AUTODESK Instructables

How to Make Smart Glasses!

By maker_soup in CircuitsRaspberry Pi

Introduction: How to Make Smart Glasses!

[Note from future me: This is a very software intensive project and my code is not the greatest. It may take a couple of modifications to fit your exact situation as libraries and other third party applications may have had updates that create bugs in the code. If you notice any such bugs, please contact me and I will try to update the code.]

Also check the very end of this tutorial!

Hello everyone, today I'm going to show you all how to build your very own smart glasses with a RaspberryPi 0w! I was inspired to build this project after watching Spiderman FFH. I was thinking about how I could build the E.D.I.T.H. glasses and quite honestly I thought it was an impossible task and pretty soon thereafter dismissed the idea. Until a couple of weeks later I saw this video by JLaservideo, which heavily inspired the first prototype. Since then I have built this iteration, which is significantly better than version one.

Before I tackled this project the most advanced thing I had done was blink an LED with an Arduino UNO or print "Hello World" in Python, so I'm sure you can imagine how difficult this project soon became. Regardless, I completed and I have learned so much about electronics because of this project and I hope you will as well. Let's go!.

Note: A quick warning, I have no professional education in.... well anything, so take everything I say with a grain of salt.

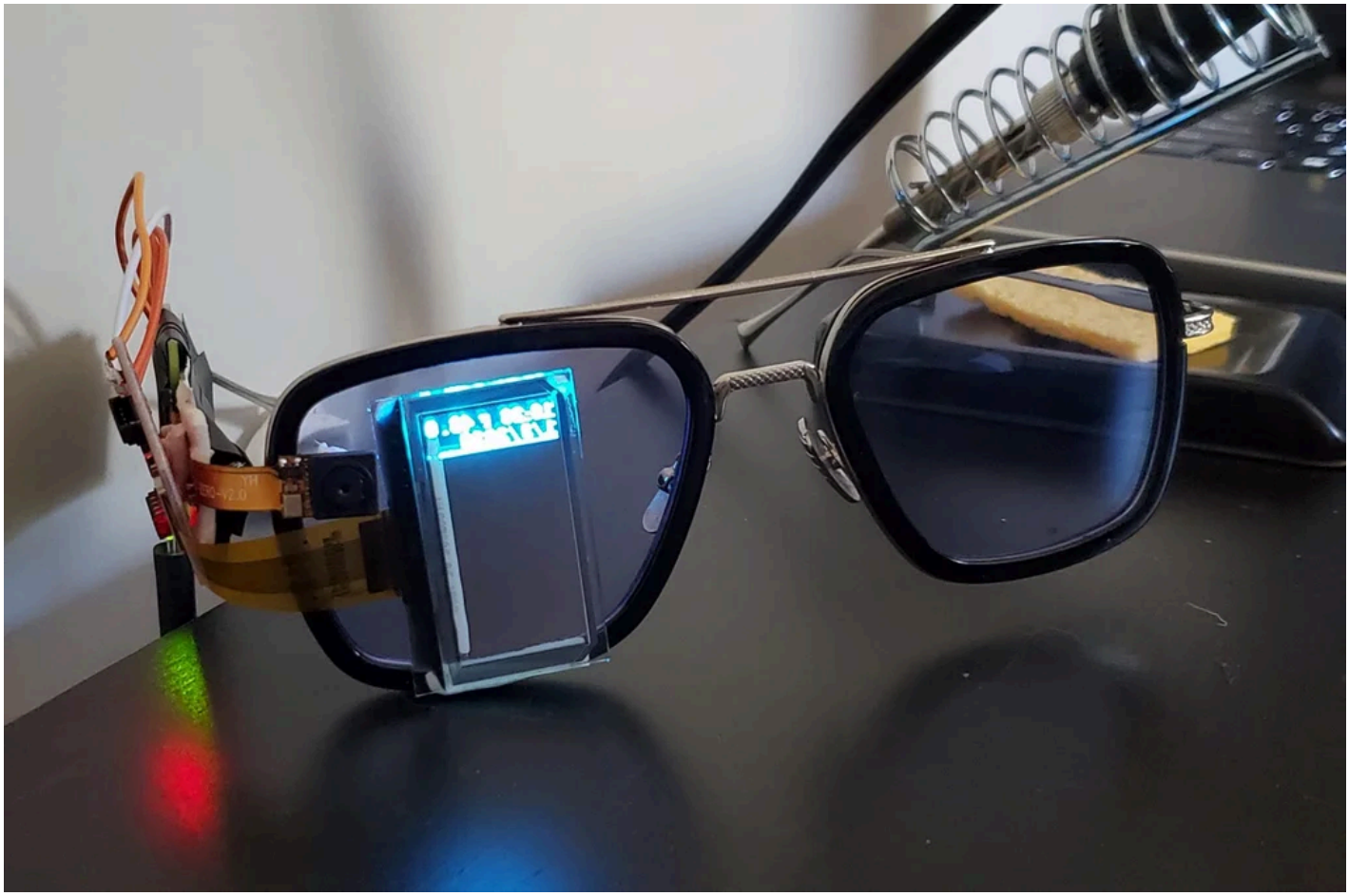
Features of the Smart Glasses

- **Displaying time/date**
- **Displaying temperature**
- **Playing music**
- **Taking pictures/videos**
- **Sending SMS(text messages)**
- **Upload photos/videos to Dropbox**
- **Answering any question**
- **Calculating any math problem**

