Hardware System Monitor

Jan Vyhnánek

May 2025

Contents

1 Introduction

This is documentation for the *HWSM* Python application. *HWSM* is a python script-based multiplatform universal open source application for professional use. The main scope of the application's functionalities is focused on system resource monitoring and maintenance. Originally this project began as an idea from need as there was very little good system monitor applications for Linux systems, especially those with highly-customizable desktop environments such as I3WM, which this all started on. Later on, this progressed to the point of semastral work for the Faculty of Nuclear Sciences on the CTU in Prague. These two factors together pushed this project from an idea into creation, and also explain why this documentation is released as an official CTU document.
It is vital to answer the question of the choice of Python. Indeed Python, as an interpreted programming language, is in no way optimal for such jobs. On the other hand Python is widely known, used and simple to debug and maintain. The primary aim is for sure on the GNU/Linux based system, where Python is nowadays always pre-installed with the kernel. Also Python applications are easy to carry around as they can run from a single script and will always run on any machine with Python installed. Therefore we don't have to worry about operating system, CPU architecture or instruction set. However this still doesn't tip the scales completely, so in the future there might be a rework into C.

2 Primary features

This section addresses the most important features of the back-end code of the application. There is still room for discussion whether the utilities and libraries used were the best choice, but for now the documentation concerns only the present libraries.

2.1 Matplotlib

Matplotlib is a widely-known library for Python that provides an interface for graphs figures. Despite being great for plotting graphs, it can also handle user interfaces fairly flawlessly. There should be a disclaimer before the continuation, and that is about the fact that Matplotlib was not designed to handle entire application GUI like this. *HWSM* is simply a clever construction based on Matplotlib that makes it possible, but there are many non-ideal features due to it.
Matplotlib is utilized in this project in basically every feature. It handles the entire GUI, including buttons, screen handling, and theme handling. Naturally, it also covers the plotting of every single graph and text in the application. More about screen and theme handling in Sections 3 and 5.

2.2 Multiprocessing

Multiprocessing is a library that provides an easy way to handle multithreading using Python. In this application, it is used as a way to monitor the usage of RAM and CPU without disrupting the GUI or handlers in any way. More about resource monitoring in Section 4.1. In short, the data are being read every 200 milliseconds, and this process would temporarily interrupt any process posted by the GUI or any handler of the application. This would cause an unstable and sluggish program that would lack usability. Therefore, multiprocessing is the way to maximize the speed and usability of the application by maintaining separate threads for these individual processes.

2.3 Psutils

Psutils is a library for Python using which you can easily get a vast multitude of system information. For us, most notable intel will be RAM, CPU and disk statistics. Nearly all measurements in the application are realized via this library and, therefore, it is a vital part in its core.

2.4 Other modules

3 Screen handling

HWSM is sectioned into so-called screens. Essentially, they are just logically partitioned segments that use their own elements and are contorted by calling a function for the given screen. The basic algorithm for each screen is always the same for modularity reasons. You can see this "blank" algorithm in picture 1.

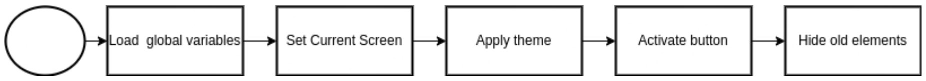

Figure 1: General screen algorithm

4 Subdirectories

4.1 Resource Monitor

4.2 Process Monitor

4.3 Disk Monitor

4.4 GPU monitor

4.5 Statistics

4.6 Hardware

4.7 Settings

5 Theme handling

6 Results