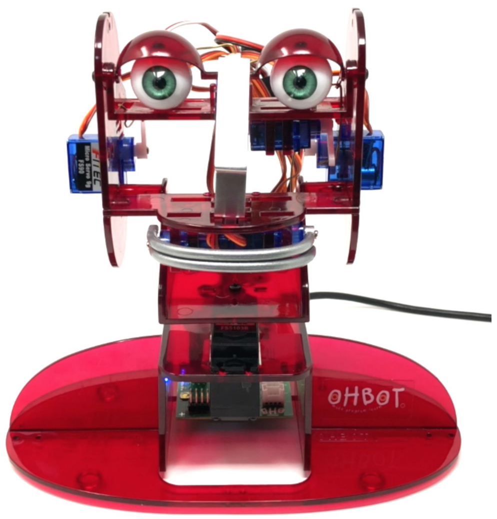
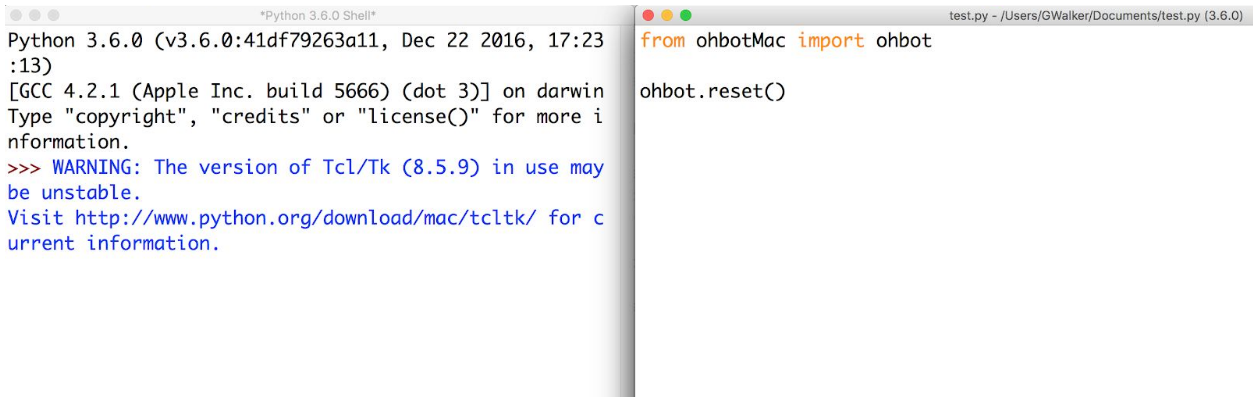
Programming Ohbot in Python

Python is a programming language with an emphasis on easy to read code. It is simple enough for beginners to get started with but powerful enough to enable complex projects. It runs on different types of computers and is widely supported on different operating systems including Windows, OSX and Raspbian.



The language is enhanced by a wide range of modules written by different programmers that help complete a range of tasks. There are modules for analysing, manipulating and generating graphics, sound and text, interacting with machines, the internet, and artificial intelligence and many other digital processes.

When Python is installed it is packaged with a code editor called IDLE which appears as an application on your computer. IDLE can be used to write, edit and run Python programs.



```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 22 2016, 17:23
:13)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more i
nformation.
>>> WARNING: The version of Tcl/Tk (8.5.9) in use may
be unstable.
Visit http://www.python.org/download/mac/tcltk/ for c
urrent information.
```

```
from ohbotMac import ohbot

ohbot.reset()
```

IDLE has two windows, on the left is the shell, this is where programs run and display any print information or requests for input.

On the right is the script area, this is where Python programs are written.

Python programs are written as text and need to be saved with the .py file extension. Programs run from the first line, line by line to the end and then stop although it is possible to loop part or all of a program.

The Ohbot library provides access to the full range of Ohbot features in a simple to use package, there are currently versions for Pi, Mac and Windows click your desired platform and follow the install instructions.

To make use of the Ohbot library in a program it needs to be imported using an import statement at the beginning of your code, these differ slightly depending on the operating system being used:

Import Statement	**Platform**
`from ohbot import ohbot`	Raspberry Pi/Raspbian
`from ohbot import ohbotMac`	Mac/Osx
`from ohbot import ohbotWin`	PC/Windows

The import statements comes at the start of a program.