Supplies

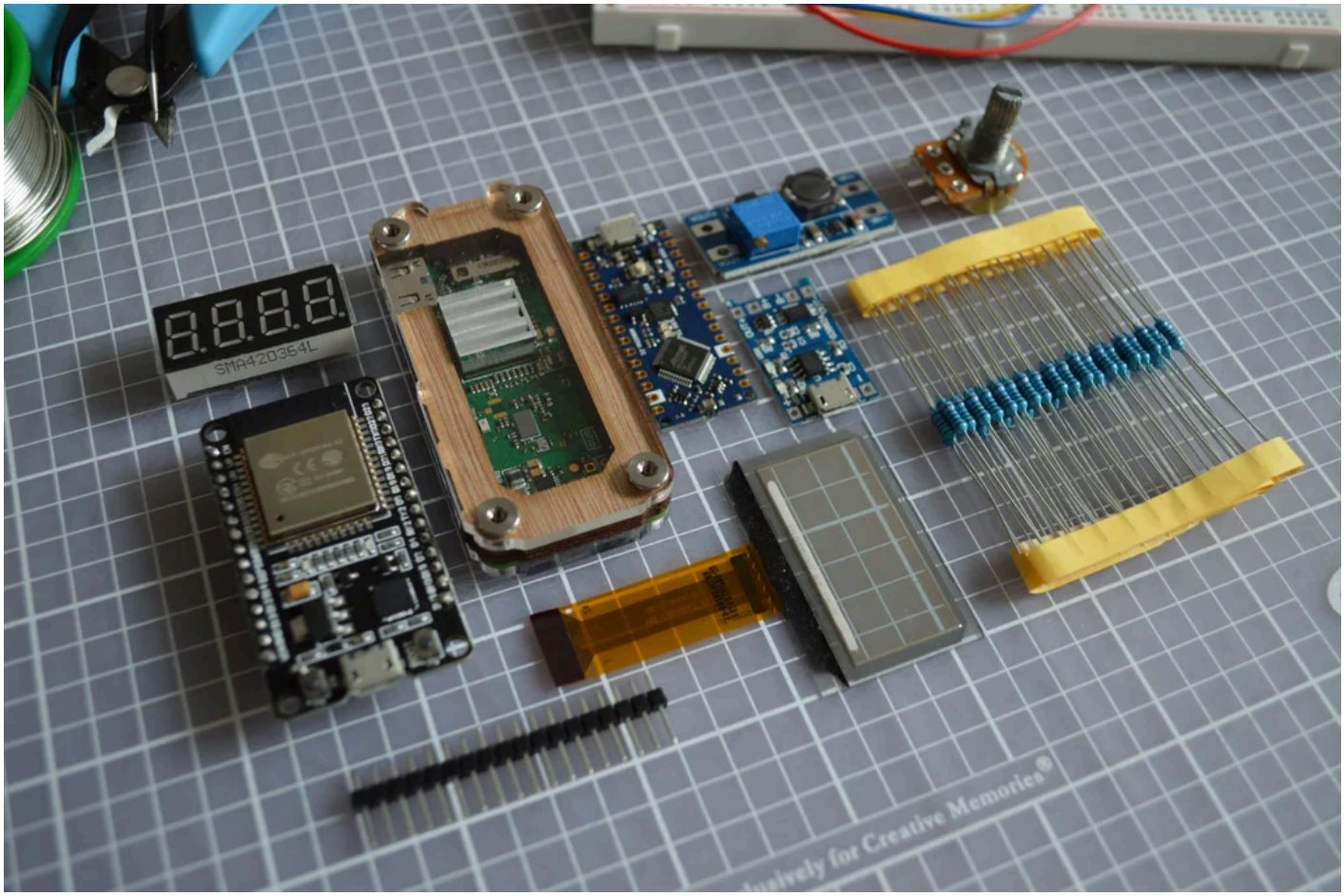
- Clear OLED
- Adafruit 500c
- RaspberryPi 0w
- Li-Po Battery
- PiCamera
- Wireless Headphones
- Soldering Iron
- Lead-Free Solder
- Ender 3 v2
- Filament

Step 1: Prototype



I built my first prototype of these glasses a while back and their main capability was taking pictures and doing basic text recognition and then giving the user some information on it with Wolfram. For example, if it took a picture of say a detour sign, the glasses would recognize the text and give the user a definition; however, these glasses were very buggy and the text recognition had a very long delay. After some consideration For the new version, I decided this feature wasn't very practical and if the user wanted that information they could just ask the chat bot anyway. I decided this iteration should focus more on reliability and efficiency while still adding more features. I think I have done that successfully considering the last prototype didn't even have a battery.

Step 2: Theory



RaspberryPi 0w

I have choosen the RaspberryPi 0w as the brains for this operation. The RaspberryPi 0w easily connects to the internet, supports Bluetooth for the wireless ear-buds, Python is pre-installed, and has an easy to use camera interface. The reason you want to use the RaspberryPi 0w opposed to a say an ESP32 is that it has significantly more ram; specifically the RaspberryPi 0w has 520mb of ram opposed to the ESP32's 512kb. For reference that is nearly one-thousand times more!

Transparent OLED

The transparent OLED from Sparkfun is the defining characteristic of the smart glasses or the 'cool factor' which will give the user feedback. It can be controlled using either the I2C or SPI protocol, but in this project we will use I2C. The display has a minimum operating voltage of 1.65v and a maximum operating voltage of 3.3v, so trying to power it from the 5v pins won't work out so well. I would like to point out that using this OLED is more or less a proof-of-concept.

Adafruit 500c

For this project, I needed a charging circuit that could also boost the voltage from 3.3v to 5v. The great thing about this board is that it can be easily switched on/off by connecting the enable pin to ground; this provides a secure and easy place to mount a slide switch. Although, you will have to chop-off the JST connector and solder the battery to the through-hole pins because it won't fit in the case with the connector.

Li-Po

After researching the recommended power supply requirements for the RaspberryPi 0w I came across this chart on the official RaspberryPi website. It tells us that the recommended PSU capacity should be 1200mah or 1.2Amps. Considering the transparent OLED only draws 400mah and the camera 250mah this should be more than enough.

Step 3: Necessary Applications & APIs

Applications you will need:

- **PuTTY**

PuTTY is the software we will use to ssh into the RaspberryPI before accessing the GUI desktop. To download it click the link and hit 'here'. This will take you to a new page where you can install either the 32bit or 64bit version.

- **BalenaEtcher**

We will need BalenaEtcher to flash the Raspbian OS to the SD-card. Click the link to get to their website where you can choose your specific operating system and 32bit or 64bit version.

- **Microsoft Remote Desktop Classic**

Remote Desktop Classic is what we need to access the RaspberryPI GUI. Remote Desktop Classic is available for download in the Microsoft store. If you are on mac a possible alternative is Chrome Remote Desktop.

- **Angry IP Scanner**

After connecting the RaspberryPI 0w to the internet we will need to access it via its IP address. Angry IP Scanner scans all the devices on your wireless network. To install it go to the link and click 'free download' then '32/64-bit Installer'. This installer will auto detect whether you have a 32bit or 64bit system.

- **Cura**

Cura is the slicing software needed to convert the stl files to gcode files which will then be loaded onto your 3D printer. Upon visiting their website you must click download then select your operating system. Cura is only available on 64bit systems, but there are alternatives like Slic3r. Although I recommend using Cura if possible.

APIs you will need:

- **Dropbox**

I'm sure most have you have heard of Dropbox. Dropbox is a server used to store any information that you would also like to access from a different computer. We will use it to store pictures and videos taken by the smart glasses, so we can access them from either our phones or computers rather than just leaving them on the RaspberryPI desktop and having to ssh in whenever we want to see them. To download Dropbox head to the link provided above and sign-up or log-in. Once you have finished click on 'app console' -> 'create app' -> 'scoped access' -> 'app folder' then finally decide on a creative title for your app and click finish. After you have finished all of that you should see a dashboard for your app. Leave this tab for now until we come back to it later.

