Package 'WebAnalytics'

October 4, 2023

Type Package

Title Web Server Log Analysis

Version 0.9.12

Date 2023-10-04

Author Greg Hunt [aut, cre, cph]

Maintainer Greg Hunt <greg@firmansyah.com>

Description Provides Apache and IIS log analytics for transaction performance, client populations and workload definitions.

License GPL-3

Depends R (>= 4.0), utils, ggplot2 (>= 3.3.5), xtable (>= 1.8.4), data.table(>= 1.14.2), scales (>= 1.1.1)

Imports brew (>= 1.0-6), fs (>= 1.5.2), reshape2 (>= 1.4.4), digest (>= 0.6.29), tinytex (>= 0.37), uaparserjs (>= 0.3.5)

Suggests whoami (>= 1.3.0), testthat (>= 3.1)

URL https://github.com/gregfrog/WebAnalytics

NeedsCompilation no

Repository CRAN

Date/Publication 2023-10-04 12:00:02 UTC

R topics documented:

`WebAnalytics-package` *Tools for web server log performance reporting*

Description

The WebAnalytics package is a simple, low-impact way of getting detailed insights into the performance of a web application and of identifying opportunities for remediation. It generates detailed analytical reports on application response time from web server logs.

The objective of the package is to extract the maximum value from web server log data and to use that information to identify problems and potential areas for remediation. It enables you to easily read web server log files; generate histograms, scatter plots and tabular reports of response times, overall and per URL; to generate some diagnostic plots; and to generate a LaTeX document that can then be formatted as a PDF. The package supplies scripts and templates to do that document generation.

Details

Package:	WebAnalytics
Type:	Package
Date:	2023-10-04
License:	GPL 3

This code was used for many years in a performance consulting/toubleshooting context, dealing with systems that were not set up with comprehensive monitoring infrastructure, and sometimes with systems that did have monitoring infrastructure but which did not generate useful measures (percentiles are difficult to calculate on stream data, sometimes too little data is retained for longitudinal analysis), or which rolled up performance metrics to mean values over long intervals, destroying the short term information in the logs. For some systems the diagnostic plots were interesting by themselves.

It is not a debugging tool, it indicates where problems are and where there are behaviours that are unexpected: the tables and histograms identifying multiple code paths that developers may not be aware of, the diagnostic plots indicating contention, the scatter plots indicating short term variations in response time that are indicative of some kind of problem. All these enable potential fixes to be worked on, and once those fixes are developed, enabling direct measurement of the impact using the baselining graphs and tables.

A sample PDF report can be generated in the current directory, with work files saved in under the R tempdir using the following code fragment:

```
library(WebAnalytics)
filesDir = paste0(tempdir(),"/ex")
configVariableSet("config.workdir", filesDir)
workingDirectoryPopulate(".")
pdfGenerate()
```

The generated report provides the following:

Response Time Overview

- Detailed Response Time Percentiles
- Response Time Change over baseline workload (if a baseline log is supplied and the baseline is read)
- Request/Response Size Percentile Breakdown
- Response Times by Time - Scatter Plot
- Response Time Histogram
- Request Status by Hour
- Top Transactions by 95th percentile response time
- Top Transactions by aggregate response time
- Top Transactions by error rate

This section addresses questions such as

- How many static, dynamic and monitoring requests are there in the logs?
- How much of total system processing time is accounted for by static, dynamic and monitoring requests?
- How much static, dynamic and monitoring data transfer is there?
- How many requests per hour are made and in what hours?
- What are the transactions with the highest 95th percentile response times?

- What are the transactions that account for the most aggregate wait time in the system?

The 95th percentile and aggregate wait time tables are useful to identify those tramnsactions that could repay some performance optimisation. Anything high in both lists is worth investigating.

Transaction Data for each URL

- response time percentiles
- response time scatter plot by time of day
- response time histogram
- error rate by hour
- and variances over a baseline dataset (useful for comparing before and after release performance)

This addresses questions such as

- What is the clock time distribution of requests and response times for a URL?
- How many distinct groups of response times are there for a URL?
- How have these metrics changed relative to a baseline set of log data?

Browser Mix Percentages

- Browser family percentiles
- Browser family and version percentiles

These percentages are useful for identifying which browsers and versions need to be tested.

Diagnostic Charts

- 95th percentile response time by request rate
- Dynamic Content Response time by degree of request concurrency
- Static Content Redirect time by degree of request concurrency
- Static Content (successful requests) time by request concurrency
- Static Content (successful requests) time by outbound data rate

These plots mostly adddress the scalability of the system.

Percentile Comparison of transaction mix with baseline reporting period

- Input Data stats
- Transaction Counts and percentages by URL
- Transaction Waits and percentages by URL

These are primarily used for callibrating test workloads to ensure that the transaction mix is similar to the production workload, or the planned workload.

Server and Session Analysis

- Server Request Counts
- Session Request Counts
- Unique Sessions by Hour

A function workingDirectoryPopulate is provided to populate a working directory with all needed supporting files and a sample R report file which can be edited as needed. The working directory contains:

- sampleRfile.R - sample report template
- sample.config - configuration file for the report
- logo.eps - a 2cm by 2cm logo graphic (a placeholder) in EPS format
- makerpt.ps1 - PowerShell script to run the report and process the output with xelatex
- makerpt.sh - bash script to run the report and process the output with xelatex
- WebAnalytics.cls - the report LaTeX class

An R function, workingDirectoryPopulate will place copies of all necessary files in a directory, already configured to generate a sample PDF report from test data supplied with the package.

