In this tutorial we will be building a **Raspberry pi WhatsApp Bot**, using Python pyautogui package. We will be launching WhatsApp web on our Pi and reading messages from our chat and responding back to those chats with appropriate answers. For demonstration purposes, I have programmed the Pi to control an LED using chat. But you can be creative and exchange any type of messages between your Pi and the people who chat with your Pi. So, let's get started.

Using WhatsApp on Raspberry Pi

If you are just [installing WhatsApp on Raspberry Pi](http://circuitdigest.com/microcontroller-projects/how-to-install-whatsapp-on-raspberry-pi), you can check out the linked tutorial where we have used the Yowsup library. But this method has many limitations and the Yowsup package also suffers from many bugs. So in this tutorial, we will be exploring other options to do that.

Why not use the official WhatsApp Business API for Automation?

It is true that WhatsApp has its own API for business purposes using which you can **automate WhatsApp messages using python**. You might have also come across various businesses like OYO, MakeMyTrip, etc.. But for a maker or small-scale project, getting permission to access the API is painstakingly expensive.

To get around this, we can put together a small hack using which we can access contents on WhatsApp web to read and respond to message for free.

Selenium vs pyautogui to Automate WhatsApp on Raspberry Pi

Selenium is a browser automation to automate anything you need over the web.

The other alternatives to selenium for WhatsApp browser automation are pyautoGUI and OpenCV. This package basically helps us to read what is on screen and navigate across the screen to perform any desired action. So most things that you can do with a keyboard and mouse on your raspberry pi can be done using pyautogui and paperclip package on python.

**In this tutorial, we have used pyautogui over selenium** as I found it hard to use selenium on Pi because of its webdriver. There is clearly no advantageous choice of one over the other. You can look into both the options and check which suits you best. **This tutorial will only cover pyautogui WhatsApp automation**.

Python Packages to Automate WhatsApp messages on Raspberry Pi

As mentioned earlier, this uses OpenCV, so make sure you have already installed OpenCV on your Pi. If not you can either [install OpenCV using Cmake on Pi](https://circuitdigest.com/microcontroller-projects/how-to-install-python-opencv-on-raspberry-pi) or use pip install.

pip3 install opencv-python

sudo apt-get install libcblas-dev

sudo apt-get install libhdf5-dev

sudo apt-get install libhdf5-serial-dev

sudo apt-get install libatlas-base-dev

sudo apt-get install libqt4-test

sudo apt-get install libqtgui4

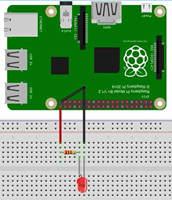
sudo apt-get install libjasper-dev

Next, we have to install Pyperclip and pyautogui to automate our WhatsApp messages. Pyautogui is used to read elements on the screen and to create mouse and keyboard events, while pyperclip is used to copy and paste information between the screen and the python program.

pip3 install pyperclip

pip3 install pyautogui

Hardware set-up to control LED lights from WhatsApp using Raspberry Pi

****

Python Program to Send and Receive WhatsApp Messages on Raspberry Pi

The complete Python code for this tutorial can be downloaded from the link below.

* [**Raspberry pi WhatsApp Automation Python code files**](https://circuitdigest.com/sites/default/files/Python_RaspberryPi_Code_WhatsApp_Automation.zip)

Inside the folder, you will find the main Python code along with five other images. These images help us in identifying different parts on the WhatsApp page. Always make sure to execute your Python file in the same directory as these images are present. Also, if you are adding new images for your program make sure they are in **.JPG format**, Raspberry Pi will have problems with other formats.

Now, let's take a closer look at our code to get a better understanding.

**Importing all required packages:**

Let's start the project by importing all the required packages. Most of these have been explained and installed already in the previous step. The new packages are the time and webbrowser packages. The time package is used to create a delay and the webbrowser package is used to open a web browser and launch WhatsApp automatically.

import pyautogui as pygu #To move mouse cursor and make click and keyboard strokes

from time import sleep #for delay

import pyperclip #to copy and past data

import webbrowser #to open webbrowser

import RPi.GPIO as IO # calling header file for GPIO’s of PI

import time # calling for time to provide delays in program

**Set GPIO pins and open WhatsApp Browser:**

The GPIO pin 40 is used to control the LED here, so we have set it as output and we have launched a web browser, which is Chromium by default on Pi. Then we have opened web.whatsapp.com to being our WhatsApp automation. Make sure you have already scanned your QR code and have checked “always login” if this is your first time

IO.setmode (IO.BOARD) # programming the GPIO by BOARD pin numbers, GPIO21 is called as PIN40

IO.setup(40,IO.OUT) # initialize digital pin40 as an output.

