

Cross Platform and Configurable Performance Monitoring Widget Library (qw-performance-monitoring)

DEVELOPED BY: BOB K DANANI – BOBDANANI@GRAZIACOMPUTING.COM

Current version: 0.5 (beta)

Latest Updated: November 7th, 2011

Table of Contents

| | |
|--|----|
| COVER PAGE | 1 |
| TABLE OF CONTENTS | 2 |
| INTRODUCTION (BACKGROUND, TECHNOLOGY DEPENDENCY) | 3 |
| SUPPORTED FEATURES | 4 |
| TUTORIALS AND SCREENSHOTS | 7 |
| SUPPORTS, BUG REPORTS, CONTACT | 11 |

I. Introduction

Background

The purpose of this project is to build a configurable user interface widget library, which can be used to monitor the performance of any general technical application. Many developers often want to monitor the performance rate of an application, a process, or an operation, and display them in a nice looking graph.

Some existing implementations exist as a solution of this problem. Qwt is a good candidate for this. The Qwt library enables developers to visualize the performance of some processes or applications in nice looking graph plots. However many of us would probably want to have a dynamic moving performance manager widget, something similar to the performance graph in Windows Task Manager. I found a good performance chart library, which looks similar to the Windows Task Manager's performance graph; it is called C2DPushGraph. I should say that this library is good; however it is highly dependent on Microsoft Foundation Classes (MFC); a library that wraps portions of the Windows API in C++ classes, including the default Windows Application Framework. This would be useful for applications developed for Windows platform. However, how if we want to develop applications that are intended to work under multi platforms environment?

This motivated me to create my own implementation of performance manager widget, which is built based on Qt libraries, one of the most widely-used User Interface (UI) library that is available on market today. Qt is a cross-platform user interface framework with API for C++ programming, which is used by various industries around the world, both business industries and research industries. This performance widget library is called **qw-performance-monitoring**, and currently it is distributed as a *static library*, under the MIT license. Header files, libraries, and example codes are included in the code distribution.

Technology used for the development of this library, and several dependencies for compilation and linking:

This Performance Widget library is built to work for C++. All the development works were done using the Qt Creator SDK, which is provided by Qt. The following dependencies need to be installed to be able to use the qw-performance-monitoring library:

- C++ compiler (latest version of gcc is recommended).
- Qt4 library (version 4.7 or above is recommended). You need to link the QtCore, QtOpenGL, and QtGui libraries to be able to use this QWPerformance Monitoring library in your existing projects.

II. Supported Features

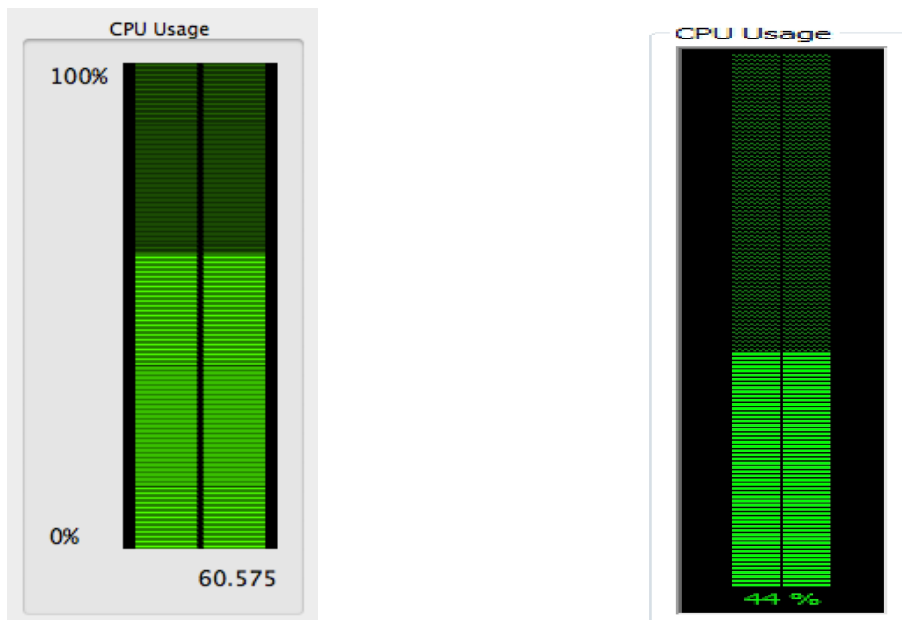
The library has been made almost fully configurable, which means that users can easily modify the widget parameters and have control on how they want to display this widget. Some of the parameters that have been made configurable include but not limited to:

- Widget size, grid width and grid height
- Ability to add multiple data groups to represent multiple processes
- Statistical data: mean, highest peak, and lowest peak performance value
- Number of timeframe per grid
- Text, color, and visibility of all the labels
- Vertical grid behavior (moving or unmoving grid)
- Grid color, line color, background color, performance bar color
- Line thickness for each data group representation
- User-friendly legend (users can change the legend headers, as well as setting up the visibility for each data group line, and its corresponding properties).
- Flexibility to update the frequency duration (how frequent should the chart refresh the display).
- Option to use anti-aliasing (The effect is very minimum though, due to the nature of this project that the rendering is not that complicated).

Currently there are 3 main widgets available in this qw-performance-monitoring library: ResourceUsageWidget, ResourceHistoryWidget, and ResourceHistoryLegendWidget.

1. ResourceUsageWidget

This widget is similar as the CPU Usage and Memory Usage widget that we see on Windows Task Manager.



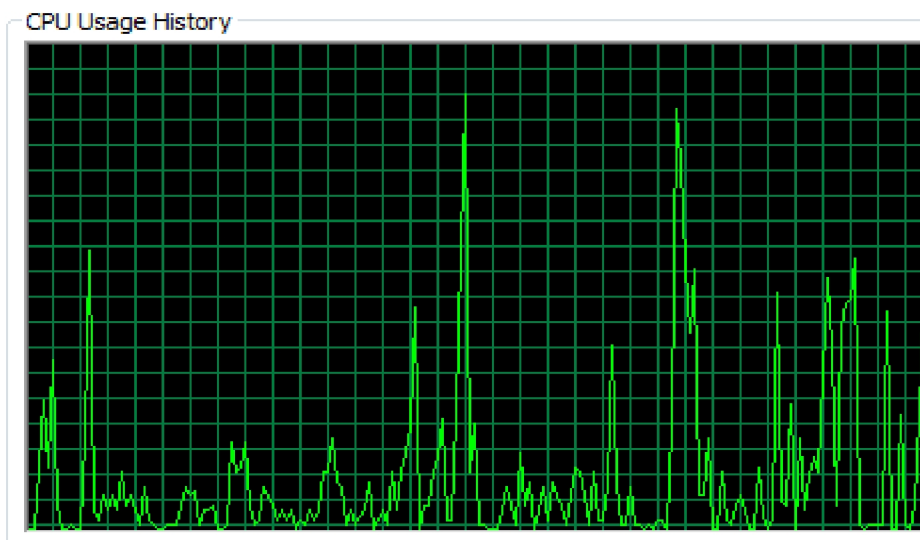
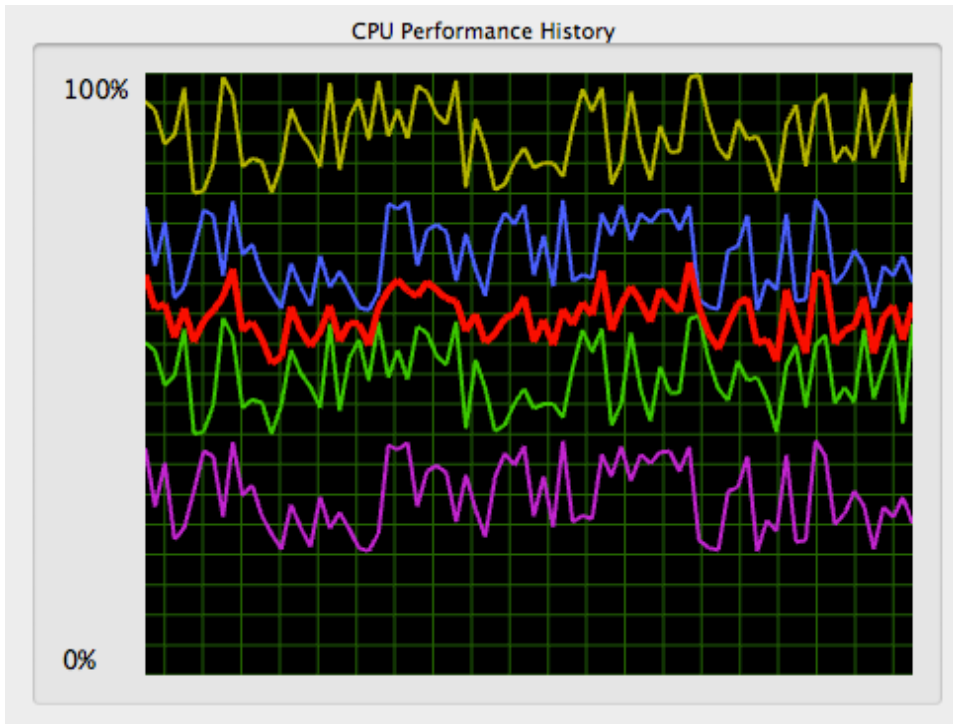
The one on the left represents the ResourceUsageWidget object from this qw-performance-monitoring library. The one on the right represents the Windows Task Manager version.