The supplied configuration file sample.config read by the report script provides enough flexibility for most purposes. Switches are provided to turn on or off different sections of the report. Edit the config file, sample.config to update the list of column names and data types (documented in logFileRead or use the IIS log utility function logFileFieldsGetIIS. The directory structure that it assumes is that there is a data directory identified in config.current.dataDir with multiple log directories under it (config.current.dirNames). This applies to both current data and the baseline log. The default behaviour of the script is to read the lexically last file name with a .log extension from each log directory and it checks that the log names are the same in each directory. This is consistent with a structure in which logs are regularly copied into a log directory for processing or where some pre-processing is required, for example where the log is being written with a varying number of fields as a result of sme other configuration by network or admin teams. Additional functions are provided to select all or some files: logFileNamesGet, logFileNamesGetAll, logFileNamesGetLast, and logFileNamesGetLastMatching and these can be substituted in the report template as needed.

There are multiple ways to run xelatex on the generated template. A bash script and a Powershell script are provided to do that if you have LaTeX already installed. Run the sample script and config file that are created in that directory using the command . ./makerpt.sh sample or powershell -f makerpt.ps1 sample to generate a sample PDF from the test data supplied as part of the package. If you do not have a LaTeX installation, The R package **tinytex** can be used to install LaTeX and a function pdfGenerate is provided in this package to do the PDF generation from within R.

The package uses the CRAN package **brew** to produce the LaTeX source from a Brew template and comes with its own LaTeX document class and a blank logo graphic, both of which can be tailored as needed.

The generated LaTeX document has been tested with xelatex and is known not to work with plain LaTeX because of font issues.

The package requires Apache or IIS log files to contain elapsed times in addition to timestamps, HTTP verbs, HTTP response codes and URLs. In Apache the elapsed time is provided by the %d or %D format specifier in a log format specification string. In IIS the time-taken field must be added to the log format. If supplied, the request and response sizes are also used by the report. For WebSphere applications, adding the JSESSIONID cookie to the log enables server-level session statistics (the server ID is parsed out of the WebSphere JSESSIONID cookie value, if the JSESSIONID cookie is not of the format serverID:sessionID the server distribution will be represented by a single server. To get session-level information without the cookie being present, it

might be possible to use the client IP (depending on the structure of the network), in which case, adding

```
b$jsessionid = b$userip
b$serverid = 1
```

to the `config.fix.data` function, in the sample configuration file, will provide some useful information.

The `config.fix.data` function is used to classify URLS as dynamic (the URL is retained), static or monitoring. The script depends on the literals that are used and the function must use those literals to identify Static and monitoring requests.

Author(s)

Maintainer: Greg Hunt <greg@firmansyah.com>

Examples

```
## Not run:
# find the *.log files in the directory
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

# get the columns from an IIS log
cols = logFileFieldsGetIIS(logFileName)

# read the log file as the current data
logdf = logFileListRead(logFileName,
          readFunction=logFileRead,
          columnList=cols)

# read a baseline data set
logbasedf = logFileListRead(logFileName,
          readFunction=logFileRead,
          columnList=cols)

# compare percentage counts and delays between
#   baseline and current, useful for load test callibration
plotWriteFilenameToLaTexFile(
  plotSaveGG(
    # convert elapsed time to seconds
    percentileBaselinePrint(logdf$elapsed/1000,
              logbasedf$elapsed/1000,
              columnNames = c("Delta", "Current", "Baseline", "Percentile"))
    , "xxx")
    )

## End(Not run)
```

`calculatePercentiles` *calculate quantile values from a column.*

Description

Calculate quantile values for a supplied numeric list. This is a wrapper around the R quantile function.

Usage

```
calculatePercentiles(column,
  percentileList=c(0.7, 0.8, 0.9, 0.95, 0.96, 0.97, 0.98, 0.99, 1))
```

Arguments

`column`	a vector of numeric values. The values will be rounded to two decimal places before calculation.
`percentileList`	a list of the quantile values that are to be calculated (as decimal values in the range 0 to 1)

Value

Returns a list of the quantile calculated by quantile.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
dataValues = c(1,1,1,2,3,4,5,6,7,8,9,10,10,10,10)
calculatePercentiles(dataValues, percentileList=c(0.5))
calculatePercentiles(dataValues)
```

`configFilesDirectoryNameGet` *get the path of the temporary directory used for storing work files*

Description

This directory may be an absolute or relative path and ends with a / character

Usage

```
configFilesDirectoryNameGet()
```

Value

Returns a string that is the path of the work directory

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
configFilesDirectoryNameGet()
```

`configVariablesLoad` *Read a configuration file and store the variables*

Description

These are functions to read, validate and execute a report configuration file placing its output as a series of variables in a hidden scope; they also access and check existence of the variables. Variables whose names begin 'config.' will be printed at load time along with their values.

Usage

```
configVariablesLoad(fileName="report.config")
configVariableGet(name)
configVariableIs(name)
configVariablesAll()
configVariableSet(name, value)
```

Arguments

`fileName`	The name of the configuration file
`name`	The name of a variable from the config file.
`value`	The value to be assigned to the config variable name.

Details

The configuration file is an R script that is intended to be used to define the variables and helper functions that control the supplied sample report script. The config file may be executed more than once as part of validating its content.

String values in the config file should be quoted using double quote characters. Lists of values are written using the R `c()` function, for example `c("a","b")`

Switches controlling behaviour

`config.projectName` The Project name for the cover page of the document. This is also printed on the internal page headers.

- Required

- No Default Value

`config.documentName` The name to be printed on the document cover page.