#Open the default webbrowser and open web.whatsapp

webbrowser.open\_new('https://web.whatsapp.com/')

**Prepare Message Templates which has to be sent to the user as Response:**

Next, we have prepared three messages which we will send to our users as a response to the chat. You can create as many as you need. All WhatsApp formatting features like **\*bold\*** for bold and **:smiley**for smiley will also work here as you can see below.

default\_message = [

"Hi I am your Whatsapp Bot :robot \n from RaspberryPi. I can help you with basic home automation. You can try any of the following :notes \n commands",

"\*turn on light\* - \_Turns on the led connected to pi\_",

"\*turn off light\* - \_Turns off the led connected to pi\_"]

turn\_on\_light = [

"Sure, your :bulb \n Light is now turned on"

]

turn\_off\_light = [

"Okay, Your LED is not turned off"

]

**Check if WhatsApp has opened successfully on web** **browser:**

The below method is used to check if the WhatsApp chat page is opened successfully on a web browser. It uses the **whatsapp\_header.jpg** (can be found in the code file above) image to check if it is the right screen. We will proceed to the program only if the page has opened successfully.

Basically, we are searching for the image **“whatsapp\_header.jpg”** on our screen. If we find that image, we can confirm that this particular page has opened.

#Wait for whatsapp page to

def open\_whatsapp():

# check if whatsapp opened successfully

find\_whatsapp\_header = None

while find\_whatsapp\_header is None:

find\_whatsapp\_header = pygu.locateOnScreen("Whatsapp\_header.JPG", confidence=.8)

use\_here\_button\_pos = pygu.locateOnScreen("use\_here\_button.JPG", confidence=.8)

if (use\_here\_button\_pos):

print("Whatsapp is being used somewhere else, clicking on use here")

sleep(2)

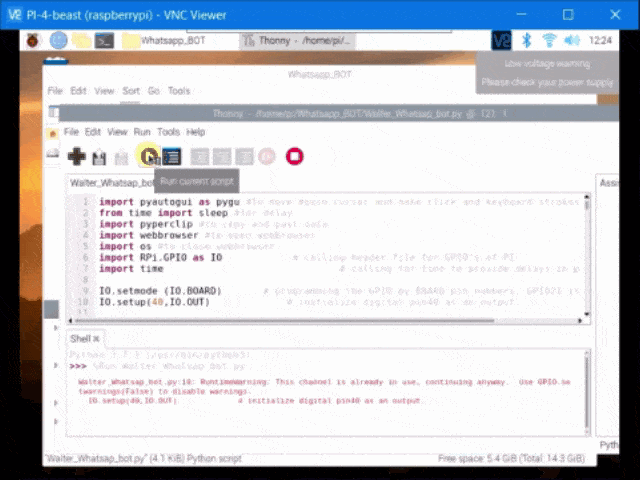
pygu.moveTo(use\_here\_button\_pos[0], use\_here\_button\_pos[1], duration=0.5)

pygu.click()

print(".")

sleep(2)

return 1



**Check if a new chat is available:**

Similarly, in this method, we will search for the**“green\_circle.jpg”** image. If that image is found, we will navigate to the circle and click on it. If the user is chatting for the first time, we might want to click on the ok button before proceeding, otherwise, we can just continue.

This method will return 1 if a new chat is available and will return 0 if not. It will also open the chat and get it ready for reading or sending a message.