- **Twillio**

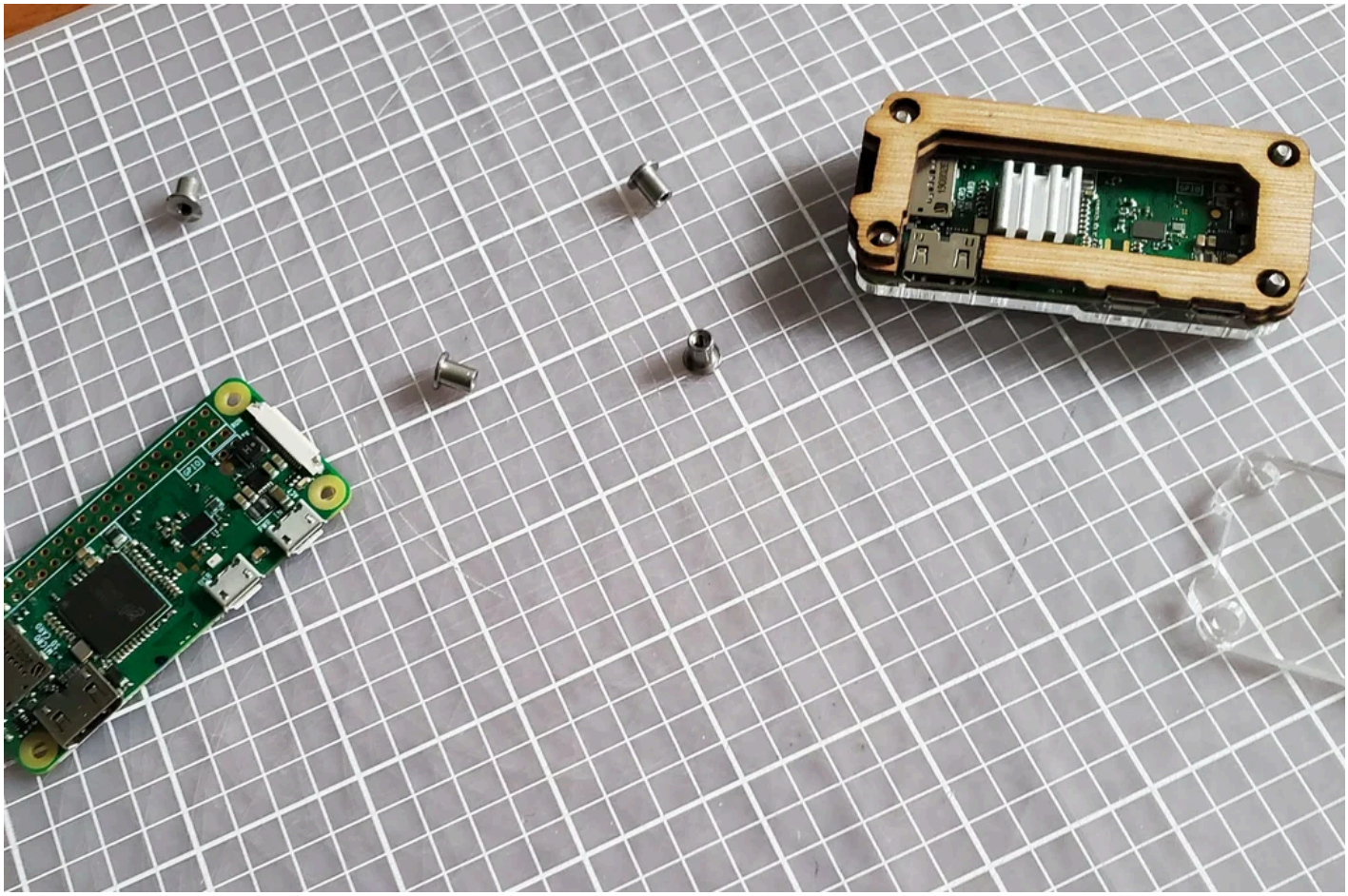
One of the many features the smart glasses includes sending SMS or text messages. To accomplish this task we will use Twillio which is a 'free' to use communication API that is capable of sending or receiving SMS or calls. The reason I say 'free' in brackets is that this service is only free until the graciously provided trial balance of $15.00 runs out; although this might not sound like a lot on the surface until you realize that sending SMS only costs a little over one cent per message. That is approximately five-hundred messages.

The set-up process is simple and easy, but may take a while at first. Go the website and sign-up for a free trial account. After verifying your phone number Twilio will ask you some questions. For the first two questions just put other then select 'with code' then 'Python' and finally 'No, I want to use my own hosting service'. Now you are inside your console which will display anything and everything about your account and your apps. Click 'Get a trial number' this is your phone number Twilio will use to send SMS. For the trial account we have to verify the numbers we would like to send SMS to. In order to do that go to your console and on the right you should see a link that says 'Verified Numbers' your number will already be there, but if you are ever going to want to message someone else you will have to add their phone number here. That should be it for now, leave the console open we will come back to it later when we build the SMS feature in the code.

- **Wolfram Alpha**

Last but not least is Wolfram Alpha which is a computational search engine. Wolfram Alpha will be the brains of the smart glasses. Wolfram Alpha allows us to get answers to any question we may ask or math problem we present it with. Sign-up is very straight forward just head to the link and sign-up for an account then click on your profile in the top right then 'My Apps (API)'. Now click 'Get AppID' where it will ask you for some description about your app then finally it will give you your app-ID. Keep this screen open we will come back to it later in the code.

Step 4: Setting Up the RaspberryPI



Flashing The Operating System

First things first, install the Raspbian OS from the official website; you will want to choose the **second** option(OS with recommended software) unless you know what your doing. We will use BalenaEtcher to flash the OS to a micro-SD card. Plug your micro-SD card into your computer via an adapter then open BalenaEtcher. Select the .zip folder containing the OS, then select the proper SD-card and click flash, this will take ~20min. When it's finished disconnect the SD-card because BalenEtcher might automatically eject it, but if it doesn't click eject anyway through file explorer or finder.

Connecting the RaspberryPI to your Wi-Fi

Now that the OS image is flashed onto the SD-card we need to add a couple of files that will enable us to wirelessly access the RaspberryPI from our computer. Plug the SD-card back into your computer and navigate to its directory in File Explorer or Finder. Create an empty text file named ssh, but remove the file name extension so that you get a file named 'ssh'. If you don't see a .txt at the end you may have to enable file name extensions. To do this open file explorer then click view at the top of the window and then check the box labeled file name extensions in the top right.

The next file will be used by the RaspberryPi to know which network is yours and how to connect to it. Create a new text file within the SD-card directory named 'wpa_supplicant.txt'. Insert the following lines of text within this file and replace 'WIFI_SSID' with your network name and 'WIFI_PASSWORD' with your network password then save it. Now replace the txt file extension with conf.

```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant Group=netdev update_config=1
network={ ssid="WIFI_SSID"
scan_ssid=1
psk="WIFI_PASSWORD"
key_mgmt=WPA-PSK }
```

Step 5: SSH Into the RaspberryPI

Accessing with SSH

Now that your RaspberryPI is ready to connect to your network power it up by connecting the micro-usb port labeled 'PWR' to your laptop. Please note that the RaspberryPI 0w is a 2.4ghz device meaning that if you only have a 5ghz network you won't be able to connect to the RaspberryPI wirelessly; however, most networks have a 2.4ghz and 5ghz band. If your computer isn't already connected to the 2.4ghz band go ahead and connect it. Next you are going to have to figure out the IP-address of the RaspberryPI by scanning all of the devices on your network. Open Angry IP Scanner and scan your network by clicking the start button in the top right. Your RaspberryPI should have a host name of 'raspberrypi', so find that device in the list and note the IP-address. Now that you have the IP-address open PuTTY. Enter the IP-address in the text-bar at the top and check the box labeled ssh below it then click open.

Updating the RaspberryPI 0w

You should now be prompted with a window that says log-in. The default username is 'pi' and the password is 'raspberry', be careful when typing the password as you won't be able to see it for security reasons.

As with any new RaspberryPI we need to do some updates first. Type these commands into the terminal one-by-one.

