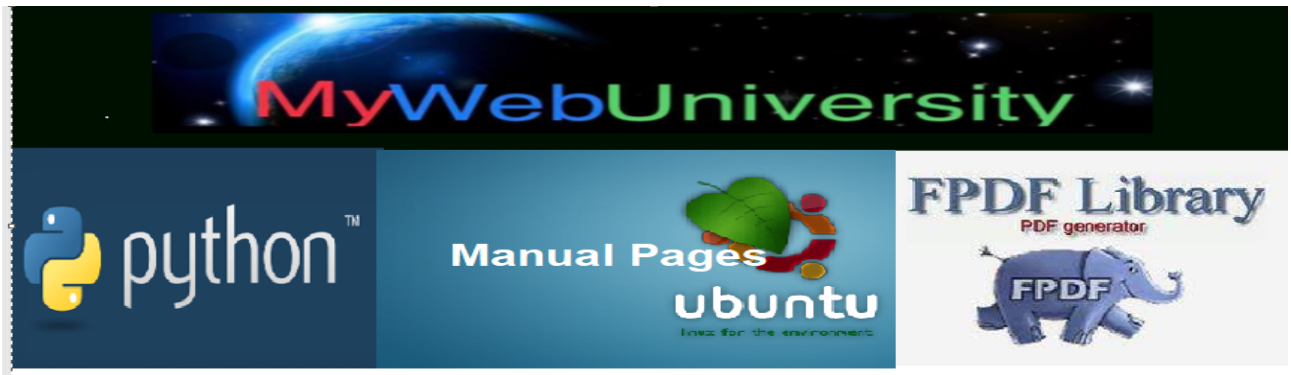

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'gprofng-display-text.1'

$ man gprofng-display-text.1

GPROFNG-DISPLAY-TEXT(1) User Commands GPROFNG-DISPLAY-TEXT(1)

NAME

gprofng-display-text - Display the performance data in plain text format

SYNOPSIS

gprofng display text [option(s)] [commands] [-script script-file] experiment(s)

DESCRIPTION

Print a plain text version of the various displays supported by gprofng.

The input consists of one or more experiment directories. Through commands, the user controls the output.

There is a rich set of commands to control the display of the data. The NOTES section lists the most common ones. The gprofng user guide lists all the commands supported.

Commands specified on the command line need to be prepended with the dash ('-') symbol.

In this example, a function overview will be shown, followed by the source code listing of function my-func, annotated with the performance metrics that have been recorded during the data collection and stored in experiment directory my-exp.er:

$ gprofng display text -functions -source my-func my-exp.er

Instead of, or in addition to, specifying these commands on the command line, commands may also be included in a file called the script-file.

Note that the commands are processed and interpreted from left to right, so the order matters.

If this tool is invoked without options, commands, or a script file, it starts in interpreter mode. The user can then issue the commands interactively. The session is terminated with the exit command in the interpreter.

OPTIONS

--version

Print the version number and exit.

--help

Print usage information and exit.

-script script-file

Execute the commands stored in the script file. This feature may be combined with commands specified at the command line.

NOTES

Many commands are supported. Below, the more common ones are listed in mostly alphabetical order, because sometimes it is more logical to swap the order of two entries.

There are many more commands. These are documented in the user guide.

"callers-callees"

In a callers-callees panel, it is shown which function(s) call the target function (the callers) and what functions it is calling (the callees). This command prints the callers-callees panel for each of the functions, in the order specified by the function sort metric.

"calltree"

Display the dynamic call graph from the experiment, showing the hierarchical metrics at each level.

"compare {on | off | delta | ratio}"

By default, the results for multiple experiments are aggregated. This command changes this to enable the comparison of experiments for certain views (e.g. the function view). The first experiment specified is defined to be the reference. The following options are supported:

"on"

For each experiment specified on the command line, print the values for the metrics that have been activated for the experiment.

"off"

Disable the comparison of experiments. This is the default.

"delta"

Print the values for the reference experiment. The results for the other experiments are shown as a delta relative to the reference (current-reference).

"ratio"

Print the values for the reference experiment. The results for the other experiments are shown as a ratio relative to the reference (current/reference).

"disasm function-name"

List the source code and instructions for the function specified. The instructions are annotated with the metrics used.

"fsingle function-name [n]"

Write a summary panel for the specified function. The optional parameter n is needed for those cases where several functions have the same name.

"fsummary"

Write a summary panel for each function in the function list.

"functions"

Display a list of all functions executed. For each function the used metrics (e.g. the CPU time) are shown.

"header"

Shows several operational characteristics of the experiment(s) specified on the command line.

"limit n"

Limit the output to n lines.

"lines"

Write a list of source lines and their metrics, ordered by the current sort metric.

"metric_list"

Display the currently selected metrics in the function view and a list of all the metrics available for the target experiment(s).

"metrics metric-spec"

Define the metrics to be displayed in the function and callers-callees overviews.

The metric-spec can either be the keyword default to restore the default metrics selection, or a colon separated list with metrics.

A special metric is "hwc". It automatically expands to the active set of hardware event counters used in the experiment(s).

If both instructions and clock cycles have been measured, the "CPI" and "IPC" metrics can be used to see the Clockcycles Per Instruction and Instructions Per Clockcyle values, respectively.

The gprofng user guide has more details how to define metrics.

"name {short | long | mangled}[:{soname | nosoname}]"

Specify whether to use the short, long, or mangled form of function names. Optionally, the load object that the function is part of can be included in the output by adding the soname keyword. It can also be ommitted (nosoname), which is the default.

Whether there is an actual difference between these types of names depends on the language.

Note that there should be no (white)space to the left and right of the colon (:).

This option should not be confused with the keyword name in a metric definition, which is used to specify that the names of functions should be shown in the function overview.

"overview"

Shows a summary of the recorded performance data for the experiment(s) specified on the command line.

"pcs"

Write a list of program counters (PCs) and their metrics, ordered by the current sort metric.

"sort metric-spec"

Sort the function list on the metric-spec given.

@IndexSubentry{Sort, Reverse order} The data can be sorted in reverse order by prepending the metric definition with a minus (-) sign.

For example sort -e.totalcpu.

@IndexSubentry{Sort, Reset to default} A default metric for the sort operation has been defined and since this is a persistent command, this default can be restored with "default" as the key (sort default).

"source function-name"

List the source code for the function specified, annotated with the metrics used.

"viewmode {user | expert | machine}"

This command is only relevant for Java programs. For all other languages supported, the viewmode setting has no effect.

The following options are supported:

"user"

Show the Java call stacks for Java threads, but do not show housekeeping threads.

The function view includes a function called <JVM-System>. This represents the

aggregated time from non-Java threads. In case the JVM software does not report a Java call stack, time is reported against the function <no Java callstack recorded>.

"expert"

Show the Java call stacks for Java threads when the user Java code is executed, and machine call stacks when JVM code is executed, or when the JVM software does not report a Java call stack. Show the machine call stacks for housekeeping threads.

"machine"

Show the actual native call stacks for all threads. This is the view mode for C, C++, and Fortran.

SEE ALSO

gprofng(1), gprofng-archive(1), gprofng-collect-app(1), gprofng-display-html(1), gprofng-display-src(1)

The user guide for gprofng is maintained as a Texinfo manual. If the info and gprofng programs are correctly installed, the command info gprofng should give access to this document.

COPYRIGHT