

Python Network IO Monitor Tutorial

Contents

Python Network IO Monitor Tutorial	1
psutil	1
Tutorial 1 - Research	2
Tutorial 2 - Total Network Statistics.....	2
Tutorial 3 - Kilobits per Second	3
Tutorial 4 - Network IO Monitor.....	5
Tutorial 5 - Getting Rich with Network IO	7
Assignment Submission.....	8

Time required: 60 minutes

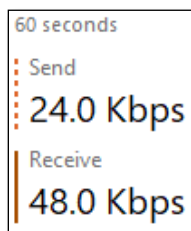
psutil

psutil (python system and process utilities) is a cross-platform library for retrieving information on running processes and system utilization (CPU, memory, disks, network, sensors) in Python. It is useful mainly for system monitoring, profiling, limiting process resources and the management of running processes.

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

We are going to use **psutil** to build a local device network IO monitor. This program will monitor the network input output on your local computer.

We want something like the Windows Task Manager → Performance tab.



The following tutorials show a development process from research to testing to final product.

Tutorial 1 - Research

We want to be able to measure network traffic on our local computer.

1. Google **python psutil**
2. Psutil documentation and example programs are among the results.
3. After a little reading through the documentation, we find this:
<https://psutil.readthedocs.io/en/latest/#network> We will start by getting network statistics.

Tutorial 2 - Total Network Statistics

This program will display the total network statistics since OS install.

Create a Python program named **network_io_monitor_1.py**

```
1  """
2      Name: network_io_monitor_1.py
3      Author:
4      Created:
5      Purpose: Network bytes sent and received since OS install
6  """
7  # https://pypi.org/project/psutil/
8  # pip install psutil
9  import psutil
10
11 # Get total local device network bytes
12 # sent and received since OS install
13 net_io = psutil.net_io_counters()
14
15 # Get the separate sent and and received statistics
16 # as properties of net_io_counters
17 sent = net_io.bytes_sent
18 recv = net_io.bytes_recv
19
20 # Display statistics
21 print(f" Sent: {sent:,.1f} Bytes")
22 print(f" Recv: {recv:,.1f} Bytes")
```

Example run:

```
Sent: 8,320,575,142.0 Bytes
Recv: 6,881,916,659.0 Bytes
```

Tutorial 3 - Kilobits per Second

Copy the last file and rename it as **network_io_monitor_2.py**

Keeping previous versions of programs is helpful if the changes you are making break the program.

- Add a loop to get new network io statistics every second using the Python sleep method
- Subtract the first readings from the second readings to get bytes per second
- Convert from bytes to bits to Kbps (kilobits per second)

```

1  """
2      Name: network_io_monitor_2.py
3      Author:
4      Created:
5      Purpose: Measure bytes per second from Network statistics
6      Convert bytes to bits to megabits per second
7  """
8  # pip install psutil
9  import psutil
10 import time
11
12 while True:
13     # Get total local device network bytes
14     # and received since OS install
15     net_io = psutil.net_io_counters()
16
17     # Get the separate sent and and received statistics
18     # as properties of net_io_counters
19     sent_1 = net_io.bytes_sent
20     recv_1 = net_io.bytes_recv
21
22     # Wait one second for the next reading
23     time.sleep(1)
24
25     # Get total local device network bytes sent and received
26     net_io = psutil.net_io_counters()
27
28     # Get the separate sent and and received statistics
29     # as properties of net_io_counters
30     sent_2 = net_io.bytes_sent
31     recv_2 = net_io.bytes_recv
32
33     # Subtract first reading from second reading
34     # gives us how many bytes were sent/recv per second
35     sent = sent_2 - sent_1
36     recv = recv_2 - recv_1
37
38     # Convert from bytes per second
39     # to megabits per second, Mbps
40     sent = sent / 125000
41     recv = recv / 125000
42
43     # Display statistics
44     print(f" Sent: {sent:,.2f} Mbps - Recv: {recv:,.2f} Mbps")

```

Example run:

```
Sent: 4.05 Mbps - Recv: 0.04 Mbps  
Sent: 3.45 Mbps - Recv: 0.05 Mbps  
Sent: 4.25 Mbps - Recv: 0.08 Mbps  
Sent: 1.27 Mbps - Recv: 0.06 Mbps  
Sent: 2.60 Mbps - Recv: 0.06 Mbps
```

Tutorial 4 - Network IO Monitor

- Add **end="\r"** to the print statement. This does not move to the next line, it overwrites the current line. This does not work in Idle.
- Add a nice program title.
- Add a try except to allow CTRL-C to exit the program nicely.

```

1  """
2      Name: network_io_monitor_3.py
3      Author:
4      Created:
5      Purpose: Monitor local device network IO
6  """
7
8  import time
9  import sys
10 # pip install psutil
11 import psutil
12
13
14 # TODO: Print program title
15 print(" +-----+")
16 print(" |      Python Network I/O Monitor      |")
17 print(" +-----+")
18
19 while True:
20     try:
21         # Get total local device network bytes
22         # sent and received since OS install
23         net_io = psutil.net_io_counters()
24
25         # Get the separate sent and and received statistics
26         # as properties of net_io_counters
27         sent_1 = net_io.bytes_sent
28         recv_1 = net_io.bytes_recv
29
30         # Wait one second for the next reading
31         time.sleep(1)
32
33         # Get total local device network bytes sent and received
34         net_io = psutil.net_io_counters()
35
36         # Get the separate sent and and received statistics
37         # as properties of net_io_counters
38         sent_2 = net_io.bytes_sent
39         recv_2 = net_io.bytes_recv
40
41         # Subtract first reading from second reading
42         # gives us how many bytes were sent/recv per second
43         sent = sent_2 - sent_1
44         recv = recv_2 - recv_1
45
46         # Convert from bytes per second
47         # to megabits per second, Mbps
48         sent = sent / 125000
49         recv = recv / 125000
50
51         # Add end="\r" to print statement to remove the return
52         # Each line overwrites itself on the console
53         # The extra spaces are to make sure any longer data is cleared out
54         print(
55             f" Sent: {sent:,.2f} Mbps - Recv: {recv:,.2f} Mbps"
56         )
57     except KeyboardInterrupt:
58         # Catch CTRL-C for clean exit
59         sys.exit(0)

```

Example run:

```
+-----+
|  ----  Python Network I/O Monitor  ----  |
+-----+
Sent: 10.0 Kbps - Recv: 15.9 Kpbs
```

Tutorial 5 - Getting Rich with Network IO

We added some rich formatting to this version. There are only a couple of changes.

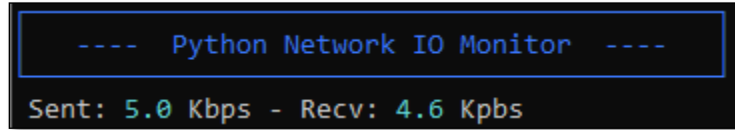
These changes replace the code before the while loop.

```
1  """
2      Name: network_io_monitor_4.py
3      Author:
4      Created:
5      Purpose: Monitor local device network IO
6  """
7
8  import time
9  import sys
10 # pip install psutil
11 import psutil
12 # Windows: pip install rich
13 # Import Console for console printing
14 from rich.console import Console
15 # Import Panel for title displays
16 from rich.panel import Panel
17 # Initialize rich.console
18 console = Console()
19
20 # TODO: Print program title
21 console.print(
22     Panel.fit(
23         "  ----  Python Network IO Monitor  ----  ",
24         style="bold blue")
25 )
```

Add **console** to the beginning of the output print line.

```
# Add end="\r" to print statement to remove the return
# Each line overwrites itself on the console
# The extra spaces are to make sure any longer data is cleared out
console.print(
    f" Sent: {sent:,.1f} Kbps - Recv: {recv:,.1f} Kpbs", end="\r")
```

Example run:

A screenshot of a terminal window with a black background. The text is displayed in a monospaced font. The first line is "---- Python Network IO Monitor ----" in blue. The second line is "Sent: 5.0 Kbps - Recv: 4.6 Kpbs" in green.

```
---- Python Network IO Monitor ----  
Sent: 5.0 Kbps - Recv: 4.6 Kpbs
```

Assignment Submission

1. Attach all program files.
2. Attach a screenshot of each successful program run.
3. Submit the assignment in Blackboard.