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get autoremove
sudo reboot
```

Note: This could take a while, so make sure you have a stable internet connection before you start.

Enabling the GUI

Before we enable the GUI we need to change the hostname so if you ever get another RaspberryPI 0w you won't have to worry about connecting to the wrong one. Type 'sudo raspi-config' after which you will be prompted with a new window. Using your arrow keys navigate to the second option and hit enter then type in your new hostname. Don't exit out of this window quite yet, we also need to enable the SSH server so go to 'Interfacing options' then scroll down to where it says SSH then hit enter. For now we are done working inside this window so click finish and then reboot. Now SSH is enabled and ready to use, but we still have to install another application, xrdp. This essentially communicates to the remote desktop application on your computer. To install it start another Putty session and then type 'sudo apt-get install tightvncserver' the 'sudo apt-get install xrdp'.

Viewing the GUI

You are now good to go ahead and access the GUI, so close out of that Putty session and head over to remote desktop. Hit 'add' in the top right corner then 'Desktop' and enter the IP-address of the RaspberryPI 0w. As long as your RaspberryPI is powered up click on the tab with the IP-address and it will wirelessly access the GUI desktop.

Step 6: Downloading Packages & Music

Installing Libraries

Although Python comes pre-installed on the RaspberryPI we still need to install some libraries necessary for the glasses. Below is the command for installing all of the packages not included in the initial Python install.

```
pip install bs4 urllib3 luma.core luma.oled wolframalpha SpeechRecognition PyAudio wikipedia picamera dropbox twilio
```

Transfering the Music

Now we have to get some music files onto the RaspberryPI, so we can play some dank tunes on our spectacular glasses. We are going to first download them from our PC then copy pasta onto the RaspberryPI. Technically, you could download them straight from the Pi, but that would be very slow. First download this song and convert it to wav format with an online converter. Now rename it to 'Imagine.mp3' and copy pasta onto the RaspberryPi desktop. Although, you can download any song you like just make sure you save it on the desktop in wav format and you change the name in the code so it matches the file name.

Step 7: Connecting Wireless Headphones

In order to get user input and provide feedback we will need to connect the wireless headphones to the RaspberryPI 0w. There are multiple ways to do this, I will show you two ways, so if the first method doesn't work you can always try the second option.

First Method

This method is by far the easiest as you can accomplish it with only two clicks. First make sure your bluetooth headphones are in pair mode, if you are using the same ones that I recommended just hold down the power button until the LED starts rapidly flashing. Now that your headphones are in pair mode click on the bluetooth icon in the top right of the screen and select tab labeled GS. If it pairs you should see a green check mark in that menu and the LED should stop flashing. If you were able to get this method to work then you can move on to the next step. For the rest of us who weren't so lucky try the second method.

Second Method

This method is a bit more in-depth, but shouldn't be to difficult.

First we need to initialize the Bluetooth agent. Type these commands into the shell editor one-by-one.

```
bluetoothctl
power on
agent on
default-agent
```

Turn the headphones to pair mode by holding down the power button. Now type this command.

```
scan on
```

While it's scanning you should see the MAC address of your headset appear. If you don't know what the MAC address of your device is I recommend connecting them to your phone where it should show the address in your wireless device settings. After you have seen your MAC address type this command to stop the scan.

```
scan off
```

Finally, all that is left is to pair the device and connect it. Type these commands in the shell editor one-by-one making sure to replace the asterisk with the MAC address for your device.

```
pair *
trust *
connect *
```

Step 8: The RaspberryPI Can Talk?

Text-To-Speech

The main way the smart glasses will relay the information is through the chat bot. The chat bot will work with what is called a text-to-speech library that just takes a normal input string or text and converts that into an MP3 file. Originally, I wanted to use the pyttsx3 library to drive the chat bot and I have gotten it to work in the past on the RaspberryPi 0w; however, when I tried it more recently it didn't work. I believe the problem arose from updates to the RaspberryPI's operating system and various Python libraries. After some quick research I discovered a different speech engine, Espeak, and while I don't think Espeak is as good as pyttsx3 it will accomplish what we need it to.

Download

To download it open the shell editor and enter the following command.

```
sudo apt-get install espeak
```

Step 9: Making the RaspberryPI Speak!

Talking Pi?

To test Espeak we will give it some text which it will hopefully dictate back to us! If your headphones aren't connected anymore go ahead and power them on. Type the following line into the shell editor and you should hear a talking Pi!

Note: Wherever you see a 'XX:XX:XX:XX:XX:XX' replace it with the MAC address of your wireless headset.

```
espeak "Hello World" -ven+f3 -k5 -s125 --stdout | aplay -D bluealsa:DEV=XX:XX:XX:XX:XX:XX,PROFILE=sco<br>
```

Modify the Voice

On it's own Espeak has a pretty robotic sound to it, but thankfully we can modify the tone, gender, speed, and much more. Below are a couple of modifications you can make, but if you would like a full list visit this link. All it takes a couple of modifications and an imagination, so make it yours!

- -ven ~ Specifies the language. Change the en to the abbreviation of whatever language you should desire.
- -f ~ Speaks a text file
- -a ~ Set amplitude, range 0-200.
- -s ~ Changes the speed. Lower limit 80 and no maximum.
- -g ~ Increases the pause between words in 10ms.
- -f3 ~ Changes to the third female voice. There are ten options, five male and five female. (e. m1, f2...)

Step 10: Beautiful Pictures!

Setup

Another one of the amazing features of the smart glasses includes taking videos and pictures! Taking pictures with the RaspberryPI 0w is very straight forward! All you have to do is plug the camera into the ribbon cable connector by pulling back on the black lever located near the end. To actually take the picture we first have to enable the camera, so click on the RaspberryPI icon in the top left and scroll down to preferences->Raspberry PI Configuration->Interfaces. If you are wondering where you are this is just the GUI version of the raspi-config menu we accessed earlier in the tutorial. Inside of the menu you should see an option to enable camera, then we need to reboot. Go to the shell editor and type sudo reboot.

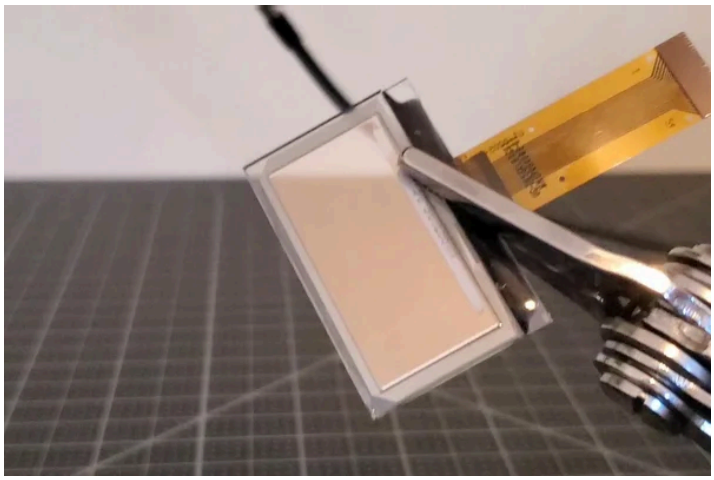
Taking Pictures

You are now ready to start taking beautiful pictures. Open up the shell editor and type the following.

