Compatible with the Root® and Create® 3 Robots

iRobot Python Web Playground

Explore the following code snippets to program iRobot Education's Root® or Create® 3 robots using the iRobot Python Web Playground.

Events

@event(robot.when_play) async def when_play(robot):	Commands will run on the robot when the Play button is pressed.
@event(robot.when_bumped, [True, True]) async def bumped(robot): @event(robot.when_bumped, [True, False]) async def bumped(robot):	The robot will sense input from its left and/ or right bumper sensors. Example #1: Any bumper is activated. Example #2: Front left bumper is activated.
@event(robot.when_touched, [True, True, True, True]) async def touched(robot): @event(robot.when_touched, [False, True, False, True]) async def touched(robot): @event(robot.when_touched, [True, False]) async def touched(robot): @event(robot.when_touched, [False, True]) async def touched(robot):	The robot will sense input from its touch sensors. On the Root robots, there are four touch top quadrants. In order, they are: front left, front right, back left, and back right. Example #1: Any touch quadrant is activated. Example #2: Front right and back right quadrants are activated. On the Create 3 robot, there are two buttons. In order, they are: left (•) and right (••). Example #3: Left button is activated (•). Example #4: Right button is activated (••).
@event(robot.when_color_scanned, [ColorID.GREEN]) async def scanned(robot): @event(robot.when_color_scanned, [ColorID.SKIP, ColorID.SKIP, ColorID.GREEN, ColorID.SKIP, ColorID.SKIP]) async def scanned(robot):	The robot will sense input from its color scanner. (Root Pro rt1 robots only). On the Root robots, there are five color scanning zones from left to right: Example #1 will activate when any color zone registers green. Example #2 will activate when the middle of five color zones registers green.
@event(robot.when_light_seen, [robot.LightEvent.BRIGHTER]) async def seen(robot): @event(robot.when_light_seen, [robot.LightEvent.DARKER]) async def seen(robot):	The robot will sense ambient light changes. (Root robots only). Example #1 will trigger when the robot detects that the lighting has become brighter. Example #2 will trigger when the robot detects that the lighting has become darker.

Commands

await robot.move(16)	Command the robot to move forward a specified number of centimeters. Example shows 16 cm distance.
await robot.turn_left(90) await robot.turn_right(90)	Command the robot to rotate clockwise (right) or counter-clockwise (left) for a specified number of degrees. Example #1 shows robot driving counter-clockwise 90°.
await robot.arc_left(90, 4) await robot.arc_right(90, 4)	Command the robot to drive forward or backward in an arc, either clockwise (right) or counter-clockwise (left) for a specified number of degrees along a curve of a specified radius. Example #1 shows robot driving counter-clockwise 90° around a curve with a 4 cm radius.
await robot.reset_navigation()	Command resets the robot's navigation grid origin of (0, 0, 90°) with the center of the robot at the origin whose X axis goes through the right wheel and Y axis goes through the bumper.
await robot.navigate_to(16, 16) await robot.navigate_to(0, 0, 45)	Command the robot to navigate across an invisible grid of centimeter increments. The robot's position is reset to (0, 0, 90°) when the program is started. Example #1 shows a robot navigating to (16 cm, 16 cm) without setting a final heading. Example #2 shows a robot navigating to (0 cm, 0 cm) with a final heading of 45 degrees.
await robot.get_position()	Command the robot to give its current location based on an invisible grid of centimeter increments.
await robot.play_note(440, 0.25) await robot.play_note(Note.A4, 0.25)	Command the robot to play a musical note of a specified frequency for a specified duration. Example #1 and #2 play the note A4 for .25 seconds.
await robot.say('Hello')	Command the robot to communicate in beeps out loud. (Root robots only).
await robot.stop_sound()	The robot will stop sounds currently playing.

Setters

`await robot.set_marker_down()`
`await robot.set_marker_and_eraser_up()`
`await robot.set_eraser_down()`
	Set the robot to either raise or lower its marker or eraser (Root robots only).
	Example #1 will lower the robot's marker.
	Example #2 will raise the robot's marker and eraser.
	Example #3 will lower the robot's eraser.
	Note: Only the Root Pro robot features an eraser.

