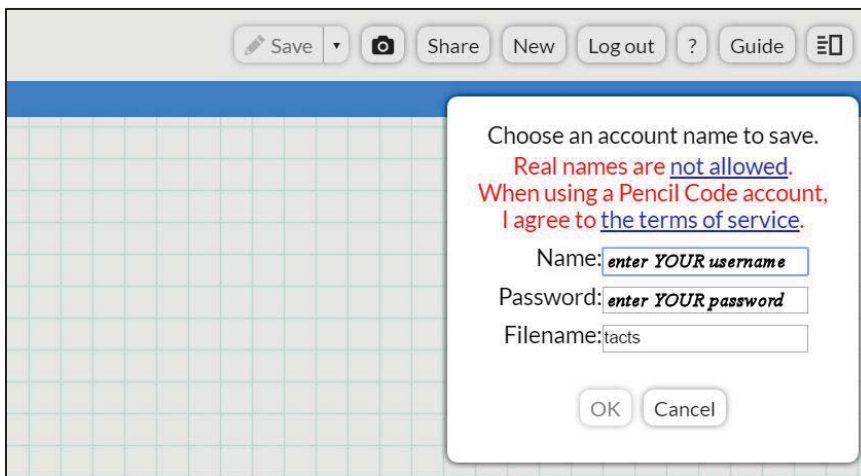
1. Open your browser and type **drjay.pencilcode.net/edit/tacts** in the address field.
2. After that page opens, click the down arrowhead (▼) just to the right of the **Save** button.



3. This will open a dropdown menu. Click **Copy and Save As...**

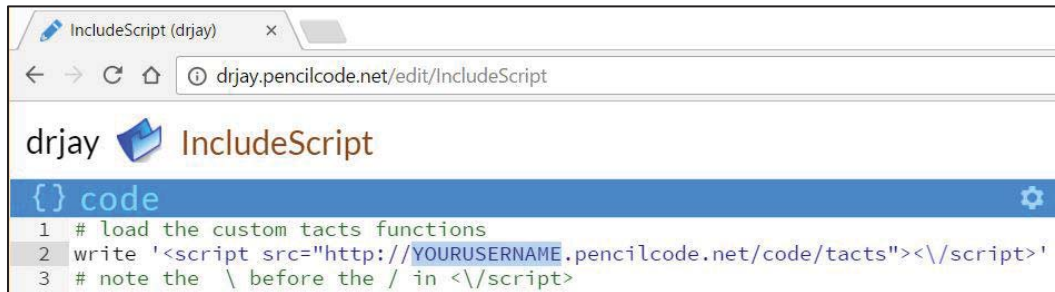


4. This will open a dialog box. Enter **your** user Name and Password and click the OK button.



5. This will save the **tacts** functions to your own account.

6. Now type **drjay.pencilcode.net/edit/IncludeScript** in the address field of your browser window. After that page opens you will see the following 3-line program.



```
{ } code
1 # load the custom tacts functions
2 write '<script src="http://YOURUSERNAME.pencilcode.net/code/tacts"></script>'
3 # note the \ before the / in </script>
```

7. Copy-and-paste these 3 lines (or at least line 2) into your own program. (The critical code is line 2. Lines 1 and 3 are comments.)
8. Replace **YOURUSERNAME** with your own user name.
9. You can now use the **sing** and **singsay** functions as before, with one small change: all functions need to be preceded by **tacts**. (including the dot). Thus, to use the **sing** function, you now type:

```
tacts.sing 1, "C D E F G"
```

And to use the **singsay** function you now type:

```
tacts.singsay 1, "C", "do"
```

**Important Note:** The **tacts** functions cannot be mixed with the standard Pencil Code functions. If you choose to use the **tacts** functions in your program, you must do everything with these functions.