#checks for new message and opens it

def new\_chat\_available():

# Check for new messages

green\_circle\_pos = pygu.locateOnScreen("green\_circle.JPG", confidence=.8)

if (green\_circle\_pos):

sleep(2)

pygu.moveTo(green\_circle\_pos[0], green\_circle\_pos[1], duration=0.5)

pygu.click()

sleep(1)

ok\_button\_pos = pygu.locateOnScreen("ok\_button.JPG", confidence=.8)

if (ok\_button\_pos):

pygu.moveTo(ok\_button\_pos[0], ok\_button\_pos[1], duration=0.5)

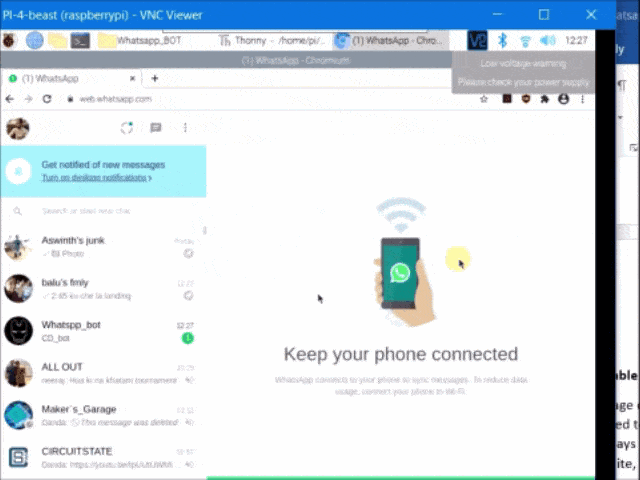
pygu.click()

return 1

else:

sleep(1)

return 0



**Read the last message sent by the user:**

This method reads the last message by searching for **“smily\_paperclip.jpg”** image on the screen. Then from there, it moves to the text box using the **py.mpveTo()**. Note that the values 50 and 35 used in this method are hard-coded to work for Raspberry Pi. You might want to tweak or change it as needed.

Then we can read the last message by triple-clicking on the white box. This will select the complete message, we can then copy the message using *ctrl+C* and then later use pyperclip to get this value. As you can see, this function returns the copied information using **pyperclip.paste().**

def read\_last\_message():

smily\_paperclip\_pos = pygu.locateOnScreen("smily\_paperclip.JPG", confidence=.6)

pygu.moveTo(smily\_paperclip\_pos[0], smily\_paperclip\_pos[1])

pygu.moveTo(smily\_paperclip\_pos[0] + 50, smily\_paperclip\_pos[1] - 35, duration=0.5)

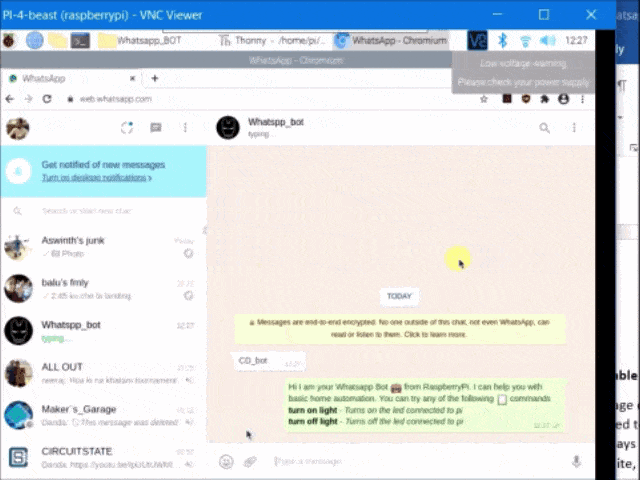
sleep(1)

pygu.tripleClick()

pygu.hotkey('ctrl', 'c')

sleep(0.1)

return (pyperclip.paste())



**Get a proper response for the last received message:**

The next method is used to find an appropriate replay for the last message the user has sent. This method takes in the last read message as input and compares it with a set of pre-defined answers which we are expecting from the customer. In our case, we are expecting “CD\_bot” or “turn on light” or “turn off light” if the user has sent any one of the above messages, we will respond with the pre-defined text. We will also toggle the GPIO pin of the LED as per the command received.

def get\_response(incoming\_message):

if "CD\_bot" in incoming\_message:

return default\_message

if "turn on light" in incoming\_message:

IO.output(40,1) # turn the LED on

return turn\_on\_light

if "turn off light" in incoming\_message:

IO.output(40,0) # turn the LED off

return turn\_off\_light

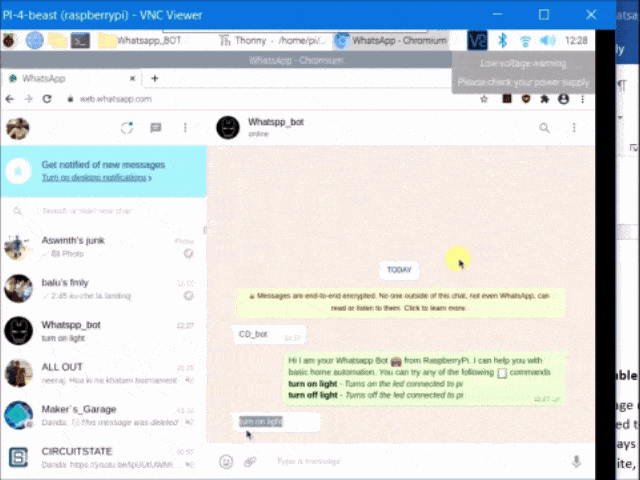
else:

return ""

**Sending a WhatsApp Message as Response Automatically:**

Now, we have already read the last message from a chat and have prepared the message to be sent back. All that is left is the sent that response as a chat replay. To do that, we have the below method called **send\_message()**. This function takes in the response as a parameter and types it on the chat windows using typewrite method in pyautogui. Then we have used *shift+enter* to go to the new line and then finally press enter to send the message.

def send\_message(message\_content):  
    for content in message\_content:  
        pygu.typewrite(content, interval=.02)  
        pygu.hotkey('shift', 'enter')  
    pygu.hotkey('enter') #Enter key to send the message



**Check if a new message is available from the current chat:**

We can look for the *green\_circle* image only if the chat is not opened yet. If the currently opened chat user sends a new message, we will need to detect it by checking the color of the pixel under the cursor. Since WhatsApp received messages always have white color as a background color. We can simply look for the color under the cursor, if it is white, we will proceed with reading the message as discussed earlier.

def new\_message\_available():

current\_mouse\_pos = pygu.position()

pointer\_color = pygu.pixel(current\_mouse\_pos[0], current\_mouse\_pos[1])

if (pointer\_color == (255, 255, 255)):

return 1

else:

return 0

**Open WhatsApp, read the last message, and send a response message:**

Now that we have prepared a different method for each action, all we have to do is, use them appropriately inside a while loop so that our python script will keep looking for new messages and reply to them accordingly.

if (open\_whatsapp()): #if whatsapp page is opened successfully

print("##Whatsapp page ready for automation##")

while(1):

if (new\_chat\_available() or new\_message\_available()):

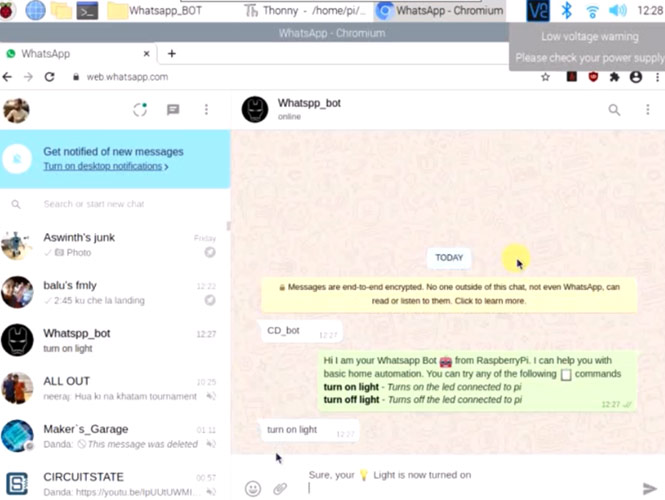
print("New chat or message is available")

incoming\_message = read\_last\_message() #read the last message that we received

message\_content = get\_response(incoming\_message) #decide what to respond to that message

send\_message(message\_content) #send the message to person

That is it, now you can control any devices or get information like the temperature and humidity of your room on your WhatsApp like chatting to your friend. The best part is you can easily share this with anyone who has WhatsApp installed, all they have to do to access your bot is to call your bot name like**“CD\_bot”**. You can also add this to a WhatsApp group so that anyone in the group can access this bot.



Check out the below video for a full working demonstration, if you have any questions, use our forums to post your questions. Also, let me know what you will be building with the help of the tutorial, if it sounds very interesting, we might also build one for our next tutorial.