- Required
- No Default value

`config.current.dataDir` Data is assumed to be stored in a directory hierarchy, with a root directory and a series of child directories corresponding with the individual web servers, this variable specifies the root directory of that hierarchy. For example a directory structure might be a series of server specific logs under `/var/logs/apache`, with the individual servers' logs being located in `/var/logs/apache/server1`, `/var/logs/apache/server2` and `/var/logs/apache/server3`. the data dir in this case is `/var/logs/apache` with the `config.current.dirNames` being specified as `c("server1", "server2", "server3")`.

- Required
- No Default Value

`config.current.dirNames` This is the list of child directory paths, often corresponding with a list of server names, it must be a concatenation of strings, for example `c("PRODMAW1","PRODMAW2")`. Multiple directory names can be coded here: `c("PRODMAW1/app1","PRODMAW2/app1")`.

- Required
- No Default Value

`config.current.columnList` The list of column names in the log file, for example

`c("Apache", "elapsedus", "jsessionid")`.

See `logFileRead` for the list of valid column names. The name `Apache` is an abbreviation for the Apache common log format:

`c("userip", "ignored column 1", "username", "ApacheTimestamp", "url", "httpcode", "responsebytes")`

- Required
- No Default Value

`config.readBaseline` Read a baseline log to be compared with current behaviour. Valid values are either `TRUE` or `FALSE`.

- Optional
- Default value: `FALSE`

`config.baseline.dataDir` The root directory for the baseline data files

- Must be supplied if `config.readBaseline=TRUE`
- No Default Value

`config.baseline.dirNames` The list of baseline log server-specific directories, usually a list of server names, it must be a concatenation of strings, for example `c("PRODMAW1","PRODMAW2")`

- Must be specified if `config.readBaseline=TRUE`
- No Default Value,

`config.baseline.columnList` The list of columns to be read. It is the baseline version of `config.current.columnList`

- Must be specified if `config.readBaseline=TRUE`
- No Default Value

`config.generateGraphForTimeOver` Response time graphs and histograms are generated for URLs whose maximum elapsed time exceeds this number of milliseconds

- Optional
- Default Value: 10000

`config.generateServerSessionStats` Generate histograms and counts of requests by server. These are only generated if `jsessionid` is also one of the column names in the current data

- Optional
- Default Value: `TRUE`

`config.generatePercentileRankings` Generate tables that compare frequencies and total elapsed times of URLs in the baseline and current data. Intended to be used for calibrating performance test workloads

- Optional
- Default Value: `FALSE`

`config.fix.data` An R function definition that is used to adjust the data read from the log files. This is provided in the sample report configuration file. The function must categorise records using the literals `"Static Content Requests"` and `"Monitoring"`. The function supplied in the sample.config file created by `workingDirectoryPopulate` is a good starting point and can be used to subset or correct the log data as it is read to focus on a smaller subset of records.

- Optional
- No Default Value

`config.fix.current.data` The function to be used to adjust baseline data if different cleaning functions are to be applied to current and baseline data

- Optional
- No Default Value

`config.fix.baseline.data` The function to use if different functions are to be applied to current and baseline data

- Optional
- Default value: `config.fix.current.data`

`config.tempdir` The name of the temp dir to be used for storage of generated graphics.

- Optional
- Default Value: ./txdata/

`config.useragent.generateFrequencies` Generate the User Agent Frequency Section of the report. Setting this to FALSE suppresses the report, which in any case is only produced if the current dataset contains user agent strings.

- Optional
- Default Value: TRUE

`config.useragent.minimumPercentage` The minimum percentage that a User Agent family or version must represent to be considered.

- Optional
- Default Value: 2

`config.useragent.maximumPercentile` The maximum cumulative percentile to report. The last few percent is made up of very low frequency of occurrence User agents that are not feasible (or in many cases possible) to test.

- Optional
- Default Value: 96

`config.useragent.discardOther` Discard browser family "Other" records. These are typically monitoring or heartbeat sources whose frequencies distort the percentile calculations.

- Optional
- Default Value: TRUE

`config.author` Name of the author of the report. Displayed on the default first page and in the page footer.

- Optional
- Default Value: Author

`config.securityClass` The security classification of the document. Displayed on the default first page and in the page footer.

- Optional
- Default Value: Commercial-In-Confidence

`config.longurls.threshold` The length in characters of a URL above which the URL text is replaced by a placeholder and the URL content is logged. Increasing this number will allow processing of longer URL text, but can lead to problems in LaTeX's processing the file.

- Optional
- Default Value: 1000

Value

configVariableIs returns a boolean to indicate existence of a named variable configVariableGet returns the value of the variable configVariableAll returns a list of all variables configVariableSet does not return a value

Author(s)

Greg Hunt <greg@firmansyah.com>

See Also

`workingDirectoryPopulate` `logFileRead` `logFileListRead`

Examples

```
configVariablesLoad(fileName=paste0(tempdir(),"/xx/sample.config"))
if(configVariableIs("config.documentName"))
{
  print(configVariableGet("config.documentName"))
}
allvars = configVariablesAll()
```

laTeXEscapeString *Escapes a string to enable it to be embedded in a LaTeX document*

Description

The string parameter is an R string that will have a regex applied to it so backslashes will need to be escaped before calling this.

Usage

```
laTeXEscapeString(nameString)
```

Arguments

nameString	The string to have escapes applied.