Configurable Parameters:

Labels (minimum label, maximum label, title label), widget size, grid color, performance bar color, text color, grid density, visibility of performance value, option for anti-aliasing, and the update time frequency.

2. ResourceHistoryWidget

This widget is similar as the CPU Usage History display that we see on Windows Task Manager, and this one gives us the ability to add multiple line groups (which represents the performance of multiple processes).



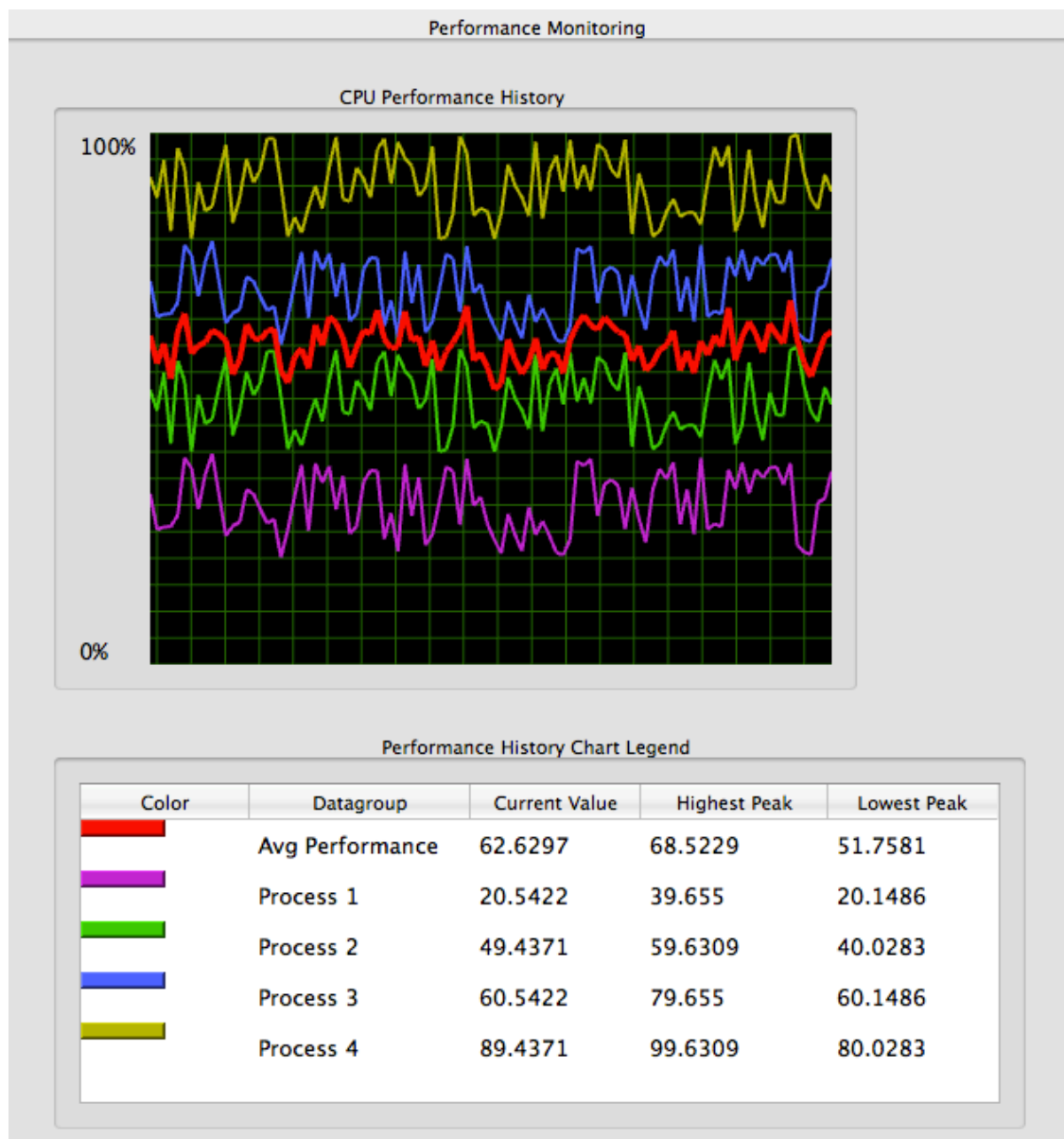
The one on the top represents the ResourceHistoryWidget object from this qw-performance-monitoring library. The one on the bottom represents the Windows Task Manager version.

