## How to Convert Speech to Text in Python

### Sound converter link:

<https://voice-recorder-online.com/> for voice record in online

<https://www.zamzar.com/convert/mp3-to-wav/> for convert one format to wav or other

Learning how to use Speech Recognition Python library for performing speech recognition to convert audio speech to text in Python.

Speech recognition is the ability of a computer software to identify words and phrases in spoken language and convert them to human readable text. In this tutorial, you will learn how you can convert speech to text in Python using [SpeechRecognition](https://pypi.org/project/SpeechRecognition/) library.

As a result, we do not need to build any machine learning model from scratch, this library provides us with convenient wrappers for various well known public speech recognition APIs (such as Google Cloud Speech API, IBM Speech To Text, etc.).

Alright, let's get started, installing the library using pip:

**pip3 install SpeechRecognition pydub**

Okey, open up a new Python file and import it:

import speech\_recognition as sr

The nice thing about this library is it supports several recognition engines:

* [CMU Sphinx](http://cmusphinx.sourceforge.net/wiki/) (offline)
* Google Speech Recognition
* [Google Cloud Speech API](https://cloud.google.com/speech/)
* [Wit.ai](https://wit.ai/)
* [Microsoft Bing Voice Recognition](https://azure.microsoft.com/en-us/services/cognitive-services/speech-services/)
* [Houndify API](https://www.houndify.com/)
* [IBM Speech To Text](https://www.ibm.com/watson)
* [Snowboy Hotword Detection](https://snowboy.kitt.ai/) (offline)

We gonna use Google Speech Recognition here, as it's straightforward and doesn't require any API key.

### Reading from a File

Make sure you have an audio file in the current directory that contains english speech (if you want to follow along with me, get the audio file [here](https://github.com/x4nth055/pythoncode-tutorials/blob/master/machine-learning/speech-recognition/16-122828-0002.wav)):

filename = "16-122828-0002.wav" #16-122828-0002.wav is sample wav audio file name

This file was grabbed from [LibriSpeech](https://www.openslr.org/12) dataset, but you can use any audio WAV file you want, just change the name of the file, let's initialize our speech recognizer:

# initialize the recognizer

r = sr.Recognizer()

The below code is responsible for loading the audio file, and converting the speech into text using Google Speech Recognition:

# open the file

with sr.AudioFile(filename) as source:

# listen for the data (load audio to memory)

audio\_data = r.record(source)

# recognize (convert from speech to text)

text = r.recognize\_google(audio\_data)

print(text)

This will take few seconds to finish, as it uploads the file to Google and grabs the output, here is my result:

**import** speech\_recognition **as** sr

filename **=** "lol.wav"

*# initialize the recognizer*

r **=** sr.Recognizer()

*# open the file*

**with** sr.AudioFile(filename) **as** source:

*# listen for the data (load audio to memory)*

    audio\_data **=** r.record(source)

*# recognize (convert from speech to text)*

    text **=** r.recognize\_google(audio\_data)

    print(text)

The above code works well for small or medium size audio files. In the next section, we gonna write code for large files.

Below two format could not understood so not ok

### Reading Large Audio Files

### Reading from the Microphone

## How to Convert Text to Speech in Python

Online synthesis: In this system need internet connection ,otherwise text not convert to file

Offline synthesis: In this system no need to internet connection

### Country language short form link:

<https://meta.wikimedia.org/wiki/Template:List_of_language_names_ordered_by_code>

Learn how you to perform speech synthesis by converting text to speech both online and offline using gTTS and pyttsx3 libraries in Python.

[Speech synthesis](https://en.wikipedia.org/wiki/Speech_synthesis) is the computer-generated simulation of human speech. It converts human language text into human-like speech. In this tutorial, you will learn how you can convert text to speech in Python.

In this tutorial, we won't be building neural networks and train the model in order to achieve results, as it is pretty complex and hard to do it. Instead, we gonna use some APIs and engines that offer it. There are a lot of APIs out there that offers this service, one of the commonly used services is Google Text to Speech, in this tutorial, we will play around with it along with another offline library: [pyttsx3](https://github.com/nateshmbhat/pyttsx3).

To make things clear, this tutorial is about converting text to speech and not the other way around, if you want to [convert speech to text](https://www.thepythoncode.com/article/using-speech-recognition-to-convert-speech-to-text-python) instead, check [this tutorial](https://www.thepythoncode.com/article/using-speech-recognition-to-convert-speech-to-text-python).

To get started, let's install required modules:

pip3 install gTTS pyttsx3 playsound

### Online Synthesis

Online synthesis: In this system need internet connection ,otherwise text not convert to file

As you may guess, [gTTS](https://github.com/pndurette/gTTS) stands for Google Text To Speech, it is a Python library that wraps the original API to ease the work for us.

Open up a new Python file and import:

It's pretty straightforward to use this library, you just need to pass text to gTTS object that is an interface to [Google Translate](https://translate.google.com/)'s Text-to-Speech API:

import gtts

from playsound import playsound

It's pretty straightforward to use this library, you just need to pass text to gTTS object that is an interface to [Google Translate](https://translate.google.com/)'s Text-to-Speech API:

# make request to google to get synthesis

tts = gtts.gTTS("Hello world")

Up to this point, we have sent the text and retrieved the actual audio speech, let's save this audio to a file:

# save the audio file

tts.save("hello.mp3")

Awesome, you'll see a new file appear in the current directory, let's play it using [playsound](https://pypi.org/project/playsound/) module installed previously:

#### Final code:

#### Example1:

এই প্রোগ্রাম এর মাধ্যমে যে কোনো টেক্সট , অডিও ফাইল এ সেভ এন্ড পড়া যাবে

*#online text to  audio file convert*

**import** gtts

**from** playsound **import** playsound

*# make request to google to get synthesis*

tts **=** gtts.gTTS("Hello worlde i am mahmud hossain how are you")

#tts **=** gtts.gTTS("Hello worlde i am mahmud hossain how are you,প্রোগ্রাম এর মাধ্যমে ")

*# save the audio file*

tts.save("hello.mp3")

