BUILD AN ADVANCED KEYLOGGER_ <WITH PYTHON>

Crash Course Application and Notes

cybercademy

Grant Collins

DOCUMENT OVERVIEW

WHY IT'S IMPORTANT TO TRY BEFORE FOLLOWING A COURSE

The online learning community presents a great opportunity to learn almost anything new for free or a relatively low cost. It's great to have online information, courses, books, and other resources freely available to increase your knowledge and skillset in a topic. However, one very important and critical piece of the learning process is being able to apply what you learn from a course to the real-world. And often, it can be challenging to apply what you have learned. It's easier to go through the course which has all the answers provided right away than trying to solve the problem on your own…

But being able to apply what you are learning is critical. Therefore, before you walk through a tutorial, online course, or guide, I highly recommend you try building your own version of the topic… It doesn't have to be the same exact implementation as the instructor. It's more about the thought process that goes into the idea over the answer. Through time, you will find it easier to solve and create your very own implementations of a concept you think of.

With all of this being said, I encourage you to build your very own version of the advanced keylogger in python before watching the online crash course.

Not only will you be able to better understand the functionalities of the keylogger through your own application and thought processes, but you will also be able to retain what you are learning.

Use this small guide as a way of helping you think through each functionality of the keylogger. Sometimes it's nice to have a "launch pad" which helps you get started in the thinking process and once you consider the process, you will start to come up with ideas of implementation. I hope this helps!

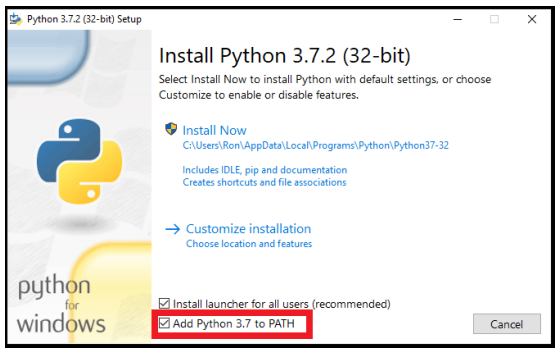
- Grant Collins

GETTING STARTED – PYTHON, PYCHARM AND MODULES

Before you start writing code, it's important to ensure you have Python installed as well as the proper modules.

Step One: Go to https://www.python.org, navigate to the downloads section and download the latest version of python.

Step Two: Go through the setup wizard and make sure to install pip as well as add python to the path (screenshot credit: Data to Fish)



Step Three: Go to https://www.jetbrains.com/pycharm/download/#section=windows, under Community, choose the free download option. Go through the setup wizard using default options.

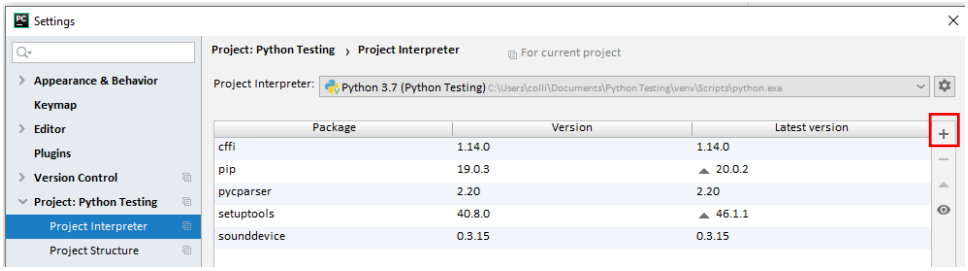
Step Four: Open PyCharm once downloaded and select Create New Project (screenshot credit: BeginnersBook).

Step 5: Now you will download all packages / modules / dependencies for the project. There are multiple methods to do this, including using the pip tool, or directly importing through PyCharm. We will be directly importing all packages in Python (because often permission and file paths can get messed up when using the pip tool).

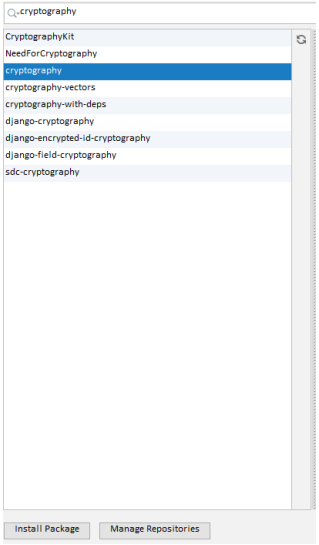
To install a package through PyCharm, navigate to File --> Settings (CTRL + ALT + S).

Under settings, navigate to Project: *Project Name*, and select Project Interpreter.

In the Project Interpreter, click the + icon to add a new module.



When you have clicked on the + icon, a new window will pop open named Available Packages. We can search for each module / package and install directly into our project. For example, to install the cryptography module, simply search “cryptography”, click the package which says cryptography, then click Install Package and wait for it to install.



Once package has been successfully installed, we can move onto the next module to install. For this project, install all of the following modules (name is exactly the name of the package)

- pywin32
- pynput
- scipy
- cryptography
- requests
- pillow
- sounddevice

Once you have imported all modules, exit out all of settings windows and wait a few minutes for each package to install.

You have successfully installed python, PyCharm, and all required modules.