## Summary of Function Headers

*Note that parentheses are required after all the **Get** functions.*

```
tacts.CreatePianos nPianos, vertical_separation
nPianos = tacts.GetNPianos()
tacts.MovePiano nPiano, dx, dy
tacts.SetKey "key"
current_key = tacts.GetKey()
tacts.SetTempo tempo
current_tempo = tacts.GetTempo()
tacts.SetTime "timesig"
current_time_signature = tacts.GetTime()
tacts.sing nPiano, "series_of_notes"
tacts.singsay nPiano, "note", "lyric"
picture_handle = tacts.ShowPicture "url", width, height, x, y
tacts.SizePicture picture_handle, multiplier
```

## Index to Functions

tacts.CreatePianos <i>nPianos</i> , <i>vertical_separation</i> .....	3
<i>nPianos</i> = tacts.GetNPianos() .....	3
tacts.MovePiano <i>nPiano</i> , <i>dx</i> , <i>dy</i> .....	3
tacts.SetKey "key" .....	4
<i>current_key</i> = tacts.GetKey() .....	4
tacts.SetTempo <i>tempo</i> .....	5
<i>current_tempo</i> = tacts.GetTempo() .....	5
tacts.SetTime "timesig" .....	5
<i>current_time_signature</i> = tacts.GetTime() .....	5
tacts.sing <i>nPiano</i> , "notes" .....	6
tacts.singsay <i>nPiano</i> , "note", "lyric" .....	6
<i>picture_handle</i> = tacts.ShowPicture "url", <i>width</i> , <i>height</i> , <i>x</i> , <i>y</i> .....	7
tacts.SizePicture <i>picture_handle</i> , <i>multiplier</i> .....	8

### tacts.CreatePianos *nPianos*, *vertical\_separation*

*Purpose* This function creates 1 or more pianos for playing notes. It must be called before calling the `sing` or `singsay` or `MovePiano` functions.

*Parameters* ***nPianos*** = the number of pianos that you wish to create. If *nPianos* is not supplied, it will be set to 1.  
***vertical\_separation*** = the distance (in pixels) that each piano will be separated from each other in the vertical direction. If *vertical\_separation* is not supplied, it will be set to 150 (pixels).

*Example* `tacts.CreatePianos 3, 125`  
This function call creates 3 pianos and separates them vertically by 125 pixels.

### *nPianos* = tacts.GetNPianos()

*Purpose* This function returns the number of pianos that the system knows about.

*Parameters* This function takes no parameters. Therefore, the name of the function **must** be followed by parentheses.

*Return Value* This function returns the number of active pianos.

*Examples* `write "There are currently " + tacts.GetNPianos() + " pianos."`

## **tacts.MovePiano** *nPiano, dx, dy*

*Purpose* This function moves a piano **dx** pixels to the right and **dy** pixels down. If **dx** or **dy** is negative, the piano is moved to the left or up, respectively.

*Parameters* **nPiano** = number of piano to move. This parameter is required and must be less than or equal to the number of pianos created.

**dx** = number of pixels to move the piano to the right. If **dx** is negative, the piano moves to the left. If the **dx** is not supplied, it is set to 0, which indicates no move in the horizontal direction.

**dy** = number of pixels to move the piano down. If **dy** is negative, the piano moves up. If the **dy** parameter is not supplied, it is set to 0, which indicates no move in the vertical direction.

*Example* `tacts.MovePiano 2, 100, -50`

This function call moves piano #2 100 pixels to the right and 50 pixels up.

*Conditions* The number of the piano in the first parameter must be less than or equal to the number of pianos you created with the `tacts.CreatePianos` function.

## **tacts.SetKey** "key"

*Purpose* This function sets the key (in ABC notation format) of the music to be played. If you do not call this function, the key will be set to "C".

*Parameters* **key** = an ABC notation key specification, enclosed in double quotes. This parameter is required.

*Examples*

<code>tacts.SetKey "F"</code>	# key of F (1 flat)
<code>tacts.SetKey "Bb"</code>	# key of B-flat (2 flats)
<code>tacts.SetKey "F#"</code>	# key of F-sharp (6 sharps)

## *current\_key* = **tacts.GetKey()**

*Purpose* This function returns the current key in which music will be played.