```
raspistill –o Image.jpg
```

After hitting enter the indicator LED on the camera should turn on for a few seconds then the picture named Image.jpg will get saved in /home/pi directory.

Step 11: Crystal Clear!



Enabling I2C

Finally, we have made it to one the coolest parts about the smart glasses, the transparent OLED. This is where we are able to display information to the user such as, time, temperature, and just general info. As I said previously, this display will work via I2C and the luma.core python library. By default, I2C isn't enabled, so we must enable it. Go to the same place you enabled the camera module and enable I2C then reboot.

Wire It!

To connect up the wires me must solder them from the driver to the RaspberryPi. I recommend a soldering temperature of 725°F/385°C. Don't forget to power off your RaspberryPi and disconnect any peripherals. Here are the connections.

OLED---RaspberryPI

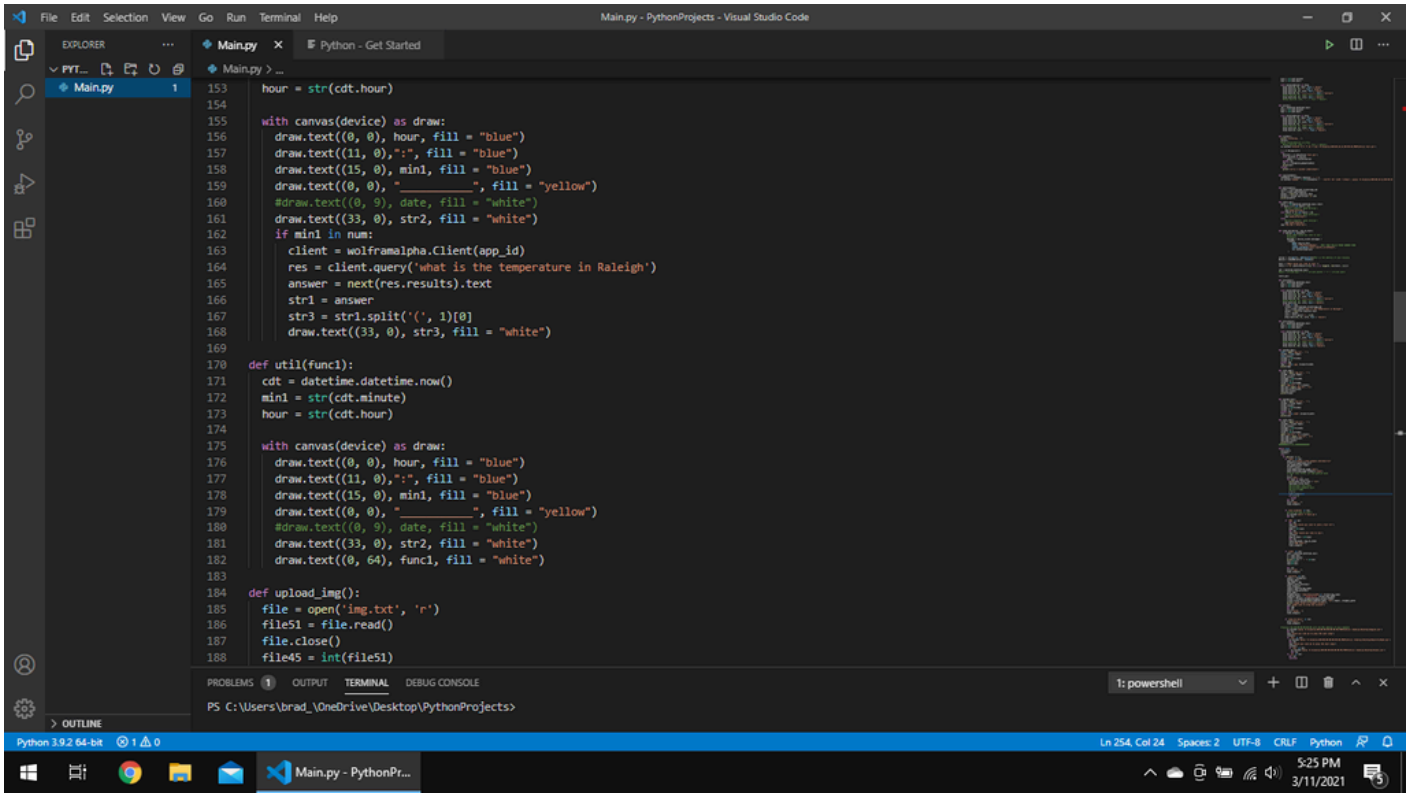
GND-GND

3v3-3v3

SDA-GPIO_2(SDA)

SCL-GPIO3_(SCL)

Step 12: Time to Code!



Prepare the Python File

Now that the RaspberryPI, transparent display, and camera are up and running we are ready to start writing the code. I coded the majority of this while in the terminal because the Thonny editor doesn't run very well on the RaspberryPi 0w; nevertheless, it's still an option. To open a new python script, we need to be located in the /home/pi directory, where all of are libraries are. If you aren't and you dont know how to get back just open the shell editor and enter, 'cd ~'. While in the shell editor type 'nano main.py' this will open our main script. Below are brief explanations of each part of the code. Each part of the code is marked by section.

#1 Importing the Libraries

These are the libraries that we installed earlier in the tutorial, so all you have to do is copy pasta.

#2 AppIDs and Authorization Tokens

Here are all of our App_IDs and account tokens as well as some other variables. Replace the contact numbers and names with people you would like to send SMS to. Put all of your specific appIDs and auth tokens that we got from step three into their indicated places.

#3 Functions

In this section we define a lot of tasks for the display that would be rather tedious to type out each time. This is also where the start up sequence is defined and initiated.

#4 While Loop and If Statements

This is the part of the code that infinitely loops until you end it with the key word, *'exit program'*. Taking a deeper look into this section we see the glasses can tell you the news, send SMS, tell time, take pictures, take videos, and play music based of off the trigger words inside of the if-statement condition.

#5 Catch All

If you didn't say any of the trigger words then the program will search whatever you said on Wolfram, but if that fails it will search Wikipedia and return a short summary.

Full Code

```
########Section_1##########
import sys
import bs4
from bs4 import BeautifulSoup as soup
from urllib.request import urlopen
import os
import sys
from luma.core.interface.serial import i2c
from luma.core.render import canvas
from luma.oled.device import ssd1306, ssd1325, ssd1331, sh1106
import time
import datetime
import wolframalpha
import speech_recognition as sr
import pyaudio
import wikipedia
from picamera import PiCamera
import dropbox
import picamera
from twilio.rest import Client
```

```
##########Section_2###########
contacts= {'Pizza Hut': "718-822-2909", 'Papa John's': "415-586-7272", 'Friend3': "316-316-316"}

#Wolfram
app_id = ('') #WolfRam Alpha ID

#Dropbox
dropbox_access_token= "" #Unique Dropbox token

#Twilio
twilio_account_sid = '' #You will find this information in your Twilio account
twilio_auth_token = ''
twilio_client = Client(twilio_account_sid, twilio_auth_token)

client = wolframalpha.Client(app_id)
res = client.query('what is the temperature in San Fransisco') #Replace with the name of your city
answer = next(res.results).text
str1 = answer
str2 = str1.split('(', 1)[0]

bad_chars = [';', '|', '(', ')', '+', '=', '1'] #We use this to remove all of the characters that Espeak can't say

num = ["05", "10", "15", "20", "25", "30", "35", "40", "45", "50", "55", "0"] #At these points in every hour the glasses will refresh the