Value

Returns the escaped string

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
laTeXEscapeString("a$b")
```

laTexFilePercentileComparisonsWrite *Write percentile count comparison table*

Description

Writes a LaTeX table listing URLs in decreasing order of request count for a baseline workload compared with a current workload on stdout for incorporation in a LaTeX report

Usage

```
laTexFilePercentileComparisonsWrite(latest,
          baseline,
          headingLaTeX="\\section{Transaction Count Percentile Ranking}")
```

Arguments

latest	data frame of log records for the latest (test) workload
baseline	data frame of log records for the baseline workload
headingLaTeX	LaTeX section heading for this table

Value

Does not return a value.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

laTexFilePercentileComparisonsWrite(logdf,
                logdf)
```

---

laTeXParagraphWrite *Writes a LaTex paragraph on stdout*

Description

A convenience function to write a paragraph (with optional text) on stdout. This is useful in code blocks in Brew files, for example between graphics.

Usage

```
laTeXParagraphWrite(string="")
```

Arguments

string	Text to be inserted into the paragraph

Value

Does not return any value

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
laTeXParagraphWrite()
laTeXParagraphWrite("blah blah")
```

---

logFileFieldsGetIIS *Get field names from an IIS log file*

Description

Retrieves and validates the log fields from an IIS Log file. An IIS log file contains one or more comments records (leading hash) that identify the software that produced the log and the fields that were written. There may be multiple fields records, the code does not attempt to handle the case where different fields are written in different parts of the log. The MS names are mapped to the names used by this package.

Fields that the package does not use have names which begin with 'ignored: ' and these are dropped when the file is read.

Usage

```
logFileFieldsGetIIS(fileName)
```

Arguments

fileName	name of the file to be examined

Value

Returns a list of field names mapped to the

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileFieldsGetIIS(system.file("extdata", "compressed.log", package = "WebAnalytics"))
```

`logFileListRead` *Given a list of file names, read them as log files*

Description

This function calls logFileRead to read the individual log files.

Usage

```
logFileListRead(fileNameList, readFunction=logFileRead, columnList=NULL)
```

Arguments

`fileNameList`	A list of character file names
`readFunction`	This function is called to read the file name.
`columnList`	The columnList is a list of predefined column names. See `logFileRead` for the list of valid values.

Value

The function returns a dataframe that is the concatenation (rbind) of the read log files. If the default read function is used the data frame will contain the standard column set required by the other functions in this package.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
fileNameList = logFileNamesGetAll(dataDirectory=datd)

logdf = logFileListRead(fileNameList,
          readFunction=logFileRead,
          columnList=logFileFieldsGetIIS(fileNameList[[1]]))
```

`logFileNamesGet` *Base function for retrieval of file names from a base directory and a list of data directories*

Description

The function searches in a list of log file directories for log file names and returns a list of names found. A typical directory topology for log access is to have a number of per-server directories mounted or mapped under some common root and this function's parameters reflect that.

Switches are provided to return all or only the lexically last file names in each directory. Returning the lexically last file name works where a report is scheduled to be run late in a day so it will pick up the current day's file - this works for IIS logs and for typical Apache log rotation schemes.

Usage

```
logFileNamesGet(dataDirectory = getwd(),
            directoryNames=c("."),
            fileNamePattern=".*[.]log",
            allNamesMustMatch = TRUE,
            getLastFileName = TRUE)
```

Arguments

`dataDirectory`	a string containing the root directory under which the log file directories are found
`directoryNames`	a list of directory names that is concatenated with the data directory name. This is intended to support the structure where logs are collected into a series of per-server log directories.
`fileNamePattern`	a string containing a regular expression for the log file names that are to be processed
`allNamesMustMatch`	when processing files named for the period that they contain (for example a date in a typical IIS log file name) this ensures that the same log data period is processed from each log file directory. This does not guarantee that the content of the log files are complete, it just ensures that the log was written for the period. This check is only applicable when getLastFileName is true. If there are a number of files returned, there could be gaps in logging on one or more servers or servers may have failed over, so it does not make sense to check that the sets of logs from each directory are the same.
`getLastFileName`	Returns the lexically last file name from each data directory.

Value

Returns a list of character string file names.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
datd = dirname(system.file("extdata", "compressed.log", package = "WebAnalytics"))
logFileNamesGet(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log", allNamesMustMatch=FALSE)
```

---

logFileNamesGetAll *Get the list of file names matching a regex (default picks .log files) from a list of log directories*

Description

The function returns a list of names found in a list of directories.

Usage

```
logFileNamesGetAll(dataDirectory = getwd(),
  directoryNames=c("."),
  fileNamePattern=".*[.]log")
```

Arguments

dataDirectory	a string containing the root directory under which the log file directories are found
directoryNames	a list of directory names that is concatenated with the data directory name. This is intended to support the structure where logs are collected into a series of per-server log directories.
fileNamePattern	a string containing a regular expression for the log file names that are to be processed

Value

Returns a list of character string file names.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
datd = dirname(system.file("extdata", "compressed.log", package = "WebAnalytics"))
logFileNamesGetAll(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")
```

`logFileNamesGetLast` *Get lexically last file names from a list of log directories.*

Description

The function searches in a list of log file directories for log file names and returns a list of names found (the last name in each directory). This function does not check that the found file names are all the same from each directory. This is intended to be used to locate the most recent day's log file for reporting.

Usage

```
logFileNamesGetLast(dataDirectory = getwd(),
  directoryNames=c("."),
  fileNamePattern=".*[.]log")
```

Arguments

`dataDirectory`	a string containing the root directory under which the log file directories are found
`directoryNames`	a list of directory names that is concatenated with the data directory name. This is intended to support the structure where logs are collected into a series of per-server log directories.
`fileNamePattern`	a string containing a regular expression for the log file names that are to be processed

Value

Returns a list of character string file names.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
datd = dirname(system.file("extdata", "compressed.log", package = "WebAnalytics"))
logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")
```

logFileNamesGetLastMatching *Get lexically last file names from a list of log directories, checking that the file names are the same in all directories.*

Description

The function searches in a list of log file directories for log file names and returns a list of names found (the last name in each directory). This is intended to be used to locate the most recent day's log file for reporting.