#### example2:

এই প্রোগ্রাম এর মাধ্যমে যে কোনো টেক্সট , অন্য যে কোনো ভাষার মতো করে উচ্চারণ করে অডিও ফাইল এ সেভ এন্ড পড়া যাবে

*#read and speake any text*

**import** gtts

**from** playsound **import** playsound

*# make request to google to get synthesis*

*# in spanish*

tts **=** gtts.gTTS("আল্লাহ আপনি আমাকে হেদায়েত দান করুন ", **lang=**"uk")

tts.save("hello.mp3")

playsound("hello.mp3")

#### how know possible language and lang code

To get the list of available languages, use this:

# all available languages along with their IETF tag

print(gtts.lang.tts\_langs())

*#how much lang code*

**import** gtts

*# all available languages along with their IETF tag*

print(gtts.lang.tts\_langs())

### Offline Synthesis

Offline synthesis: In this system no need to internet connection

Now you know how to use Google's API, but what if you want to use text to speech technologies offline ? Well, [pyttsx3](https://github.com/nateshmbhat/pyttsx3) library comes into the rescue, it basically looks for TTS engines pre-installed in your platform, here are the text-to-speech synthesizers that this library uses:

* [SAPI5](https://en.wikipedia.org/wiki/Microsoft_Speech_API) on Windows XP, Windows Vista, 8, 8.1 and 10
* [NSSpeechSynthesizer](https://developer.apple.com/documentation/appkit/nsspeechsynthesizer) on Mac OS X 10.5 and 10.6
* [espeak](http://espeak.sourceforge.net/) on Ubuntu Desktop Edition 8.10, 9.04 and 9.10

We'll see in a minute how to use different drivers and voices in this library.

To get started with this library, open up a new Python file and import it:

To get started with this library, open up a new Python file and import it:

import pyttsx3

Now we need to initialize the TTS engine:

# initialize Text-to-speech engine

engine = pyttsx3.init()

Now to convert some text, we need to use say() and runAndWait() methods:

# convert this text to speech

text = "Python is a great programming language"

engine.say(text)

# play the speech

engine.runAndWait()

say() method adds an utterance to speak to the event queue, while runAndWait() method runs the actual event loop until all commands queued up. So you can call multiple times the say() method and run a single runAndWait() method in the end, in order to hear the synthesis, try it out!

This library provides us with some properties that we can tweak based on our needs. For instance, let's get the details of speaking rate:

# get details of speaking rate

rate = engine.getProperty("rate")

print(rate)

Alright, let's change this to 300 (make the speaking rate much faster):

# setting new voice rate (faster)

engine.setProperty("rate", 300)

engine.say(text)

engine.runAndWait()

Or slower:

# slower

engine.setProperty("rate", 100)

engine.say(text)

engine.runAndWait()

#### example1:

এই প্রোগ্রাম এর মাধ্যমে যেকোনো Only english টেক্সট ফাইল ডিফল্ট rate উচ্চারণ হবে

*#this proram ,can easily read any english text as default spaking rate in offline*

**import** pyttsx3

*# initialize Text-to-speech engine*

engine **=** pyttsx3.init()

*# convert this text to speech*

text **=** "Python is a great programming language"

engine.say(text)

*# play the speech*

engine.runAndWait()

#### example2:

এই প্রোগাম এর মধ্যে বই ডিফল্ট কত স্পিড এ রিড হোসে সেটা দেখা যাবে

*#by this proram ,can easily read any english text and show the default speaking rate in offline*

**import** pyttsx3

*# initialize Text-to-speech engine*

engine **=** pyttsx3.init()

*# convert this text to speech*

text **=** "Python is a great programming language"

engine.say(text)

*# play the speech*

engine.runAndWait()

*# get details of speaking rate*

rate **=** engine.getProperty("rate")

print(rate)

#### example3:

এই প্রোগাম এর মধ্যম এ ,স্পিকিং রাতে চেঞ্জ করা যাবে

*#by this program, easily read any english text with specific speaking rate in offline*

**import** pyttsx3

*# initialize Text-to-speech engine*

engine **=** pyttsx3.init()

*# convert this text to speech*

text **=** "hello i am mahmud hossain"

*# setting new voice rate (faster)*

engine.setProperty("rate", 100)

engine.say(text)

engine.runAndWait()

### Conclusion

Great, that's it for this tutorial, I hope that will help you build your application, or maybe your own virtual assistant in Python.

To conclude, if you want to use more reliable synthesis, Google TTS API is your choice, if you just want to make it work a lot faster and without Internet connection, you should use [pyttsx3](https://github.com/nateshmbhat/pyttsx3) library.

Here are the documentation for both libraries:

* [gTTS (Google Text-to-Speech)](https://gtts.readthedocs.io/en/latest/)
* [pyttsx3 - Text-to-speech x-platform](https://pyttsx3.readthedocs.io/en/latest/)

## How to Translate Text by Python

### Country language short form link:

<https://meta.wikimedia.org/wiki/Template:List_of_language_names_ordered_by_code>

Google translate is a free service that translates words, phrases and entire web pages into more than 100 languages. You probably already know it and you have used it many times in your life.

In this tutorial, you will learn how to perform text translation in Python using [Googletrans](https://py-googletrans.readthedocs.io/en/latest/) library. Googletrans is a free and unlimited Python library that make unofficial [Ajax](https://en.wikipedia.org/wiki/Ajax_(programming)) calls to Google Translate API in order to detect languages and translate text.