#######Section_3#################

def util2():
  cdt = datetime.datetime.now()
  min1 = str(cdt.minute)
  hour = str(cdt.hour)

  with canvas(device) as draw:
    draw.text((0, 0), hour, fill = "blue")
    draw.text((11, 0),":", fill = "blue")
    draw.text((15, 0), min1, fill = "blue")
    draw.text((0, 0), "___________", fill = "yellow")
    #draw.text((0, 9), date, fill = "white")
    draw.text((33, 0), str2, fill = "white")
    #draw.text((0, 115), "...", fill = "white")


def util3():
  cdt = datetime.datetime.now()
  min1 = str(cdt.minute)
  hour = str(cdt.hour)

  with canvas(device) as draw:
    draw.text((0, 0), hour, fill = "blue")
    draw.text((11, 0),":", fill = "blue")
    draw.text((15, 0), min1, fill = "blue")
    draw.text((0, 0), "___________", fill = "yellow")
    #draw.text((0, 9), date, fill = "white")
    draw.text((33, 0), str2, fill = "white")
    draw.text((0, 115), "...", fill = "white")


def listen():
  print('listening...')
  util3()
  #with canvas(device) as draw:
    #draw.text((0, 115), "...", fill = "white")
  os.system('arecord -d 4 -f cd -t wav -D bluealsa:DEV=A0:18:12:00:9D:55,PROFILE=sco test.wav') #Replace the MAC address with that of you

  r = sr.Recognizer()
  try:
    harvard = sr.AudioFile('test.wav')
    with harvard as source:
        audio = r.record(source)
    global val
    val = r.recognize_google(audio)
    print(val)
  except:
    print('sorry I couldnt understand')


def say(statment):
  statment1 = statment.replace(" ", "_")
  os.system('espeak ' + str(statment1) + ' -ven+f3 -k5 -s130 --stdout | aplay -D bluealsa:DEV=A0:18:12:00:9D:55,PROFILE=sco') #Change the



def query(query):
  client = wolframalpha.Client(app_id)
  res = client.query(query)
  answer = next(res.results).text
  answer1 = answer.partition('\n')[0]
  print(answer1)

def start_up():
  hour1 = int(datetime.datetime.now().hour)
  if hour1 >= 0 and hour1 < 12:
      #print('computer: good morning')
      say('good_morning')
  elif hour1 >= 12 and hour1 < 20:
      #print('computer: good afternoon')
      say('good_afternoon')
  else:
      #print('computer: good evening')
      say('good_evening')
  say('how may i help you')


def send_sms(person, msg_to_text):
```

```
    if person in contacts:
        #say('What would you like to say')
        #listen()
        message = twilio_client.messages \
        .create(
             body= msg_to_text,
             from_='+YourTwilioNumber',  #PUT YOUR TWILIO PHONE NUMBER HERE, sry for caps
             status_callback='http://postb.in/1234abcd',
             to= contacts[person]
         )


serial = i2c(port=1, address=0x3C)#Put in the address of your display
device = ssd1306(serial, rotate=1)

text = ("What would you like to say? ")
text1 = ('\n'.join([text[i:i+11] for i in range(0, len(text), 11)]))

cdt = datetime.datetime.now()
#date = str(cdt.day) + "/" + str(cdt.month) + "/" + str(cdt.year)

start_up()

def utilitys():
  cdt = datetime.datetime.now()
  min1 = str(cdt.minute)
  hour = str(cdt.hour)

  with canvas(device) as draw:
    draw.text((0, 0), hour, fill = "blue")
    draw.text((11, 0),":", fill = "blue")
    draw.text((15, 0), min1, fill = "blue")
    draw.text((0, 0), "___________", fill = "yellow")
    #draw.text((0, 9), date, fill = "white")
    draw.text((33, 0), str2, fill = "white")
    if min1 in num:
      client = wolframalpha.Client(app_id)
      res = client.query('what is the temperature in San Fransisco')
      answer = next(res.results).text
      str1 = answer
      str3 = str1.split('(', 1)[0]
      draw.text((33, 0), str3, fill = "white")

def util(func1):
  cdt = datetime.datetime.now()
  min1 = str(cdt.minute)
  hour = str(cdt.hour)

  with canvas(device) as draw:
    draw.text((0, 0), hour, fill = "blue")
    draw.text((11, 0),":", fill = "blue")
    draw.text((15, 0), min1, fill = "blue")
    draw.text((0, 0), "___________", fill = "yellow")
    #draw.text((0, 9), date, fill = "white")
    draw.text((33, 0), str2, fill = "white")
    draw.text((0, 64), func1, fill = "white")

def upload_img():
  file = open('img.txt', 'r')
  file51 = file.read()
  file.close()
  file45 = int(file51)
  file45 += 1
  global img
  img = 'img{}.jpg'.format(file45)
  print(img)

def save_img():
  file1 = open('img.txt', 'r')
  file99 = file1.read()
  file1.close()
  file100 = int(file99)
  file100 += 1
  file2 = str(file100)
  open('img.txt', 'w').close()
  new_num = open('img.txt', 'w')
  new_num.write(file2)
  new_num.close()
  print(file2)


def upload_vid():
  file3 = open('vid.txt', 'r')
  file61 = file3.read()
  file3.close()
  file47 = int(file61)
  file47 += 1
  global vid
  vid = 'vid{}.h264'.format(file47)
  print(vid)

def save_vid():
  file3 = open('vid.txt', 'r')
  file101 = file3.read()
  file3.close()
  file102 = int(file101)
  file102 += 1
  file7 = str(file102)
  open('vid.txt', 'w').close()
  new_num2 = open('vid.txt', 'w')
  new_num2.write(file7)
  new_num2.close()
  print(file7)
```

```
########Section_4#############

while True:
  utilitys()
  listen()
  try:
    if len(val) >= 2:
      if 'news' in val:
        news_url="https://news.google.com/news/rss"
        Client=urlopen(news_url)
        xml_page=Client.read()
        Client.close()
        soup_page=soup(xml_page,"xml")
        news_list=soup_page.findAll("item")
        # Print news title, url and publish date
        i = 0
        util('news...')
        for news in news_list:
          print(news.title.text + "\n")
          say(news.title.text)
          #print(news.link.text)
          #print(news.pubDate.text)
          #print("-"*60)
          i += 1
          time.sleep(1)
          if i == 3:
            break
        del val
        util("news...")
        time.sleep(3)


      if 'exit program' in val:
        os.system('pkill -f main.py')
        del val

      if 'SMS' in val:
          del val
          say('who would you like to send a text to?')
          listen()
          dude = str(val)
          del val
          say('what would you like to say')
          listen()
          msg_to_send = str(val)
          del val
          send_sms(dude, msg_to_send)
          util("SMS sent!")
          time.sleep(3)


      if 'time' in val:
        cdt1 = datetime.datetime.now()
        h = cdt1.hour
        m = cdt1.minute
        print(str(h) + ':' + str(m))
        say(str(h))
        say(str(m))

        del val
        util("time...")
        time.sleep(3)

      if 'picture' in val:
        camera = PiCamera()
        upload_img()
        img_name = img
        camera.start_preview()
        time.sleep(5)
        camera.capture(img_name)
        camera.stop_preview()
        save_img()
        dropbox_path= "/SmartGlassesAPI/{}".format(img_name)   #Go to your Dropbox profile and make a folder named SmartGlassesAPI
        computer_path = r'/home/pi/{}'.format(img_name)
        client = dropbox.Dropbox(dropbox_access_token)
        client.files_upload(open(computer_path, "rb").read(), dropbox_path)
        print('image saved to dropbox account')
        say('image save to drop box account')
        del img
        del val
        util("saved...")
        time.sleep(3)


      if 'play my music' in val:
        util("music...")
        time.sleep(2)