Usage

```
logFileNamesGetLastMatching(dataDirectory = getwd(),
  directoryNames=c("."),
  fileNamePattern=".*[.]log")
```

Arguments

dataDirectory	a string containing the root directory under which the log file directories are found
directoryNames	a list of directory names that is concatenated with the data directory name. This is intended to support the structure where logs are collected into a series of per-server log directories.
fileNamePattern	a string containing a regular expression for the log file names that are to be processed

Value

Returns a list of character string file names.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
datd = dirname(system.file("extdata", "compressed.log", package = "WebAnalytics"))
logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")
```

`logFileRead` *Given a list of file names, read them as log files*

Description

This function reads a file, parsing it for the fields specified, and normalises the values that have been read.

The log file is assumed to be space delimited, which is the case for Apache and IIS.

Usage

```
logFileRead(fileName,
  columnList=c("MSTimestamp", "clientip", "url", "httpcode", "elapsed"),
  logTimeZone = "",
  timeFormat = "")
```

Arguments

`fileName`	The name, including path, of the file to read
`columnList`	The columns in the file, in order. Columns are:

`ApacheTimestamp`	Optional	Apache log format timestamp
`MSTimestamp`	Optional	IIS log format timestamp
`servername`	Optional	Name of the web server
`serverip`	Optional	IP of the server
`httpop`	Optional	HTTP verb
`url`	Required	Path part of the request
`parms`	Optional	Query string
`port`	Optional	TCP/IP port that the request arrived on
`username`	Optional	User name logged by the web server
`userip`	Optional	IP that the request was seen to originate from.
`useragent`	Optional	User agent string in the request
`httpcode`	Required	HTTP response code
`windowscode`	Optional	Windows return code recorded by IIS
`windowssubcode`	Optional	Windows sub code recorded by IIS
`responsebytes`	Optional	Number of bytes in the HTTP response
`requestbytes`	Optional	Number of bytes in the HTTP request
`elapsedms`	Optional	Request elapsed time in milliseconds
`elapsedus`	Optional	Request elapsed time in microseconds (will be rounded to milliseconds)
`elapseds`	Optional	Request elapsed time in seconds (not recommended, will be expanded to milliseconds)
`jsessionid`	Optional	User session identifier
`ignore*`	Optional	Columns with names starting with 'ignore' are dropped

One timestamp and one elapsed time column name must be specified.

The Apache URL is handled partly in the fix data procedure in the config file because it wraps the operation and URL path in one field. The IIS URL does

not need this additional parsing.

`logTimeZone` The timezone to use to adjust the timestamps in the log. This is used primarily for IIS logs where the log may be either UTC or local time.

`timeFormat` If the timestamp in the log is not in the default for IIS or Apache this can be used to override the timestamp parsing. The format is the r strptime format.

Value

The function returns a dataframe that contains the contents of the file.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")
```

---

pdfGenerate *Generate a PDF using the R API for TinyTeX*

Description

This function calls **brew** to generate a TeX document and then calls the TinyTeX API to generate a PDF from the LaTeX document in the specified work directory.

The function does not attempt to install TinyTex itself, this must be done by the user calling the `tinytex::install_tinytex()` function. The first time that this function is called to generate a report, provided that TinyTeX is has been installed with default settings, it will install the required LaTeX packages for the report. This may take some time.

Tinytex has been observed to run glacially slowly on Windows. For some users it may be preferable (and far quicker) to install some version of xelatex and run it with it set to automatically download packages.

Usage

```
pdfGenerate(configFile, templateFile="sampleRfile.R", workDir=".")
```

Arguments

configFile	The name of the config file to use. If this parameter is omitted, the work directory is searched for a file whose name ends, case insensitively, in .config. If there is only one file with the extnsion .config it is used. If there is no config file or multiple config files an error is signalled. The leading part of the config file name, before .config, is used as the prefix for the gemerated PDF name
templateFile	This defaults to sampleRfile.R which is the default name of the template.
workDir	The directory to use for document generation. This directory must contain the template and any associated support files, such as the LaTeX document class. workingDirectoryPopulate will populate this directory correctly and create a temporary files directory under it.

Value

Returns the name of the generated PDF.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
## Not run:
filesDir =  paste0(tempdir(),"/ex")
wkDir = paste0(tempdir(),"/ey")
configVariableSet("config.workdir", filesDir)
# setup the work directory
workingDirectoryPopulate(wkDir)

pdfGenerate(workDir=wkDir)


## End(Not run)
```

---

percentileBaselinePrint

Print a LaTeX table comparing current and baseline values and return a bar graph of the same data

---

Description

Calculate quantile values for a supplied numeric list. This is a wrapper around the R quantile function.

Usage

