

Quick Reference

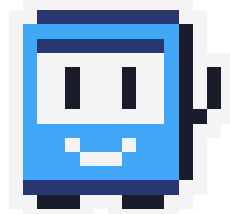
for

Lua



and

TIC-80



www.tic80.com/learn



www.lua.org/manual/5.4/

TIC-80 Terminal Commands

new lua

Creates a new Lua cartridge.

edit

Switches to the code editor for the loaded cartridge.

save

Saves the loaded cartridge to its file.

save [filename]

Write the loaded cartridge to the filename. The .tic extension is automatically added.

load [filename]

Loads a cartridge with the filename. The .tic extension is automatically added.

export html [filename]

Exports to a webpage.

export win|linux|mac [filename]

Exports an executable to run natively on the desired OS.

exit

Exits the TIC-80 terminal.

del [filename]

Deletes a file.

folder

Opens the folder with the cartridges in the file explorer

Lua Quick Reference

Comments:

Single-line Comments	-- <i>comment</i>	-- This is a comment.
Multi-line Comments	--[[<i>comment</i>]]	--[[This is a comment.]]

Arithmetic Operators:

Addition	+	21+21 -- yields 42
Subtraction	-	3-2 -- yields 1
Multiplication	*	5*4 -- yields 20
Division	/	64/2 -- yields 32
Modulus	%	8%3 -- yields 2
Exponent	^	2^3 -- yields 8
Negation	-	-5

Comparison Operators:

Is Equal	==	5 == 5 -- yields true
Is Not Equal	~=	5 ~= 5 -- yields false
Greater Than	>	42 > 32 -- yields true
Less Than	<	42 < 32 -- yields false
Greater Than Or Equal to	>=	7 >= 7 -- yields true
Less Than Or Equal to	<=	7 <= 7 -- yields true

Logical Operators:

and	true and false -- yields false	
or	true or false -- yields true	
not	not 5 == 5 -- yields false	

Other Operators:

Concatenation	..	"Hello ".."World" -- yields "Hello World"
Length	#	#"Hello World" -- yields 11

Variable Declarations:

Global Variables	<i>variable_name</i> = <i>value</i>	x = 5
Local Variables	<i>local variable_name</i> = <i>value</i>	local x = 5

Data Types:

Boolean	<i>true or false</i>
Number	<i>1, 42, 3.14</i>
String	<i>"Hello World" or 'Hello World'</i>
Table	<i>{'foo', 'bar', 'baz'} {['banana'] = 'yellow', ['apple'] = 'red'} {first = 'Andrew', last = 'Teesdale'}</i>

Flow Control:

if	Executes if the condition is true	<i>if condition then statement elseif condition then statement else statement end</i>
for	Sets a variable to the start value and executes until the stop value is reached	<i>for var = start, stop, step do statement end</i>
while	Executes while the condition is true	<i>while condition do statement end</i>
repeat-until	Executes until the condition is true	<i>repeat statement until condition</i>

Function Declaration:

```
function function_name(arg1, arg2, arg3, ...)
    statements
    return value
end
```

Function Calling:

```
function_name(arg1, arg2, arg3, ...)
```

Error Catching:

```
local status, err = pcall(function, arg1, arg2, ...)
if not status then
    statement
end
```

Error Throwing:

```
function throw_error()
    error(error_value)
end
```

TIC-80 System Calls

Quick Reference

Main Functions

TIC()

The code entry point. This function is called 60 times every second.

exit()

Stops the cartridge.

reset()

Restarts the cartridge.

trace(message color=15)

Prints a debug message to the console.

Input Functions

btn(id) -> pressed

Returns true if the button specified by id is currently being pressed.

btnp(id hold=-1 period=-1) -> pressed

Returns true if the button specified by id is currently being pressed and wasn't pressed the previous tick. The hold parameter indicates how long in ticks until the function can read the specified button again. The period parameter indicates how often the function can check the button in ticks.

key(code=-1) -> pressed

Returns true if the key specified by code is currently being pressed.

keyp(code=-1 hold=-1 period=-1) -> pressed

Returns true if the key specified by code is currently being pressed and wasn't pressed the previous tick. The hold parameter indicates how long in ticks until the function can read the specified button again. The period parameter indicates how often the function can check the button in ticks.

mouse() -> x y left middle right scrollx scrolly

This function returns the mouse coordinates and a boolean value for the state of each mouse button, with true indicating that a button is pressed.

Drawing Functions

circ(x y radius color)

This function draws a filled circle of the desired radius and color with its center at x, y.

circb(x y radius color)

Draws the circumference of a circle with its center at x, y using the radius and color requested.

elli(x y a b color)

This function draws a filled ellipse of the desired a, b radiuses and color with its center at x, y.

ellib(x y a b color)

This function draws an ellipse border with the desired radiuses a b and color with its center at x, y. It uses the Bresenham algorithm.

line(x0 y0 x1 y1 color)

Draws a straight line from point (x0,y0) to point (x1,y1) in the specified color.

rect(x y w h color)

This function draws a filled rectangle of the desired size and color at the specified position.

rectb(x y w h color)

This function draws a one pixel thick rectangle border at the position requested.

tri(x1 y1 x2 y2 x3 y3 color)

This function draws a triangle filled with color, using the supplied vertices.

trib(x1 y1 x2 y2 x3 y3 color)

This function draws a triangle border with color, using the supplied vertices.

Time Functions:

time() -> ticks

Returns the number of milliseconds elapsed since the cartridge began execution.

tstamp() -> timestamp

Returns the number of seconds elapsed since January 1st, 1970.

Rendering Functions:

cls(color=0)

Clears the screen with the specified color.

map(x=0 y=0 w=30 h=17 sx=0 sy=0 colorkey=-1 scale=1 remap=nil)

Renders the map cells from (x,y) with the specified width and height on the screen with the specified screen coordinates (sx,sy).

print(text x=0 y=0 color=15 fixed=false scale=1 smallfont=false) -> width

Prints text to the screen.

spr(id x y colorkey=-1 scale=1 flip=0 rotate=0 w=1 h=1)

Draws the sprite number index at the x and y coordinate. The scale parameter indicates how many times larger the sprite will be rendered. The flip parameter indicates how to flip the image. The rotate parameter indicates how many times to rotate the sprite by 90 degrees. The width and height parameters indicate how many sprite tiles are being rendered at once.

Sound Functions:

music(track=-1 frame=-1 row=-1 loop=true sustain=false tempo=-1 speed=-1)

Starts playing a track created in the Music Editor. Call without arguments to stop the music.

sfx(id note=-1 duration=-1 channel=0 volume=15 speed=0)

This function will play the sound with id created in the sfx editor. Calling the function with id set to -1 will stop playing the channel.