To reset Ohbot use the ohbot.reset() command. Putting the import and reset commands together at the start of program will connect to ohbot and reset it to the rest position leaving it ready to be programmed.

Open IDLE, click File ->New File and type or paste in the following:

```
from ohbot import ohbot
ohbot.reset()
```

remember to use the correct import statement for your platform

Next click Run -> Run Module to run the script, you will be prompted to save your file and can save it wherever you like on your computer. Once saved the program should run, unless the motors have been moved from their home position Ohbot won't move quite yet, if they have been moved Ohbot's motors should all move back to their home position. As you go through the guide, click run again each time you change your program to see the results on Ohbot.

Move Ohbot

To make Ohbot move use the following command:

```
ohbot.move(ohbot.EYETURN, 10)
```

The ohbot.move function has two parameters; these are the bits of information the function needs to perform its job. Enter values for each parameter inside brackets after the function. First the function needs the name of the motor to move and second the position to move it to. The two parameters are separated by a comma.

The following motors can be moved:

Motor No	**Motor Name**	**Command (Replace 10 with any number 0-10)**
0	Head Nod	`ohbot.move(ohbot.HEADNOD, 10)`
1	Head Turn	`ohbot.move(ohbot.HEADTURN, 10)`
2	Eye Turn	`ohbot.move(ohbot.EYETURN, 10)`
3	Lid Blink	`ohbot.move(ohbot.LIDBLINK, 10)`
4	Top Lip	`ohbot.move(ohbot.TOPLIP, 10)`
5	Bottom Lip	`ohbot.move(ohbot.BOTTOMLIP, 10)`
6	Eye Tilt	`ohbot.move(ohbot.EYETILT, 10)`

Add the following code to your program with your import and reset code:

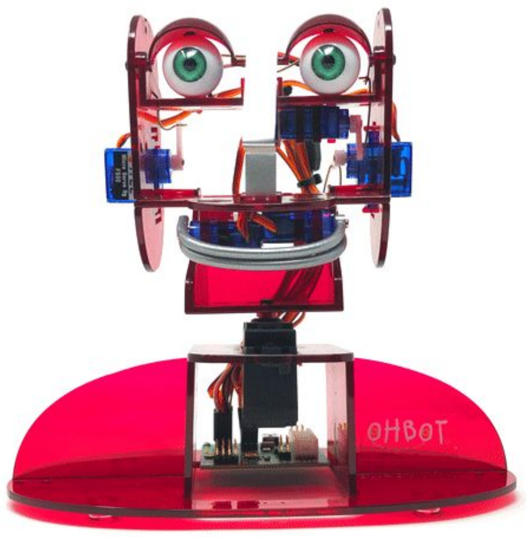
```
ohbot.move(ohbot.HEADTURN,0)

ohbot.wait(2)

ohbot.move(ohbot.HEADTURN,10)

ohbot.wait(0.5)

ohbot.move(ohbot.HEADTURN,5)
```

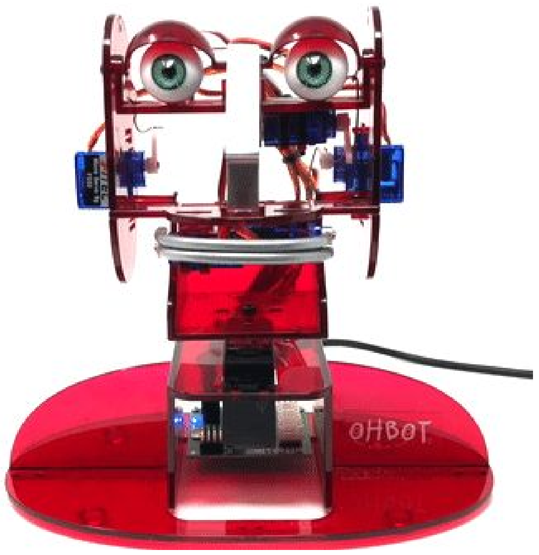
This program also includes the ohbot.wait() command. This command makes the program wait for the specified number of seconds before moving to the next line. In this example the wait gives the motor time to move before the next move command is sent.