Configurable Parameters:

Labels (minimum label, maximum label, title label), widget size, grid color, background color, line color, grid size (grid width and grid height), timeframe per grid, ability to add multiple line groups, line width, visibility of the grid, option to have a dynamic moving grid or static (un-moving) grid, option to show / hide statistics (mean line), option for anti-aliasing, and the update time frequency.

3. ResourceHistoryLegendWidget

This widget is to be used in complementary with the ResourceHistoryWidget. This widget offers a user-friendly table-view display, showing the legend information for each line group represented in the ResourceHistoryWidget. This legend shows the line color, labels, current value, highest peak value, and lowest peak value for each of the line data-group represented in ResourceHistoryWidget.



Configurable parameters:

Legend Widget title, table headers, option to show / hide a particular legend information (color, data-group, current value, highest peak, lowest peak), option to show / hide line data-groups, and widget size.

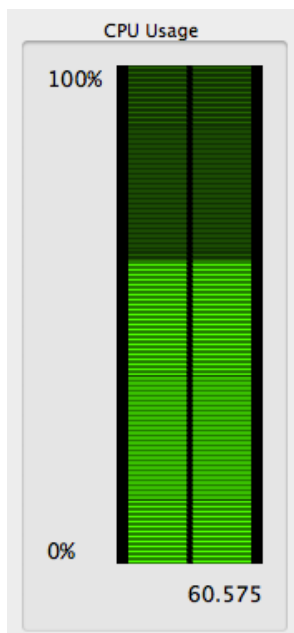
III. Tutorials and Screenshots

I believe that good class design is a very important aspect in software development. All the classes in this qw-performance-monitoring library are fully object oriented. Most of the implementation details are encapsulated in functions, which means that user will be able to use these widgets by just inserting few lines of codes for initializing and configuring the widget parameters.