        #replace the 00:00:00:00:00:00 with the MAC address of your headset
        os.system('aplay -D bluealsa:DEV=00:00:00:00:00:00,PROFILE=sco /home/pi/Desktop/Imagine.wav')
        del val
        say('would you like me to play the next song')
        listen()
        if 'yes' in val:                #Dang I guess I should have stored this in a variable
          os.system('aplay -D bluealsa:DEV=00:00:00:00:00:00,PROFILE=sco /home/pi/Desktop/BeautifulName.wav')
          del val
          say('would you like me to play the next song')
          listen()
          if 'yes' in val:                      #Lol, you have to do it again
            os.system('aplay -D bluealsa:DEV=00:00:00:00:00:00,PROFILE=sco /home/pi/Desktop/Oceans.wav')
            del val
        if 'no' in val:
          del val
```

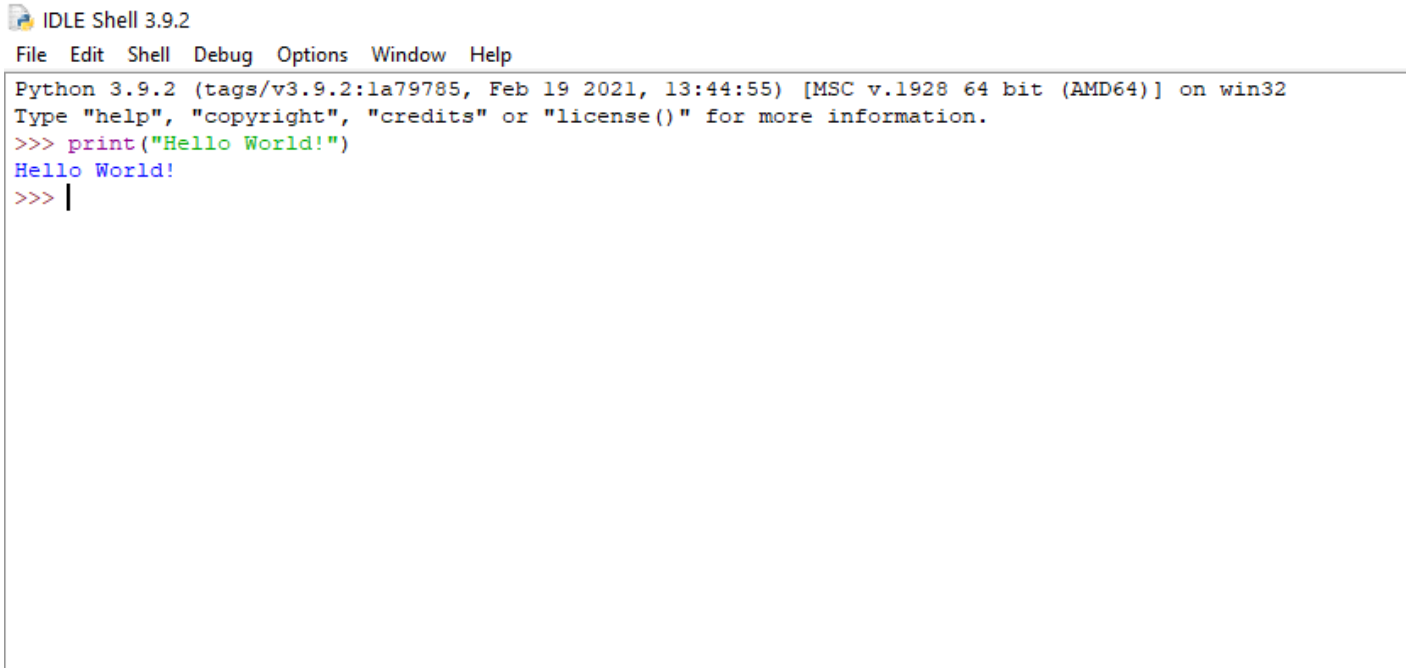
```
            continue


        if 'video' in val:
          upload_vid()
          vid_name = vid
          #vid_name = str(vid_name)
          with picamera.PiCamera() as camera:
            camera.start_recording(vid_name)
            time.sleep(30)
            camera.stop_recording()
            save_vid()
            new_vid_name = vid_name.replace('.h264', '.mp4')
            command = "MP4Box -add {} {}".format(vid_name, new_vid_name)
            subprocess.call([command], shell=True)
            dropbox_path= "/SmartGlassesAPI/{}".format(new_vid_name)
            computer_path = r'/home/pi/{}'.format(new_vid_name)
            client = dropbox.Dropbox(dropbox_access_token)
            client.files_upload(open(computer_path, "rb").read(), dropbox_path)
            time.sleep(3)
            print('video saved to dropbox account')
            say('video save to drop box account')
            del vid
            del val
            util("saved...")
            time.sleep(3)

  ############section_5######################

        try:
          client1 = wolframalpha.Client(app_id)
          res1 = client1.query(val)
          answer1 = next(res1.results).text
          answer2 = answer1.partition('\n')[0]
          for i in bad_chars:
            answer2 =  answer2.replace(i, '')
          print(answer2)
          say(answer2)
        except:
          try:
            print(wikipedia.summary(val, sentences = 1))
            say(wikipedia.summary(val, sentences = 1))
          except:
            print('')
        del val
    except:
      print(' ')
      #del val
  ###############THE_END#########################
```

Step 13: The Code Works!



How to Use

Before trying to execute the code make sure your wireless headphones are connected. If all goes well the display should turn on and the chatbot will ask what you would like for it to do. It will listen for your response whenever the three dots at the bottom of the screen are present. Below is a full list of the commands and what they will do. When you are done just say, *'exit program'* then flip the power switch. It's that simple!

Commands

news - The chatbot will tell you the top three headlines from Google News.

exit program - This will stop the script and then all you have to do is flip the switch.

SMS - This will send any message to anyone you tell it to.

Time - This one is pretty self-explanatory and is a little more descriptive then the clock on the screen.

Picture - This will take a picture and upload it to your Dropbox account.

Play my music - This will play the first song in the list and upon finishing will ask if you'd like to play the next one.

Video - This will take a 30 sec video then convert it from h264 format to MP4 format.

Step 14: Power Supply Setup

Explanation

You should now have the skeleton of the smart glasses. Although we are still powering through the micro-usb port. We should probably fix this because most of us don't want to have to plug our glasses into the outlet to turn them on.

For this step, you will need your soldering iron, solder, Adafruit 500c, slide switch, and Li-Po. Below are the connections. Again, I recommend a soldering temperature of 725°F/385°C, but be careful around the battery with the soldering iron and do not short the battery terminals.

Connections

Adafruit 500c - Slide Switch

en - side pin

gnd - middle pin

Adafruit 500c - Li-Po

Bat - Positive

Gnd - Negative

Step 15: Printing & Cura Settings

Printer Settings

Now the guts of our smart glasses are finished one of the last steps is printing the assembly. The 3D-printed case consists of 3 parts, the left ear-piece, right ear-piece, and the front piece. I printed using PLA on the Creality Ender 3 v2. If you don't have access to a 3D-printer, card-board or MDF board could be used for an enclosure. Before you print don't forget to level your print bed. I use a a standard 3"x5" note card to slide under the nozzle at each corner; there should be a good amount of resistance, but it should not be leaving any indents or marks on the card. If your print bed is dirty I recommend cleaning it with a wet wipe before attempting any prints. My 3D printer has had some trouble with accuracy. To fix this, I made sure the belt on the axis the extruder travels along is tight.

Cura Settings

For figuring out the perfect Cura settings I followed this tutorial although I made a couple of tweaks for my specific filament. You may want to do a couple of test prints to find the perfect nozzle and bed temperature if you are not using the same filament I listed in the beginning. Below is a brief summary of what I think is a good profile for the Ender 3.

- *Printing Temperature 215*
- *Bed Temperature 70*
- *Speed 100mm/s*
- *Layer Height .12mm*
- *Retraction 6mm*
- *Infill 30%*
- *Initial Speed 20mm/s*

STL Files

Step 16: Assembly

Connections

You should have three different parts the battery circuit, RaspberryPI circuit, and the 3D prints. Before we connect the battery to the RaspberryPI we need to thread the wires through the cover plates. First solder wires to the 5v pin and GND pin of the Adafruit 500c. Now thread those wires through the holes in both cover plates and then to the power input pins of the RaspberryPi. Make sure the cover plate with the thin slit is on the right. Below are the connections for the circuit.

Adafruit 500c - RaspberryPI

5v - 5v(Pin 2)

GND - GND(Pin 6)

Now we should be ready to assemble them fully. First, line the Adafruit 500c charging port with the hole in the bottom of the left ear piece and hot-glue the bottom to the case. Now put the battery next to the charger and then slide the cover plate down the wires and hot-glue it in place. Set the RaspberryPI down in the right ear piece, and slide it all the way back, if it's not fully slid all the way back the camera won't fit. With a bit of wiggling around you should be able to get the camera in its place. Now un-plug the transparent OLED from the driver by pulling up on the black lever. Thread the ribbon cable through the slit in the cover and then plug the screen back into the driver. Now, pull the cover down and glue it in place. I didn't have any lenses that would fit, but if you do go ahead and glue them into the front. All that's left to do is connect the two ear pieces to the front. I used hot glue, but make sure none is leaking out the sides like on mine. While still putting enough on for support.

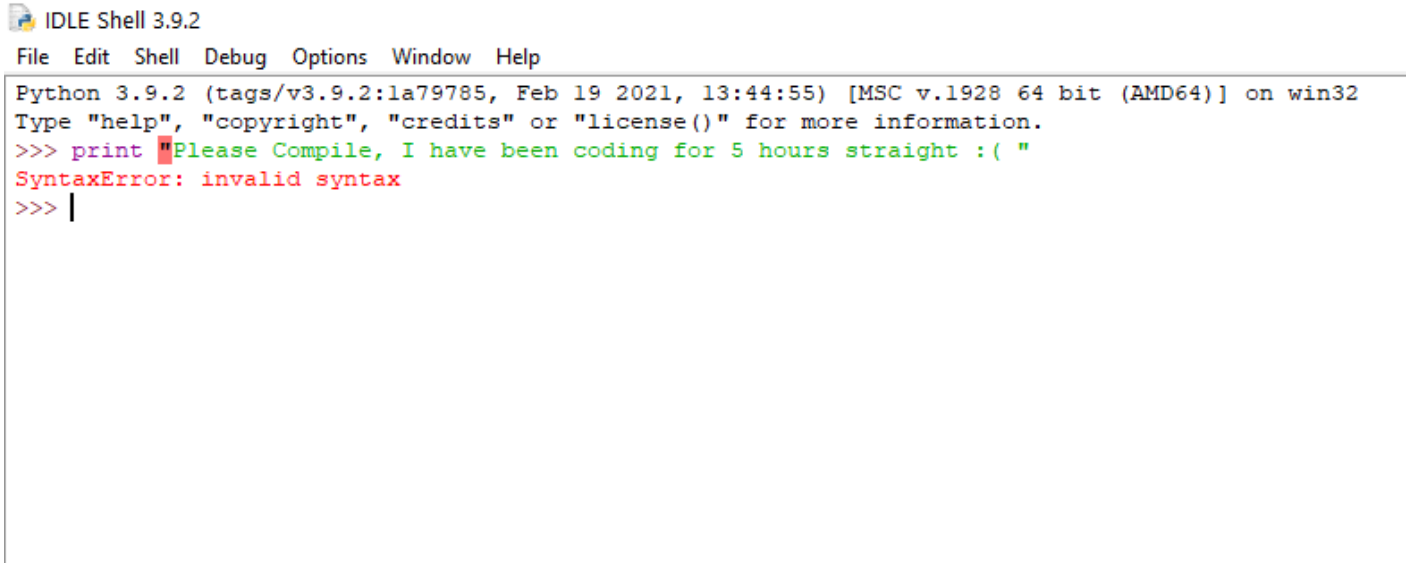
Step 17: Test

We should now be ready to do a final test! Turn on your RaspberryPi by flipping the power switch. You should be able to see a dim green and red light inside the right ear piece if all is connected well. After a minute SSH into the Raspberry Pi through Remote Desktop. Now turn on the wireless headphones which should connect automatically. Finally, open the shell editor and type,

```
python3 main.py
```

This will run the script and after a brief start-up delay the transparent OLED will turn on and the chatbot will start talking! The smart glasses are now yours! Feel free to ask it a couple of test questions, such as "What's the news?", "What's the time?", "Send an SMS?", or "Play my music!"

Step 18: Troubleshooting



Errors

If you have any problems or errors, message me and I will put it in here and try to help you fix it. That way everyone could benefit thanks to you!

Trouble Installing xrdp

This problem may arise if you have already tried to install it before. To fix it open the shell editor and type in:

```
sudo apt-get remove xrdp vnc4server tightvncserver
```

Trouble Taking Pictures

If you weren't able to take pictures with the *'raspistill'* command it's possible the changes to raspi-config were never made or they didn't get saved. Check this by clicking the Raspberry Pi icon in the top left and navigate to preferences->Raspberry Pi Configuration->Interfaces. Make sure the box next to camera labeled 'enable' is checked. After that is done go to the shell editor and type *'sudo reboot'*.

Step 19: Conclusion

Congratulations! You have successfully built your very own smart glasses with the Raspberry Pi 0w. Thanks, for taking your time to read my Instructable. I hope that I have inspired some of you to tackle this project or another and if you have any any questions please put them in the comments down below where I will try and get back to you quickly.

> *For the wages of sin is death, but the gift of God is eternal life in Christ Jesus our Lord. -Romans 6:23*

If you want to read an instructable that has real value, check this out!