First, let's install it using pip:

**pip3 install googletrans**

### Translating Text

এই প্রোগ্রাম এর মাধ্যমে যে কোনো ল্যাঙ্গুয়েজে by ডিফল্ট ভাবে ইংলিশ এ কনভার্ট হয়ে যাবে

*#by default any string convert to english*

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

translator **=** Translator()

*# translate a spanish text to english text (by default)*

translation **=** translator.translate("আমি তোমাকে ভালো বাসি") *#example: bangla to english*

*#translation = translator.translate("मैं तुम्हें पसंद करता हूं")     #example: hindi to english*

*#translation = translator.translate("私はあなたが好きです")    #example: japan to english*

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

*#any string or language translating a specific language*

এই প্রোগ্রাম এর মাধ্যমে যে কোনো ল্যাংগুয়েজ কোনো নিদ্ধিষ্ট ল্যাঙ্গুয়েজে এ কনভার্ট করা যাবে

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

translator **=** Translator()

translation **=** translator.translate("আমি তোমাকে ভালো বাসি", **dest=**"ar") *#bangla to arabic(ar)*

*#translation = translator.translate("আমি তোমাকে ভালো বাসি", dest="bn") #bangla to bangla(bn)*

*#translation = translator.translate("আমি তোমাকে ভালো বাসি", dest="en") #bangla to english(en)*

*#translation = translator.translate("আমি তোমাকে ভালো বাসি", dest="el") #bangla to greek(el)*

*#translation = translator.translate("আমি তোমাকে ভালো বাসি", dest="de") #bangla to garman(de)*

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

*#any specific source language  translating  to English*

এই প্রোগ্রাম এর মাধ্যমে যে কোনো নিদ্ধিষ্ট ল্যাংগুয়েজ এবং তার কোড দিয়া , নিদ্ধিষ্ট ইংলিশ ল্যাঙ্গুয়েজে কনভার্ট করা যাবে

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

translator **=** Translator()

*# specify source language*

*#translation = translator.translate("আমি তোমাকে ভালো বাসি?", src="bn") #bangla*

*#translation = translator.translate("私はあなたが好きです", src="ja")    #japani*

*#translation = translator.translate("EGO amo te", src="la")  #latin*

translation **=** translator.translate("أنا أحبك", **src=**"ar")   *#arabic*

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

*#to get extra information ,we use extra data funtion al last line*

এই প্রোগ্রাম এর মাধ্যমে কোনো ল্যাংগুয়েজ কনভার্সন এর সাথে সাথে এর ডিটেলস জানা যাবে

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

translator **=** Translator()

*# specify source language*

translation **=** translator.translate("আমি তোমাকে ভালো বাসি?", **src=**"bn") *#bangla*

*#translation = translator.translate("私はあなたが好きです", src="ja")    #japani*

*#translation = translator.translate("EGO amo te", src="la")  #latin*

*#translation = translator.translate("أنا أحبك", src="ar")   #arabic*

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

*# print all translations and other data*

pprint(translation.extra\_data)

Importing necessary libraries:

from googletrans import Translator, constants

from pprint import pprint

Googletrans provides us with a convenient interface, let's initialize our translator instance:

#init the Google API translator

translator = Translator()

Now we simply use translate() method to get the translated text:

# translate a spanish text to english text (by default)

translation = translator.translate("Hola Mundo")

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

This will print the original text and language along with the translated text and language:

Hola Mundo (es) --> Hello World (en)

It automatically detects the language and translate to english by default, let's translate to another language, arabic for instance:

# translate a spanish text to arabic for instance

translation = translator.translate("Hola Mundo", dest="ar")

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

"ar" is the language code for arabic, here is the output:

Hola Mundo (es) --> مرحبا بالعالم (ar)

Now let's set a source language and translate to English:

# specify source language

translation = translator.translate("Wie gehts ?", src="de")

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

Output:

Wie gehts ? (de) --> How are you ? (en)

You can also check other translations and some other extra data:

# print all translations and other data

pprint(translation.extra\_data)

See the output:

{'all-translations': [['interjection',

['How are you doing?', "What's up?"],

[['How are you doing?', ["Wie geht's?"]],

["What's up?", ["Wie geht's?"]]],

"Wie geht's?",

9]],

'confidence': 1.0,

'definitions': None,

'examples': None,

'language': [['de'], None, [1.0], ['de']],

'original-language': 'de',

'possible-mistakes': None,

'possible-translations': [['Wie gehts ?',

None,

[['How are you ?', 1000, True, False],

["How's it going ?", 1000, True, False],

['How are you?', 0, True, False]],

[[0, 11]],

'Wie gehts ?',

0,

0]],

'see-also': None,

'synonyms': None,

'translation': [['How are you ?', 'Wie gehts ?', None, None, 1]]}

A lot of data to benefit from, you have all the possible translations, confidence, definitions and even examples.

### Translating List of Phrases

এই প্রোগ্রামটা দিয়ে আমরা মাল্টিপল সেন্টেন্স অথবা phase অন্য কোনো নিদ্ধিষ্ট ল্যাঙ্গুয়েজে এ ট্রান্সলেট করতে পারি ,এবং কোড এর শেষ লাইন দিয়া ডিটেলস জানতে পারি

#### Final code:

*#convert english phase or more sentence to specific language with extra . function*

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

translator **=** Translator()

*# specify source language*

*# translate more than a phrase*

sentences **=** [

    "Hello everyone",

    "How are you ?",

    "Do you speak english ?",

    "Good bye!"

]

translations **=** translator.translate(sentences, **dest=**"en")

**for** translation **in** translations:

    print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

pprint(translation.extra\_data)

You can also pass a list of text to translate each sentence individually:

# translate more than a phrase

sentences = [

"Hello everyone",

"How are you ?",

"Do you speak english ?",

"Good bye!"

]

translations = translator.translate(sentences, dest="tr")

for translation in translations:

print(f"{translation.origin} ({translation.src}) --> {translation.text} ({translation.dest})")

Output:

Hello everyone (en) --> herkese merhaba (tr)

How are you ? (en) --> Nasılsın ? (tr)

Do you speak english ? (en) --> İngilizce biliyor musunuz ? (tr)

Good bye! (en) --> Güle güle! (tr)

### Language and language code Detection

এই প্রোগ্রামটা দিয়ে আমরা কোন ল্যাঙ্গুয়েজ টেক্সট লিখেছি তার আউটপুট ল্যাঙ্গুয়েজে শর্ট কোড পাবো

#### Final code:

##### Language detection:

*#in here give any language sentence and found the language code, and fun language*

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

*# detect a language*

translator **=** Translator()

