Package 'googleErrorReportingR'

October 27, 2022

Title Send Error Reports to the Google Error Reporting Service API

Version 0.0.4

Description Send error reports to the Google Error Reporting service <https://cloud.google.com/error-reporting/> and view errors and assign error status in the Google Error Reporting user interface.

License MIT + file LICENSE

URL https://github.com/ixpantia/googleErrorReportingR,
https://ixpantia.github.io/googleErrorReportingR/

Encoding UTF-8

Imports jsonlite, httr, magrittr

Suggests knitr, rmarkdown, pkgdown, testthat (>= 3.0.0)

RoxygenNote 7.2.1

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author ixpantia, SRL [cph],
Frans van Dunné [cre, aut] (<https://orcid.org/0000-0002-7853-2811>),
Andrea Vargas [ctb]

Maintainer Frans van Dunné <frans@ixpantia.com>

Repository CRAN

Date/Publication 2022-10-27 15:25:13 UTC

R topics documented:

config_message 2
format_error_message 2
list_errors 4
report_error 4

Index **6**

config_message *Format messages before sending to Google Error Reporting*

Description

Format messages before sending to Google Error Reporting

Usage

```
config_message(message, req, status_code)
```

Arguments

message	Error message returned by validation
req	Request object used by plumber
status_code	Valid HTTP status code

Value

formatted message

Examples

```
## Not run:
your_function_call <- tryCatch(
  your_function(),
  error = function(e) {
    message$message <- as.character(e)
    googleErrorReportingR::report_error(message)
    message <- config_message(message, req, "401")
    stop("Error", call. = FALSE)
  })

## End(Not run)
```

format_error_message *Format Error Message for Google Error Reporting*

Description

Format Error Message for Google Error Reporting

Usage

```
format_error_message(
  message = "Error description",
  service = "My Service",
  version = "0.0.1",
  method = "GET",
  url = "https://example.com",
  user_agent = "",
  referrer = "",
  response_status_code = "500",
  remote_ip = "192.178.0.0.1",
  user_id = "UserID",
  filepath = "/",
  line_number = 0,
  function_name = "my_function"
)
```

Arguments

`message`	the error message you want in the logs
`service`	the name of your service
`version`	the version of the service
`method`	the http method used for hte call
`url`	hte unique resource identifier that was called
`user_agent`	the user agente identifier
`referrer`	the referrer to the service
`response_status_code`	http response code
`remote_ip`	remote ip
`user_id`	user id
`filepath`	filepath of the code where the error originates
`line_number`	line number where the error originates
`function_name`	function name where the error originates

Value

message object, a list to be formated as JSON in the error report body

Examples

```
## Not run:
message <- format_error_message()

message$serviceContext$service <- "A demo service"
message$serviceContext$version <- "v0.3.4"

## End(Not run)
```

list_errors *Get list of errors from Google Error Reporting*

Description

Get list of errors from Google Error Reporting

Usage

```
list_errors(project_id, api_key)
```

Arguments

project_id	the project ID of your project on GCP
api_key	an API key with permissions to write to Google Error Reporting

Value

No return, we focus on side effect

report_error *Report error to Google Error Reporting*

Description

Report error to Google Error Reporting

Usage

```
report_error(message, project_id = NULL, api_key = NULL)
```

Arguments

message	the error report to be written out to the
project_id	the project id where you want to monitor the error reports
api_key	the google API key with authorisation to write to the Google Error Reporting API

Value

No return, we focus on side effect

Examples

```
## Not run:
report_error(project_id, api_key, message)

#If you have set the environmental variables "PROJECT_ID" and
#"ERROR_REPORTING_API_KEY" then you can make shorter calls like so
report_error(message)

## End(Not run)
```

Index

config_message, 2

format_error_message, 2

list_errors, 4

report_error, 4