```
percentileBaselinePrint(column,
        baselineColumn,
        columnNames = c("Delta", "Current", "Baseline", "Percentile"))
```

Arguments

column	a vector of numeric values from the current dataset The values will be rounded to two decimal places before calculation.
baselineColumn	a vector of numeric values from the baseline data. The values will be rounded to two decimal places before calculation.
columnNames	names of the columns in the table that is printed by this function.

Value

Returns a ggplot graph of the data.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
fileNameList = logFileNamesGetAll(dataDirectory=datd)

logdf = logFileListRead(fileNameList,
          readFunction=logFileRead,
          columnList=logFileFieldsGetIIS(fileNameList[[1]]))

logbasedf = logFileListRead(fileNameList,
          readFunction=logFileRead,
          columnList=logFileFieldsGetIIS(fileNameList[[1]]))
plotWriteFilenameToLaTexFile(
  plotSaveGG(
    percentileBaselinePrint(logdf$elapsed,
              logbasedf$elapsed,
              columnNames = c("Delta", "Current", "Baseline", "Percentile"))
    , "xxx")
    )
```

plotByRate *Generates a plot that compares how percentile values in a metric of interest vary as an underlying rate metric changes.*

Description

Data is supplied as separate columns for time, data, base rate,

Usage

```
plotByRate(timecol,
      datacol,
      baseratecol,
      percentile,
      breaksString,
      baseratetimes = timecol,
      xlab = "Rate",
      ylab="Variation from overall 95th percentile",
      title="",
      baseTimeCol = NULL,
      baseDataCol=NULL,
      baseBaseRateCol = NULL,
      outlierPercentile=NULL)
```

Arguments

timecol	the timestamps for the data column
datacol	the data to be compared with the rate column (y axis)
baseratecol	the underlying rate (x axis on the plot), as a list of counts (setting a dataframe column to 1 works for a web server log dataframe). The rate is calculated over the breaksString interval, meaning that if you want to aggregate over a ten minute interval and report an indicative per-second rate the counts in this column must be scaled by dividing them by 600, the number of seconds in ten minutes.
percentile	the percentile value that the values are calculated at, the percentiles are calculated for each breaksString interval and converted to differences from the value of the percentile calculated over the whole period.
breaksString	the time interval used for calculation of percentile values, it is supplied to the cut function to group the data values.
baseratetimes	the timestamps for the base data if they are not the same as the timestamps for the data column (defaults to the data column timestamps). The two sets of timestamps must correspond in some way and the extent to which they do not align must be accounted for in the breaks interval, for example a few seconds or minutes difference would not be an issue for an aggregation interval of an hour, but would be a problem if aggregation is being done by minute or second
xlab	label for the rate metric

ylab	label for the data metric
title	the plot title
baseTimeCol	Use this parameteter and the associated baseDataCol as an alternative time base and data to summarise the data.
baseDataCol	Data values associated with the rate column above.
baseBaseRateCol	Rate column for the prevoious two columns.
outlierPercentile	discard data above this percentile

Value

Returns an R base graphics plot. This function is intended to be wrapped in a call to plotSave

Author(s)

Greg Hunt <greg@firmansyah.com>

See Also

cut

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

logdf$numrequestsinthisrecord = 1

plotByRate(logdf$ts,
      logdf$elapsed,
      logdf$numrequestsinthisrecord,
      0.95,
      "10 mins",
      xlab="request rate (10 minutes)",
      ylab="variance from overall 95th percentile response time (milliseconds)")
```

plotDataRateImpactOnResponse
Get list of latest files from log directories

Description

Generates a plot of 95th percentile response time for a specified combination of transaction and response status against aggregate data rate, for ten minute intervals in the dataframe provided.

Usage

```
plotDataRateImpactOnResponse(dataFrame, filterURL, status)
```

Arguments

dataFrame	a transaction data frame
filterURL	the URL to be examined
status	the status of the request: 'Success', 'Redirect', 'Client Error' or 'Server Error'

Value

Returns an R base graphics plot. This function is intended to be wrapped in a call to plotSave

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")
plotDataRateImpactOnResponse(logdf, "/QWERTYTest/XRMServices/2011/Organization.svc", "Success")
```

`plotDataRateImpactOnStaticResponse` *Plot static object response time against aggregate data rate*

Description

Generates a plot of 95th percentile response time for static objects against aggregate data rate, for ten minute intervals in the dataframe provided.

Usage

```
plotDataRateImpactOnStaticResponse(dataFrame)
```

Arguments

`dataFrame`	a transaction data frame

Value

Returns a base graphics plot. This function is intended to be wrapped in a call to `plotSave`

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotDataRateImpactOnStaticResponse(logdf)
```

`plotErrorRateByHour` *Plots rates of HTTP response code groups by hour*

Description

Generates a stacked bar plot of http response code types (2xx Success, 3xx Redirect, 4xx User Error and 5xx System Error) by hour.

The x-axis is hours and the plot is limited to 24 axis labels (optimally this is one day) regardless of how many days are being reported. This ensures that the labels are readable.

Usage

```
plotErrorRateByHour(dataFrame)
```

Arguments

`dataFrame`	a transaction data frame created by `logFileRead` or `logFileListRead`

Value

Returns a ggplot2 plot. This function is intended to be wrapped in a call to `plotSaveGG`

Author(s)

Greg Hunt <greg@firmansyah.com>

See Also

`logFileRead` `logFileListRead`

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotErrorRateByHour(logdf)
```

plotFrequencyHistogram *Plot histogram of response times for a transaction dataframe*

Description

Generates a plot of response time frequencies.

Times are expressed in seconds.

The histogram bin width is a minimum of 0.1 seconds, or 1/200 of the maximum elapsed time. The graph tries to show a minimum of 20 bins (2 seconds), for data with very small elapsed times this can lead to graphs with significant empty space on the right.

If the maximum elapsed is greater than 99 seconds, the x axis labels are rotated so that they do not overlap.

