

Package ‘gfonts’

January 11, 2021

Title Offline 'Google' Fonts for 'Markdown' and 'Shiny'

Version 0.1.2

Description Download 'Google' fonts and generate 'CSS' to use in 'rmarkdown' documents and 'shiny' applications. Some popular fonts are included and ready to use.

URL <https://github.com/dreamRs/gfonts>

BugReports <https://github.com/dreamRs/gfonts/issues>

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 2.10)

Imports utils, htmltools, shiny, crul, jsonlite, glue, usethis

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, testthat (>= 2.1.0), vcr, covr

VignetteBuilder knitr

NeedsCompilation no

Author Victor Perrier [aut, cre],
Fanny Meyer [aut],
Mario Ranftl [ctb, cph] (google-webfonts-helper)

Maintainer Victor Perrier <victor.perrier@dreamrs.fr>

Repository CRAN

Date/Publication 2021-01-11 11:10:02 UTC

R topics documented:

download_font	2
generate_css	3
get_all_fonts	4
get_font_info	5
gfonts	5

included_fonts	6
setup_font	6
tag_example	7
use_font	8
use_pkg_gfont	10

Index	12
--------------	-----------

download_font	<i>Download font files</i>
---------------	----------------------------

Description

Download font files

Usage

```
download_font(id, output_dir, variants = NULL, ..., http_options = list())
```

Arguments

id	Id of the font, correspond to column id from get_all_fonts .
output_dir	Output directory where to save font files.
variants	Variant(s) to download, default is to includes all available ones.
...	Additional parameters to API query.
http_options	Arguments passed to <code>curl::HttpClient\$new</code> .

Value

a character vector of the filepaths extracted to, invisibly.

Examples

```
if (interactive()) {

# For example, we use a temporary directory
path_to_dir <- tempfile()
dir.create(path_to_dir)

# Download Roboto font
download_font(
  id = "roboto",
  output_dir = path_to_dir
)

# Get only regular, italic and bold
download_font(
  id = "roboto",
  output_dir = path_to_dir,
```

```

    variants = c("regular", "300italic", "700")
  )

  # Clean up
  unlink(path_to_dir, recursive = TRUE)

}

```

generate_css

Generate CSS to import fonts

Description

Generate CSS to import fonts

Usage

```

generate_css(
  id,
  variants = NULL,
  subsets = NULL,
  output = NULL,
  font_dir = "../fonts/",
  prefer_local_source = TRUE,
  ...
)

```

Arguments

id	Id of the font, correspond to column id from get_all_fonts .
variants	Variant of font to use.
subsets	Subsets to use.
output	Specifies path to output file for CSS generated.
font_dir	Fonts directory relative to ouput.
prefer_local_source	Generate CSS font-face rules in which user installed fonts are preferred. Use FALSE if you want to force the use of the downloaded font.
...	Arguments passed to <code>curl::HttpClient\$new</code> .

Value

a character string with CSS code (invisibly).

Examples

```
if (interactive()) {  
  
  # Generate CSS code to use Roboto font  
  cat(generate_css("roboto", "regular"))  
  
}
```

get_all_fonts	<i>Get infos about all fonts available</i>
---------------	--

Description

Retrieve from API all fonts currently available. Use the id field in other functions to reference the font you want to use.

Usage

```
get_all_fonts(...)
```

Arguments

... Arguments passed to `curl::HttpClient$new`.

Value

a data.frame.

Examples

```
if (interactive()) {  
  
  # Retrieve all fonts currently available  
  all_fonts <- get_all_fonts()  
  
}
```

get_font_info	<i>Get detailed information about a font</i>
---------------	--

Description

Get detailed information about a font

Usage

```
get_font_info(id, subsets = NULL, ...)
```

Arguments

id	Id of the font, correspond to column id from get_all_fonts .
subsets	Select charsets, for example "latin".
...	Arguments passed to <code>curl::HttpClient\$new</code> .

Value

a data.frame.

Examples

```
if (interactive()) {  
  # Info about Roboto  
  roboto <- get_font_info("roboto")  
}
```

gfonts	<i>Use Google fonts offline</i>
--------	---------------------------------

Description

Download Google fonts and generate CSS to use in rmarkdown documents and shiny applications. Some popular fonts are included and ready to use.

Download a font

Use [setup_font](#) to get a font inside your current project, then in a {shiny} application or {rmarkdown} document, you can use [use_font](#) to import the font.

Ready-to-use fonts

Some fonts are included in this package and can be used directly with [use_pkg_gfont](#).

Author(s)

Victor Perrier (@dreamRs_fr)

included_fonts *Detail about included fonts.*

Description

Id and version of fonts included and available through [use_pkg_gfont](#).

Usage

```
included_fonts
```

Format

A data frame with 8 rows and 5 variables:

id Id for the font.

family Name of the font.

category Category.

version Version number.

lastModified Last modified date.

Source

<https://google-webfonts-helper.herokuapp.com/fonts>

setup_font *Setup a font to be used in Shiny or Markdown*

Description

This function will download the specified font into a directory of your project and generate CSS code to use it in a Shiny application or RMarkdown document.