Try expanding on the example above and including a longer sequence of movements and waits.

Remember to separate move commands for the same motor with wait commands to give them time to move.

Can you make Ohbot Smile?
Can you make Ohbot Frown?
Can you make Ohbot Blink?





Make Ohbot speak

To enable Ohbot to speak the library includes a function, ohbot.say(). For simple use the say function only needs one parameter, the text you want Ohbot to say, enclosed in "".

```
ohbot.say("Hello my name is Ohbot")
```

The function takes the text and converts it into an audio file using a text to speech synthesizer. It then generates a file containing a series of times and lip positions in order to be able to move the lips in time with the speech. The command will also prevent the program from moving to the next line until the speech has concluded. This all happens in the background, a single ohbot.say() function is enough to move the lips, generate and play the speech audio and wait long enough for the speech to finish.

remember to use the correct import statement for your platform

```
from ohbot import ohbot

ohbot.reset()

ohbot.wait(1)

ohbot.say("Hello my name is Ohbot, good to meet you")

ohbot.wait(1)

ohbot.say("Goodbye")

ohbot.move(ohbot.HEADNOD,3)

ohbot.move(ohbot.LIDBLINK,1)

ohbot.wait(1)

ohbot.close()
```

This program also includes the ohbot.close() function which simply powers down Ohbot’s motors.

The ohbot.say() command has two more optional parameters.

The say function has a parameter called untilDone, which controls whether the program waits while the speech is spoken. Enter False for the second parameter in the say command to prevent the program from waiting while the speech is being spoken:

remember to use the correct import statement for your platform

```
from ohbot import ohbot

ohbot.reset()
ohbot.wait(1)
ohbot.say("Don’t finish speaking before I move my eyes",False)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,5)

ohbot.wait(3)

ohbot.say("Wait until i’ve finished speaking before moving my eyes",True)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,5)
```

The say command has one more parameter you can use. The third parameter turns the lip sync on or off. These say commands have all three parameters one after another, separated by commas:

remember to use the correct import statement for your platform

```
from ohbot import ohbot

ohbot.reset()
ohbot.say("Speak without moving my lips, then move my eyes", True, False)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,5)
ohbot.wait(1)

ohbot.say("Speak with my lips moving, then move my eyes.", True, True)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,5)
ohbot.wait(1)

ohbot.say("Speak without my lips moving, and move my eyes at the same time", False,
False)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,5)
ohbot.wait(1)

ohbot.say("Speak with my lips moving, and move my eyes at the same time", False,
True)
ohbot.move(ohbot.EYETURN,10)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,0)
ohbot.wait(0.6)
ohbot.move(ohbot.EYETURN,5)
ohbot.wait(1)

ohbot.close()
```

Loops

To make a program repeat indefinitely a simple option is a while loop.

```
from ohbot import ohbot
ohbot.reset()
ohbot.wait(1)

while True:
    ohbot.move(ohbot.HEADNOD,10)
    ohbot.wait(2)
    ohbot.move(ohbot.HEADNOD,0)
    ohbot.wait(2)
```

Notice that the lines inside the loop need to be indented, four spaces in from the left. Many programming languages use brackets to denote blocks within lines of code, Python uses indentations instead.

To loop something a certain number of times you can use a for loop.

The `for x in range()` code will run the program once for each value in the range so range(0,5) will run 5 times. The first time x will equal 0, then 1 and so on up to 4, the last whole number within the specified range.

Again the section inside the loop is indented one tab from the left.

```
from ohbot import ohbot
ohbot.reset()
ohbot.wait(1)

for x in range(0,5):
    ohbot.say("I will say this 5 times")

for x in range(0,2):
    ohbot.say("I will say this 2 times")

ohbot.close()
```

The variable x can also be used inside the loop, for example to move through all of a motor’s positions:

```
from ohbot import ohbot
ohbot.reset()
ohbot.wait(1)

for x in range(0,10):
    ohbot.move(ohbot.HEADTURN,x)
    ohbot.wait(1)
```

Try changing the values in the range(,) command and see the effect this has on the movements.