*#detection = translator.detect("नमस्ते दुनिया") #hindi*

detection **=** translator.detect("আমি তোমাকে ভালো বাসি ") *# bangla*

*#detection = translator.detect("EGO amo te") #latin*

*#detection = translator.detect("أنا أحبك")   #arabic*

print("Language:", constants.LANGUAGES[detection.lang])

##### language code detection:

*#in here give any language sentence and found the language code*

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# init the Google API translator*

*# detect a language*

translator **=** Translator()

*#detection = translator.detect("नमस्ते दुनिया") #hindi code(hi)*

detection **=** translator.detect("আমি তোমাকে ভালো বাসি ") *# bangla code(bn)*

*#detection = translator.detect("EGO amo te") #latin code(la)*

*#detection = translator.detect("أنا أحبك")   #arabic code(ar)*

print("Language code:", detection.lang)

print("Confidence:", detection.confidence)

Google Translate API offers us language detection call as well:

# detect a language

detection = translator.detect("नमस्ते दुनिया")

print("Language code:", detection.lang)

print("Confidence:", detection.confidence)

This will print the code of the detected language along with confidence rate (1.0 means 100% confident):

Language code: hi

Confidence: 1.0

This will return the language code, to get the full language name, you can use the LANGUAGES dictionary provided by Googletrans:

print("Language:", constants.LANGUAGES[detection.lang])

Output:

Language: hindi

### Supported Languages

পাইথনে টোটাল কত ধরনের ল্যাঙ্গুয়েজ ট্রান্সলেশন সাপোর্ট করে ,নিচের প্রোগ্রাম টা রান করলেই তা শো করবে

#### Final program:

*#by this program we can find totall python language avaliable*

**from** googletrans **import** Translator, constants

**from** pprint **import** pprint

*# print all available languages*

print("Total supported languages:", len(constants.LANGUAGES))

print("Languages:")

pprint(constants.LANGUAGES)

As you may know, Google Translate supports more than 100 languages, let's print all of them:

# print all available languages

print("Total supported languages:", len(constants.LANGUAGES))

print("Languages:")

pprint(constants.LANGUAGES)

Here is a truncated output:

Total supported languages: 106

{'af': 'afrikaans',

'am': 'amharic',

'ar': 'arabic',

'az': 'azerbaijani',

'be': 'belarusian',

'bg': 'bulgarian',

...

<SNIPPED>

...

'yo': 'yoruba',

'zh-cn': 'chinese (simplified)',

'zh-tw': 'chinese (traditional)',

'zu': 'zulu'}

### Conclusion

There you have it, this library is a great deal for everyone that wants a quick way to translate text in an application. However, this library is unofficial as mentioned earlier, the author noted that the maximum character length on a single text is 15K.

It also doesn't guarantee that the library would work properly at all times, if you want to use a stable API you should use the [official Google Translate API](https://cloud.google.com/translate/docs).

Using this library a lot, Google translate may block your IP address, consider [using proxies](https://www.thepythoncode.com/article/using-proxies-using-requests-in-python) by passing a proxy dictionary to proxies parameter in Translator() class, or use the official API as discussed.

I've written a quick Python script that will allow you to translate text as well as documents in the command line, check it [here](https://www.thepythoncode.com/code/translate-text-in-python#translate_doc).

Finally, I encourage you to further explore the library, check out its [official documentation](https://py-googletrans.readthedocs.io/en/latest/).

## How to Play and Record Audio in Python

<https://www.thepythoncode.com/article/play-and-record-audio-sound-in-python>

Package install help link:

<https://stackoverflow.com/questions/52283840/i-cant-install-pyaudio-on-windows-how-to-solve-error-microsoft-visual-c-14>

Learn how to play and record sound files using different libraries such as playsound, Pydub and PyAudio in Python.

Many of the applications out there records your voice as well as playing sounds, if you want to do that as well, then you came into the right place, in this tutorial, we will be using different Python libraries to play and ..

### Audio Player by playsound liabrary

First, we gonna start with the most straightforward module here, [playsound](https://pypi.org/project/playsound/):

Code:

**from** playsound **import** playsound

playsound("cool1.mp3")

Yes, that's it required for this module, it is basically a pure Python, cross platform, single function module. The [documentation](https://pypi.org/project/playsound/) says that WAV and MP3 extensions are known to work, and it may work for other formats as well.

playsound() function plays the sound in the audio file and blocks until the file reading is completed, you can pass block=False to make the function run asynchronously.

### Audio Player by pydub liabrary (not work)

Another alternative is to use [Pydub](https://github.com/jiaaro/pydub) library:

<https://github.com/jiaaro/pydub>

<https://pythonawesome.com/manipulate-audio-with-a-simple-and-easy-high-level-interface/>

## How to Get Geolocation in Python

<https://www.thepythoncode.com/article/get-geolocation-in-python>

<https://geopy.readthedocs.io/en/stable/>

<https://pypi.org/project/geopy/>

<https://nominatim.openstreetmap.org/ui/search.html>

<https://developers.google.com/maps/documentation/geocoding/overview>

এই প্রোগ্রাম এর মাধ্যমে খুব সহজেই পৃথিবীর যে কোনো দেশ,শহর,গ্রাম .. এর ডিটেলস বের করা যাই ,কি এমনকি পোস্টকোড ,জিপকোড

Learn how to use GeoPy library to geocode physical addresses into latitude and longitude and vice versa; getting latitude and longitude from addresses, towns, cities and more in Python.

Geocoding is the process of transforming a description of a location (such as a physical address, or a name of a place) into a pair of latitude and longitude on the Earth's surface for that place.

It also refers to converting geographic coordinates to a description of a location (such as an address), this is often called reverse geocoding. In this tutorial, we will learn how to do both with the help of GeoPy library in Python.

[GeoPy](https://pypi.org/project/geopy/) is a Python client that provides several popular geocoding web services, it makes it easy for Python developers to locate the coordinates of an address, a city, or a country and vice-versa.

To get started, let's install it:

pip3 install geopy

GeoPy provides many geocoding service wrappers, such as [OpenStreetMap Nominatim](https://nominatim.openstreetmap.org/), [Google Geocoding API V3](https://developers.google.com/maps/documentation/geocoding/overview), Bing Maps and more. In this tutorial, we gonna stick with OpenStreetMap Nominatim.

