profiling

Line by line profiling and code coverage for **GAP**

1.1.0

01/11/2016

Christopher Jefferson

Christopher Jefferson

Email: caj21@st-andrews.ac.uk
Homepage: http://caj.host.cs.st-andrews.ac.uk/
Address: St Andrews
      Scotland
      UK

Contents

Chapter 1

profiling automatic generated documentation

1.1 profiling automatic generated documentation of global functions

1.1.1 ReadLineByLineProfile

▷ ReadLineByLineProfile(*filename*)   (function)

Returns:

Read *filename*, a line by line profile which was previously generated by GAP, using ProfileLineByLine or CoverageLineByLine functions from core GAP. A parsed profile can be transformed into a human-readable form using either OutputAnnotatedCodeCoverageFiles (1.1.5) or OutputFlameGraph (1.1.3)

1.1.2 MergeLineByLineProfiles

▷ MergeLineByLineProfiles(*filenames*)   (function)

Returns:

Read *filenames*, a list of line by line profiles which were previously generated by GAP, using ProfileLineByLine or CoverageLineByLine functions from core GAP. The elements of *filenames* can be either filenames, or files previously parsed by ReadLineByLineProfile (1.1.1).

1.1.3 OutputFlameGraph

▷ OutputFlameGraph(*codecover[, filename]*)   (function)

Returns:

Generate an 'svg' file which represents a 'flame graph', a method of visualising where time is spent by a program.

codecover should either be a profile previously read by ReadLineByLineProfile, or the filename of a profile.

The flame graph input will be written to *filename* (or returned as a string if *filename* is not present).

1.1.4 OutputFlameGraphInput

▷ OutputFlameGraphInput(*codecover[, filename]*)   (function)

    Returns:

Generate the input required to draw a 'flame graph', a method of visualising where time is spent by a program. One program for drawing flame graphs using this output can be found at https://github.com/brendangregg/FlameGraph

codecover should either be a profile previously read by ReadLineByLineProfile, or the filename of a profile.

The flame graph input will be written to *filename* (or returned as a string if *filename* is not present).

1.1.5 OutputAnnotatedCodeCoverageFiles

▷ OutputAnnotatedCodeCoverageFiles(*codecover[, indir], outdir*)   (function)

    Returns:

Takes a previously generated profile and outputs HTML which shows which lines of code were executed, and (if this was originally recorded) how long was spent executing these lines of code.

codecover should either be a profile previously read by ReadLineByLineProfile, or the filename of a profile which will first be read with ReadLineByLineProfile.

Files will be written to the directory *outdir*.

The optional second argument gives a filter, only information about filenames starting with *indir* will be outputted.

1.1.6 OutputJsonCoverage

▷ OutputJsonCoverage(*cover, outfile*)   (function)

    Returns:

Takes a previously generated profile and outputs a json coverage file which is amongst other things accepted by codecov.io.

codecover should either be a profile previously read by ReadLineByLineProfile, or the filename of a profile which will first be read with ReadLineByLineProfile.

The output will be written to the file *outfile*.

1.1.7 LineByLineProfileFunction

▷ LineByLineProfileFunction(*function, arguments*)   (function)

    Returns:

Calls *function* with the list of arguments *arguments*, and opens a time profile of the resulting call in the default web browser.

Index