Please note, only numbers 0-10 will work as this is the full range of the motor.

Sensors

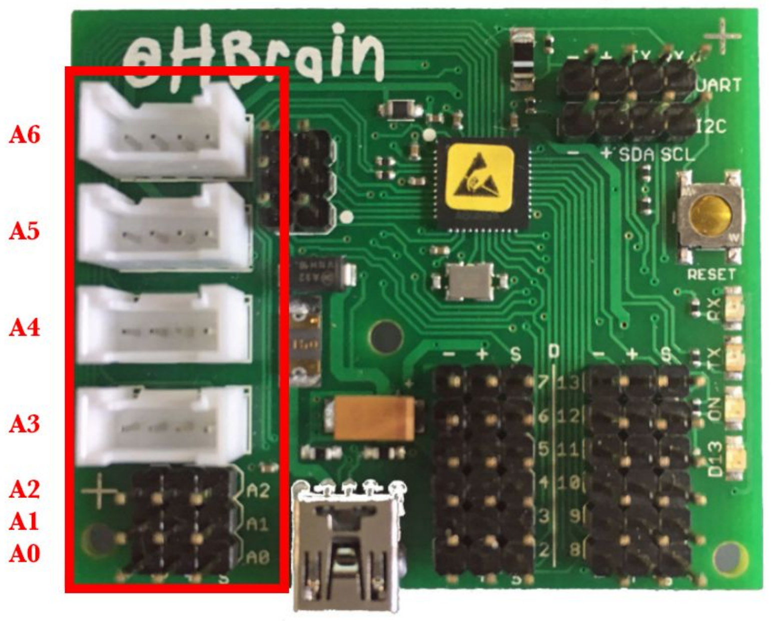
The Ohbot library includes a command to help interface with sensors. Ohbots are compatible with most arduino sensors, digital or analog. We also sell the Ohbot Sensor Pack which includes 3 sensors (light, touch and tilt) that work well for a range of projects.

Ohbot has 6 inputs for sensors - 3 with servo headers and 3 with Grove connectors.

To read a sensor use the ohbot.readSensor() command.

The command only takes one parameter, the sensor number to be read, and returns the requested sensor value as a float mapped to the range 0-10.

Look at the Ohbrain to find out which pin a sensor is connected to.



It is a good idea to store a sensor value in a variable before using it in a program.

```
reading = ohbot.readSensor(0)
ohbot.move(ohbot.HEADTURN,reading)
```

This code snippet will read the sensor connected to pin 0 and set the HEADTURN motor to this reading.

Please note, the code above will only read the sensor once, to read a sensor over and over again the snippet should be inside a loop:

```
while True:
    reading = ohbot.readSensor(0)
    ohbot.move(ohbot.HEADTURN,reading)
```

Conditionals (if statements) can also be used when writing programs using sensors.

if statements will evaluate the statement after the if and respond differently if it is True or False.

In this example the light sensor is plugged into pin 5.

Ohbot will say one phrase if the reading is below 3 another if it is between 3 and 7 and a third phrase if the value is above 7.

```
while True:
    reading = ohbot.readSensor(5)
    if reading > 7:
        ohbot.say("Wow it is very bright")
    elif reading > 3:
        ohbot.say("Not too bright not too dark")
    else:
        ohbot.say("Who turned out the lights?")
        ohbot.wait(2)
```

The Ohbot Python library uses the same command for digital and analog sensors.

Analog sensors, depending on the input, will give a range of values between 0 and 10. Digital sensors will give only 2 values as they only have two states , one value will be close to 0 and the other close to 10.

Use an if statement to evaluate a digital sensor, in this example a digital touch sensor is connected to pin 3.

```
while True:
    reading =  ohbot.readSensor(3)
    if reading > 2:
        ohbot.say("Don't touch my nose")
    ohbot.wait(1)
```

If you have any questions or queries about controlling Ohbot from Python, want to share a project with us or want to suggest improvements to the library please feel free to get in contact on info@ohbot.co.uk.