Usage

```
plotFrequencyHistogram(theDf)
```

Arguments

theDf	a transaction data frame

Value

Returns a ggplot2 plot. This function is intended to be wrapped in a call to plotSaveGG

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotFrequencyHistogram(logdf)
```

plotFrequencyHistogramOutlierCutoff *Plot frequencies of elapsed times up to a percentile cutoff*

Description

Given a column of values or a data frame created by readFileList or readFile, generate a ggplot of a frequency histogram excluding values above a specified percentile

Usage

```
plotFrequencyHistogramOutlierCutoff(theDf, outlierCutoff)
```

Arguments

theDf	Either a data frame created by readFile or readFileList or a vector of numeric values (assumed to be elapsed times).
outlierCutoff	A value between 0 and 1 which specifies the percentile above which values are excluded.

Details

The function is used to produce histogram plots of elapsed times with outliers excluded. It can accept either a list (which is converted to a data frame with the column named 'elapsed' or a data frame from a log file.

Value

Returns a ggplot2 plot. This function is intended to be wrapped in a call to plotSaveGG

Author(s)

Greg Hunt <greg@firmansyah.com>

See Also

logFileRead logFileListRead

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
```

```
            logTimeZone = "", timeFormat = "")

plotFrequencyHistogramOutlierCutoff(logdf, 0.95)
```

`plotParallelismRateImpactOnResponse` *Plot response time against degree of parallelism*

Description

Generates a plot of the effect of overall parallelism on response time possibly limited to a single URL in the overall background, the URL whose time is calculated, and by http response type and status.

Usage

```
plotParallelismRateImpactOnResponse(b,
      intervalLength = 600,
excludeURLOverall="",
includeURLOverall="",
excludeResponse="",
includeResponse="",
excludeStatus="",
includeStatus="",
percentileCutoff = 1,
title="Degree of Parallelism and Response Time",
subtitle="")
```

Arguments

`b`	a transaction data frame
`intervalLength`	length of the intervals (in seconds) that parallelism is calculated over
`excludeURLOverall`	a URL to be deleted from the dataset
`includeURLOverall`	the URL to be included from the dataset
`excludeResponse`	a URL to be excluded from the response time calculation
`includeResponse`	the URL to be used for the response time calculation
`excludeStatus`	a status to be excluded from the response time calculation, for example success statuses could be excluded. Possible status values are: 'Success', 'Redirect', 'Client Error' or 'Server Error'
`includeStatus`	a status to filter URLs by, for example, only include success responses in the response time calculation

`percentileCutoff`
: exclude values above the specified quantile, intended for use in excluding outlier events that would distort the elapsed time calculation

`title`
: the plot title

`subtitle`
: the plot subtitle

Value

Returns an R base graphics plot. This function is intended to be wrapped in a call to `plotSave`

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")
plotParallelismRateImpactOnResponse(logdf,
            includeStatus="Success",
            excludeResponse="Static Content Requests",
            percentileCutoff=0.95)
```

---

plotResponseTimeScatter

Generates a scatter plot of response times

Description

Scatter plot (base graphics object) of response times from a data. The log form uses a log10 y axis.

Usage

```
plotLogResponseTimeScatter(times, elapsed,timeDivisor = 1000, ylabText = "Time (log sec)")
plotResponseTimeScatter(times, elapsed,timeDivisor = 1000, ylabText = "Time (sec)")
```

Arguments

times	POSIXct timestamps
elapsed	elapsed milliseconds
timeDivisor	divisor to adjust times to seconds (the default value of 1000) or some other interval
ylabText	divisor to adjust times to seconds (the default value of 1000) or some other interval

Value

Generates a base graphics plot. This function is intended to be wrapped in a call to plotSave

Author(s)

Greg Hunt <greg@firmansyah.com>

See Also

savePlot

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]] # only want one

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotResponseTimeScatter(logdf$ts, logdf$elapsed)
plotLogResponseTimeScatter(logdf$ts, logdf$elapsed)
```

---

plotSave *Save a plot to a file with a generated name*

Description

These two functions save a base graphics plot function call or a ggplot object to a file with a generated name in the format (eps or jpg) specified and return the generated file name.

Usage

```
plotSaveGG(thePlot, fileID, fileType = "jpg")
plotSave(thePlot,
      fileID,
      fileType = "jpg",
      imageQuality=90,
      imageDefaultWidth=600,
      imageDefaultHeight=400)
```

Arguments

thePlot	Either a base graphics plot function call or a ggplot plot object. The base graphics function call is valuated within the function, not at the time of the call (A peculiarity of the R language)
fileID	A unique ID for the file. This ID is used to generate a hash which is used as the file name. The ID may contain any characters and can, for example be a URL which would not otherwise be a valid filename.
fileType	Either eps or jpg depending on the file format required. EPS files grow significantly as the number of data points grows. For very large data sets jpg is preferable.
imageQuality	The percent quality for JPG file construction. EPS files are metafiles and do not have a percent quality.
imageDefaultWidth	The width in pixels of the image
imageDefaultHeight	The height in pixels of the image

Value

The function returns the generated file name after creating the file.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotSaveGG(plotErrorRateByHour(logdf),"xxx", "eps")
```

```
plotSave(plotResponseTimeScatter(logdf$ts, logdf$elapsed), "yyy", "jpg")
```

`plotTransactionRateImpactOnDynamicContentResponse`
Generate a plot of mean transaction rate by interval against dynamic content response

---

Description

Calls `plotByRate` internally to generate a rate plot.

Usage

```
plotTransactionRateImpactOnDynamicContentResponse(b)
```

Arguments

b	a transaction data frame created by `logFileRead` or `logFileListRead`

Value

Returns an R base graphics plot. This function is intended to be wrapped in a call to `plotSave`

Author(s)

Greg Hunt <greg@firmansyah.com>

See Also

`plotByRate` `savePlot` `logFileRead` `logFileListRead`

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotTransactionRateImpactOnDynamicContentResponse(logdf)
```

plotWriteFilenameToLaTexFile *Write an includegraphic element to the generated LaTeX file*

Description

Writes an includegraphic element to the brew-generated LaTeX file. Written to be used in a series of nested calls as shown in the example.