`await robot.set_lights_on_rgb(0, 0, 255)`
`await robot.set_lights_spin_rgb(0, 0, 255)`
`await robot.set_lights_blink_rgb(0, 0, 255)`
`await robot.set_lights_off()`
	Set the robot's lights to show various colors and patterns.
	Examples #1-3 set the robot's lights blue, using the fully on, spinning, and blinking animations, respectively.
	Example #4 turns the robot's lights off.

`await robot.set_wheel_speeds(10, -10)`	Set the robot's wheel speed (in cm/s) and direction on either side (left/right). Example shows the robot's left wheel driving forward 10 cm/s and right wheel driving backward 10 cm/s. The robot will rotate in place, clockwise.

`await robot.set_left_speed(5)`
`await robot.set_right_speed(-5)`
	Sets one of the robot's wheel motor speeds (in cm/s); the other will continue unchanged.

`await robot.set_name('Flea')`	Set the robot's name to "Flea."

Flow Control

await robot.wait(0.5)	Specify a delay in seconds before the robot moves onto the next line of code. Example tells robot to wait .5 seconds before reading next line.
await hand_over()	Python appears to run all tasks (@events) concurrently. The tasks are not truly running at the same time, but instead are sharing time on your processor. Tasks automatically hand over processor time to other tasks whenever await is called. If you have written a task which spends a lot of time without calling a function which awaits (for example in a very long loop), it's important to call await hand_over() to tell the program that it is okay to hand time over to other tasks. To see this in action, view the 24_tasks_and_loops.py tutorial under the Examples tab.

Getters

await robot.get_version_string()	Gets the robot's version(s) as a human-readable string.
await robot.get_name()	Gets the robot's name.
await robot.get_serial_number()	Gets the robot's serial number.
await robot.get_battery_level()	Gets the robot's battery level.
await robot.get_position()	Gets the robot's current internal position.
await robot.get_accelerometer()	Gets the current values reported by the robot's accelerometer.
await robot.get_bumpers()	Gets the current state of the bumper sensors.
await robot.get_touch_sensors()	Gets the current values reported by the robot's touch sensors.
await robot.get_ipv4_address()	Gets the robot's IP addresses for all of its interfaces. (Create 3 robot only).
await robot.get_ir_proximity()	Gets the values of the robot's IR proximity sensors. (Create 3 robot only).
await robot.get_light_values()	Gets the values of the robot's ambient light sensors. (Root robots only).

Compatible with the Create® 3 Robot

RGB Color Reference Sheet

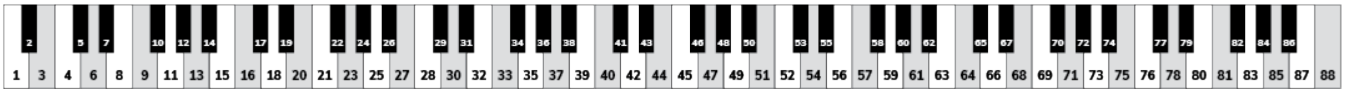
Use these RGB values when setting the color of the LED's in the Ring Light for the Create 3 robot in the iRobot Python Web Playground.

100% Brightness	50% Brightness
(255, 0, 0)	(127, 0, 0)
(255, 64, 0)	(127, 32, 0)
(255, 115, 0)	(127, 57, 0)
(0, 255, 0)	(0, 127, 0)
(101, 197, 181)	(50, 98, 90)
(0, 0, 255)	(0, 0, 127)
(111, 71, 127)	(55, 35, 63)
(255, 255, 255)	(127, 127, 127)
(60, 60, 60)	(0, 0, 0)

Compatible with the Create® 3 Robot

Note Frequencies for Python

The following note values are for use in the iRobot Python Web Playground.



