check for pc's pids names and usage stats and send them to a file to search later .py v2

```
import os
import psutil
import datetime

def check_high_memory_usage(threshold=50):
    high_memory_usage_processes = []
    total_memory = psutil.virtual_memory().total
    for proc in psutil.process_iter(['pid', 'name', 'memory_info']):
        try:
            memory_percent = (proc.info['memory_info'].rss / total_memory) * 100
            if memory_percent > threshold:
                high_memory_usage_processes.append((proc, memory_percent))
        except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):
            pass

    return high_memory_usage_processes

def write_processes_to_file():
    current_pid = os.getpid()
    now = datetime.datetime.now()
    file_name = f"processes_{now:%Y-%m-%d_%H-%M-%S}.txt"
    with open(file_name, 'w') as f:
        f.write(f"List of all running processes on {now}:\n\n")
```

```
        for proc in psutil.process_iter(['pid', 'name', 'memory_percent',
'cpu_percent']):

            try:

                if proc.info['pid'] != current_pid:  # Exclude the current script

                    cpu_percent = proc.info['cpu_percent']

                    mem_percent = proc.info['memory_percent']

                    f.write(f"PID: {proc.info['pid']} - Name: {proc.info['name']}
 - CPU%: {cpu_percent:.2f} - Memory%: {mem_percent:.2f}\n")

                    f.write(f"\tDisk usage: {psutil.disk_usage('/').percent:.2f}%\n
")

                    f.write(f"\tNetwork usage: {psutil.net_io_counters().bytes_sent
/1024:.2f}KB sent/{psutil.net_io_counters().bytes_recv/1024:.2f}KB received\n")

            except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess
):

                pass



        f.write("\n\n
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=\n\n")

        f.write(f"List of highest CPU usage processes on {now}:\n\n")

        for proc in sorted(psutil.process_iter(['pid', 'name', 'memory_percent',
'cpu_percent']), key=lambda p: p.info['cpu_percent'], reverse=True):

            try:

                if proc.info['pid'] != current_pid:  # Exclude the current script

                    cpu_percent = proc.info['cpu_percent']

                    if cpu_percent > 0.0:

                        f.write(f"PID: {proc.info['pid']} - Name: {proc.info['name'
]} - CPU%: {cpu_percent:.2f} - Memory%: {proc.info['memory_percent']:.2f}\n")

                        f.write(f"\tDisk usage: {psutil.disk_usage('/').percent:.2f
}%\n")

                        f.write(f"\tNetwork usage: {psutil.net_io_counters().
bytes_sent/1024:.2f}KB sent/{psutil.net_io_counters().bytes_recv/1024:.2f}
KB received\n")

            except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess
):

                pass
```

```
        f.write("\n\n
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=\n\n")

        f.write(f"List of highest memory usage processes on {now}:\n\n")

        for proc, mem_percent in sorted(check_high_memory_usage(), key=lambda p: p[
1], reverse=True):

            f.write(f"PID: {proc.info['pid']} - Name: {proc.info['name']}
 - Memory%: {mem_percent:.2f}\n")

            f.write(f"\tDisk usage: {psutil.disk_usage('/').percent:.2f}%\n")

            f.write(f"\tNetwork usage: {psutil.net_io_counters().bytes_sent/1024
:.2f}KB sent/{psutil.net_io_counters().bytes_recv/1024:.2f}KB received\n")



def main():

    write_processes_to_file()



if __name__ == '__main__':

    main()
```

Revision #3
Created 2023-04-28 08:02:18 UTC by naruzkurai
Updated 2023-04-28 11:33:49 UTC by naruzkurai