Code

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111 | import pyautogui as pygu #To move mouse cursor and make click and keyboard strokes  from time import sleep #for delay  import pyperclip #to copy and past data  import webbrowser #to open webbrowser  import os #to close webbrowser  import RPi.GPIO as IO            # calling header file for GPIO’s of PI  import time                              # calling for time to provide delays in program    IO.setmode (IO.BOARD)       # programming the GPIO by BOARD pin numbers, GPIO21 is called as PIN40  IO.setup(40,IO.OUT)             # initialize digital pin40 as an output.    #Open the default webbrowser and open web.whastapp  webbrowser.open\_new('<https://web.whatsapp.com/>')    default\_message = [      "Hi I am your Whatsapp Bot :robot \n from RaspberryPi. I can help you with basic home automation. You can try any of the following :notes \n commands",      "\*turn on light\* - \_Turns on the led connected to pi\_",      "\*turn off light\* - \_Turns off the led connected to pi\_"]    turn\_on\_light = [      "Sure, your :bulb \n Light is now turned on"  ]    turn\_off\_light = [      "Okay, Your LED is not turned off"  ]    #Wait for whatsapp page to  def open\_whatsapp():      # check if whatsapp opened successfully      find\_whatsapp\_header = None      while find\_whatsapp\_header is None:          find\_whatsapp\_header = pygu.locateOnScreen("Whatsapp\_header.JPG", confidence=.8)          use\_here\_button\_pos = pygu.locateOnScreen("use\_here\_button.JPG", confidence=.8)          if (use\_here\_button\_pos):              print("Whatsapp is being used somewhere else, clicking on use here")              sleep(2)              pygu.moveTo(use\_here\_button\_pos[0], use\_here\_button\_pos[1], duration=0.5)              pygu.click()          print(".")          sleep(2)      return 1    #checks for new message and opens it  def new\_chat\_available():      # Check for new messages      green\_circle\_pos = pygu.locateOnScreen("green\_circle.JPG", confidence=.8)      if (green\_circle\_pos):          sleep(2)          pygu.moveTo(green\_circle\_pos[0], green\_circle\_pos[1], duration=0.5)          pygu.click()          sleep(1)          ok\_button\_pos = pygu.locateOnScreen("ok\_button.JPG", confidence=.8)          if (ok\_button\_pos):              pygu.moveTo(ok\_button\_pos[0], ok\_button\_pos[1], duration=0.5)              pygu.click()            return 1      else:          sleep(1)          return 0    def read\_last\_message():      smily\_paperclip\_pos = pygu.locateOnScreen("smily\_paperclip.JPG", confidence=.6)      pygu.moveTo(smily\_paperclip\_pos[0], smily\_paperclip\_pos[1])      pygu.moveTo(smily\_paperclip\_pos[0] + 50, smily\_paperclip\_pos[1] - 35, duration=0.5)      sleep(1)      pygu.tripleClick()      pygu.hotkey('ctrl', 'c')      sleep(0.1)      return (pyperclip.paste())    def get\_response(incoming\_message):      if "CD\_bot" in incoming\_message:          return default\_message      if "turn on light" in incoming\_message:          IO.output(40,1)                      # turn the LED on          return turn\_on\_light      if "turn off light" in incoming\_message:          IO.output(40,0)                      # turn the LED off          return turn\_off\_light      else:          return ""      def send\_message(message\_content):      for content in message\_content:          pygu.typewrite(content, interval=.02)          pygu.hotkey('shift', 'enter')      sleep(1)      pygu.hotkey('enter') #Enter key to send the message    def new\_message\_available():      current\_mouse\_pos = pygu.position()      pointer\_color = pygu.pixel(current\_mouse\_pos[0], current\_mouse\_pos[1])      if (pointer\_color == (255, 255, 255)):          return 1      else:          return 0    if (open\_whatsapp()): #if whatsapp page is opened successfully      print("##Whatsapp page ready for automation##")    while(1):        if (new\_chat\_available() or new\_message\_available()):          print("New chat or message is available")          incoming\_message = read\_last\_message() #read the last message that we received            message\_content = get\_response(incoming\_message) #decide what to respond to that message          send\_message(message\_content) #send the message to person |