Key #	Note	Hz	Constant
1	A$_0$	**27.50**	
2	A♯$_0$/B♭$_0$	29.14	
3	B$_0$	30.87	
4	C$_1$	32	C1
5	C♯$_1$/D♭$_1$	34	C1_SHARP
6	D$_1$	36	D1
7	D♯$_1$/E♭$_1$	38	D1_SHARP
8	E$_1$	41	E1
9	F$_1$	43	F1
10	F♯$_1$/G♭$_1$	46	F1_SHARP
11	G$_1$	48	G1
12	G♯$_1$/A♭$_1$	51	G1_SHARP
13	A$_1$	**55**	**A1**
14	A♯$_1$/B♭$_1$	58	A1_SHARP
15	B$_1$	61	B1
16	C$_2$	65	C2
17	C♯$_2$/D♭$_2$	69	C2_SHARP
18	D$_2$	73	D2
19	D♯$_2$/E♭$_2$	77	D2_SHARP
20	E$_2$	82	E2
21	F$_2$	87	F2
22	F♯$_2$/G♭$_2$	92	F2_SHARP
23	G$_2$	97	G2
24	G♯$_2$/A♭$_2$	103	G2_SHARP
25	A$_2$	**110**	**A2**
26	A♯$_2$/B♭$_2$	116	A2_SHARP
27	B$_2$	123	B2
28	C$_3$	130	C3
29	C♯$_3$/D♭$_3$	138	C3_SHARP
30	D$_3$	146	D3
31	D♯$_3$/E♭$_3$	155	D3_SHARP
32	E$_3$	164	E3
33	F$_3$	174	F3
34	F♯$_3$/G♭$_3$	184	F3_SHARP
35	G$_3$	195	G3
36	G♯$_3$/A♭$_3$	207	G3_SHARP
37	A$_3$	**220**	**A3**
38	A♯$_3$/B♭$_3$	233	A3_SHARP
39	B$_3$	246	B3
40	C$_4$	261	C4
41	C♯$_4$/D♭$_4$	277	C4_SHARP
42	D$_4$	293	D4
43	D♯$_4$/E♭$_4$	311	D4_SHARP
44	E$_4$	329	E4
45	F$_4$	349	F4
46	F♯$_4$/G♭$_4$	369	F4_SHARP
47	G$_4$	391	G4
48	G♯$_4$/A♭$_4$	415	G4_SHARP
49	A$_4$	**440**	**A4**
50	A♯$_4$/B♭$_4$	466	A4_SHARP
51	B$_4$	493	B4
52	C$_5$	523	C5
53	C♯$_5$/D♭$_5$	554	C5_SHARP
54	D$_5$	587	D5
55	D♯$_5$/E♭$_5$	622	D5_SHARP
56	E$_5$	659	E5
57	F$_5$	698	F5
58	F♯$_5$/G♭$_5$	739	F5_SHARP
59	G$_5$	783	G5
60	G♯$_5$/A♭$_5$	830	G5_SHARP
61	A$_5$	**880**	**A5**
62	A♯$_5$/B♭$_5$	932	A5_SHARP
63	B$_5$	987	B5
64	C$_6$	1046	C6
65	C♯$_6$/D♭$_6$	1108	C6_SHARP
66	D$_6$	1174	D6
67	D♯$_6$/E♭$_6$	1244	D6_SHARP
68	E$_6$	1318	E6
69	F$_6$	1396	F6
70	F♯$_6$/G♭$_6$	1479	F6_SHARP
71	G$_6$	1567	G6
72	G♯$_6$/A♭$_6$	1661	G6_SHARP
73	A$_6$	**1760**	**A6**
74	A♯$_6$/B♭$_6$	1864	A6_SHARP
75	B$_6$	1975	B6
76	C$_7$	2093	C7
77	C♯$_7$/D♭$_7$	2217	C7_SHARP
78	D$_7$	2349	D7
79	D♯$_7$/E♭$_7$	2489	D7_SHARP
80	E$_7$	2637	E7
81	F$_7$	2793	F7
82	F♯$_7$/G♭$_7$	2959	F7_SHARP
83	G$_7$	3135	G7
84	G♯$_7$/A♭$_7$	3322	G7_SHARP
85	A$_7$	**3520**	**A7**
86	A♯$_7$/B♭$_7$	3729	A7_SHARP
87	B$_7$	3951	B7
88	C$_8$	4186	C8