CREATING FILES AND APPENDING TO FILES

For multiple parts of the keylogger, we will be appending data to files. Before we append data to files, we must first create variables with the proper extensions. Here are the variables you will need with the proper extensions.

```
system_information = "system.txt"
audio_information = "audio.wav"
clipboard_information = "clipboard.txt"
screenshot_information = "screenshot.png"
keys_information = "key_log.txt"
```

We will also need 3 addition files for encryption, I simply used the e_file_name syntax for each file.

```
system_information_e = 'e_system.txt'
clipboard_information_e = 'e_clipboard.txt'
keys_information_e = 'e_keys_logged.txt'
```

To open and append to files, use the with ***open(file_path, "a") as f***:

To write to the file, simply use the ***f.write(data)*** method

LOGGING KEYS

To log keys using python, we will be using the pynput module.

Module to install:

```
from pynput.keyboard import Key, Listener
```

Key Ideas with pynput:

- pynput has multiple functions including on_press, write_file, and on_release
- to understand pynput, follow this tutorial: https://www.youtube.com/watch?v=TbMKwl11itQ

EMAIL

To add an email functionality, we will be using the email module.

Modules to install:

```
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
import smtplib
```

Key Ideas with email:

- To send with email, follow this tutorial: https://www.geeksforgeeks.org/send-mail-attachment-gmail-account-using-python/?ref=lbp

COMPUTER INFORMATION

To gather computer information, we will use socket and platform modules.

Modules to install:

```
import socket
import platform
```

Key Ideas with socket:

- The ***hostname = socket.gethostname()*** method gets the hostname
- To get the internal IP address, use ***socket.gethostbyname(hostname)*** method

Key ideas with platform:

- To receive processor information, use the ***platform.processor()*** method
- To get the system and version information use ***platform.system()*** and ***platform.version()***
- To get the machine information, use the **platform.machine()** method

To get external (public facing) IP address, use api.ipify.org

- Use the ***get('https://api.ipify.org').text*** to get external ip

CLIPBOARD

To get the clipboard information, we will be using the win32clipboard module, which is a submodule of pywin32

Module to install:

```
import win32clipboard
```

Key ideas with win32clipboard:

- The person may not have any writeable data for the clipboard (could have copied an image), so make sure to use a try – except block just in case information could not be copied.
- To open clipboard, use the ***win32clipboard.OpenClipboard()***
- To get clipboard information, use the ***win32clipboard.GetClipboardData()***
- To close the clipboard, use the ***win32clipboard.CloseClipboard()***

MICROPHONE

To record with microphone, we will be using the sounddevice module and writing to a .wav file using the scipy.io.wavefile module.

Module to install:

```
from scipy.io.wavfile import write
import sounddevice as sd
```

Key ideas with sounddevice:

- Ensure to set the fs variable: ***fs = 44100***
- Ensure to add a seconds variable: ***seconds = microphone_time***

- To record, use the following code:
  myrecording = sd.rec(int(seconds * fs), samplerate=fs, channels=2)
  sd.wait()
- To write the recording to a .wav file, use the following code: ***write(filepath, fs, myrecording)***

SCREENSHOT

To take a screenshot, we will use the ImageGrab from the Pillow Module.

Modules to install:

```
from multiprocessing import Process, freeze_support
from PIL import ImageGrab
```

Key Ideas with ImageGrab:

- The ***ImageGrab.grab()*** method takes a screenshot
- To save the image, use the ***image_variable.save()*** method

To ensure only one screenshot is taken at a time, add ***freeze_support().*** Use the following code below:

```
if __name__ == "__main__":
    freeze_support()
    Process(target=screenshot).start()
```

BUILD THE TIMER

To build a timer which goes through a certain number of iterations before the keylogger ends, we will be using the timer function.

Use the following process:

1. Create an iterations variable and set its value to zero (***iterations = 0***)
2. Create an ***end_iterations*** variable which sets to a certain amount of iterations before ending the keylogger (***end_iterations = 5***)
3. Get the current time using the time.time() function, set this equal to a variable (***currentTime = time.time()***)
4. Create a ***time_iteration*** variable which collects the keylogs for a certain period of time in seconds (***time_iteration = 15***)
5. Get the stoppingTime by adding the time.time() function + time_iteration to stop, set this equal to a variable (***stoppingTime = time.time() + time_iteration***)
6. while iterations is less than (<) the ending_iterations…
    a. log keys
7. If the current time is greater than (>) the stopping time…
    a. Take a screenshot
    b. Send screenshot to email
    c. Gather clipboard contents
    d. Add 1 to iterations variable
    e. Get new current time
    f. Get new stopping time

ENCRYPTION OF FILES

To encrypt files, we will use the cryptography.fernet module.

Module to import:

```
from cryptography.fernet import Fernet
```

Key ideas to remember for encryption module:

- Use the following tutorial
    - How to Encrypt Strings and Files in Python: https://www.youtube.com/watch?v=H8t4DJ3Tdrg

EXEXCUTABLE

Converting our Python Keylogger to an executable can be a little tricky. Python isn't very good at converting scripts and programs into executables because there are often many dependencies (modules) which have to be downloaded. To turn this keylogger into an executable, I recommend using one of two programs… I have included some helpful tutorials which can help you convert your Python programs into executables.

- pyinstaller: https://www.youtube.com/watch?v=lOIJIk_maO4
- Auto PY to EXE: https://www.youtube.com/watch?v=OZSZHmWSOeM