Usage

```
plotWriteFilenameToLaTexFile(graphicFileName)
```

Arguments

graphicFileName	The name of the file to be included in the LaTeX

Value

Does not return a value

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

plotWriteFilenameToLaTexFile(plotSaveGG(plotErrorRateByHour(logdf),"xxx", "eps"))
```

printPercentiles *calculate quantile values from a column and print as an xtable vertically*

Description

Calculate quantile values for a supplied numeric list. .

Usage

```
printPercentiles(column,
      dataName,
      percentileList=c(0.7, 0.8, 0.9, 0.95, 0.96, 0.97, 0.98, 0.99, 1))
```

Arguments

column	a vector of numeric values. The values will be rounded to two decimal places before calculation.
dataName	the literal value to be used as the data name in the table
percentileList	a list of the quantile values that are to be calculated (as decimal values in the range 0 to 1)

Value

Does not return a value. It prints the xtable.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
dataValues = c(1,1,1,2,3,4,5,6,7,8,9,10,10,10,10)
printPercentiles(dataValues, "Random Data", percentileList=c(0.5, 0.75, 0.9))
```

summaryTxDataFrameCreate *Creates a dataframe containing summary URL performance metrics*

Description

From a dataframe containing log data, calculate 95th percentile response, total wait and error counts, embed TeX hyperlinks referencing the URL. Return these in a dataframe intended for printing in a report.

Usage

```
summaryTxDataFrameCreate(logDataframe)
```

Arguments

logDataframe	a dataframe created by the functions that read log files.

Value

Returns a dataframe containing columns

Response (sec, 95th pctl) 95th Percentile response time for the URL

Transaction The URL

Count Number of requests for that URL

Total Wait (sec) Total wait time for the URL in seconds

Server Errors Number of HTTP 5xx errors

Client Error Number of HTTP 4xx errors

Redirect Number of HTTP 3xx responses

Success Number of HTTP 200 responses

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)

logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

summarydf = summaryTxDataFrameCreate(logdf)
```

summaryTxTablePrint *Prints a transaction summary table generated by summaryTxDataFrameCreate*

Description

Formats and prints the LaTeX for a transaction summary table as generated by summaryTxDataFrameCreate.

The first column in the table is assumed to be a URL and is 0.6 times the width of the text page, subsequent column are formatted left or right aligned as appropriate for numeric or character variables.

Usage

```
summaryTxTablePrint(dataFrame, formatFunctions=NULL)
```

Arguments

dataFrame	transaction summary data frame generated by summaryTxDataFrameCreate
formatFunctions	a list of formatting functions that accept a value and return a formatted value. The sample report includes functions f0, f2 that format numbers with 0 or 2 respectively decimal places.

Value

Does not return a value. It prints the xtable.

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
f0<-function(n)
{
return(format(n,digits=1,nsmall=0,big.mark=","))
}
f2<-function(n)
{
return(format(n,digits=2,nsmall=2,big.mark=","))
}
logFileName = logFileNamesGetLast(dataDirectory=datd,
  directoryNames=c(".", "."),
  fileNamePattern="*[.]log")[[1]]

cols = logFileFieldsGetIIS(logFileName)
```

```
logdf = logFileRead(logFileName, columnList=cols,
            logTimeZone = "", timeFormat = "")

summarydf = summaryTxDataFrameCreate(logdf)

summaryTxTablePrint(summarydf[1:40,c(2,1,4,3)],list(format, f2, f2, f0))
```

`workingDirectoryPopulate` *Create files in the working directory to be used for report generation*

Description

The function creates the specified directory (if it does not already exist) and creates the configured file directory (by default the name is 'txdata') if that does not exist. If the files directory has an absolute path it is created, if it is a relative path it is appended to the work directory path and created. The work directory contains the report template and scripts, the files directory holds the images that are generated by the report template. It creates the following files:

makerpt.ps1 PowerShell script to generate the report. The script is run by specifying its name followed by the stem part of the R file that contains the script `.\makerpt.ps1 reportname`

makerpt.sh bash script to generate the report. The script is run by specifying its name followed by the stem part of the R file that contains the script `./makerpt.sh reportname`

size10.clo A file used by LaTex that specifies the page layout

webanalyticsreport.cls A latex document class that the report uses

logo.eps A 2cm square eps logo placed in the top right corner of each report page

sampleRfile.R An outline R report file

If a file exists and has different content to the new file it will be renamed (a timestamp suffix added to the name) before the new file is created.

Usage

```
workingDirectoryPopulate(directoryName = ".")
```

Arguments

`directoryName`	A character string that specifies the name of the directory to be created and populated

Value

The function does not return a value

Author(s)

Greg Hunt <greg@firmansyah.com>

Examples

```
# in the current directory run it as
# workingDirectoryPopulate()


workingDir = paste0(tempdir(),"/wk")

workingDirectoryPopulate(workingDir)
```

Index