Usage

```
setup_font(  
  id,  
  output_dir,  
  variants = NULL,  
  subsets = NULL,  
  prefer_local_source = TRUE,  
  ...  
)
```

Arguments

id	Id of the font, correspond to column id from get_all_fonts .
output_dir	Output directory where to save font and CSS files. Must be a directory.
variants	Variant(s) to download, default is to includes all available ones.
subsets	Subsets to download.
prefer_local_source	Generate CSS font-face rules in which user installed fonts are preferred. Use FALSE if you want to force the use of the downloaded font.
...	Arguments passed to <code>curl::HttpClient\$new</code> .

Value

None.

Note

Two directories will be created (if they do not exist) in the `output_dir` specified: **fonts/** and **css/**.

Examples

```
if (interactive()) {

  # For example, we use a temporary directory
  path_to_www <- tempfile()
  dir.create(path_to_www)

  # In a Shiny app, you can use the www/ directory
  # in Markdown, use a subfolder of your Rmd directory
  setup_font(
    id = "open-sans-condensed",
    output_dir = path_to_www
  )

  # Clean up
  unlink(path_to_www, recursive = TRUE)

}
```

tag_example

Generate HTML tags used in examples

Description

Generate HTML tags used in examples

Usage

```
tag_example(class = NULL)
```

Arguments

class Class of the main div.

Value

HTML tags.

Examples

```
tag_example()
```

use_font

Use a downloaded font in Shiny or Markdown

Description

Use a downloaded font in Shiny or Markdown

Usage

```
use_font(id, css_path, selector = "body", css = NULL)
```

Arguments

id Id of the font downloaded.

css_path Path to CSS generated by [setup_font](#).

selector CSS selector for which to use the font, usually an HTML tag, default to "body" (all document).

css CSS variables needed to use font, normally this should be automatic.

Value

an HTML tag with an HTML dependency ([htmlDependency](#)).

Examples

```
if (interactive()) {
  library(gfonts)

  # Here we use a temp directory
  # but in Shiny, it can be www/ folder
  directory <- tempfile()
  dir.create(directory)

  # Setup a font (only needed once)
  setup_font(
    id = "dancing-script",
```



```
    output_dir = directory
  )

  library(shiny)

  ui <- fluidPage(

    # Use font
    use_font(
      id = "dancing-script",
      css_path = file.path(directory, "css/dancing-script.css")
    ),

    tags$p(
      paste(letters, collapse = "")
    ),
    tags$p(
      paste(LETTERS, collapse = "")
    ),
    tags$p(
      style = "font-weight: bold;",
      paste(letters, collapse = "")
    ),
    tags$p(
      style = "font-weight: bold;",
      paste(LETTERS, collapse = "")
    ),
    tags$p(
      style = "font-style: italic;",
      paste(letters, collapse = "")
    ),
    tags$p(
      style = "font-style: italic;",
      paste(LETTERS, collapse = "")
    ),
    tags$h1("First level title"),
    tags$h2("Second level title"),
    tags$h3("Third level title"),
    tags$h4("Fourth level title"),
    tags$h5("Fifth level title"),
    tags$h6("Sixth level title")
  )

  server <- function(input, output, session) {

  }

  shinyApp(ui, server)
}
```

use_pkg_gfont	<i>Use a Google Font included in gfonts</i>
---------------	---

Description

For convenience, some fonts are included in the package, you can use them without having to download them, but only few variants are available.

Usage

```
use_pkg_gfont(
  font = c("roboto", "open-sans", "lato", "montserrat", "alegreya", "nunito-sans",
    "baloo", "happy-monkey", "henny-penny"),
  selector = "body"
)
```

Arguments

font	Name of the font to use, possible choices are: "roboto", "open-sans", "lato", "montserrat", "alegreya", "nunito-sans", "baloo", "happy-monkey", "henny-penny".
selector	CSS selector for which to use the font, usually an HTML tag, default to "body" (all document).

Value

An HTML tag with an [htmlDependency](#).

Examples

```
if (interactive()) {
  library(gfonts)
  library(htmltools)

  browsable(tags$div(
    use_pkg_gfont("open-sans"),
    tag_example(),
    tags$h1("First level title"),
    tags$h2("Second level title"),
    tags$h3("Third level title"),
    tags$h4("Fourth level title"),
    tags$h5("Fifth level title"),
```

```
        tags$h6("Sixth level title")
    ))

    browsable(tags$div(
      use_pkg_gfont("henny-penny"),
      tag_example(),
      tags$h1("First level title"),
      tags$h2("Second level title"),
      tags$h3("Third level title"),
      tags$h4("Fourth level title"),
      tags$h5("Fifth level title"),
      tags$h6("Sixth level title")
    ))
  }
```

Index

* datasets

- included_fonts, [6](#)

- download_font, [2](#)

- generate_css, [3](#)
- get_all_fonts, [2](#), [3](#), [4](#), [5](#), [7](#)
- get_font_info, [5](#)
- gfonts, [5](#)

- htmlDependency, [8](#), [10](#)

- included_fonts, [6](#)

- setup_font, [5](#), [6](#), [8](#)

- tag_example, [7](#)

- use_font, [5](#), [8](#)
- use_pkg_gfont, [5](#), [6](#), [10](#)