*Parameters* This function takes no parameters. Therefore, the name of the function **must** be followed by parentheses.

*Return Value* This function returns the current key in which music will be played.

*Examples* `write "The current key is " + tacts.GetKey()`

## **tacts.SetTempo** *tempo*

*Purpose* This function sets the tempo (number of beats per minute) of the music to be played. If you do not call this function, the tempo will be set to 120 beats per minute.

*Parameters* **tempo** = the number of beats per minute. This parameter is required.

*Examples* `tacts.SetTempo 60 # play at half normal speed`  
`tacts.SetTempo 240 # play at double normal speed`

## **current\_tempo = tacts.GetTempo()**

*Purpose* This function returns the current tempo with which music will be played.

*Parameters* This function takes no parameters. Therefore, the name of the function **must** be followed by parentheses.

*Return Value* This function returns the current tempo (in beats per minute) with which music will be played.

*Examples* `write "The current tempo is " + tacts.GetTempo() +`  
`" beats per minute."`

## **tacts.SetTime** "*timesig*"

*Purpose* This function sets the time signature (in ABC notation format) of the music to be played. If you do not call this function, the time signature will be set to "4/4".

*Parameters* **timesig** = an ABC notation time signature specification, enclosed in double quotes. This parameter is required.

*Examples* `tacts.SetTime "3/4" # 3 beats per measure, quarter note = 1 beat`  
`tacts.SetTime "6/8" # 6 beats per measure, eighth note = 1 beat`

## **current\_time\_signature = tacts.GetTime()**

*Purpose* This function returns the current time signature with which music will be played.

*Parameters* This function takes no parameters. Therefore, the name of the function **must** be followed by parentheses.

*Return Value* This function returns the current time signature with which music will be played.

*Examples* `write "The current time signature is " + tacts.GetTime()`

## tacts.sing *nPiano*, "notes"

- Purpose** This function plays a series of notes (in ABC notation) on a specific piano.
- Parameters** *nPiano* = number of piano on which to play the notes. This parameter is required and must be less than or equal to the number of pianos created.  
*notes* = series of notes to play, enclosed in double quotes. This parameter is required.
- Examples** `tacts.sing 2, "Z/2 E/2 E/2E/2 E/2F/2G/2B/2"`  
This function call plays the first singing line of Bruno Mars's song *Lighters*,  
"This one's for you\_\_ and me,\_\_\_."  
`forward = "C D E F G A B C"`  
`reverse = "C' B A G F E D C"`  
`tacts.sing 1, forward`  
`tacts.sing 2, reverse`  
These variable initializations and function calls demonstrate setting variables to series of notes and then playing those series simultaneously on two different pianos.
- Conditions** You must create at least one piano with the `tacts.CreatePianos` function (explained above) before calling this function.  
The number of the piano in the first parameter must be less than or equal to the number of pianos you created with the `tacts.CreatePianos` function.

## tacts.singsay *nPiano*, "note", "lyric"

- Purpose** This function plays a *single* note (in ABC notation) on a specific piano and prints the lyric at the same time.
- Parameters** *nPiano* = number of piano on which to play the note. This parameter is required.  
*note* = the note to play, enclosed in double quotes. This parameter is required.  
*lyric* = the lyric to display, enclosed in double quotes. This parameter is required.
- Examples** `tacts.singsay 1, "Z/2", ""`  
`tacts.singsay 1, "E/2", "This"`  
`tacts.singsay 1, "E/2", " one's"`  
`tacts.singsay 1, "E/2", " for"`  
`tacts.singsay 1, "E/2", " you"`  
`tacts.singsay 1, "F/2", "___"`  
`tacts.singsay 1, "G/2", " and"`  
`tacts.singsay 1, "B/2", " me"`  
These function calls play and show the lyrics for the first singing line of Bruno Mars's song *Lighters*, "This one's for you\_\_ and me."  
Note that there can be a blank lyric for a rest, but that must be specified as "". It cannot simply be omitted.