Here is what we'll be covering:

* [Getting Latitude and Longitude from an Address (Geocoding)](https://www.thepythoncode.com/article/get-geolocation-in-python#Geocoding)
* [Getting Address from Latitude and Longitude (Reverse Geocoding)](https://www.thepythoncode.com/article/get-geolocation-in-python#Reverse_Geocoding)

### Getting Latitude and Longitude from an Address (Geocoding)

In this section, we will be using OpenStreetMap Nominatim API to get latitude and longitude from a physical address, city, or any location name.

Let's import the library first:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

Notice we chose Nominatim geocoder, now creating a new instance of it:

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

Now let's try to get geographic data from an address:

#### Full code:

##### Example1:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

location **=** app.geocode("Dhaka, Bangladesh").raw

*# print raw data*

pprint(location)

##### output :



##### Example2:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

location **=** app.geocode("rajshahi, Bangladesh").raw

*# print raw data*

pprint(location)

##### output:



##### Example3:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

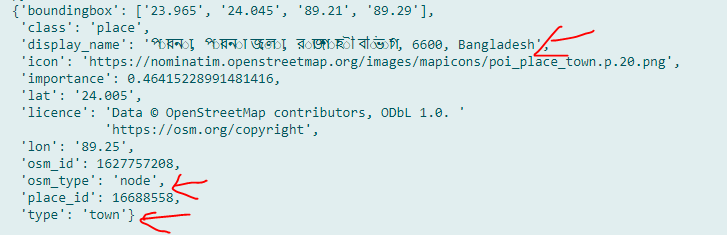
*# get location raw data*

location **=** app.geocode("pabna, Bangladesh").raw

*# print raw data*

pprint(location)

##### output:



Awesome, we have latitude at lat attribute (in which we can access by location['lat']) and longitude at lon attribute, we also have access to the bounding box of the address at the boundingbox attribute.

As you can see, the Nominatim API does not require a full address (that consists of street, house number and city), you can also pass business addresses and points of your interests, it supports that!

##### Example4:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

location **=** app.geocode("282,bangladesh").raw

*# print raw data*

pprint(location)

##### output:



##### Example5:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

location **=** app.geocode("Kolkata,India").raw

*# print raw data*

pprint(location)

##### Output:



Example6:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

location **=** app.geocode("dhaka,India").raw

*# print raw data*

pprint(location)

Output :



Example7:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

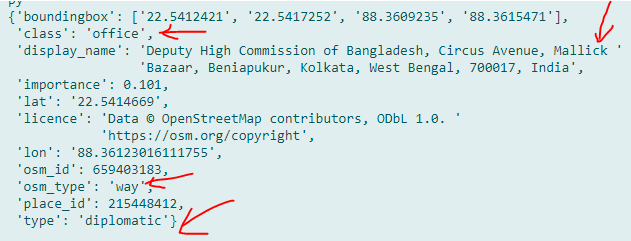
*# get location raw data*

location **=** app.geocode("kalkata, Bangladesh").raw

*# print raw data*

pprint(location)

Output :



Example8:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

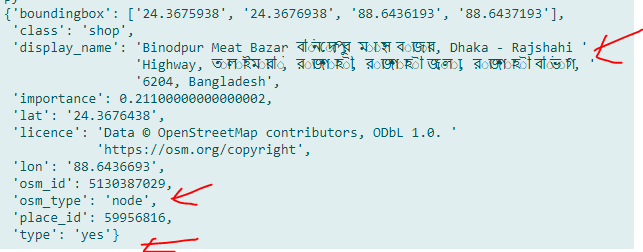
*# get location raw data*

location **=** app.geocode("Binodpur, Bangladesh").raw

*# print raw data*

pprint(location)

Output :



##### Unlimited example

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

*#location = app.geocode("Dhorompur").raw #show neyborhood*

*#location = app.geocode("Binodpur").raw #show village*

*#location = app.geocode("282").raw     #show postcode*

*#location = app.geocode("6060").raw    #show postal-code*

location **=** app.geocode("i am mahmud").raw     *#village of iraq*

*# print raw data*

pprint(location)

However, if you call this function repeatedly (such as iterating over a list of addresses), you will encounter a timed out error, and that's because if you read the [Nominatim Usage Policy](https://operations.osmfoundation.org/policies/nominatim/), it requires you to use a maximum of 1 request per second, and that's absolutely acceptable, since it's a free service.

As a result, the below function respects that requirement and sleeps for one second before making a request:

কোনো কারণে এড্রেস খুঁজে না পেলে এই ফাঙ্কশন এক সেকেন্ড ধরে সার্চ করবে এবং সিলিপ মোড চলে যাবে দেন এগেইন সার্চ করে একটা রেজাল্ট আনবে

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

*# get location raw data*

location **=** app.geocode("Dhaka, Bangladesh").raw     *#village of iraq*

*# print raw data*

pprint(location)

**def** get\_location\_by\_address(**address**):

    """This function returns a location as raw from an address

    will repeat until success"""

    time.sleep(1)

**try**:

**return** app.geocode(address).raw

**except**:

**return** get\_location\_by\_address(address)

So whenever a timed out error is raised, we catch that and call the function recursively, and this function will sleep for another second and hopefully, retrieves the result:

### Getting Address from Latitude and Longitude (Reverse Geocoding)

ওপারে আমার এড্রেস থেকে ল্যাটিচুড এন্ড লগ্গি টুডে বের করেদিলাম, যেকোন সেই বেরকরা ল্যাটিচুডে এন্ড লংগিচুদে উসে করে এড্রেস বের করতে হবে

Now to retrieve an address, city and country and various of other information, only from latitude and longitude, we simply use reverse() method instead of geocode(), which accepts coordinates (latitude and longitude) as string separated by a comma.

