Python Syntax Quick Reference for Digital Mood Ring Project

TI-NSPIRE™ CXII PYTHON

Command	Example	Behavior
from module_name import *	`from ti_hub import *`	Imports all the functions in the ti_hub module for use in the program. The ti_hub module includes all the necessary additions needed for the Mood Ring project.
color.rgb(red,green,blue)	`color.rgb(255,0,0)`	Turns the color LED on with the color red. For each color: 0 is off and 255 is full value.
# text comment	`# Sets the color LED to red`	# at the beginning of a line denotes a comment. Comments are a "best practice" by programmers to annotate their code. Comment statements are ignored when the program is run. In the TI-Nspire CXII Python editor, [ctrl]+[T] toggles the statement of the current cursor location from a comment to a statement that will be run.
sleep(seconds)	`sleep(1.5)`	Pauses program for 1.5 seconds.
name_of sensor=temperature("port")	`temp1=temperature("IN 1")`	Creates a temperature sensor object named **temp1** connected to port IN 1. Note: = is the Python operator for storing or assigning values to a variable.
var=name_of_sensor.measurement()	`t=temp1.measurement()`	Reads and stores the current measurement value of the **temp1** sensor object into variable **t** Note: **.measurement()** returns the current measured value of a sensor object.
text_at(row,"text","align")	`text_at(3,"Temperature = "+str(t),"left")`	The text_at function displays a text string on a specified row with an alignment of left, center or right. When variable **t** has a value of 26, the following is displayed on row 3, aligned to the left: Temperature = 26 Note: The **str()** function converts a numeric value to a string. The **+** operator is used to join two strings.
for index in range(start, stop): block	`for n in range(0,10):` `  print(n)`	Repeats the statements in the block ten times, printing the value of the index variable, n, as 0,1,2,…9. The index variable n starts at 0 and increases by 1 with each loop. If n is less than the stop value, 10, the loop continues to repeat. The block starts with a **colon** and includes the indented lines that follow.

Python Syntax Quick Reference for Digital Mood Ring Project

TI-NSPIRE™ CXII PYTHON

<Boolean expression>
value 1 operator value 2
	`2+3==6 (result is false)`
	`x+4>=y (if x=1 and y=3, the result is true)`
	`"enter"!="esc" (result is true)`
		Boolean expressions evaluate to either true or false. The examples show some of the relational operators available from the Built-ins Ops menu.
		Note: == is the Python operator to check equality. >= is the Python operator to check whether the value to the left is greater than or equal to the value on the right.. != is the Python operator to check inequality.

if <Boolean expression>: block	`if t<22:` `color.rgb(0,0,255)`	Checks to determine if the value of variable **t** is less than 22. If the statement is "true" then the commands in the **if** block are executed. Otherwise, the block is skipped. In the example, when the temperature is less than 22, the calculator will send a command to the TI-Innovator to set the color rgb LED to be blue.
if <Boolean expression> and <Boolean expression>: block	`if t>= 22 and t<24:` `color.rgb(0,255,0)`	If both expressions are true the **and** function is "true", then the block is executed. Otherwise, the **and** function returns false, and the block is skipped. In the example, when the temperature is greater than or equal to 22 and less than 24, the calculator will send a command to the TI-Innovator to set the color rgb LED to be green.
get_key()	`key_pressed=get_key()`	get_key() is a function that returns a string with the value associated with the last key pressed while a program is running. The value of the escape key is "esc". In the example, pressing the escape key updates the variable **key_pressed** to "esc".

while get_key() != "esc":
block
	`while get_key() != "esc":`
	`t=temp1.measurement()`
	`text_at(3,"Temperature ="+str(t),"left")`
	`sleep(1)`
		While loops repeat the statements in the block if the condition at the top of the loop is true. In the example, looping continues until the escape key is pressed. Not pressing a key or pressing any key but escape means that get_key() will return a value that is not equal to "esc". The loop condition is true and looping continues. If the escape key is pressed, get_key() returns "esc". The condition will evaluate as "esc" not equal to "esc", which is false. A false result means that the loop statements are not repeated. Program execution skips to the statement just after the loop.

See TI-Innovator Hub Technology eGuide for more background on Hub-specific commands – Link

See TI-Nspire CXII Python Programming eGuide for more background on Python commands - Link