The blank spaces before some words avoid the problem of words running into each other. That is, the above code prints “This one’s for you \_\_\_ and me” rather than “Thisone’sforyou\_\_\_andme.”

If you want a word to start on a new line, precede it with a vertical bar (`|`), as in `"|and"`. This will make the word “and” appear on the next line rather than next to the word that precedes it.

*Conditions* You must create at least one piano with the `tacts.CreatePianos` function (explained above) before calling this function.  
The number of the piano in the first parameter must be less than or equal to the number of pianos you created with the `tacts.CreatePianos` function.  
This function only works with one piano at a time. That is, unlike the `sing` function, it will only work with one part.

### *picture\_handle* = `tacts.ShowPicture "url", width, height, x, y`

*Purpose* This function displays a picture stored on a web server other than the Pencil Code web server. It can be used to display your own pictures. To do that, please see Mr. Jesse about how to get your pictures online.

*Parameters* *url* = the URL or the picture to display, enclosed in double quotes. This parameter is required.  
*width* = the width in pixels at which the picture should be displayed. This parameter is required, but if it is set to 0, the width will be proportional to the height as in the original picture.  
*height* = the height in pixels at which the picture should be displayed. This parameter is required, but if it is set to 0, the height will be proportional to the width as in the original picture.  
*x* = the *x* (horizontal) coordinate at which the upper left-hand corner of the picture should appear. This parameter is required.  
*y* = the *y* (vertical) coordinate at which the upper left-hand corner of the picture should appear. This parameter is required.

*Return Value* This function returns a *picture\_handle* that can be used to move and resize the picture. Moving a picture can be accomplished using the standard Pencil Code functions in the `Move` category, while resizing can be accomplished using the `tacts.SizePicture` function.

*Examples* `tacts.ShowPicture "tactspictures/Hessell_Taliyah_1242x1242.jpg", 0, 200, 400, 0`

This first example displays a picture with a height of 200 pixels and width proportional to the original, at a horizontal location of 400 pixels and a vertical location of 0 pixels (at the top of the display area).

```
pict2 = tacts.ShowPicture "tactspictures/Taliyah_Chan.jpg", 0, 200, 0, 150
```

This second example displays a picture with a height of 200 pixels and width proportional to the original, at a horizontal location of 0 pixels (at the left of the display area) and a vertical location of 150 pixels.

The advantage of this second example is that it saves a “handle” to the picture so that it can be moved with commands such as the following.

```
for k in [1..4]
  pict2.slide 200
  pict2.rt 90
```

Both of these examples assume that the picture to be displayed is located on the `http://tacts.000webhostapp.com/` server. If the picture is located on another server, use the full URL as follows.

```
pict3 = tacts.ShowPicture "http://multifiles.pressherald.com/
  uploads/sites/4/2017/02/1148251_Exchange_Teaching_a_Compute.jpg",
  500, 0, 0, 0
```

If pictures overlap, the pictures loaded last will overlap those loaded earlier.

## **tacts.SizePicture** *picture\_handle, multiplier*

*Purpose* This function changes to size of a picture.

*Parameters* ***picture\_handle*** = the picture handle returned by a call to the **ShowPicture** function. This parameter is required.  
***multiplier*** = the proportion by which the picture’s size is to be changed. This parameter is required. If the multiplier is greater than 1, the picture size will be increased. If it is less than 1, the picture size will be decreased.

*Examples*

```
keydown 'G', ->
  tacts.SizePicture pict1, 1.1
keydown 'H', ->
  tacts.SizePicture pict2, 1.1
keydown 'S', ->
  tacts.SizePicture pict1, 0.9
keydown 'D', ->
  tacts.SizePicture pict2, 0.9
```

These function calls set up actions when keys are pressed.

- pressing the G key increases the size of picture #1 by 10% (times 1.1)
- pressing the H key increases the size of picture #2 by 10%
- pressing the S key decreases the size of picture #1 by 10% (times 0.9)
- pressing the D key decreases the size of picture #2 by 10%