The following function reverse geocode the coordinates along with respecting Nominatim usage policy:

#### Fullcode:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

**def** get\_address\_by\_location(**latitude**, **longitude**, **language=**"en"):

    """This function returns an address as raw from a location

    will repeat until success"""

*# build coordinates string to pass to reverse() function*

    coordinates **=** f"{latitude}, {longitude}"

*# sleep for a second to respect Usage Policy*

*#time.sleep(1)*

**try**:

**return** app.reverse(coordinates, **language=**language).raw

**except**:

**return** get\_address\_by\_location(latitude, longitude)

*# define your coordinates*

latitude **=** 23.810651

longitude **=** 90.4126466

*# get the address info*

address **=** get\_address\_by\_location(latitude, longitude)

*# print all returned data*

pprint(address)

উপরে বেরকরা সেই ল্যাটিচুডে এন্ড লংগিচুদে use করে এড্রেস বের করতে হবে

##### example1:

**from** geopy.geocoders **import** Nominatim

**import** time

**from** pprint **import** pprint

*# instantiate a new Nominatim client*

app **=** Nominatim(**user\_agent=**"tutorial")

**def** get\_address\_by\_location(**latitude**, **longitude**, **language=**"en"):

    """This function returns an address as raw from a location

    will repeat until success"""

*# build coordinates string to pass to reverse() function*

    coordinates **=** f"{latitude}, {longitude}"

*# sleep for a second to respect Usage Policy*

*#time.sleep(1)*

**try**:

**return** app.reverse(coordinates, **language=**language).raw

**except**:

**return** get\_address\_by\_location(latitude, longitude)

*# define your coordinates*

latitude **=** 23.810651

longitude **=** 90.4126466

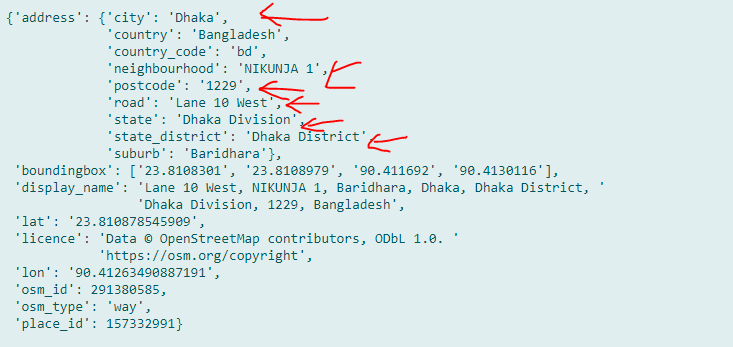
*# get the address info*

address **=** get\_address\_by\_location(latitude, longitude)

*# print all returned data*

pprint(address)

##### output:



So this will return all address data, including state, town, postcode, districts and more. If you wish to return these information in a specific language, you can set language parameter to the language you desire, or you can set it to False for the default language for that specific location.

### Welcome to Nominatim

Nominatim is a search engine for [OpenStreetMap](https://www.openstreetmap.org) data. This is the debugging interface. You may search for a name or address (forward search) or look up data by its geographic coordinate (reverse search). Each result comes with a link to a details page where you can inspect what data about the object is saved in the database and investigate how the address of the object has been computed.

For more information visit the [Nominatim home page](https://nominatim.org).

### Conclusion

As always, we only saw simple examples of what GeoPy can do, I highly suggest you read [the documentation](https://geopy.readthedocs.io/en/stable/) if you're interested in more advanced utilities.

## How to Generate and Read QR Code in Python

<https://www.thepythoncode.com/article/generate-read-qr-code-python>

<https://pypi.org/project/qrcode/>

Learning how you can generate and read QR Code in Python using qrcode and OpenCV libraries.

[**QR code**](https://en.wikipedia.org/wiki/QR_code) is a type of matrix barcode that is machine readable optical label which contains information about the item to which it is attached. In practice, QR codes often contain data for a locator, identifier, or tracker that points to a website or application, etc.

In this tutorial, you will learn how to generate and read QR codes in Python using [qrcode](https://pypi.org/project/qrcode/) and [OpenCV](https://pypi.org/project/opencv-python/) libraries.

Installing required dependencies:

pip3 install opencv-python qrcode

### Generate QR Code

First, let's start by generating QR codes, it is basically straight forward using [qrcode](https://pypi.org/project/qrcode/) library:

import qrcode

# example data

data = "https://www.thepythoncode.com"

# output file name

filename = "site.png"

# generate qr code

img = qrcode.make(data)

# save img to a file

img.save(filename)

This will generate a new image file in the current directory with the name of "site.png", which contains a QR code image of the data specified (in this case, this website).

### Read QR Code

There are many tools that reads QR code. However, we will be using [OpenCV](https://pypi.org/project/opencv-python/) for that, as it is popular and easy to integrate with the webcam or any video.

Alright, open up a new Python file and follow along with me, let's read the image that we just generated:

import cv2

# read the QRCODE image

img = cv2.imread("site.png")

Luckily for us, OpenCV already got QR code detector built in:

# initialize the cv2 QRCode detector

detector = cv2.QRCodeDetector()

We have the image and the detector, let's detect and decode that data:

# detect and decode

data, bbox, straight\_qrcode = detector.detectAndDecode(img)

detectAndDecode() function takes an image as an input and decodes it to return a tuple of 3 values: the data decoded from the QR code, the output array of vertices of the found QR code quadrangle and the output image containing rectified and binarized QR code.

We just need data and bbox here, bbox will help us draw the quadrangle in the image and data will be printed to the console!

Let's do it:

# if there is a QR code

if bbox is not None:

print(f"QRCode data:\n{data}")

# display the image with lines

# length of bounding box

n\_lines = len(bbox)

for i in range(n\_lines):

# draw all lines

point1 = tuple(bbox[i][0])

point2 = tuple(bbox[(i+1) % n\_lines][0])

cv2.line(img, point1, point2, color=(255, 0, 0), thickness=2)

cv2.line() function draws a line segment connecting two points, we retrieve these points from bbox array that was decoded by detectAndDecode() previously. we specified a blue color ( (255, 0, 0) is blue as OpenCV uses BGR colors ) and thickness of 2.