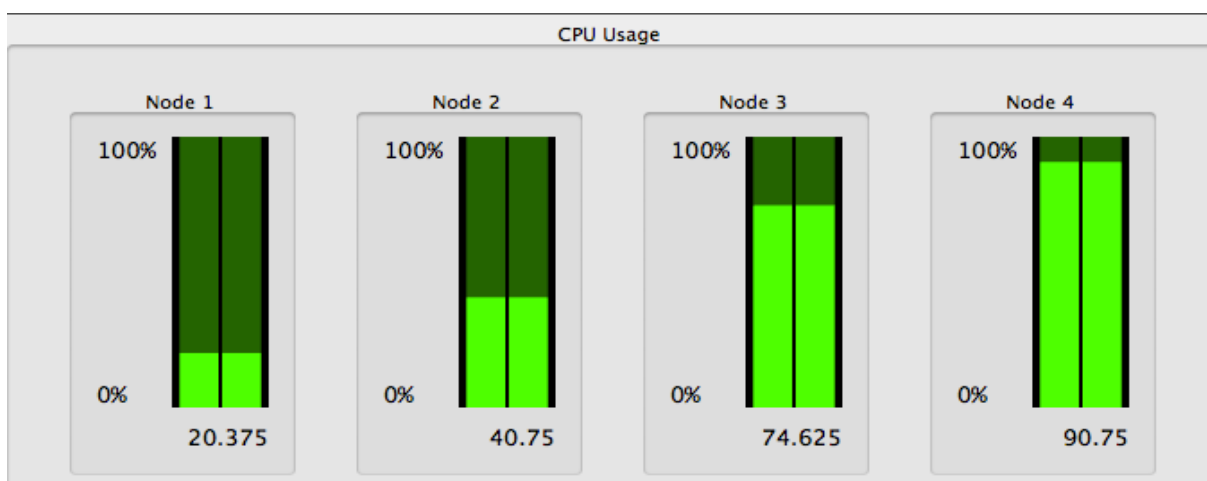
Some example programs have been given to ease users in using this library. Please see the “examples” folder in the project repository. All of these examples have been built using the QtCreator IDE. The output executable for each example is available inside the “build” folder in the corresponding “example” folder. For instance, you can find the “example1” project in: “examples/example1”, and you can find the *build output* for the “example1” project in: “examples/example1/build”.

The following are the screenshots for each example program:

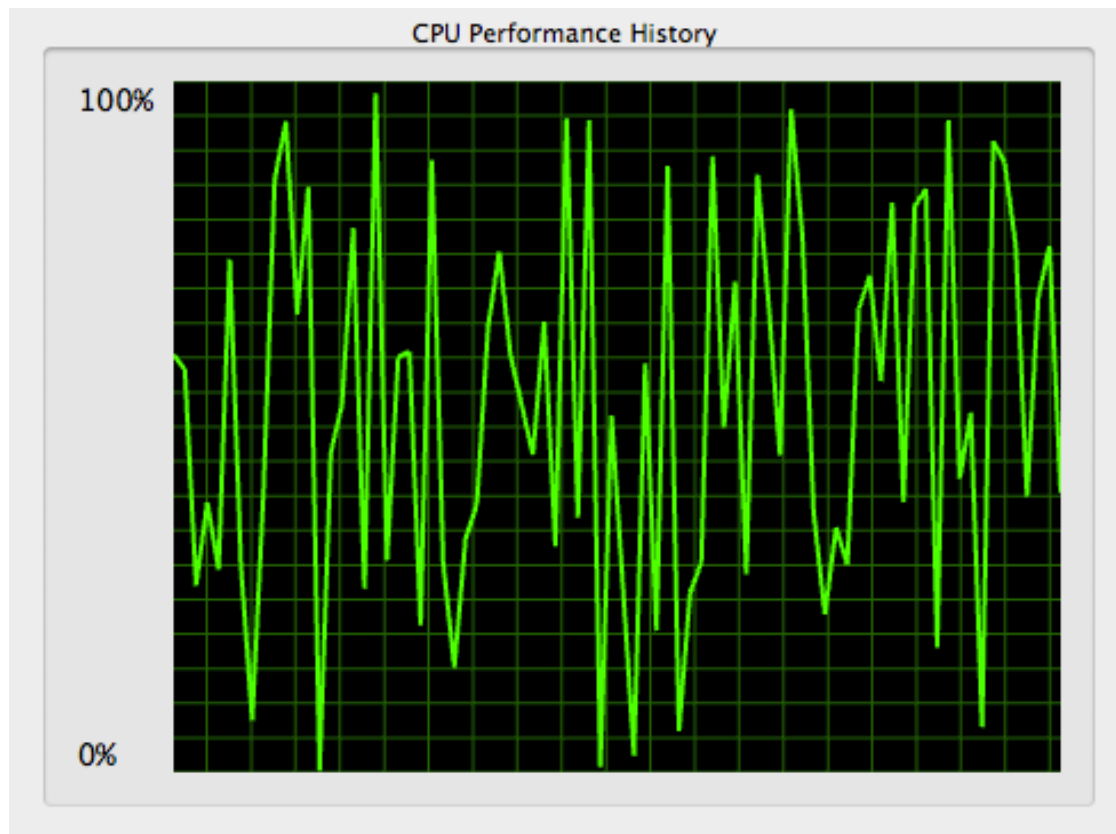
Example1



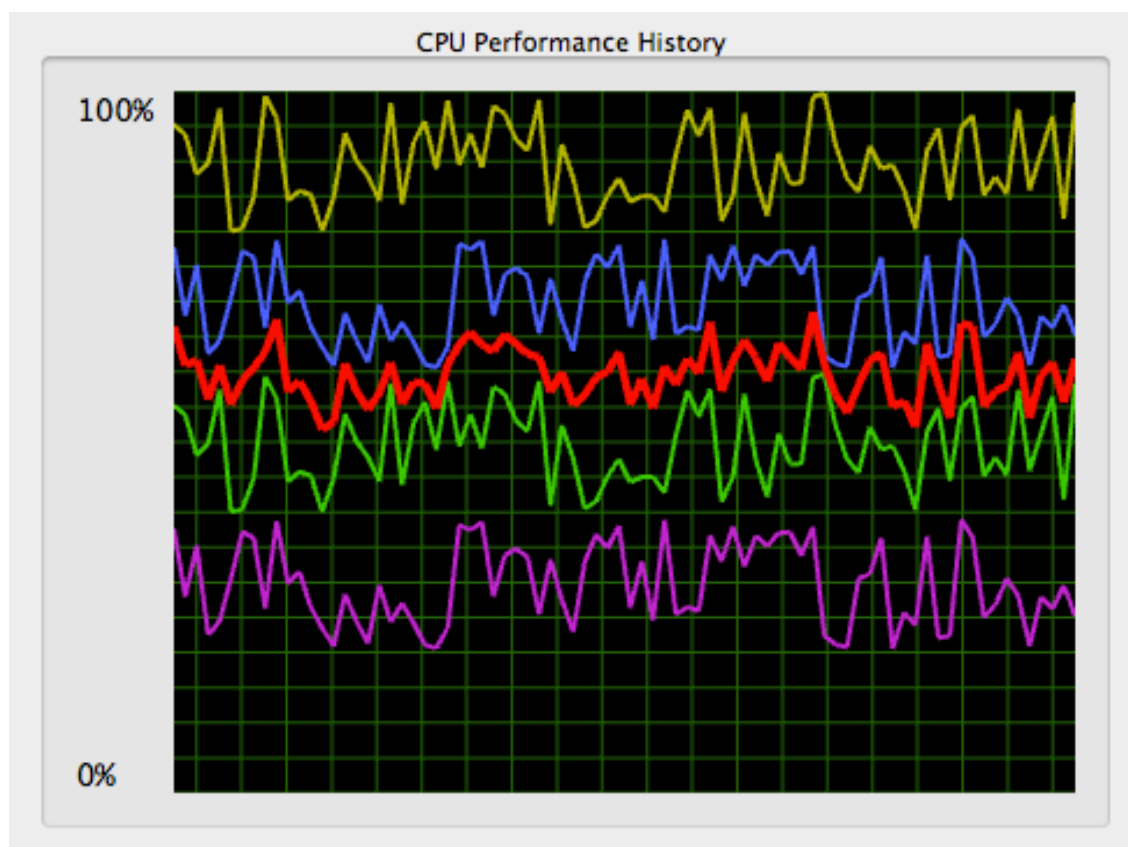
Example2



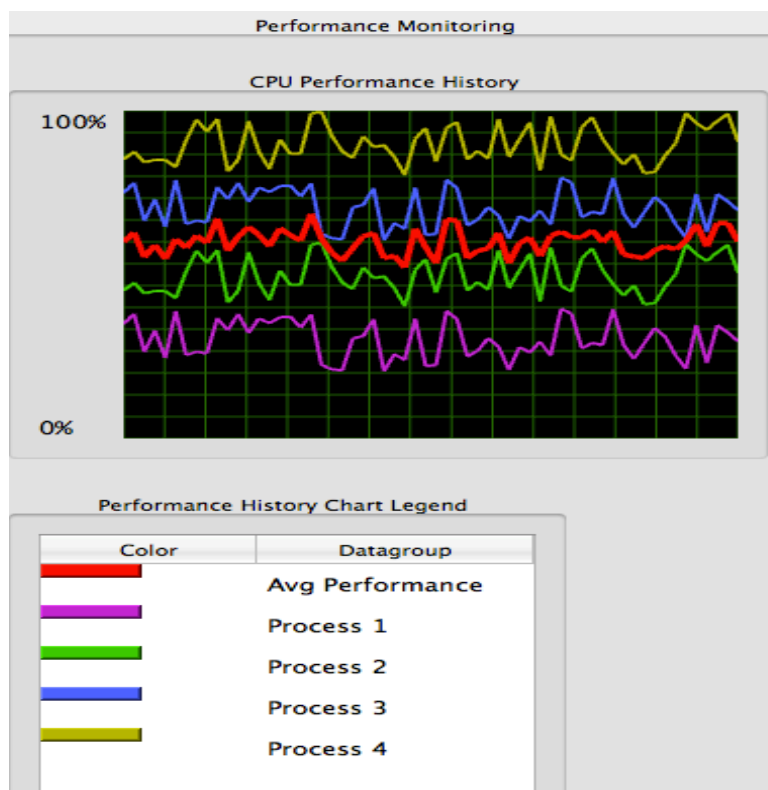
Example3



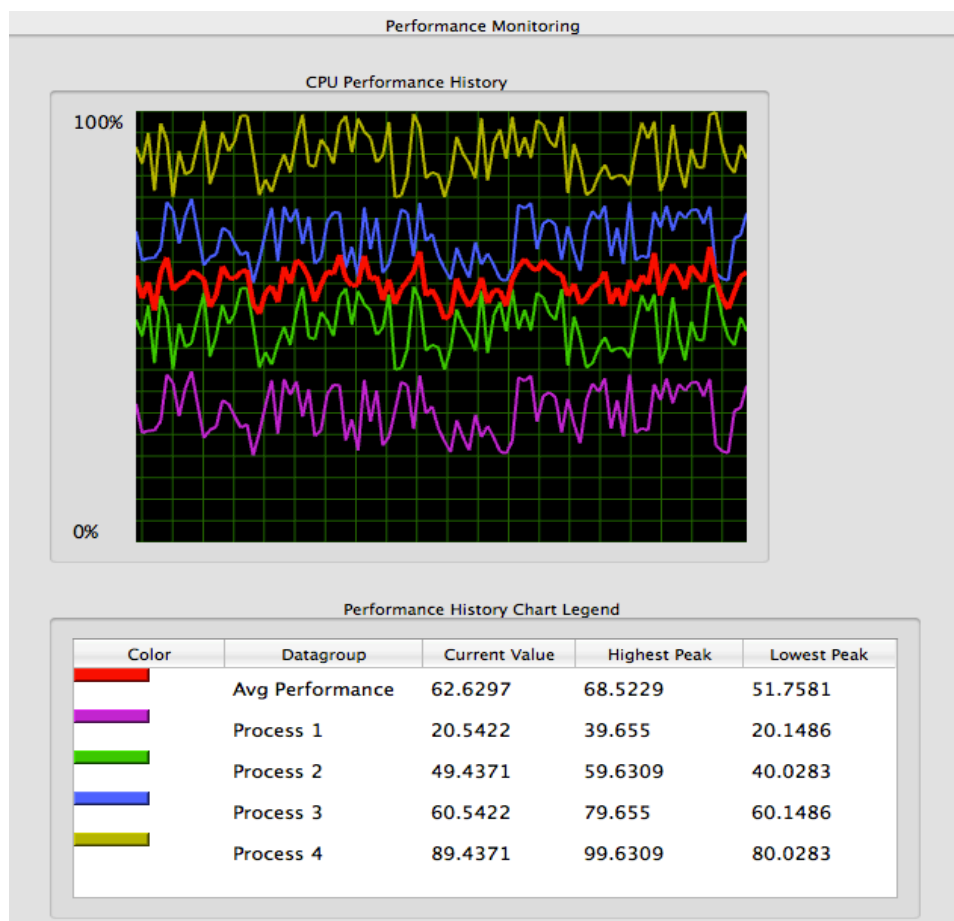
Example4



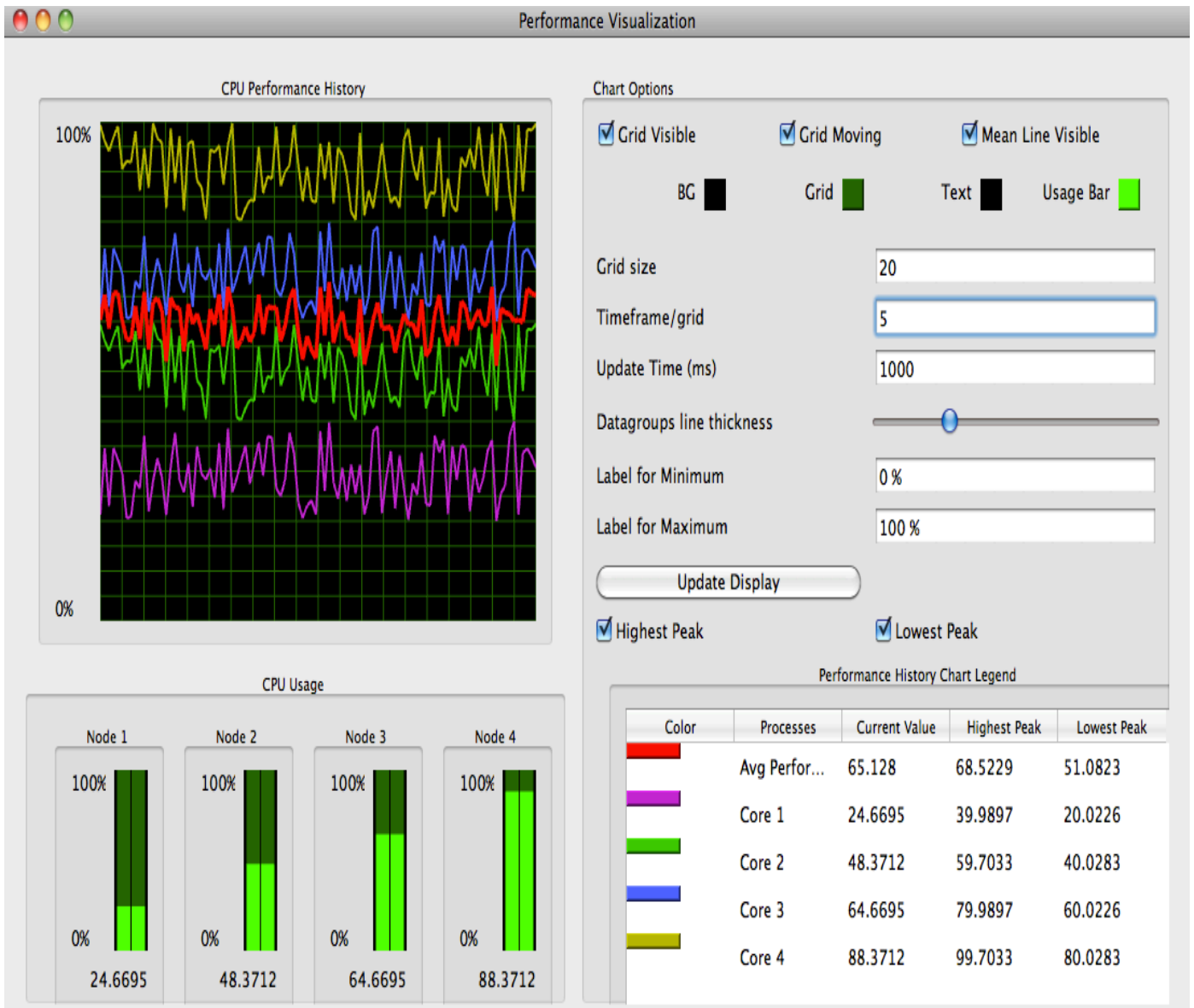
Example5



Example6



Example7



IV. Supports, Bug Reports, Contact

This library has been tested to work on Mac and Linux. Currently I am testing it on Windows platform to make sure that every feature also works on Windows. Once all the testing is completed, a stable release edition of this library will be uploaded in the project homepage. If you require any kind of support, or if you want to submit a bug report, please feel free to contact me.

Developer Contact Details:

Bob K Danani

Email: bobdanani@graziacomputing.com

Homepage: www.bobdanani.net

Project URL: <http://code.google.com/p/qw-performance-monitoring>