

# Raspberry Pi Build HAT

## Python library

Easily access LEGO®  
Technic™ motors and  
sensors in Python

# Colophon

© 2020 Raspberry Pi (Trading) Ltd.

This documentation is licensed under a Creative Commons [Attribution-NoDerivatives 4.0 International](#) (CC BY-ND).

build-date: 2021-10-22

build-version: githash: 91fd59e-dirty

## Legal Disclaimer Notice

TECHNICAL AND RELIABILITY DATA FOR RASPBERRY PI PRODUCTS (INCLUDING DATASHEETS) AS MODIFIED FROM TIME TO TIME (THE RESOURCES) ARE PROVIDED BY RASPBERRY PI (TRADING) LTD (RPTL) "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN NO EVENT SHALL RPTL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THE RESOURCES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

RPTL reserves the right to make any enhancements, improvements, corrections or any other modifications to the RESOURCES or any products described in them at any time and without further notice.

The RESOURCES are intended for skilled users with suitable levels of design knowledge. Users are solely responsible for their selection and use of the RESOURCES and any application of the products described in them. User agrees to indemnify and hold RPTL harmless against all liabilities, costs, damages or other losses arising out of their use of the RESOURCES.

RPTL grants users permission to use the RESOURCES solely in conjunction with the Raspberry Pi products. All other use of the RESOURCES is prohibited. No licence is granted to any other RPTL or other third party intellectual property right.

**HIGH RISK ACTIVITIES.** Raspberry Pi products are not designed, manufactured or intended for use in hazardous environments requiring fail safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, weapons systems or safety-critical applications (including life support systems and other medical devices), in which the failure of the products could lead directly to death, personal injury or severe physical or environmental damage (High Risk Activities). RPTL specifically disclaims any express or implied warranty of fitness for High Risk Activities and accepts no liability for use or inclusions of Raspberry Pi products in High Risk Activities.

Raspberry Pi products are provided subject to RPTL's [Standard Terms](#). RPTL's provision of the RESOURCES does not expand or otherwise modify RPTL's [Standard Terms](#) including but not limited to the disclaimers and warranties expressed in them.

## License

The Build HAT Python library is released under the MIT License.

Copyright (c) 2020-2021 Raspberry Pi Foundation

Copyright (c) 2017-2021 LEGO System A/S - Aastvej 1, 7190 Billund, DK

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the Software), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# Introduction

The Build HAT library has been created to support the Raspberry Pi Build HAT, an add-on board for the Raspberry Pi computer which allows control of up to 4 LEGO<sup>®</sup> Technic<sup>®</sup> motors and sensors included in the SPIKE<sup>®</sup> Portfolio.

Figure 1. The  
Raspberry Pi Build  
HAT

Other LEGO<sup>®</sup> devices may be supported if they use the LPF2 connector:

Figure 2. The LEGO<sup>®</sup>  
LPF2 connector

In order to drive motors, your Raspberry Pi and Build HAT will need an external 8V power supply. For best results, use the official [Raspberry Pi Build HAT power supply](#).

## ! WARNING

The API for the Build HAT is undergoing active development and is subject to change. An online version of this documentation can be found at <https://buildhat.readthedocs.io/>.

# Installation

The Python library can be installed using [pip](#),

```
$ pip3 install buildhat-*.whl
```

Alternatively it can be cloned from its [Github repository](#),

```
$ git clone https://github.com/RaspberryPiFoundation/python-build-hat.git
$ cd python-build-hat
```

and installed. If using asdf first by,

```
$ asdf install
```

and then,

```
$ pip3 install . --user
```

Building the library,

```
$ ./build.sh
```

# Usage

See the [Library](#) section for detailed documentation for the available Python objects.

```
import time
from signal import pause
from buildhat import Motor

motor = Motor('A')
motor.set_default_speed(30)

print("Position", motor.get_position())

def handle_motor(speed, pos, apos):
    print("Motor", speed, pos, apos)

motor.when_rotated = handle_motor

print("Run for degrees")
motor.run_for_degrees(360)

print("Run for seconds")
motor.run_for_seconds(5)

print("Run for rotations")
motor.run_for_rotations(2)

print("Start motor")
motor.start()
time.sleep(3)
print("Stop motor")
motor.stop()

pause()
```

# Programming Bootloader

You can use `openocd` to program the bootloader. This can be installed by,

```
$ sudo apt install automake autoconf build-essential texinfo libtool libftdi-dev  
libusb-1.0-0-dev  
$ git clone https://github.com/raspberrypi/openocd.git --recursive --branch rp2040  
--depth=1  
$ cd openocd  
$ ./bootstrap  
$ ./configure --enable-ftdi --enable-sysfsgpio --enable-bcm2835gpio  
$ make -j 4  
$ sudo make install
```

Then use the following command to program the bootloader

```
$ openocd -s /usr/local/share/openocd/scripts -f interface/raspberrypi-swd.cfg -f  
target/rp2040.cfg -c "program bootloader.elf verify reset exit"
```

# Library

## ColorSensor

The LEGO® Education SPIKE<sup>®</sup> Colour Sensor ([LEGO® Colour Sensor 45605](#)) can sort between 8 different colours and can measure reflected and ambient or natural light.

```
1 from buildhat import ColorSensor
2
3 color = ColorSensor('C')
4
5 print("HSV", color.get_color_hsv())
6 print("RGBI", color.get_color_rgb())
7 print("Ambient", color.get_ambient_light())
8 print("Reflected", color.get_reflected_light())
9 print("Color", color.get_color())
10
11 print("Waiting for color black")
12 color.wait_until_color("black")
13 print("Found color black")
14
15 print("Waiting for color white")
16 color.wait_until_color("white")
17 print("Found color white")
18
19 while True:
20     c = color.wait_for_new_color()
21     print("Found new color", c)
```

## ColorDistanceSensor

The [LEGO® Color and Distance Sensor 88007](#) can sort between six different colors and objects within 5 to 10 cm range

### ! WARNING

Support for this device is experimental and not all features are available yet.

```
1 from buildhat import ColorDistanceSensor
2
3 color = ColorDistanceSensor('C')
4
5 print("RGBI", color.get_color_rgb())
6 print("Ambient", color.get_ambient_light())
7 print("Reflected", color.get_reflected_light())
8 print("Color", color.get_color())
9
10 print("Waiting for color black")
```



```

11 color.wait_until_color("black")
12 print("Found color black")
13
14 print("Waiting for color white")
15 color.wait_until_color("white")
16 print("Found color white")
17
18 while True:
19     c = color.wait_for_new_color()
20     print("Found new color", c)

```

## DistanceSensor

The LEGO® Education SPIKE® Distance Sensor ([LEGO® Distance Sensor 45604](#)) behaves like a conventional ultrasonic range finder but also has 4 LEDs that can be used to create the *eyes* of a robot. Each LED can be controlled individually.

```

1 from signal import pause
2 from buildhat import Motor, DistanceSensor
3
4 motor = Motor('A')
5 dist = DistanceSensor('D', threshold_distance=100)
6
7 print("Wait for in range")
8 dist.wait_for_in_range(50)
9 motor.run_for_rotations(1)
10
11 print("Wait for out of range")
12 dist.wait_for_out_of_range(100)
13 motor.run_for_rotations(2)
14
15 def handle_in(distance):
16     print("in range", distance)
17
18 def handle_out(distance):
19     print("out of range", distance)
20
21 dist.when_in_range = handle_in
22 dist.when_out_of_range = handle_out
23 pause()

```

## ForceSensor

The LEGO® Education SPIKE® Prime Force Sensor ([LEGO® Force Sensor Set 45606e](#)) can measure pressure of up to 10 Newtons, but it can also be used as a touch sensor or a simple button.

NOTE

The Prime Force Sensor is also know as the LEGO® Technic Force Sensor.

```
1 from signal import pause
2 from buildhat import Motor, ForceSensor
3
4 motor = Motor('A')
5 button = ForceSensor('D', threshold_force=1)
6
7 print("Waiting for button to be pressed fully and released")
8
9 button.wait_until_pressed(100)
10 button.wait_until_released(0)
11
12 motor.run_for_rotations(1)
13
14 print("Wait for button to be pressed")
15
16 button.wait_until_pressed()
17 motor.run_for_rotations(2)
18
19 def handle_pressed(force):
20     print("pressed", force)
21
22 def handle_released(force):
23     print("released", force)
24
25 button.when_pressed = handle_pressed
26 button.when_released = handle_released
27 pause()
```

Matrix

The Spike 3x3 LED matrix has individual elements that can be set individually or as a whole.

Number	Name
0	
1	pink
2	lilac
3	blue
4	cyan
5	turquoise
6	green
7	yellow
8	orange

Number	Name
9	red

## " NOTE

Colours may be passed as string or integer parameters.

```

1 from buildhat import Matrix
2 import time
3 import random
4
5 matrix = Matrix('C')
6
7 matrix.clear(("red", 10))
8 time.sleep(1)
9
10 matrix.clear()
11 time.sleep(1)
12
13 matrix.set_pixel((0, 0), ("blue", 10))
14 matrix.set_pixel((2, 2), ("red", 10))
15 time.sleep(1)
16
17 while True:
18     out = [(int(random.uniform(0, 9)), 10) for x in range(3)] for y in range(3)]
19     matrix.set_pixels(out)
20     time.sleep(0.1)

```

## Motor

Motors from the LEGO® Education SPIKE® portfolio ([LEGO® Large angular motor 45602](#) and [LEGO® Medium angular motor 45603](#)) have an integrated rotation sensor (encoder) and can be positioned 1-degree accuracy. The encoders which can be queried to find the current position of the motor with respect to a zero mark shown on the motor itself.

Other motors such as the [LEGO® Medium Linear motor 88008](#), [Technic® Large Motor 88013](#), and [Technic® XL Motor 88014](#) without encoders will report a 0 value if queried.

```

1 from signal import pause
2 from buildhat import Motor
3 import time
4
5 motor = Motor('A')
6 motorb = Motor('B')
7
8 def handle_motor(speed, pos, apos):
9     print("Motor", speed, pos, apos)
10
11 motor.when_rotated = handle_motor
12 motor.set_default_speed(50)
13

```

```
14 print("Run for degrees 360")
15 motor.run_for_degrees(360)
16 time.sleep(3)
17
18 print("Run for degrees -360")
19 motor.run_for_degrees(-360)
20 time.sleep(3)
21
22 print("Start motor")
23 motor.start()
24 time.sleep(3)
25 print("Stop motor")
26 motor.stop()
27 time.sleep(1)
28
29 print("Run for degrees - 180")
30 motor.run_for_degrees(180)
31 time.sleep(3)
32
33 print("Run for degrees - 90")
34 motor.run_for_degrees(90)
35 time.sleep(3)
36
37 print("Run for rotations - 2")
38 motor.run_for_rotations(2)
39 time.sleep(3)
40
41 print("Run for seconds - 5")
42 motor.run_for_seconds(5)
43 time.sleep(3)
44
45 print("Run both")
46 motor.run_for_seconds(5, blocking=False)
47 motorb.run_for_seconds(5, blocking=False)
48 time.sleep(10)
49
50 print("Run to position -90")
51 motor.run_to_position(-90)
52 time.sleep(3)
53
54 print("Run to position 90")
55 motor.run_to_position(90)
56 time.sleep(3)
57
58 print("Run to position 180")
59 motor.run_to_position(180)
60 time.sleep(3)
```

## MotorPair

```
1 from buildhat import MotorPair
2
3 pair = MotorPair('C', 'D')
4 pair.set_default_speed(20)
5 pair.run_for_rotations(2)
6
7 pair.run_for_rotations(1, speedl=100, speedr=20)
8
9 pair.run_to_position(20, 100, speed=20)
```



**Raspberry Pi**

Raspberry Pi is a trademark of the Raspberry Pi Foundation

---

**Raspberry Pi Trading Ltd**