Finally, let's show the image and quit when a key is pressed:

# display the result

cv2.imshow("img", img)

cv2.waitKey(0)

cv2.destroyAllWindows()

Once you run this, the decoded data is printed:

QRCode data:

https://www.thepythoncode.com

And the following image is shown:



As you can see, the blue lines are drawn in the exact QR code borders. Awesome, we are done with this script, try to run it with different data and see your own results !

If you want to detect and decode QR codes live using your webcam (and I'm sure you do), here is a code for that:

import cv2

# initalize the cam

cap = cv2.VideoCapture(0)

# initialize the cv2 QRCode detector

detector = cv2.QRCodeDetector()

while True:

\_, img = cap.read()

# detect and decode

data, bbox, \_ = detector.detectAndDecode(img)

# check if there is a QRCode in the image

if bbox is not None:

# display the image with lines

for i in range(len(bbox)):

# draw all lines

cv2.line(img, tuple(bbox[i][0]), tuple(bbox[(i+1) % len(bbox)][0]), color=(255, 0, 0), thickness=2)

if data:

print("[+] QR Code detected, data:", data)

# display the result

cv2.imshow("img", img)

if cv2.waitKey(1) == ord("q"):

break

cap.release()

cv2.destroyAllWindows()

Awesome, we are done with this tutorial, you can now integrate this in your own applications!

Check [qrcode's official documentation](https://pypi.org/project/qrcode/).

Finally, many of the Python concepts aren't discussed in detail here, if you feel you want to dig more to Python, I highly suggest you get one of these amazing courses:

### Exapmle1:

যেকোনো text ,link,... কে QR কোড এ কনভার্ট করা যাই

*#create any thing to QR-code*

**import** qrcode

*# example data*

data **=** "https://www.facebook.com/mahmud.hossain.12135/" *#my fb id link*

data**=**'i am mahmud hossain' *#any sentence*

*# output file name*

filename **=** "lol.png" *#give file name and type*

*# generate qr code*

img **=** qrcode.make(data)

*# save img to a file*

img.save(filename)

### Example2:

যে কোনো কোড QR-code ডিকোড করে ইন্টারনাল ইনফরমেশন শো করে

*#it take a QR code and decode it and show its data and also show the QR code also*

**import** cv2

*# read the QRCODE image*

img **=** cv2.imread("si.png")

*# initialize the cv2 QRCode detector*

detector **=** cv2.QRCodeDetector()

*# detect and decode*

data, bbox, straight\_qrcode **=** detector.detectAndDecode(img)

print(data)

*# display the result*

cv2.imshow("img", img)

cv2.waitKey(0)

cv2.destroyAllWindows()

### Example3:

ডিকোড করার QR-Codeএর অল সাইড ব্লু ব্রডের শো করে

*#if want to show blue or other color all side or QR code use this program*

**import** cv2

*# read the QRCODE image*

img **=** cv2.imread("site.png")

*# initialize the cv2 QRCode detector*

detector **=** cv2.QRCodeDetector()

*# detect and decode*

data, bbox, straight\_qrcode **=** detector.detectAndDecode(img)

*# if there is a QR code*

**if** bbox **is** **not** None:

    print(f"QRCode data:\n{data}")

*# display the image with lines*

*# length of bounding box*

    n\_lines **=** len(bbox)

**for** i **in** range(n\_lines):

*# draw all lines*

        point1 **=** tuple(bbox[i][0])

        point2 **=** tuple(bbox[(i**+**1) **%** n\_lines][0])

        cv2.line(img, point1, point2, **color=**(0,0,0), **thickness=**5)

*# display the result*

cv2.imshow("img", img)

cv2.waitKey(0)

cv2.destroyAllWindows()

### example4:

ল্যাপটপ ক্যামেরা use করেQR-Code স্ক্যান

*#this code scan QR code by laptop camera*

**import** cv2

*# initalize the cam*

cap **=** cv2.VideoCapture(0)

*# initialize the cv2 QRCode detector*

detector **=** cv2.QRCodeDetector()

**while** True:

    \_, img **=** cap.read()

*# detect and decode*

    data, bbox, \_ **=** detector.detectAndDecode(img)

*# check if there is a QRCode in the image*

**if** bbox **is** **not** None:

*# display the image with lines*

**for** i **in** range(len(bbox)):

*# draw all lines*

            cv2.line(img, tuple(bbox[i][0]), tuple(bbox[(i**+**1) **%** len(bbox)][0]), **color=**(255, 0, 0), **thickness=**2)

**if** data:

            print("[+] QR Code detected, data:", data)

*# display the result*

    cv2.imshow("img", img)

**if** cv2.waitKey(1) **==** ord("q"):

**break**

cap.release()

cv2.destroyAllWindows()

## How to Make a Screen Recorder in Python

I cannot working with the project

<https://www.thepythoncode.com/article/make-screen-recorder-python>

Using pyautogui and OpenCV to record display screen video and save it to a file in Python.

Screen recording enables you to create demonstration videos, record gaming achievements and create videos that can be shared online on social media. Many industrial softwares exists out there that can help you do that very easily though. In this tutorial, you will learn how to make your own simple screen recorder in Python that you can further extend to your own needs.

Let's get started, first, install the required dependencies for this tutorial:

pip3 install numpy opencv-python pyautogui

Let's initialize the format we gonna use to write our video file ( named "output.avi" ):

**Note: You need to get the correct SCREEN\_SIZE from your operating system, that is the screen resolution, otherwise writing to the file won't work (alternatively, you can use pyautogui.size() function to get the size of the primary monitor).**

fourcc is the video codec library that OpenCV will use to write the video file, we specified [XVID](https://en.wikipedia.org/wiki/Xvid) here. The **20.0** float value passed as third parameter to cv2.VideoWriter corresponds to the FPS (**F**rame **P**er **S**econd).

Now we need to keep capturing screenshots and writing to the file in a loop until the user clicks the "q" button, here is the main loop for that:

First, we use the screenshot() function which returns an image object, so we need to convert it to a proper numpy array. After that, we need to convert that frame into **RGB**, that's because OpenCV uses **BGR** by default.

As mentioned in [pyautogui's official documentation](https://pyautogui.readthedocs.io/en/latest/), you can also record only regions of your screen, by passing region keyword argument which is a four-integer tuple representing the **top**, **left**, **width** and **height** of the region to capture, here is how it's done:

After you are done with recording, just click **"q"**, it will destroy the window and finish writing to the file, try it out!

Also, you can replace the while True statement with a for loop as follows:

This will record your screen for 10 seconds, that's because we set the FPS to 20 (which makes sense because 200 is 20 times 10)

Alright, there are endless of ideas you can use to extend this. For example, you can combine this with an [audio recorder](https://www.thepythoncode.com/article/play-and-record-audio-sound-in-python), and you'll come up with a Python tool that records your screen and voice simultaneously, you will need to [use a thread](https://www.thepythoncode.com/article/using-threads-in-python) that records audio and another for the screen recorder, let us know your progress in the comments below !

Or you can create keyboard shortcuts that starts, pauses and stops recording, [this tutorial](https://www.thepythoncode.com/article/control-keyboard-python) can help you.

### Total code by using while loop:

**import** cv2

**import** numpy **as** np

**import** pyautogui

*# display screen resolution, get it from your OS settings*

SCREEN\_SIZE **=** (1366, 768)

*# define the codec*

fourcc **=** cv2.VideoWriter\_fourcc(**\***"XVID")

*# create the video write object*

out **=** cv2.VideoWriter("output.avi", fourcc, 20.0, (SCREEN\_SIZE))

**while** True:

*# make a screenshot*

    img **=** pyautogui.screenshot()

*# convert these pixels to a proper numpy array to work with OpenCV*

    frame **=** np.array(img)

*# convert colors from BGR to RGB*

    frame **=** cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

*# write the frame*

    out.write(frame)

*# show the frame*

    cv2.imshow("screenshot", frame)

*# if the user clicks q, it exits*

**if** cv2.waitKey(1) **==** ord("q"):

**break**

*# make sure everything is closed when exited*

cv2.destroyAllWindows()

out.release()

*# make sure everything is closed when exited*

cv2.destroyAllWindows()

out.release()

img **=** pyautogui.screenshot(**region=**(0, 0, 300, 400))

this record stop when press ‘Q’ button

### total code by using for loop:

**import** cv2

**import** numpy **as** np

**import** pyautogui

*# display screen resolution, get it from your OS settings*

SCREEN\_SIZE **=** (1366, 768)

*# define the codec*

fourcc **=** cv2.VideoWriter\_fourcc(**\***"XVID")

*# create the video write object*

out **=** cv2.VideoWriter("output.avi", fourcc, 20.0, (SCREEN\_SIZE))

**for** i **in** range(50):

*# make a screenshot*

    img **=** pyautogui.screenshot()

*# the rest of the code...*

*# convert these pixels to a proper numpy array to work with OpenCV*

    frame **=** np.array(img)

*# convert colors from BGR to RGB*

    frame **=** cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

*# write the frame*

    out.write(frame)

*# show the frame*

    cv2.imshow("screenshot", frame)

*# if the user clicks q, it exits*

**if** cv2.waitKey(1) **==** ord("q"):

**break**

*# make sure everything is closed when exited*

cv2.destroyAllWindows()

out.release()

img **=** pyautogui.screenshot(**region=**(0, 0, 300, 400))

this record stop when loop is end

## How to Download All Images from a Web Page in Python

<https://www.thepythoncode.com/article/download-web-page-images-python>

## Face Detection using OpenCV in Python

<https://www.thepythoncode.com/article/detect-faces-opencv-python>

<https://www.codingforentrepreneurs.com/blog/opencv-python-web-camera-quick-test/>

<https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html>

<https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition/>

## How to Convert Python Files into Executables

<https://www.thepythoncode.com/article/building-python-files-into-stand-alone-executables-using-pyinstaller>

Using PyInstaller package to bundle Python applications and all its dependencies into a single stand-alone executables.

Have you ever wanted to quickly build your Python application into a single Windows .exe file ? or you want to bundle a large Python project into a single executable package ? Well, in this tutorial, you will learn how you can use [PyInstaller](https://www.pyinstaller.org/) package to be able to do that.

PyInstaller freezes Python applications into a stand-alone executables under Windows, GNU/Linux, Mac OS X, FreeBSD, Solaris and AIX. The user can run and execute the resulting packaged application without installing a Python interpreter or any modules. It supports Python 2.7 and Python 3.4+ and successfully bundles the major and famous Python packages such as numpy, PyQT, Django and more.

PyInstaller isn't cross-compiler, which means if you want to make a Windows executable (.exe), you need to run Pyinstaller on Windows, and if you want a GNU/Linux app, you do the same, etc.

Pyinstaller analyzes your code to discover every other module and library your script is using in order to execute, then it collects copies of all those files and puts them with your script in a single folder (or a single executable).

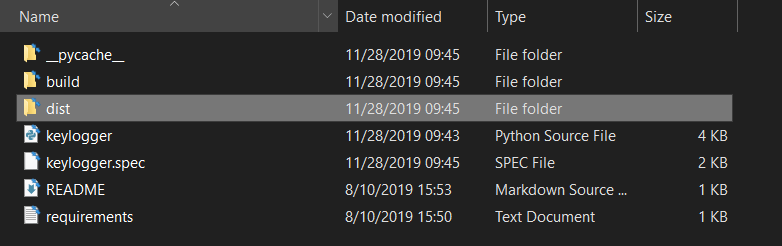
Alright, it is pretty simple to install it:

pip3 install pyinstaller

To get started, I'm gonna use an example Python file from [writing a keylogger](https://www.thepythoncode.com/article/write-a-keylogger-python) tutorial, to show you how you can exactly use Pyinstaller.

So, to run it, you need to be in the same directory of the Python file you wanna build and write the following:

This will build keylogger.py file in a couple of seconds, here is my resulting folders:



As you can see, two new folders appeared ("build" and "dist"), you'll find most of the metadata and log files into build folder. As a result, we're more interested in the dist folder, after opening it up, you'll see something like this:

